

**DATA EXCHANGE STANDARDS FOR  
REGISTRY SYSTEMS UNDER THE KYOTO PROTOCOL**

**TECHNICAL SPECIFICATIONS (Version 1.1.11)**

**24 November 2013**

*[This page intentionally left blank.]*

# Revision History

<b>Date</b>	<b>Version</b>	<b>Description</b>	<b>Author</b>
31/10/2003	1.0	Draft #1	Andrew Howard
14/11/2003	1.0	Draft #2	Andrew Howard
21/11/2003	1.0	Draft #3	Andrew Howard
27/05/2004	1.0	Draft #4	Andrew Howard
08/06/2004	1.0	Draft #5	Andrew Howard
30/07/2004	1.0	Draft #6	Andrew Howard
03/11/2004	1.0	Draft #7	Andrew Howard
18/12/2004	1.0	Final	Andrew Howard
29/11/2006	1.1	Final	Andrew Howard
21/08/2008	1.1.1	Final	David Sturt
26/03/2009	1.1.2	Final	Frank Ng
13/11/2009	1.1.3	Final	Frank Ng
20/01/2010	1.1.4	Draft	Frank Ng
16/03/2010	1.1.5	Final	Goran Milenkovic, Frank Ng
07/05/2010	1.1.6	Draft	Frank Ng
30/09/2010	1.1.7	Final	Frank Ng
16/12/2008	1.1.8	Final	Frank Ng
13/09/2012	1.1.9	Final	Markwin Pieters
25/01/2013	1.1.9	Final Formatted in UNFCCC template	Markwin Pieters, Ling Ling Federhen
24/04/2013	1.1.10	Final	Youssef Wabi
24/11/2013	1.1.11	Final	Youssef Wabi

**DES 1.1.11**

Section	Version	Description	Comment
Main document	1.1.11	Added check 5166 - New check to ensure use of units with adequate applicable period for net source cancellations	RSNCM-48
Annex E	1.1.12	Added check 5166 - New check to ensure use of units with adequate applicable period for net source cancellations	RSNCM-48

## Table of Contents

<b>1</b>	<b>Introduction .....</b>	<b>1</b>
1.1	PURPOSE .....	1
1.2	INTENDED AUDIENCE .....	1
1.3	SCOPE .....	1
1.4	DEFINITIONS, ACRONYMS, ABBREVIATIONS AND TERMINOLOGY .....	4
1.5	DERIVATION DOCUMENTS .....	4
1.6	MULTIPLE LANGUAGE SUPPORT .....	4
1.7	VALIDITY OF DATA .....	4
<b>2</b>	<b>Assumptions and Constraints .....</b>	<b>5</b>
2.1	ASSUMPTIONS .....	5
2.2	CONSTRAINTS .....	5
<b>3</b>	<b>Data Exchange Mechanism Specifications .....</b>	<b>6</b>
3.1	GENERAL REQUIREMENTS .....	6
3.2	COMMUNICATIONS SPECIFICATIONS .....	7
3.3	DATA TRANSFER SECURITY .....	7
3.3.1	Firewall and Network Interconnection .....	7
3.3.2	Virtual Private Network (VPN) .....	7
3.3.3	SSL .....	8
3.3.4	Certificate Authority .....	8
3.4	THE COMMUNICATIONS HUB AND MESSAGE QUEUE .....	8
3.5	DATA TRANSFER FORMAT SPECIFICATIONS .....	8
3.6	USER ACCOUNTS .....	8
3.7	TIME VALIDATION SPECIFICATIONS .....	9
3.8	WEB SERVICE TIMEOUT BEHAVIOUR .....	10
3.8.1	Timeout processing .....	10
3.8.2	What to do before sending the http response .....	10
3.8.3	Extending the timeout period past 60 seconds .....	11
<b>4</b>	<b>Unit Transaction Processes .....</b>	<b>13</b>
4.1	UNIT TRANSACTION TYPES .....	13
4.1.1	Issuance .....	13
4.1.2	Conversion .....	13
4.1.3	External Transfer .....	13
4.1.4	Cancellation .....	14
4.1.5	Replacement .....	15
4.1.6	Retirement .....	15
4.1.7	Carry-over Process .....	15
4.1.8	Expiry Date Change .....	16
4.1.9	Internal Transfers and Other Transactions Routed to an STL .....	16
4.2	DESCRIPTION OF DATA EXCHANGE FLOW .....	16
4.2.1	UML Notation for Transactional Message Flow .....	17
4.3	SINGLE REGISTRY MODEL .....	19
4.3.1	Single Registry Transaction Sequence Diagrams .....	20
4.3.2	Single Registry Transaction State Transitions .....	23
4.4	TWO REGISTRY TRANSACTION MODEL .....	28
4.4.1	Two Registry Transaction Sequence Diagrams .....	29
4.4.2	Two Registry Transaction State Transitions .....	33
4.5	LIST OF FUNCTIONS FOR TRANSACTION DATA EXCHANGE .....	42
4.5.1	Registry Web Services and Functions .....	42
4.5.2	ITL Web Services and Functions .....	42
4.6	TRANSACTION CHECKS .....	43
4.6.1	Version and Authentication Checks .....	43
4.6.2	Message Viability Checks .....	43
4.6.3	System Status Checks .....	43
4.6.4	Data Integrity Checks for Transactions .....	44
4.6.5	Message Sequence Checks for Transactions from Registries .....	44

4.6.6	General Transaction Checks .....	45
4.6.7	Transaction-specific Checks .....	45
<b>5</b>	<b>Reconciliation Process .....</b>	<b>46</b>
5.1	RECONCILIATION PROCESS FLOW .....	46
5.1.1	Reconciliation Behaviour Diagrams .....	50
5.2	RECONCILIATION STAGE TABLES .....	51
5.3	LIST OF FUNCTIONS FOR RECONCILIATION PROCESS .....	54
5.3.1	Registry Web Services and Functions .....	54
5.3.2	ITL Web Services and Functions .....	54
5.4	RECONCILIATION CHECKS AND RESPONSES .....	55
5.4.1	Version and Authentication Checks for Reconciliation .....	55
5.4.2	Registry Validation Checks for Reconciliation .....	55
5.4.3	Data Integrity Checks for Reconciliation .....	55
5.4.4	Message Sequence Checks for Reconciliation Messages Received from Registries ....	55
5.4.5	Other Reconciliation Checks and Messages .....	55
<b>6</b>	<b>ITL Administrative Functions .....</b>	<b>56</b>
6.1	24-HOUR TRANSACTION CLEAN-UP .....	56
6.2	NOTIFICATIONS .....	57
6.2.1	Net Source Cancellation .....	59
6.2.2	Non-compliance Cancellation .....	59
6.2.3	Impending tCER or ICER Expiry .....	59
6.2.4	Reversal of Storage for CDM Project .....	59
6.2.5	Non-submission of Certification Report for CDM Project .....	59
6.2.6	Excess Issuance for CDM Project .....	60
6.2.7	Commitment Period Reserve .....	60
6.2.8	Unit Carry-over .....	60
6.2.9	Expiry Date Change .....	60
6.2.10	Notification Update .....	60
6.2.11	EU15 Commitment Period Reserve .....	61
6.3	GENERAL MESSAGES .....	63
6.4	TRANSACTION STATUS SERVICE .....	64
6.5	TIME SYNCHRONIZATION .....	64
6.6	HEARTBEAT CONNECTION HEALTH MONITORING BY STLS .....	65
<b>7</b>	<b>Data Logging Specifications .....</b>	<b>66</b>
7.1	TRANSACTION LOG .....	66
7.2	RECONCILIATION HISTORY LOG .....	67
7.3	NOTIFICATION LOG .....	69
7.4	INTERNAL AUDIT LOG .....	70
7.5	MESSAGE ARCHIVE .....	70
<b>8</b>	<b>Change Management Specifications .....</b>	<b>71</b>
8.1	OBJECTIVES .....	71
8.2	TECHNICAL SPECIFICATIONS .....	71
8.2.1	Version Definition .....	71
8.3	ITL WEB PORTAL .....	71
8.3.1	Web Service Modifications .....	71
8.3.2	Support Table Content Modification .....	72
<b>9</b>	<b>Registry Initialization Specifications .....</b>	<b>73</b>
9.1	STAFF IDENTIFICATION AND PLANNING .....	73
9.2	DOCUMENTATION .....	74
9.2.1	Database and Application Backup .....	74
9.2.2	Disaster Recovery Plan .....	74
9.2.3	Security Plan .....	75
9.2.4	Application Logging Documentation .....	75
9.2.5	Time Validation Plan .....	76
9.2.6	Version Change Management .....	76
9.2.7	Test Plan and Test Report .....	76

9.2.8	Operational Plan .....	77
9.3	INITIALIZATION TESTS .....	77
9.4	COMMUNICATION INITIALIZATION.....	78
9.5	ACCESS TO ITL WEBSITE.....	79
9.5.1	Public ITL Website .....	79
9.5.2	Access to ITL Extranet.....	79
9.6	WEB SERVICES TESTING .....	79
9.7	REQUEST FOR OTHER DATA .....	79
9.8	DATA IDENTIFIER INITIALIZATION.....	80
9.9	FULL SYSTEM TEST .....	80
9.10	RECONCILIATION SERVICES AND SCHEDULE .....	80
9.11	GOVERNMENT ACCOUNT INFORMATION .....	80

## List of Figures

Figure 1.1: Communication Via the Data Exchange Standards .....	2
Figure 3.1: Data Exchange Architecture .....	6
Figure 3.2: Process Time Stamps .....	10
Figure 3.3: Timeout period .....	10
Figure 3.4: When to send the http response .....	11
Figure 4.1: Single registry transaction: no STL, successful transaction finalisation .....	21
Figure 4.2: Single registry transaction: STL, successful transaction finalisation .....	21
Figure 4.3: Single registry transaction: ITL rejected .....	22
Figure 4.4: Single registry transaction: STL rejected .....	22
Figure 4.5: State Transition Diagram: Single Registry Transactions, not involving STLs .....	23
Figure 4.6: State Transition Diagram: Single Registry Transactions, involving STLs .....	25
Figure 4.7: Two registry transaction: no STL, successful transaction finalisation .....	30
Figure 4.8: Two registry transaction: STL, successful transaction finalisation .....	30
Figure 4.9: Two registry transaction: ITL rejected .....	31
Figure 4.10: Two registry transaction: STL rejected .....	31
Figure 4.11: Two registry transaction: no STL, Acquiring Registry rejected .....	32
Figure 4.12: Two registry transaction: STL, Acquiring Registry rejected .....	32
Figure 4.13: State Transition Diagram: Two Registry Transactions, not involving STLs .....	33
Figure 4.14: State Transition Diagram: Two Registry Transactions, not involving STLs .....	37
Figure 5.1: Reconciliation Behaviour Diagram .....	50
Figure 5.2: Send Reconciliation Results Behaviour Diagram .....	51
Figure 6.1: Transaction Clean-up Diagram .....	57
Figure 6.2: ITL Notification .....	58
Figure 6.3: Get Transaction Status Diagram .....	64
Figure 6.4: Time Synchronization Diagram .....	65
Figure 7.1: Transaction Log Entity Relationship Diagram .....	67
Figure 7.2: Reconciliation History Log Entity Relationship Diagram .....	69

## List of Tables

Table 4.1: Cancellation Transaction Information .....	14
Table 4.2: Replacement Transaction Information .....	15
Table 4.3: Key to UML Sequence Diagrams in Transactional Message Flow Descriptions .....	17
Table 4.4: Key to State Transition Diagrams in Transaction Message Flow Descriptions .....	18
Table 4.5: State Descriptions - Single Registry Non-STL Transactions .....	23
Table 4.6: ITL State Transition Event Descriptions - Single Registry Non-STL Transactions .....	24
Table 4.7: State Descriptions - Single Registry STL Transactions .....	25
Table 4.8: ITL State Transition Event Descriptions - Single Registry STL Transactions .....	27
Table 4.9: State Descriptions - Two Registry Non-STL Transactions .....	33
Table 4.10: ITL State Transition Event Descriptions - Two Registry Non-STL Transactions .....	35
Table 4.11: State Descriptions - Two Registry STL Transactions .....	37
Table 4.12: ITL State Transition Event Descriptions - Two Registry STL Transactions .....	40
Table 4.13: Registry Public Web Service Methods .....	42
Table 4.14: Registry Internal Functions .....	42
Table 4.15: System Status Descriptions .....	43
Table 5.1: Notation used in Reconciliation Behaviour Diagrams .....	49
Table 5.2: Confirm Reconciliation .....	51
Table 5.3: Reconciliation Phase 1 - Validate Account Totals .....	52
Table 5.4: Reconciliation Phase 2 - Validate Unit Blocks .....	52
Table 5.5: Reconciliation Phase 3 - Review Audit Logs .....	53
Table 5.6: Registry Public Web Service Methods .....	54
Table 5.7: Registry Internal Functions .....	54
Table 6.1: ITL Notification Data Elements .....	61
Table 7.1: Transaction Log Attributes .....	66
Table 7.2: Reconciliation History Log Attributes .....	68
Table 7.3: Notification Log Attributes .....	69
Table 7.4: Internal Log Attributes .....	70
Table 9.1: Table of Initialization Tests .....	78
Table 9.2: Look-up Table Initialization .....	80
Table 9.3: Government Account File Specifications .....	81



# 1 Introduction

## Purpose

This document contains technical specifications for data exchange between registries and the Independent Transaction Log (ITL) under the Kyoto Protocol. This exchange of data forms the technical basis for transactions under the mechanisms defined in Articles 6, 12 and 17 of the Kyoto Protocol and the modalities for the accounting of assigned amounts (to demonstrate compliance with emission targets) under Article 7.4 of the Kyoto Protocol.

These technical specifications contain full information on *how* the data exchange standards are to be implemented. They are based on the functional specifications for data exchange, which define in broader terms *what* data are exchanged and *by whom*. The technical specifications are necessary to ensure that the registries and the ITL employ consistent data exchange and messaging functionality.

The design of the ITL provides for the complementary functioning of supplementary transaction logs (STLs) developed by groups of Parties under the Kyoto Protocol. Such STLs are to conduct additional activities in relation to the transactions of those Parties under the Kyoto Protocol and under regional trading schemes. This complementary functionality is designed to avoid the duplication of validity checks and ensure consistent results between transaction logs. It further serves to integrate electronic communications between the relevant registries.

At time of writing, the only STL undergoing development is the Community Independent Transaction Log (CITL) for the European Union greenhouse gas emissions trading scheme. This is being developed under Article 20 of EU Directive 2003/87/EC.

## Intended Audience

This document is to guide technical experts in the design, development and implementation of communication functionality in registries and the ITL.

## Scope

The data exchange standards define how data are to be exchanged between national registries, the CDM Registry and the ITL under the Kyoto Protocol, as well as any STLs established. The Technical Specifications include the communication protocols to be used and a messaging architecture that includes an overall design for message management, message content, and data transfer formats. They define in detail the specific data elements to be exchanged between registry systems to support designated functionality throughout the process.

The diagram in Figure 1.1 demonstrates how both national registries and the CDM Registry will both send and receive messages enabling two-way communications exchanges to the ITL through a Communications Hub. The figure further demonstrates how messages will be forwarded to any STLs and will be returned by them to the ITL.

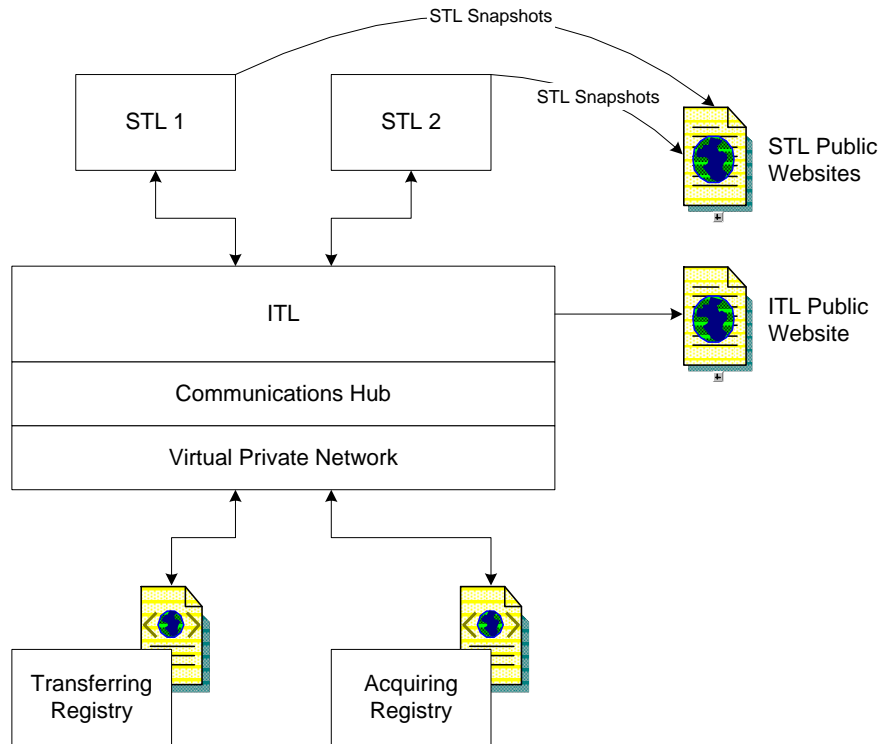
These technical specifications include:

### **Section 2 Assumptions and Constraints**

Facts and constraints identified in the Functional Specifications and held to be true for the Technical Specifications to be valid.

### **Section 3 Data Exchange Mechanism Specifications**

Specifications relating to registration, authentication and communication protocols required.



**Figure 1.1: Communication Via the Data Exchange Standards**

**Section 4 Unit Transaction Processes**

Specifications for message exchange relating to unit transactions.

**Section 5 Reconciliation Processes**

Specifications relating to the process of reconciliation.

**Section 6 ITL Administrative Processes**

Specifications regarding notifications to registries, time checks and message cancellations.

**Section 7 Data Logging Specifications**

Specifications for retaining records, utilizing internal logs and communicating transaction data for reconciliation.

**Section 8 Change Management Specifications**

Specifications to manage and distribute information on changes in the data content, messages, or message sequences to accommodate new requirements.

**Section 9 Registry Initialization Specifications**

Specifications on the start-up processes a registry will be required to complete before initiating communication and data exchange with the ITL.

**Annex A Glossary of Terms**

This annex provides definitions, acronyms and abbreviations relevant to this document.

**Annex B Web Service Operations and Functions for Transaction Processing**

This annex contains the detailed specifications for the Web services and programming functions which receive and/or generate transaction messages.

**Annex C Web Services Operations and Functions for Reconciliation**

This annex contains the detailed specifications for the Web services and programming functions relating to reconciliation.

**Annex D Web Service Operations and Functions for Administrative Processes**

This annex contains the detailed specifications for the Web services and programming functions which receive and/or generate administrative messages.

**Annex E List of Checks and Response Codes for Transaction Processing**

This annex identifies the categories of transaction responses and provides a numeric list of responses.

**Annex F Definition of Identifiers**

This annex provides detailed specifications and rules for creating and using identifiers for entities for which information is exchanged.

**Annex G List of Codes**

This annex identifies the codes which are used to represent a variety of categories, types, and statuses which may be contained in messages.

**Annex H Test Protocols for Data Exchange Specification Implementation**

This annex addresses the test requirements for verifying conformance with the Data Exchange Specifications Version 1.0.

**Annex I Messaging Service Specification**

This annex provides information on the required XML message structure.

**Annex J QA Checklist by Requirement**

This annex lists the requirements in the "Data Exchange Standards for Registry Systems under the Kyoto Protocol: Functional Specifications, Draft Version <7.0>" and cross references the sections of the Technical Specifications which address them.

**Annex K Descriptive Language (WSDL) Documentation**

This annex provides the WSDLs for the Web services required for message exchange between a registry and the ITL and examples for each Transaction Type.

**Annex L WSDL Examples and Instructions**

This annex provides additional information and examples about how data should be provided for each transaction and notification type.

## Definitions, Acronyms, Abbreviations and Terminology

See the glossary in Annex A for definitions, acronyms and abbreviations relating to the Kyoto Protocol and related policy documents defining how the Protocol is to be implemented.

This list is intended to promote a common understanding of terminology which is critical to understanding and interpreting the Technical Specifications, and to ensure that developers and policy analysts use a common vocabulary for describing and discussing the specifications for data exchange.

It is important that readers familiarize themselves with these definitions, as the usage of many terms in this document is specific to these technical specifications and are not generic.

Note in particular that the term "registries" refers to both national registries and the CDM Registry. "Registry systems" refers to registries, the ITL, and STLs.

## Derivation Documents

- Data Exchange Standards for Registry Systems under the Kyoto Protocol: Functional Specifications (Version 1.0)  
→ <http://unfccc.int/sessions/workshop/281103/documents.html>
- Decisions 15-18/CP.7 on the mechanisms under the Kyoto Protocol  
→ Document FCCC/CP/2001/13/Add.2  
→ <http://unfccc.int/resource/docs/cop7/13a02.pdf>
- Decision 19/CP.7 containing general requirements for the ITL and registries and modalities for the accounting of assigned amounts under the Kyoto Protocol  
→ Document FCCC/CP/2001/13/Add.2  
→ <http://unfccc.int/resource/docs/cop7/13a02.pdf>
- Decision 24/CP.8 containing general design requirements for the data exchange standards  
→ Document FCCC/CP/2002/7/Add.3  
→ <http://unfccc.int/resource/docs/cop8/07a03.pdf>
- Decision 19/CP.9 on the modalities and procedures for afforestation and reforestation Project activities under the clean development mechanism in the first Commitment Period of the Kyoto Protocol  
→ Document FCCC/CP/2003/6/Add.2  
→ <http://unfccc.int/resource/docs/cop9/06a02.pdf>

## Multiple Language Support

With the exception of the country codes which utilize the alpha codes in ISO3166, all message content exchanged is represented as numeric values. The numeric codes are listed in Annex G. Therefore, the content of all messages is independent of a specific language.

## Validity of Data

The non-functional requirements for registries and the ITL require accuracy and data integrity. These requirements are addressed throughout these technical specifications, including in particular, the requirements for data elements and message content. The reconciliation process also provides assurance that these non-functional requirements will be met.

## 2 Assumptions and Constraints

These technical specifications are based upon the derivation documents specified in Section 1.5. In particular, they are based upon the constraints and requirements contained in the Functional Specifications for the Data Exchange Standard. A detailed cross reference of these technical specifications and the specific requirements is included in Annex J.

### Assumptions

The ITL will have access to the Compilation and Accounting Database (C & A Database) of the Secretariat.

The ITL will receive information regarding CDM Projects from the CDM Executive Board and information regarding track 2 joint implementation Projects from the Joint Implementation Supervisory Committee.

### Constraints

These data exchange standards utilize the following standards:

- SOAP  
<http://www.w3.org/TR/2000/NOTE-SOAP-20000508>
- XML  
<http://www.w3.org/TR/2000/REC-xml-20001006>
- WSDL  
<http://www.w3.org/TR/wsdl>

### 3 Data Exchange Mechanism Specifications

#### General Requirements

Communications between the registries and the ITL must be secure and processed as real-time transactions. The Functional Specifications for the Data Exchange Standards specify the use of TCP/IP connections using encrypted messages over the Internet.

Communications must be protected from modification or interception in transit. Users must be authenticated to ensure their identity and associated permissions. Communications will be initiated by either registries or the ITL and an immediate response will be expected.

To provide this functionality, the registries and ITL shall utilize a consistent and coordinated set of technical solutions. The technical specifications require:

- Web services using Simple Object Access Protocol (SOAP);
- Hardware-based Virtual Private Network (VPN);
- XML formats adhering to the described standards in Annexes I and K;
- Digital signature authentication; and
- Network time protocols.

Each of these technologies, with the exception of the VPN requirement, is platform and language independent. As discussed below, the hardware specifications for the VPN will take into account cost, interoperability and existing registry hardware, to the extent feasible.

Figure 3.1 provides a diagram of the basic architecture of the data exchange mechanism required for communications between registries and the ITL.

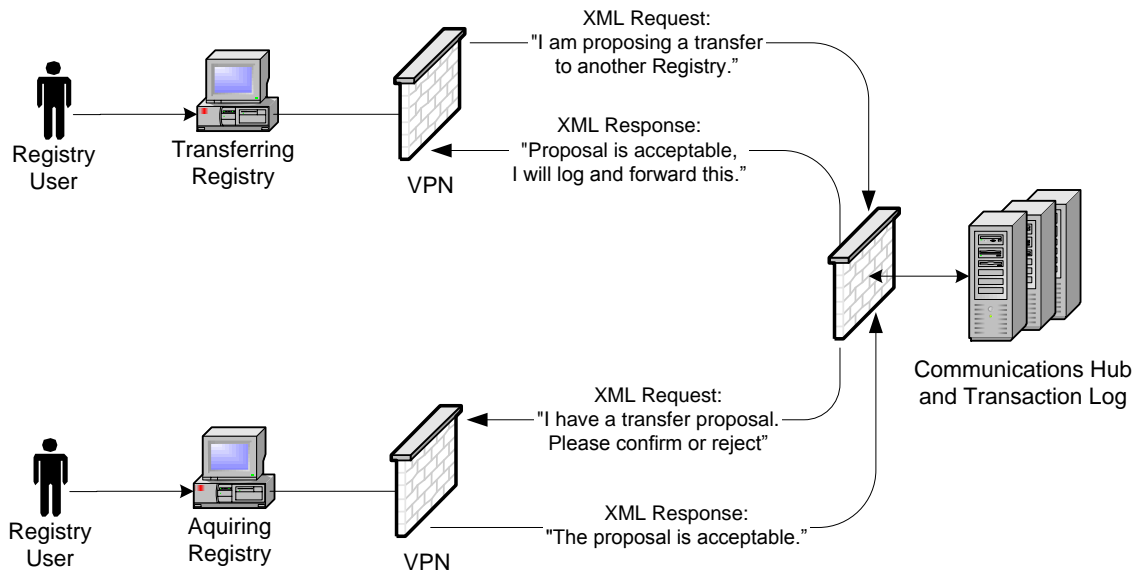


Figure 3.1: Data Exchange Architecture

## Communications Specifications

All registries and the ITL shall use Web services to support the sending and receiving of messages. Web services enable disparate applications running on different machines to easily exchange data with one another without requiring additional proprietary third-party software or hardware. Web services depend upon a standard XML messaging systems and SOAP and therefore are not tied to any one operating system or programming language. Based on common industry standards and existing technology such as XML and HTTP, Web services costs very little to deploy. Any information that is exchanged to and from both registries and the ITL shall be through the use of XML exchanged via SOAP. For technical specifications on the construct of these documents, see Annexes I and K.

SOAP is one of several an XML-based protocols for exchanging information between computers and is widely used in the internet community. Since SOAP runs primarily on top of HTTP and XML, all communications are encrypted using Secure Socket Layer (SSL).

Both the ITL and all registries shall be available for requests via the Internet. The technical specifications for the functionality of these Web services are defined in the Web services and functions specified in Annexes B, C, and D.

## Data Transfer Security

### 3.1.1 Firewall and Network Interconnection

The ITL shall be located on an Internet-connected network protected by a hardware-based firewall. The firewall shall be configured with rules such that only "registered" clients can make connections to the VPN server. This is achieved through Registry Systems having fixed public IP addresses and the ITL only accepting communications originating from these IP addresses. Registry Systems shall implement hardware-based VPN end-points for use in connecting to the system. These VPN end-points shall be configured with the appropriate credentials as provided by the ITL Administrator.

### 3.1.2 Virtual Private Network (VPN)

All communications to and from Registry Systems and the ITL shall be protected using hardware-based virtual private network (VPN) technology. The VPNs shall be configured to maintain the VPN connections permanently, in order to allow reliable, two-way, real-time communication between the ITL and Registry Systems at all times. VPN technologies provide the ability to "tunnel" through the Internet from one point to another, protecting all communications. Prior to the creation of the VPN tunnel, a digital certificate is issued to a prospective client end-point, allowing the client to provide proof of identity during the negotiation of the connection. The client installs the certificate into their VPN end-point. The client initiates the connection and is authenticated by the VPN server. Using digital certificates, the VPN server will access a central authority to negotiate authentication credentials. During the tunnel creation process, encryption is negotiated, ensuring that all communications through the tunnel are protected.

The ITL shall be located on an Internet-connected network protected by a hardware-based firewall. The firewall shall be configured with rules such that only "registered" clients can make connections to the VPN server. This is achieved through client registries having fixed public IP addresses and the ITL only accepting communications originating from these IP addresses. Client registries shall implement hardware-based VPN end-points for use in connecting to the system. These VPN end-points shall be configured with the appropriate credentials as provided by the ITL Administrators. The client VPN end-points shall be configured to maintain the VPN tunnel permanently, in order to allow reliable, two-way, real-time communication between the ITL and a client registry at all times.

#### 3.1.2.1 Client VPN Specifications

VPN equipment at the client registries shall be dedicated devices that can reliably terminate the VPN connection to the ITL as well as maintain acceptable performance levels. The

recommended VPN equipment adequate for a client registry VPN connectivity is a Cisco PIX firewall/VPN device. Information on the Cisco PIX firewall is available at <http://www.cisco.com/en/US/products/hw/vpndevc/ps2030/index.html>.

### **3.1.2.2 IPsec Implementation**

The use of the IPsec protocol to form the site-to-site VPN infrastructure will provide for site-to-site authentication, data integrity, and data encryption. IPsec VPN configurations provide for authentication between two end-points in a VPN connection. The ITL will identify and authenticate the remote client via the IPsec connection using a digital certificate provided by a Certificate Authority.

IPsec also ensures data integrity of all communications passed through the VPN tunnel. Packets of data are hashed and signed using the authentication information established by the VPN. Data confidentiality is also ensured by IPsec encrypting the data using Triple DES (3DES). This encryption applies to all network traffic between the ITL and Registry Systems.

### **3.1.3 SSL**

SSL shall be used for all communications between the Registry Systems and the Communications Hub. Since IPsec VPN provides only site-to-site authentication, a method is required to authenticate the actual registry communications to the ITL, in particular where multiple registries are hosted on a single site. SSL will provide application server-to-application server authentication as well as data encryption for all communications between registries. This will be achieved using client and server digital certificates that are authenticated during the establishment of the SSL connection. Additionally, the use of SSL will protect any communications that may pass over the networks at the registry site before transport through the VPN on to the ITL.

### **3.1.4 Certificate Authority**

IPsec and SSL require the use of a trusted Certificate Authority in order to realize the full benefit of positive authentication and secure encryption. Trusted Certificate Authority services are provided commercially by several vendors, such as Verisign and Thawte. These vendors verify identity and issue certificates which can be used to positively identify an organization and encrypt data communications between the organization and other certificate holders. These vendors are already widely used and trusted worldwide, with a large percentage of on-line transactions via SSL using their certificates.

Due to the number of registry end-points and size of the VPN, a third-party managed Certificate Authority will be used, as specified by the ITL.

## **The Communications Hub and Message Queue**

The security layer and supporting hardware and software between the VPN and ITL database is the Communications Hub. The Communications Hub receives and logs all messages passed through the VPN. The Communications Hub hosts a message queue which processes all incoming messages. The purpose of the queue is to receive and store messages and to provide scalability during peak transaction times.

## **Data Transfer Format Specifications**

All message packages must utilize XML and conform to the standards in Annex I. WSDL specifications for these XML messages are defined in Annex K.

## **User Accounts**

The ITL VPN shall register and maintain user IDs and passwords for users who are logging in directly to the ITL's Web application. A user account is valid for an indefinite period of



time. The ITL may revoke or replace a user's registration or password if there is a suspected breach of security or rules of behaviour by a user.

## Time Validation Specifications

To ensure that transaction rules are accurately and consistently applied to all proposed transactions, the ITL and registries shall use a consistent convention for recording time, and shall also utilize time synchronization practices and procedures to ensure accurate logging and sequencing of all transactions. Accurate and consistent time clocks are essential to the reconciliation process.

All dates and times shall be recorded as Greenwich Mean Time (GMT).

Time information shall be submitted as a date, hour, minute and second in the format: YYYY-MM-DD HH:MN:SS

All registries and the ITL shall use Network Time Protocol (NTP) version 3 or better software to synchronize their clocks with well known public Stratum 2 time servers. NTP ensures that both the ITL and registries maintain consistent and accurate times. NTP software has been ported to all major computing platforms, and in some cases is bundled with the operating system. Information on NTP is found at <http://www.ntp.org/>. A list of well known international Stratum 2 NTP time servers is maintained at <http://www.eecis.udel.edu/~mills/ntp/clock2a.html>. A list of NTP ports to various computing platforms and other resources is found at <http://www.ntp.org/links.html>. Any NTP version 3 or better client is acceptable. Registries should configure their clients to contact two or more public NTP servers that are close to them in terms of traffic routing on the Internet.

As part of the ITL administrative functions described in Section 6 and Annex D, the ITL shall perform a time check service for registries in which the ITL can request the time. The ITL is responsible for recording the time in which each message is routed through its Communications Hub and enters the message queue.

Figure 3.2 outlines the sequence in which the time stamp of a message is evaluated. Sixty seconds is the recommended allowable time between Request Time Stamp (1) and Response Time Stamp (2). All Web services requests must receive an appropriate response, although this does not mean that any one or all transactions requested in the message is complete. An Acquiring Registry shall complete and send to the ITL a confirmation of the transaction as soon as possible and no later than 24 hours from Time Stamp (2). If a confirmation is not received by the ITL within 24 hours, the transaction is no longer valid and will be cancelled by the ITL.

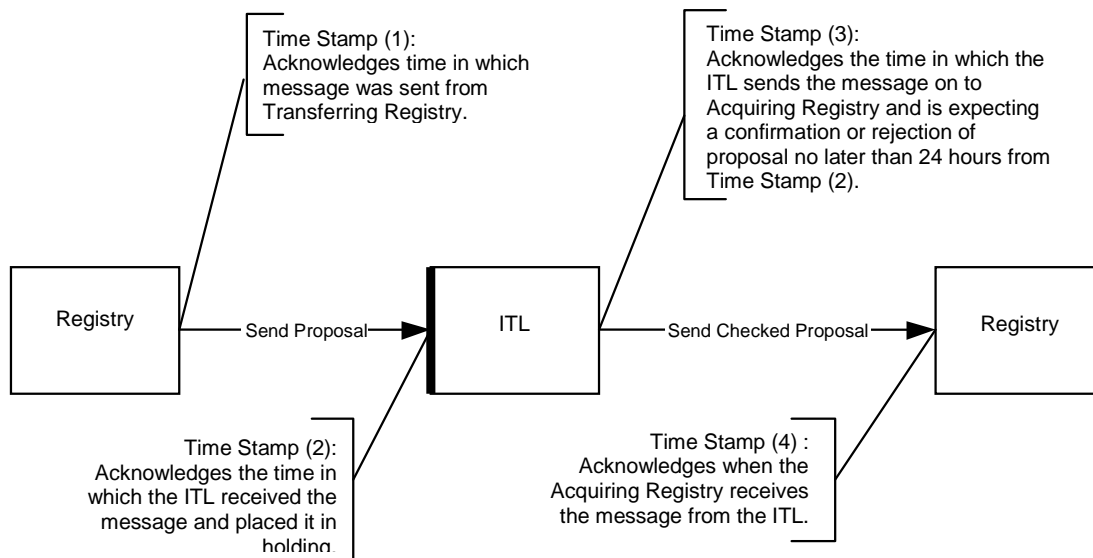


Figure 3.2: Process Time Stamps

## Web Service Timeout behaviour

Messages transferred between registries (and STLs) and the ITL use web services. Web services are invoked via http requests.

The sender/client of an http request waits for the http response from the receiver/server for a period of sixty seconds before it initiates timeout processing, as shown in Figure 3.3. This timeout behaviour applies to all web services, regardless of the type of message being transferred (transactions, reconciliations, etc)

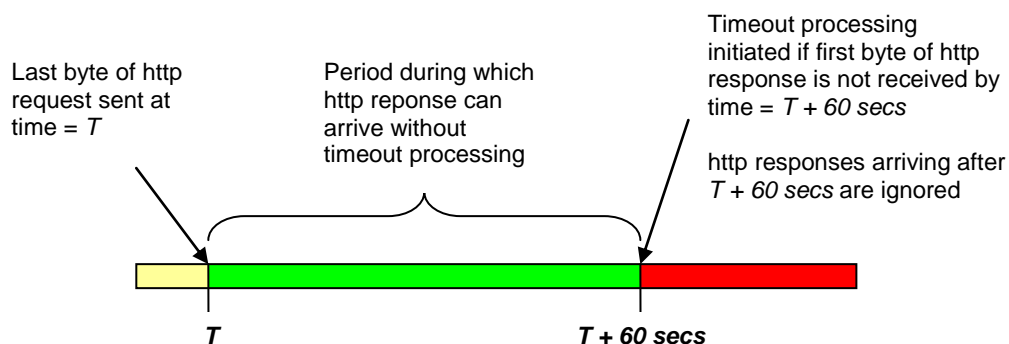


Figure 3.3: Timeout period

### 3.1.5 Timeout processing

When the ITL acts as the http client, its current timeout processing involves making multiple attempts to retransmit the http request at certain intervals. Details of the number of retransmission attempts and the intervals are published on the ITL RSA Extranet.

It is recommended that registries also include some form of retransmission in their timeout processing.

### 3.1.6 What to do before sending the http response

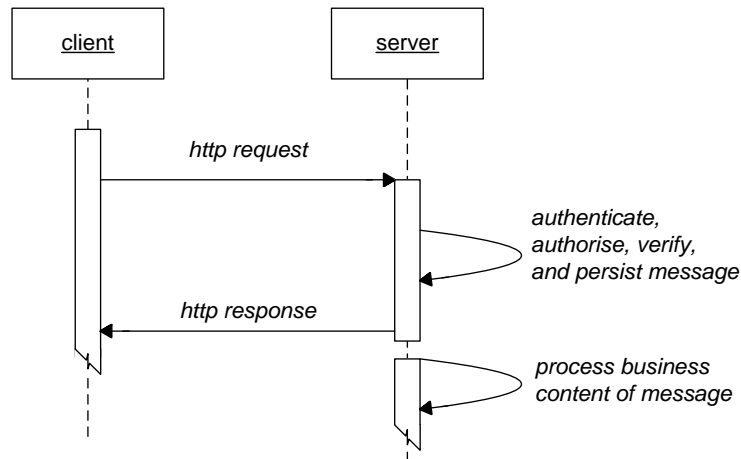
It is expected that upon receipt of the message, the server

- validate the digital signature, user account and password
- verify that the message was a well-formed<sup>1</sup> XML document, and
- ensure the message will be subsequently processed (e.g. persisting it)

before sending the http response to the client (as illustrated in Figure 3.4).

The server may perform more than the above, however, this increases the risk of exceeding the timeout, and having its response ignored by the client.

In most cases<sup>2</sup>, it is not expected that the business content of the message be processed prior to sending the synchronous http response back to the client/message sender.



**Figure 3.4: When to send the http response**

In normal circumstances, sixty seconds is sufficient time for these operations, including the sending of the http response and network latencies.

### 3.1.7 Extending the timeout period past 60 seconds

A timeout period of 60 seconds is the default timeout period the DES recommends clients (senders of the http request) use. As described above, in normal circumstances, 60 seconds is sufficient for the server (recipients of the http request) to complete the required activities and provide the http response. However, clients may optionally implement a timeout period of greater than 60 seconds.

<sup>1</sup> A *well-formed* document conforms to the XML syntax rules: It has exactly one root element, and it has matching start and end tags. *Well-formed* XML does not need to be *valid* xml, that is, it does not need to conform to an XML schema.

<sup>2</sup> Two transactions: *getTransactionStatus()* and *provideTime()*, by their nature, require additional synchronous processing in order to return a result in their http response. This is discussed in §6.4 and §6.5.

*[This page intentionally left blank.]*

## **4 Unit Transaction Processes**

### **Unit Transaction Types**

This section of the Technical Specifications addresses the messages and content requirements necessary to support submissions of unit transactions by registries and the validation of those transactions by the ITL. The unit transactions either involve the transfer of ownership of a unit, a change in an attribute of a unit, or the replacement of a tCER or ICER. This section will describe the data exchange flow, the responsibilities of registries, and the responsibilities of the ITL in order to complete a unit transaction.

The following unit transactions are described:

- Issuance;
- Conversion;
- External Transfers;
- Cancellation (Internal Transfer);
- Replacement (Internal Transfer);
- Retirement (Internal Transfer);
- Carry-over; and
- Expiry Date Change.

#### **4.1.1 Issuance**

The issuance of AAUs is undertaken by a Party in its national registry on the basis of its assigned amount under Articles 3.7 and 3.8 of the Kyoto Protocol (which is in turn calculated on the basis of greenhouse gas emissions during the base year). The issuance of RMUs is undertaken by a Party in its national registry on the basis of its net removals of greenhouse gases through LULUCF activities. The issuance of tCERs, ICERs, and CERs into a pending account is undertaken by the CDM Executive Board, in the CDM Registry, on the basis of verified and certified reductions in greenhouse gas emissions or removals of greenhouse gases from the atmosphere through a CDM Project activity. Issuance of such units is monitored and validated by the ITL.

The issuance process for AAUs, RMUs, tCERs, ICERs, and CERs follows the single registry model of data exchange described in Section 4.3.

#### **4.1.2 Conversion**

The conversion of AAUs and RMUs to ERUs is undertaken by a Party in an account in its national registry. AAUs are converted to ERUs in its national registry on the basis of verified reductions in emissions through a Joint Implementation (JI) Project. RMUs are converted to ERUs on the basis of verified removals of greenhouse gases through a JI Project. Conversion of such units is monitored by the ITL.

The conversion process of AAUs and RMUs to ERUs follows the single registry model of data exchange described in Section 4.3.

#### **4.1.3 External Transfer**

The external transfer of AAUs, RMUs, ERUs, tCERs, ICERs, and CERs to another registry is undertaken by a Party, an entity, or the CDM Executive Board, on the basis of the amount proposed by the transferor. The external transfer of such units is monitored and validated by the ITL.

The external transfer process for AAUs, RMUs, ERUs, tCERs, ICERs, and CERs follows the two registry model of data exchange described in Section 4.4.

#### 4.1.4 Cancellation

The internal transfer of AAUs, RMUs, ERUs, CERs, tCERs and ICERs to a cancellation account is undertaken by a Party, an entity or the CDM Executive Board, on the basis of the amounts proposed by the transferer. The cancellation of such units is monitored and validated by the ITL.

Any units for a Commitment Period, which remain in holding accounts after the appropriate carry-over transactions for that Commitment Period have been completed, must be cancelled to the Mandatory Cancellation Account (Account Type Code 250).

The cancellation of AAUs, RMUs, ERUs, tCERs, ICERs, and CERs typically follows the single registry model of data exchange described in Section 4.3. Cancellation from a national registry as a result of excess issuance for a CDM Project follows the two registry model of data exchange described in Section 4.4. These units must be transferred via an External Transfer (Transaction Type Code 4) to the Excess Issuance Cancellation Account in the CDM Registry (Account Type Code 240).

Table 4.1 below defines the transaction type, notification type, eligible acquiring account type, and eligible unit types associated with each specific type of cancellation.

**Table 4.1: Cancellation Transaction Information**

Transaction and Functional/Business Purpose	Transaction Type	Notification Type	Eligible Account Type (Acquiring)	Eligible Unit Types
Net Source Cancellation	Cancellation (4)	1	Net Source Cancellation Account (Account Type 210)	AAU, ERU, RMU, CER
Non-compliance Cancellation	Cancellation (4)	2	Non-compliance Cancellation Account (Account Type 220)	AAU, ERU, RMU, CER
Reversal of Storage for CDM Project	Cancellation (4)	4	Mandatory Cancellation Account (Account Type 250)	ICER (from the same Project)
Non-submission of Certification Report for CDM Project	Cancellation (4)	5	Mandatory Cancellation Account (Account Type 250)	ICER (from the same Project)
Excess Issuance for CDM Project (From a National Registry)	External Transfer (3)	6	Excess Issuance Cancellation Account in the CDM Registry (Account Type 240)	AAU, RMU, ERU, CER, tCER (for tCER excess issuance only), ICER (from the same Project only)
Excess Issuance for CDM Project (From the CDM Registry)	Cancellation (4)	6	Excess Issuance Cancellation Account in the CDM Registry (Account Type 240)	CER, tCER (for tCER excess issuance only), ICER (from the same Project only)
Administrative cancellation for CDM Project (From a National Registry)	External Transfer (3)	None	Administrative Cancellation Account in the CDM Registry (Account Type 260)	CER, tCER, ICER (from the same Project)
Administrative cancellation for CDM Project (From a CDM Registry)	Cancellation (4)	None	Administrative Cancellation Account in the CDM Registry (Account Type 260)	CER, tCER, ICER (from the same Project)
Cancellation of units for a Commitment Period which remain in holding accounts after completion of the true-up period for that Commitment Period	Cancellation (4)	None	Mandatory Cancellation Account (Account Type 250)	RMU, ERU (converted from RMU), tCER, ICER

Cancellation of units for a Commitment Period which remain in holding accounts after completion of all carry-over transactions for that Commitment Period	Cancellation (4)	None	Mandatory Cancellation Account (Account Type 250)	AAU, ERU (converted from AAU), CER
Voluntary Cancellation	Cancellation (4)	None	Voluntary Cancellation Account (Account Type 230)	AAU, ERU, RMU, CER, ICER, tCER

#### 4.1.5 Replacement

The replacement of tCERs and ICERs occurs through the internal transfer of AAUs, RMUs, ERUs, CERs, tCERs or ICERs to a replacement account and is undertaken by a Party or an entity, on the basis of the amounts proposed by the transferor. The replacement of such units is monitored and validated by the ITL.

For replacements required by a Reversal of Storage action or a Non-submission of Certification action by the CDM Executive Board, the registry must include the Notification ID associated with the replacement transaction. This Notification ID is used to determine if the replacement should be considered when the ITL performs a follow-up evaluation to assess whether the required replacement has been completed.

The replacement of tCERs and ICERs follows the single registry model of data exchange described in Section 4.3.

Table 4.2 below defines the notification type, eligible acquiring account type, and eligible unit types associated with each specific type of replacement.

**Table 4.2: Replacement Transaction Information**

Transaction and Functional/Business Purpose	Notification Type	Eligible Account Type (Acquiring)	Eligible Unit Types
tCER replacement upon expiry (held in retirement or tCER replacement account)	3	tCER Replacement Account for Expiry (411)	AAU, RMU, ERU, CER, tCER
ICER replacement upon expiry (held in retirement account)	3	ICER Replacement Account for Expiry (421)	AAU, RMU, ERU, CER
ICER replacement for storage reversal (held in holding or retirement accounts, not yet replaced)	4	ICER Replacement Account for Reversal of Storage (422)	AAU, RMU, ERU, CER, ICER (from the same Project)
ICER replacement for non-certification (held in holding or retirement account, not yet replaced)	5	ICER Replacement Account for Non-submission of Certification Report (423)	AAU, RMU, ERU, CER, ICER (from the same Project)

#### 4.1.6 Retirement

The internal transfer of AAUs, RMUs, ERUs, CERs, tCERs and ICERs to a retirement account is undertaken by a Party or an entity, on the basis of the amounts proposed by the transferor. The retirement of such units is monitored and validated by the ITL.

The retirement of AAUs, RMUs, ERUs, tCERs, ICERs, and CERs follows the single registry model of data exchange described in Section 4.3.

#### 4.1.7 Carry-over Process

The carry-over of AAUs, ERUs converted from AAUs and CERs is undertaken by a Party in an account in its national registry, on the basis of the amount of units in holding accounts after expiration of the additional period for fulfilling commitments (the "true-up period"). The units remain in the same account and the serial numbers remain unchanged. The effect of

the carry-over transaction is to give recognition, both within the registry and the ITL, to the validity of the units in the next Commitment Period. Any units in holding accounts that are not carried over in this manner must be cancelled. The carry-over of units is monitored and validated by the ITL.

At the conclusion of the true-up period for a Commitment Period, the ITL will send notifications to a registry indicating the number of units which it may carry-over to the new Commitment Period. All carry-over transactions must contain the Notification ID sent by the ITL.

The carry-over of AAUs, ERUs and CERs follows the single registry model of data exchange described in Section 4.3.

#### **4.1.8 Expiry Date Change**

The change in the expiry date is undertaken by a Party for tCERs and ICERs pursuant to notification from the ITL. For tCERs, this transaction may be necessary where they are initially issued with an expiry date which is different from that eventually agreed as the end of the subsequent Commitment Period. For ICERs, this transaction will occur when the Executive Board approves the renewal of the crediting period for a Project. The ITL ensures that these expiry date changes are consistent with the appropriate dates and updates the tCER and ICER expiry dates in the ITL database. For tCER expiry date changes, all tCERs are subject to the change. For ICER expiry date changes, the units involved must be from the Project specified in the original notification. All expiry date change transactions must contain the Notification ID sent by the ITL.

The expiry date change transaction follows the single registry model of data exchange described in Section 4.3.

#### **4.1.9 Internal Transfers and Other Transactions Routed to an STL**

For transactions routed to an STL, the ITL conducts general transaction checks necessary to check that the unit blocks are available to a pending transaction and splits the blocks as necessary. The ITL records the results of this basic step and routes them to the relevant STL for further evaluation against STL rules and requirements.

Supplemental transactions follow either the single registry model or the multiple registry model. If an STL wishes to institute a supplemental transaction to be routed through the ITL, the STL must coordinate the development of this transaction with the ITL Administrator.

### **Description of Data Exchange Flow**

The data exchange flow for each unit Transaction Type follows one of two models: the single registry model or the multiple registry model. Most of the Transaction Types follow the single registry model. External transactions, and some supplemental transactions not described in this document, follow the multiple registry model. This single registry model is described first, followed by the data exchange model for multiple registries engaged in an external transaction. The sections for both models contain the following subsections:

- UML Sequence Diagrams; and
- State Transition Diagrams.

These technical specifications describe in general terms the programming logic that should be implemented at the registries and the ITL to establish reliable communications. A list of functions needed to implement this specification is included for each model. Technical information for transaction functions, including required inputs, outputs, and responses, is included in Annex B.

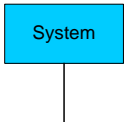
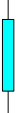
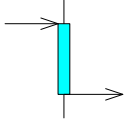
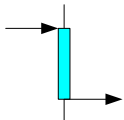
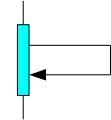
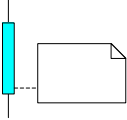

The results of a transaction evaluation conducted by the ITL, an STL, or an Acquiring Registry are returned in the XML document in the form of Response codes. Response codes and corresponding checks are grouped by the category of check, and can be found in Annex E.




#### 4.1.10 UML Notation for Transactional Message Flow

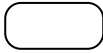

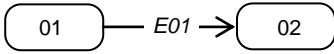
UML Sequence Diagrams are used to represent the message flow and processing during a transaction. It captures the messages sent between systems, the sequence in which they are sent, and the processing that occurs on systems between the messages. The Sequence Diagram notation is based on the standards for the Unified Modelling Language (UML), with text annotations to help non-technical readers interpret it more easily. These diagrams include the following symbols and conventions.

**Table 4.3: Key to UML Sequence Diagrams in Transactional Message Flow Descriptions**

UML Element	Description
<p><b>Systems and lifelines</b></p> 	<p>At the top of each diagram participating systems in the transaction are represented by boxes, with their name in the box. The line extending down from the box is the system's <i>lifeline</i>:</p> <ul style="list-style-type: none"> <li>arrows ending at this line represent messages to this system,</li> <li>arrows originating from this line represent messages and processing originating from this system.</li> </ul>
<p><b>Control</b></p> 	<p>A vertical rectangular box on a life line represents that the system has <i>control</i> of, or is working on, the transaction at that point in the sequence</p>
<p><b>Asynchronous messages</b></p> 	<p>A hollow arrow represents an asynchronous message from one system to another. This means that, at a business level, the sending system is not blocked, waiting for the receiving system to process the message and respond.</p> <p>In the diagrams below asynchronous messages represent web service calls. Where the <i>acceptNotification()</i> web service is called, the status of the transaction for which the call is made is shown under the name of the web service</p>
<p><b>Synchronous messages</b></p> 	<p>A solid arrow represents a synchronous message from one system to another. This means that, at a business level, the sending system is blocked, and awaits the receiving system to respond before it can continue with other processing.</p>
<p><b>Internal processing</b></p> 	<p>In sequence diagrams, when a system's internal processing depicted, it is represented by a synchronous message to from the system to itself.</p>
<p><b>Transaction States</b></p> 	<p>The status of the transaction at a particular point in the message flow is depicted by a document icon (rectangle with folded top-right corner). A line from the document icon to a life-line indicates that the system to which that status pertains.</p>
<p><b>ITL Transaction States</b></p> 	<p>The status of the transaction in the ITL are depicted in yellow. These are the states returned by <code>getTransactionStatus()</code></p> <p>The status of the transaction in other systems are depicted with other colours</p>

<p><b>Final Transaction States</b></p> 	<p>Final transaction states are depicted with thick borders.</p>
--	--

**Table 4.4: Key to State Transition Diagrams in Transaction Message Flow Descriptions**

Diagram Element	Notation
ITL Transaction States are represented by rounded rectangles	
Events causing state transitions are represented by arrows, labelled with references prefixed by "E"	
Example: In this example, Event <i>E01</i> causes a transition in State 01 to transition to State 02	

## Single Registry Model

The single registry model for transactions applies to the following Transaction Types:

- Issuance;
- Conversion;
- Cancellation (Internal Transfer);
- Replacement (Internal Transfer);
- Retirement (Internal Transfer);
- Carry-over; and
- Expiry Date Change.

The following steps apply to all the above transactions and describe the sequence of messages necessary to complete the transaction. They are also depicted diagrammatically in the next section.

### Step 1 – Proposal

The registry sends a proposal for a transaction to the ITL using the ITL's AcceptProposal web service. The proposal contains the Transaction Type, the units involved in the transaction, the appropriate transferring and acquiring account information, and the appropriate Notification ID information.

### Step 2 – ITL Review

The ITL receives the proposal and, once the incoming message is verified to be well formed and authentic, it places the message in a queue for processing. Messages are processed from the queue in the order received. The ITL validates the transaction against the business rules for the appropriate Transaction Type.

If the transaction passes ITL's validation, and if an STL is involved in the transaction, the ITL proposes the transaction to the STL using the AcceptProposal web service that STLs are required to implement. The STL validates the transaction, and notifies the ITL of the validation result by calling the ITL's AcceptNotification Web service.

### Step 3 – ITL Complete

If the transaction meets all requirements, the ITL finalises the transaction: the ITL updates its records for the units in the transaction as appropriate for the Transaction Type.

If one or more discrepancies are found, the ITL marks the transaction as terminated.

From this point, to the ITL, the transaction has completed, and registries are free to use the units in another transaction. The completed transaction has now been logged by the ITL.

If a STL is involved in the transaction, the ITL notifies the STL, via the AcceptNotification web service that STLs are required to implement, to either finalise or terminate the transaction. The ITL will await notification from the STL that it has either finalised or terminated the transaction before notifying the registry.

The ITL notifies the registry, via the AcceptNotification web service that registries are required to implement, to either finalise or terminate the transaction. If the ITL identified one or more discrepancies, response codes corresponding to the discrepancies will be included in the message.

**Note:** In order to keep compatibility with previous versions of registry software, the transactions status contained in the AcceptNotification call to the transferring registry after successful completion (Figure 4.1: message 5; Figure 4.2: message 11) is **02. Checked (No Discrepancy)**, or **09. STL Checked (No Discrepancy)** instead of **04. Completed**.

#### Step 4 – Registry Complete

Once the registry processes the ITL notification it must complete the transaction, either by finalising it (if no discrepancy was found) or by terminating it. The registry calls the ITL's AcceptNotification web service to notify the ITL that it has completed the transaction.

#### Step 5 – ITL Close Transaction

Once the ITL receives notification from the registry that it has completed (either finalised or terminated) the transaction, the ITL records this for administrative and problem management purposes. Once the ITL receives notifications from all parties involved in the transaction that they have completed, the ITL closes the transaction. Note that the transaction is considered complete, after Step 3.

### 4.1.11 Single Registry Transaction Sequence Diagrams

The UML sequence diagrams in this section describe the message flow for single registry transactions in all their variations:

Diagram describing message flow variation	Successfully finalised	Involves STL	Rejected by ITL	Rejected by STL
Figure 4.1: Single registry transaction: no STL, successful transaction finalisation Note that the transaction is completed on the ITL before the registry is notified and the transaction is completed on the registry.	y	n	n	n
Figure 4.2: Single registry transaction: STL, successful transaction finalisation Note that the transaction is completed on the ITL before the STL is notified to complete the transaction. The registry is the last system to be notified, thus the transaction is completed last on the registry.	y	y	n	n
Figure 4.3: Single registry transaction: ITL rejected Note that this is the same message flow, regardless of whether the registry is part of an STL or not, as the transaction is rejected by the ITL before it is passed to the STL.	n	n/a	y	n/a
Figure 4.4: Single registry transaction: STL rejected Note that the ITL is always the first to complete or terminate a transaction, the transferring registry, always the last.	n	y	n	y

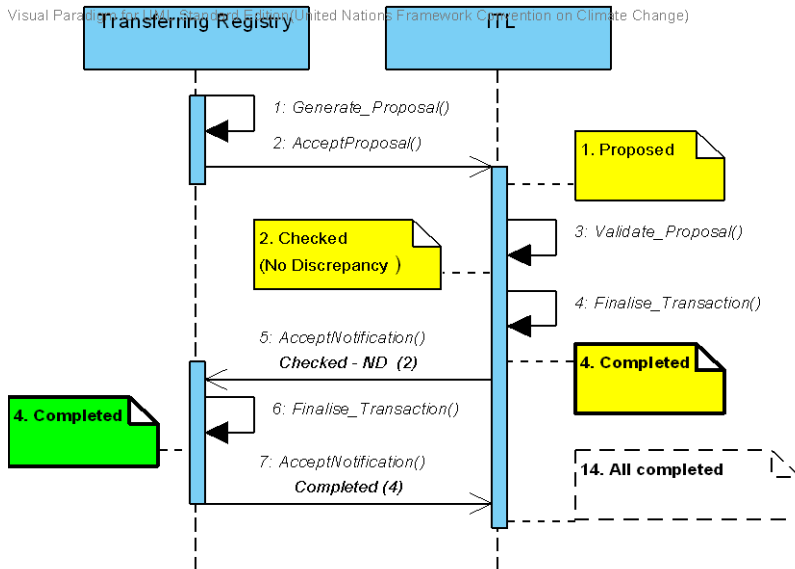


Figure 4.1: Single registry transaction: no STL, successful transaction finalisation

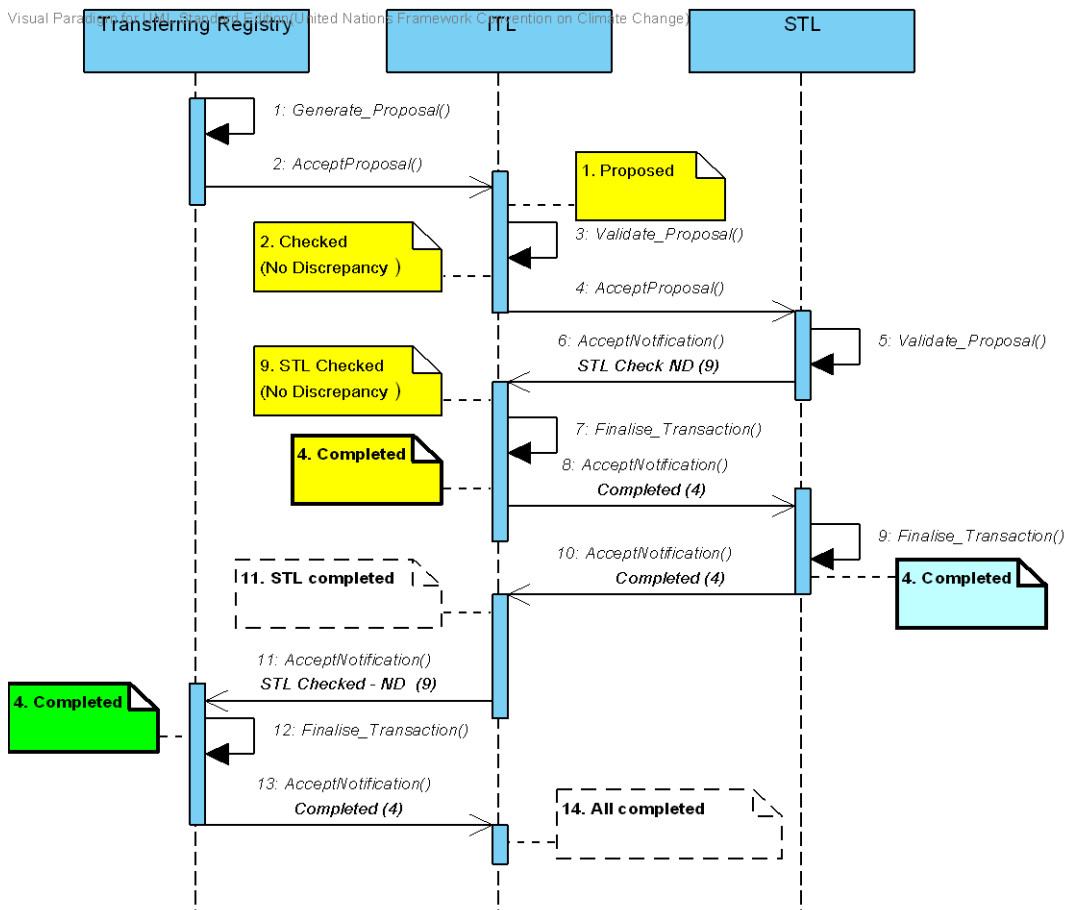
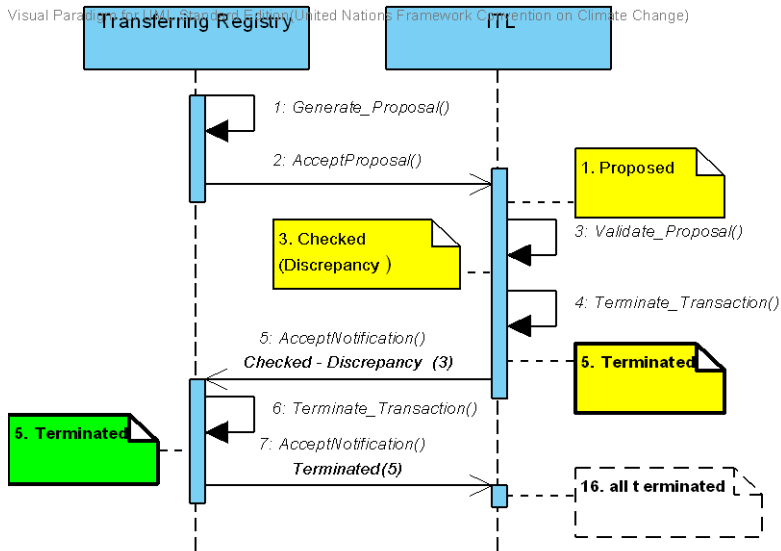
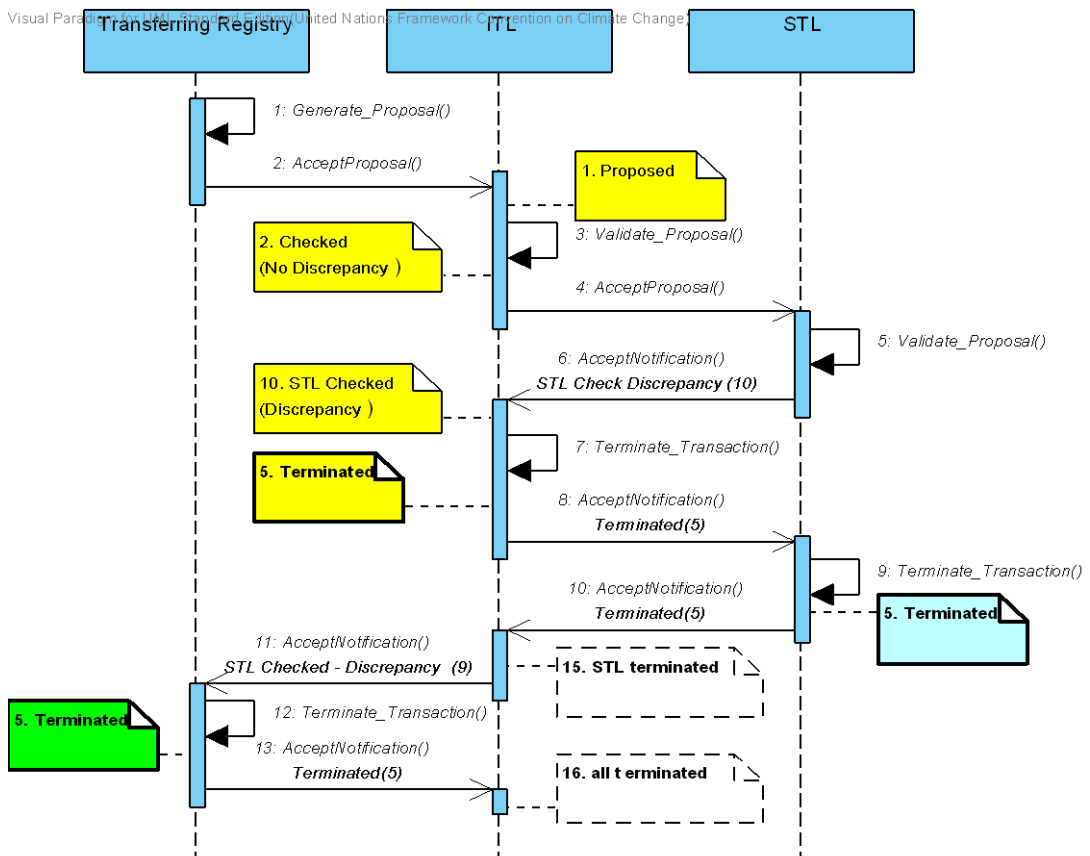


Figure 4.2: Single registry transaction: STL, successful transaction finalisation



**Figure 4.3: Single registry transaction: ITL rejected**

- same flow for STL and non-STL transactions
- same flow for single and two registry transactions



**Figure 4.4: Single registry transaction: STL rejected**

- same flow for single and two registry transactions

#### 4.1.12 Single Registry Transaction State Transitions

This section describes the states of single registry transactions, and the events that cause the transactions to transition from one state to the next state.

For clarity, two diagrams are used:

- For single registry transactions not involving STLs, and
- For single registry transactions involving STLs

##### 4.1.12.1 Single Registry Transaction State Transitions, not involving STLs

This state transition diagram, Figure 4.5, describes transactions that do not involve STLs. The descriptions of the state and events are contained in Table 4.5 and Table 4.6 below.

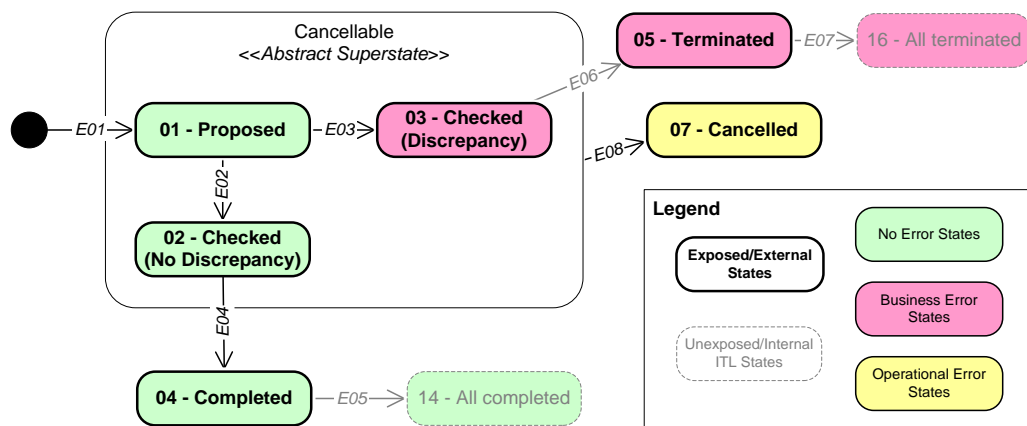


Figure 4.5: State Transition Diagram: Single Registry Transactions, not involving STLs

Table 4.5: State Descriptions - Single Registry Non-STL Transactions

State	Description	Entered via	Exits via
<b>01 - Proposed</b>	The transaction has been proposed by the Transferring registry. On entering this state, the ITL validates the transaction. The validation results in either <i>E02</i> or <i>E03</i> : <ul style="list-style-type: none"> <li>• valid (<i>E02</i>), the transaction transitions to <b>02 - Checked (No Discrepancy)</b>, or</li> <li>• invalid (<i>E03</i>), the transaction transitions to <b>03 - Checked (Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E01</i>: Idle</li> </ul>	<ul style="list-style-type: none"> <li>• <i>E02</i>: <b>02 - Checked (No Discrepancy)</b></li> <li>• <i>E03</i>: <b>03 - Checked (Discrepancy)</b></li> </ul>
<b>02 - Checked (No Discrepancy)</b>	The transaction has been checked by the ITL, and found to be valid. On entering this state, the ITL finalises the transaction. The result of completing finalisation processing is <i>E04</i> , that transitions the transaction to <b>04 - Completed</b> .	<ul style="list-style-type: none"> <li>• <i>E02</i>: <b>01 - Proposed</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E04</i>: <b>04 - Completed</b></li> </ul>
<b>03 - Checked (Discrepancy)</b>	The transaction has been checked by the ITL, and found to be invalid. On entering this state, the ITL terminates the transaction. The result of completing termination processing is <i>E06</i> , that transitions the transaction to <b>05 - Terminated</b> .	<ul style="list-style-type: none"> <li>• <i>E03</i>: <b>01 - Proposed</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E06</i>: <b>05 - Terminated</b></li> </ul>

State	Description	Entered via	Exits via
<b>04 - Completed</b>	The transaction has been completed. Units in the transaction are available for other transactions. On entering this state, the ITL notifies the registry to finalise the transaction on the registry. When the ITL receives notification from the registry that it has finalised the transaction ( <i>E05</i> ), the transaction transitions to <b>14 - All completed</b> .	<ul style="list-style-type: none"> <li><i>E04</i>: <b>02 - Checked (No Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li><i>E05</i>: <b>14 - All completed</b></li> </ul>
<b>05 - Terminated</b>	The transaction has been terminated due to discrepancies. Units in the transaction are available for other transactions. On entering this state, the ITL notifies the registry to terminate the transaction on the registry. When the ITL receives notification from the registry that it has terminated the transaction ( <i>E07</i> ), the transaction transitions to <b>16 - All terminated</b> .	<ul style="list-style-type: none"> <li><i>E06</i>: <b>03 - Checked (Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li><i>E07</i>: <b>16 - All terminated</b></li> </ul>
<b>07 - Cancelled</b>	The transaction has been cancelled by the ITL's 24 hour transaction cleanup ( <i>E08</i> )	<ul style="list-style-type: none"> <li><i>E08</i>: <b>01 - Proposed 02 - Checked (No Discrepancy) 03 - Checked (Discrepancy)</b></li> </ul>	N/A
<b>14 - All completed</b>	The transaction has been finalised by the ITL and the registry The registry has notified the ITL that it has finalised the transaction ( <i>E05</i> )	<ul style="list-style-type: none"> <li><i>E05</i>: <b>04 - Completed</b></li> </ul>	N/A
<b>16 - All terminated</b>	The transaction has been terminated by the ITL and the registry The registry has notified the ITL that it has terminated the transaction ( <i>E07</i> )	<ul style="list-style-type: none"> <li><i>E07</i>: <b>05 - Terminated</b></li> </ul>	N/A
<b>Canceltable</b>	This is a superset of states that include <ul style="list-style-type: none"> <li><b>01 - Proposed</b></li> <li><b>02 - Checked (No Discrepancy)</b>, and</li> <li><b>03 - Checked (Discrepancy)</b></li> </ul> When the transaction is in one of these states, and has not transitioned for 24 hours or more, it is cancelled by the ITL's 24 hour transaction clean-up ( <i>E08</i> )	<ul style="list-style-type: none"> <li><i>E01</i>: <b>Idle</b></li> <li><i>E02, E03</i>: <b>01 - Proposed</b></li> </ul>	<ul style="list-style-type: none"> <li><i>E04</i>: <b>04 - Completed</b></li> <li><i>E06</i>: <b>05 - Terminated</b></li> <li><i>E08</i>: <b>07 - Cancelled</b></li> </ul>

**Table 4.6: ITL State Transition Event Descriptions - Single Registry Non-STL Transactions**

Id	Type	Description	Source
<i>E01</i>	message	ITL receives a transaction <b>proposal</b> via <i>acceptProposal()</i> from the transferring registry	Transferring Registry
<i>E02</i>	validation	ITL <b>validates</b> the proposed transaction	N/A
<i>E03</i>	validation	ITL finds <b>discrepancies</b> in the proposed transaction	N/A
<i>E04</i>	process complete	ITL <b>finalises</b> the transaction	N/A
<i>E05</i>	message	ITL receives <i>acceptNotification()</i> from the transferring registry notifying the <b>transferring registry has completed</b> the transaction.	Transferring Registry
<i>E06</i>	process complete	ITL <b>terminates</b> the transaction	N/A
<i>E07</i>	message	ITL receives <i>acceptNotification()</i> from the transferring registry notifying the <b>transferring registry has terminated</b> the transaction.	Transferring Registry
<i>E08</i>	time	The proposed transaction has not transitioned for 24 hours, and is <b>cancelled</b> by the ITL's 24 hour transaction clean-up.	N/A



#### 4.1.12.2 Single Registry Transaction State Transitions, involving STLs

This state transition diagram, Figure 4.6, describes transactions that do not involve STLs. The descriptions of the state and events are contained in Table 4.7 and Table 4.8 below.

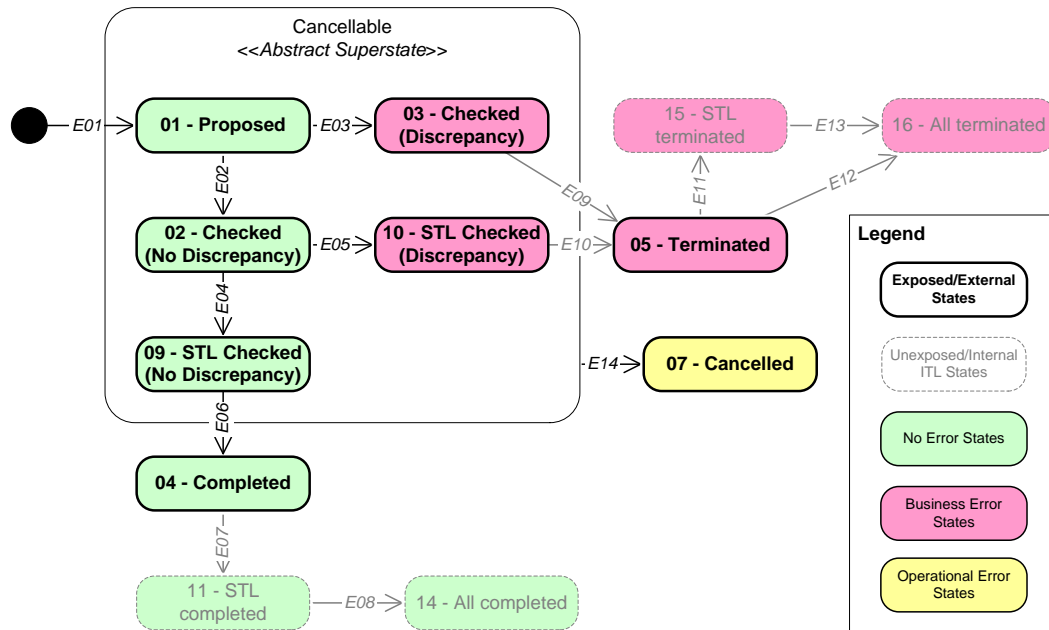


Figure 4.6: State Transition Diagram: Single Registry Transactions, involving STLs

Table 4.7: State Descriptions - Single Registry STL Transactions

State	Description	Entered via	Exits via
<b>01 - Proposed</b>	The transaction has been proposed by the Transferring registry. On entering this state, the ITL validates the transaction. The validation results in either <i>E02</i> or <i>E03</i> : <ul style="list-style-type: none"> <li>valid (<i>E02</i>), the transaction transitions to <b>02 - Checked (No Discrepancy)</b>, or</li> <li>invalid (<i>E03</i>), the transaction transitions to <b>03 - Checked (Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li><i>E01</i>: Idle</li> </ul>	<ul style="list-style-type: none"> <li><i>E02</i>: <b>02 - Checked (No Discrepancy)</b></li> <li><i>E03</i>: <b>03 - Checked (Discrepancy)</b></li> </ul>
<b>02 - Checked (No Discrepancy)</b>	The transaction has been checked by the ITL, and found to be valid. On entering this state, the ITL proposes the transaction to the STL for validation. When the ITL receives the validation result from the STL, one of two events occur: <ul style="list-style-type: none"> <li><i>E04</i>, the STL validated the transaction, in which case the transactions transitions to <b>09 - STL Checked (No Discrepancy)</b>, or</li> <li><i>E05</i>, the STL rejected the transaction, in which case, the transaction transitions to <b>10 - STL Checked (Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li><i>E02</i>: <b>01 - Proposed</b></li> </ul>	<ul style="list-style-type: none"> <li><i>E04</i>: <b>09 - STL Checked (No Discrepancy)</b></li> <li><i>E05</i>: <b>10 - STL Checked (Discrepancy)</b></li> </ul>
<b>03 - Checked (Discrepancy)</b>	The transaction has been checked by the ITL, and found to be invalid. On entering this state, the ITL terminates the transaction. The result of completing termination processing is <i>E09</i> , that transitions the transaction to <b>05 - Terminated</b> .	<ul style="list-style-type: none"> <li><i>E03</i>: <b>01 - Proposed</b></li> </ul>	<ul style="list-style-type: none"> <li><i>E09</i>: <b>05 - Terminated</b></li> </ul>

State	Description	Entered via	Exits via
<b>04 - Completed</b>	<p>The transaction has been completed. Units in the transaction are available for other transactions.</p> <p>On entering this state, the ITL notifies the STL to finalise the transaction on its systems.</p> <p>When the ITL receives notification from the STL that it has finalised the transaction (<i>E07</i>), the transaction transitions to <b>11 - STL completed</b>.</p>	<ul style="list-style-type: none"> <li>• <i>E06</i>: <b>09 – STL Checked (No Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E07</i>: <b>11 - STL completed</b></li> </ul>
<b>05 - Terminated</b>	<p>The transaction has been terminated due to discrepancies. Units in the transaction are available for other transactions.</p> <p>On entering this state,</p> <ul style="list-style-type: none"> <li>• if it was entered via <i>E10</i> (from <b>10 - STL Checked (Discrepancy)</b>), the ITL notifies the STL to terminate the transaction,</li> <li>• if it was entered via <i>E09</i> (from <b>03 - Checked Discrepancy</b>), the ITL notifies the registry to terminate the transaction.</li> </ul> <p>When the ITL receives notification from the STL that it has terminated the transaction (<i>E11</i>), the transaction transitions to <b>15 - STL terminated</b>.</p> <p>When the ITL received notification from the registry that it has terminated the transaction (<i>E12</i>), the transaction transitions to <b>16 - All terminated</b>.</p>	<ul style="list-style-type: none"> <li>• <i>E06</i>: <b>03 - Checked (Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E11</i>: <b>15 - STL terminated</b></li> <li>• <i>E12</i>: <b>16 - All terminated</b></li> </ul>
<b>07 - Cancelled</b>	<p>The transaction has been cancelled by the ITL's 24 hour transaction cleanup (<i>E14</i>)</p>	<ul style="list-style-type: none"> <li>• <i>E14</i>: <b>01 - Proposed</b> <b>02 - Checked (No Discrepancy)</b> <b>03 - Checked (Discrepancy)</b> <b>09 - STL Checked (No Discrepancy)</b> <b>10 - STL Checked (Discrepancy)</b></li> </ul>	N/A
<b>09 - STL Checked (No Discrepancy)</b>	<p>The ITL has received notification from the STL that the transaction is valid.</p> <p>On entering this state, the ITL finalises the transaction.</p> <p>The result of completing finalisation processing is <i>E06</i>, that transitions the transaction to <b>04 - Completed</b>.</p>	<ul style="list-style-type: none"> <li>• <i>E04</i>: <b>02 - Checked (No Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E06</i>: <b>04 - Completed</b></li> </ul>
<b>10 - STL Checked (Discrepancy)</b>	<p>The ITL has received notification from the STL that the transaction is invalid.</p> <p>On entering this state, the ITL terminates the transaction.</p> <p>The result of completing termination processing is <i>E10</i>, that transitions the transaction to <b>05 - Terminated</b>.</p>	<ul style="list-style-type: none"> <li>• <i>E05</i>: <b>02 - Checked (No Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E10</i>: <b>05 - Terminated</b></li> </ul>
<b>11 - STL completed</b>	<p>The ITL has received notification from the STL that it has finalised the transaction.</p> <p>On entering this state, the ITL notifies the registry to finalise the transaction.</p> <p>When the ITL receives notification from the registry that it has finalised the transaction (<i>E08</i>), the transaction transitions to <b>14 - All completed</b>.</p>	<ul style="list-style-type: none"> <li>• <i>E07</i>: <b>04 - Completed</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E08</i>: <b>14 - All completed</b></li> </ul>
<b>14 - All completed</b>	<p>The ITL has received notification from the registry that it has finalised the transaction.</p> <p>The transaction has been finalised by the ITL, the STL, and the registry.</p>	<ul style="list-style-type: none"> <li>• <i>E08</i>: <b>11 - STL Completed</b></li> </ul>	N/A

State	Description	Entered via	Exits via
<b>15 - STL terminated</b>	The ITL has received notification from the STL that it has terminated the transaction. On entering this state, the ITL notifies the registry to terminate the transaction. When the ITL receives notification from the registry that it has terminated the transaction ( <i>E13</i> ), the transaction transitions to <b>16 - All terminated</b> .	<ul style="list-style-type: none"> <li>• <i>E11</i>: <b>05 - Terminated</b></li> </ul>	<i>E13</i> : <b>16 - All terminated</b>
<b>16 - All terminated</b>	The transaction has been terminated by the ITL, the STL, and the registry. The registry has notified the ITL that it has terminated the transaction ( <i>E13</i> )	<ul style="list-style-type: none"> <li>• <i>E12, E13</i>: <b>05 - Terminated</b></li> </ul>	N/A
<b>Cancelable</b>	This is a superset of states that include <ul style="list-style-type: none"> <li>• <b>01 - Proposed</b></li> <li>• <b>02 - Checked (No Discrepancy)</b>,</li> <li>• <b>03 - Checked (Discrepancy)</b>,</li> <li>• <b>09 - STL Checked (No Discrepancy)</b>, and</li> <li>• <b>10 - STL Checked (Discrepancy)</b></li> </ul> When the transaction is in one of these states, and has not transitioned for 24 hours or more, it is cancelled by the ITL's 24 hour transaction clean-up ( <i>E14</i> )	<ul style="list-style-type: none"> <li>• <i>E01</i>: <b>Idle</b></li> <li>• <i>E02, E03</i>: <b>01 - Proposed</b></li> <li>• <i>E04</i>: <b>02 - Checked (No Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E06</i>: <b>04 - Completed</b></li> <li>• <i>E09, E10</i>: <b>05 - Terminated</b></li> <li>• <i>E14</i>: <b>07 - Cancelled</b></li> </ul>

**Table 4.8: ITL State Transition Event Descriptions - Single Registry STL Transactions**

Id	Type	Description	Source
E01	message	ITL receives a transaction <b>proposal</b> via <i>acceptProposal()</i> from the transferring registry	Transferring Registry
E02	validation	ITL <b>validates</b> the proposed transaction	N/A
E03	validation	ITL finds <b>discrepancies</b> in the proposed transaction	N/A
E04	message	ITL receives <i>acceptNotification()</i> from the STL confirming the proposed transaction has been <b>validated</b> by the CITL	STL
E05	message	ITL receives <i>acceptNotification()</i> from the STL notifying the STL has found <b>discrepancies</b> in the proposed transaction	STL
E06	process complete	ITL <b>finalises</b> the transaction	N/A
E07	message	ITL receives <i>acceptNotification()</i> from the STL notifying the <b>STL has completed</b> the transaction in the CITL and the ITL can proceed to notify the registries to complete.	STL
E08	message	ITL receives <i>acceptNotification()</i> from the transferring registry notifying the <b>transferring registry has completed</b> the transaction.	Transferring Registry
E09 E10	process complete	ITL <b>terminates</b> the transaction	N/A
E11	message	ITL receives <i>acceptNotification()</i> from STL notifying the <b>STL has terminated</b> the transaction.	STL
E12 E13	message	ITL receives <i>acceptNotification()</i> from the transferring registry notifying the <b>transferring registry has terminated</b> the transaction.	Transferring Registry
E14	time	The proposed transaction has not transitioned for 24 hours, and is <b>cancelled</b> by the ITL's 24 hour transaction clean-up.	N/A

## Two Registry Transaction Model

The two registry model for transactions applies to External Transactions in which units are transferred to a different registry. The initial steps are similar to the single registry model, but require the additional step of forwarding the proposal to the Acquiring Registry. The following steps describe the sequence of messages that complete an external transfer:

### Step 1 – Proposal

The registry sends a proposal for a transaction to the ITL using the ITL's AcceptProposal web service. The proposal contains the Transaction Type, the units involved in the transaction, the transferring and acquiring account types, and the transferring and acquiring account identifiers.

### Step 2 – ITL Review

The ITL receives the proposal and, once the incoming message is verified to be well formed and authentic, it places the message in a queue for processing. Messages are processed from the queue in the order received. The ITL validates the transaction against the business rules for external transactions. If the transaction meets all requirements, the ITL records the transaction as pending and marks the units involved in the transaction as unavailable to any other transaction.

If the transaction passes ITL's validation, and an STL is involved in the transaction, the ITL proposes the transaction to the STL using the AcceptProposal web service that STLs are required to implement. The STL validates the transaction, and notifies the ITL of the validation result by calling the ITL's AcceptNotification web service.

If the transaction passes validation to this point (the ITL's, and if appropriate, the STL's), the ITL proposes the transaction to the Acquiring Registry using the AcceptProposal web service that registries are required to implement.

### Step 3 – Registry Review

The Acquiring Registry evaluates the proposal and either accepts or rejects it. In either case, the Acquiring Registry calls the ITL's AcceptNotification web service to inform the ITL of its evaluation result.

### Step 4 – ITL Complete

If the transaction meets all requirements (the ITL's, the STL's, and the acquiring registry's), the ITL finalises the transaction: the ITL updates its records for the units in the transaction as appropriate for the Transaction Type.

If one or more discrepancies are found, the ITL marks the transaction as terminated.

From this point, to the ITL, the transaction has completed, and registries are free to use the units in another transaction. The completed transaction has now been logged by the ITL.

If a STL is involved in the transaction, the ITL notifies the STL, via the AcceptNotification web service that STLs are required to implement, to either finalise or terminate the transaction. The ITL will await notification from the STL that it has either finalised or terminated the transaction before notifying the registries

If the transaction was terminated by the ITL, the ITL notifies the transferring registry to terminate the transaction, via the AcceptNotification web service that registries are required to implement. Response codes corresponding to the transaction's discrepancies will be included in the message.

If the transaction successfully completed by the ITL, the ITL notifies both the initiating and acquiring registries to finalise the transaction via the AcceptNotification web.

**Note:** In order to keep compatibility with previous versions of registry software, the transactions status contained in the AcceptNotification call to the transferring registry after successful completion (Figure 4.7: message 9; Figure 4.8: message 15) is **08. Accepted** instead of **04. Completed**.

#### Step 4 – Registry Complete

Once registries process the ITL notification it must complete the transaction, either by finalising it (if no discrepancy was found) or by terminating it. The registry calls the ITL's AcceptNotification web service to notify the ITL that it has completed the transaction.

#### Step 5 – ITL Close Transaction

Once the ITL receives notification from registries that they have completed (either finalised or terminated) the transaction, the ITL records this for administrative and problem management purposes. Once the ITL receives notifications from all parties involved in the transaction that they have completed, the ITL closes the transaction. Note that the transaction is considered complete, after Step 3.

### 4.1.13 Two Registry Transaction Sequence Diagrams

The UML sequence diagrams in this section describe the message flow for two registry transactions in all their variations:

Diagram describing message flow variation	Successfully finalised	Involves STL	Rejected by ITL	Rejected by STL	Rejected by Acq Registry
Figure 4.7: Two registry transaction: no STL, successful transaction finalisation. Note that the transaction is completed on the ITL before the registries are notified and the transaction is completed on the registry.	y	n	n	n	n
Figure 4.8: Two registry transaction: STL, successful transaction finalisation Note that the transaction is completed on the ITL before the STL is notified to complete the transaction. The registries are the last system to be notified, thus the transaction is completed last on registries.	y	y	n	n	n
Figure 4.9: Two registry transaction: ITL rejected Note that this is the same message flow, regardless of whether the registry is part of an STL or not, as the transaction is rejected by the ITL before it is passed to the STL.	n	n/a	y	n/a	n/a
Figure 4.10: Two registry transaction: STL rejected Note that the ITL is always the first to complete or terminate a transaction, the registries, always the last.	n	y	n	y	n/a
Figure 4.11: Two registry transaction: no STL, Acquiring Registry rejected Note that the Acquiring Registry terminates the transaction immediately on rejecting the transaction.	n	n	n	n	y
Figure 4.12: Two registry transaction: STL, Acquiring Registry rejected. Note that the Acquiring Registry terminates the transaction immediately on rejecting the transaction.	n	y	n	n	y

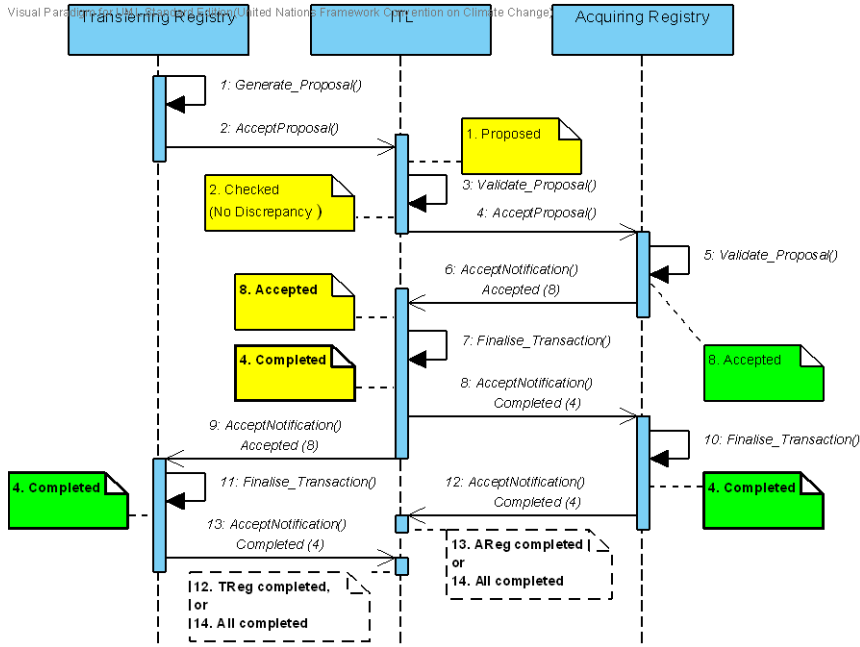


Figure 4.7: Two registry transaction: no STL, successful transaction finalisation

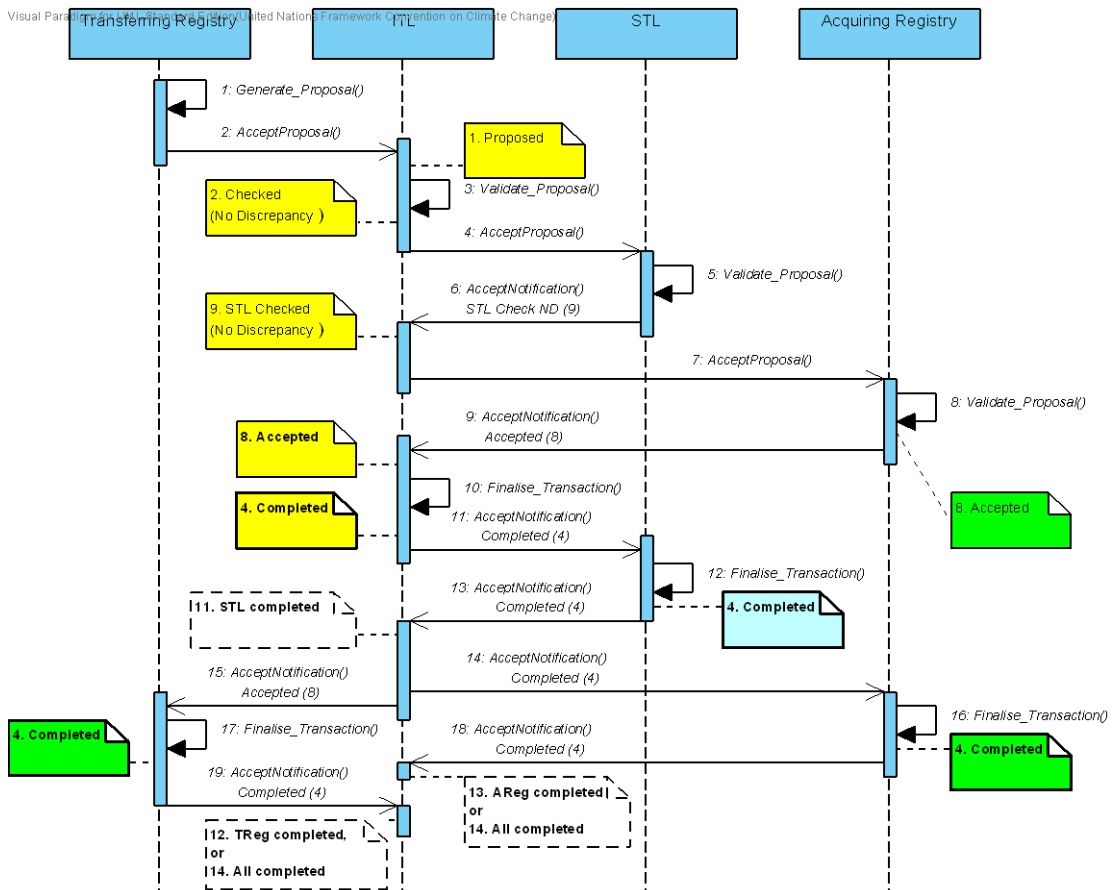
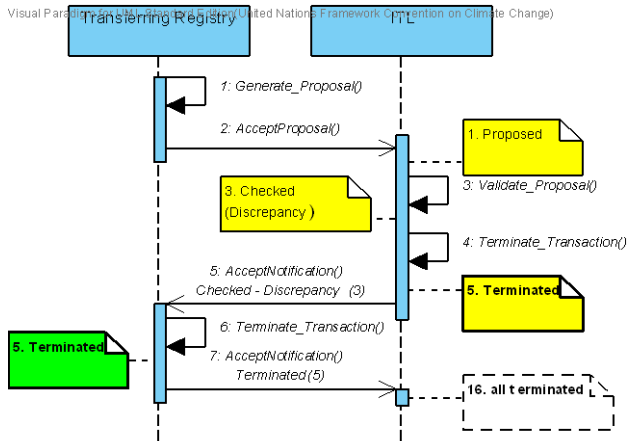
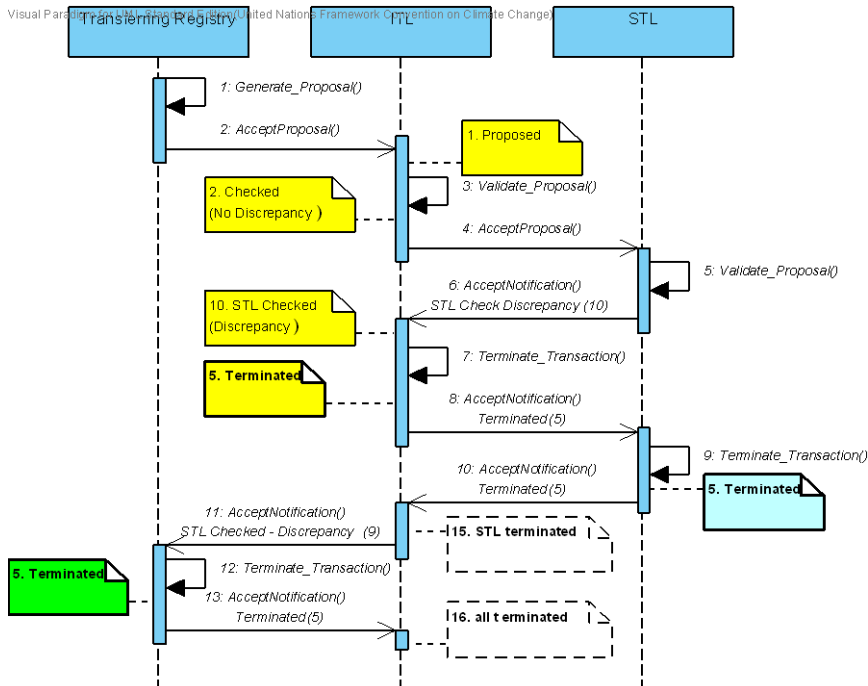


Figure 4.8: Two registry transaction: STL, successful transaction finalisation



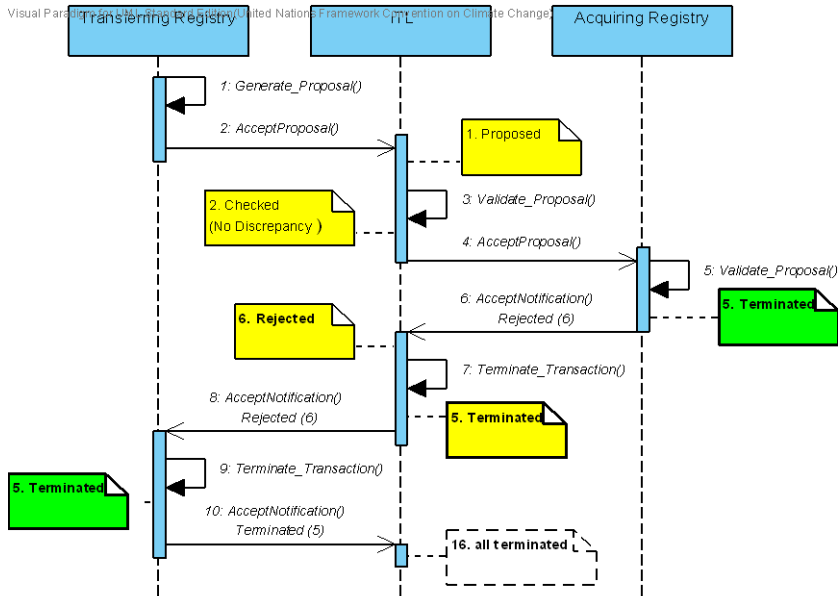
**Figure 4.9: Two registry transaction: ITL rejected**

- same flow for STL and non-STL transactions
- same flow for single and two registry transactions

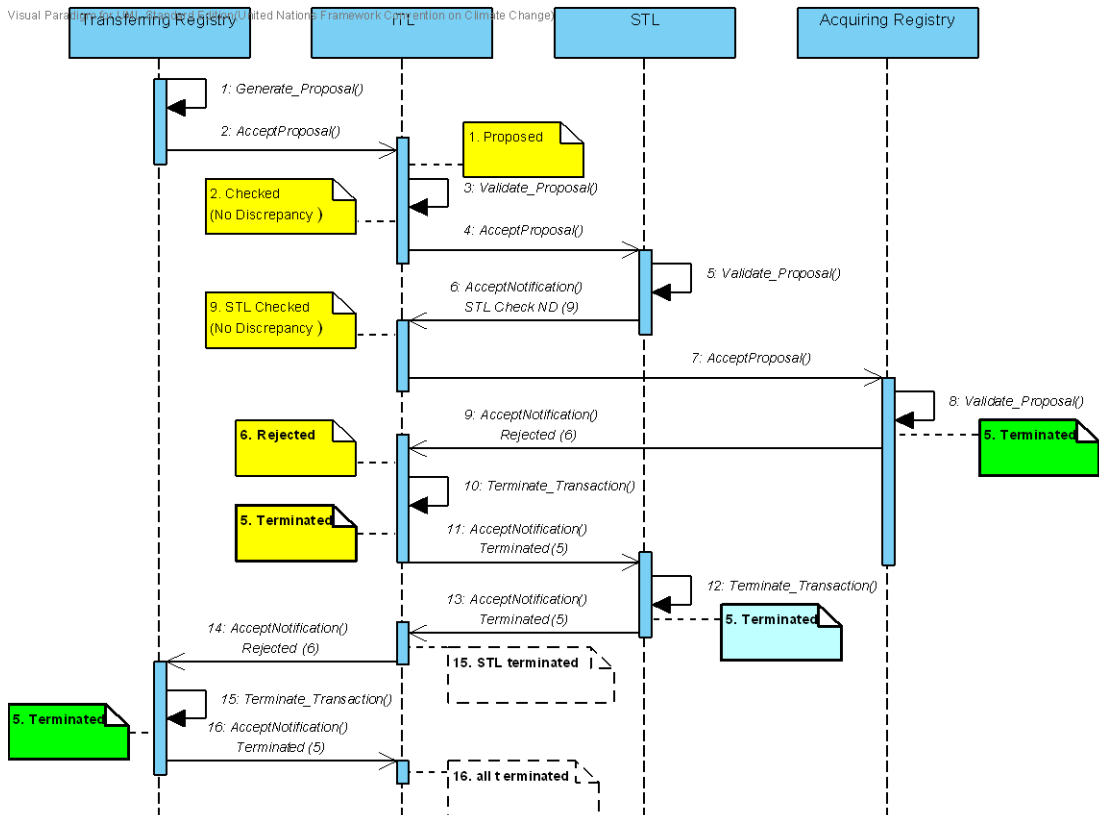


**Figure 4.10: Two registry transaction: STL rejected**

- same flow for single and two registry transactions



**Figure 4.11: Two registry transaction: no STL, Acquiring Registry rejected**



**Figure 4.12: Two registry transaction: STL, Acquiring Registry rejected**

Note: Acquiring Registry terminates transaction immediately after rejection - the ITL does not notify the Acquiring Registry to terminate.



#### 4.1.14 Two Registry Transaction State Transitions

This section describes the states of two registry transactions, and the events that cause the transactions to transition from one state to the next state.

For clarity, two diagrams are used:

- For two registry transactions not involving STLs, and
- For two registry transactions involving STLs

##### 4.1.14.1 Two Registry Transaction State Transitions, not involving STLs

This state transition diagram, Figure 4.13, describes transactions that do not involve STLs. The descriptions of the state and events are contained in Table 4.9 and Table 4.10 below.

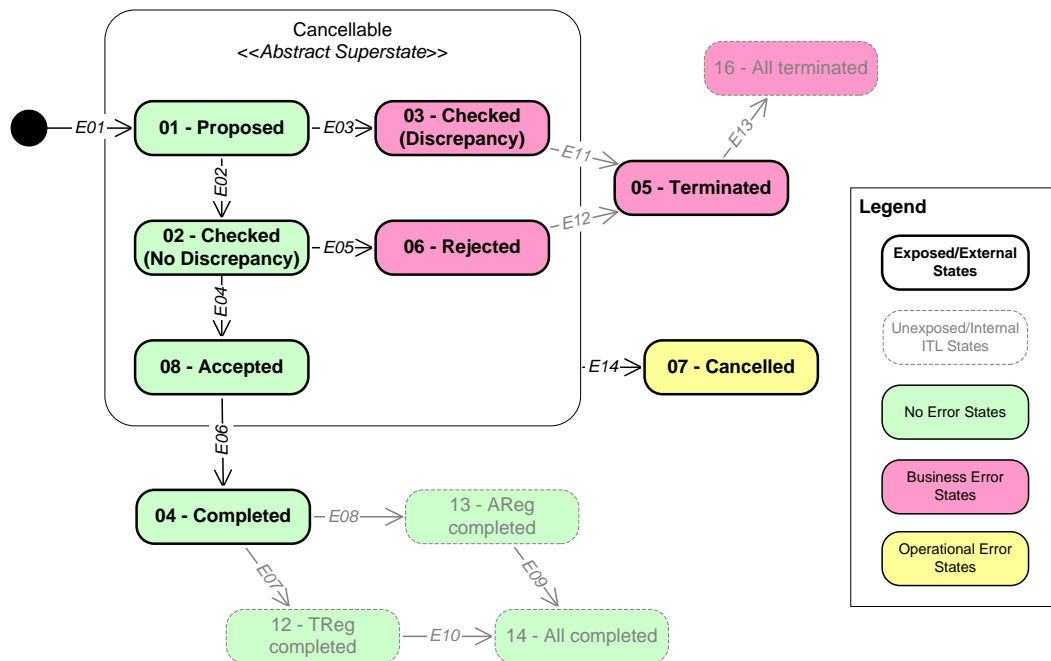


Figure 4.13: State Transition Diagram: Two Registry Transactions, not involving STLs

Table 4.9: State Descriptions - Two Registry Non-STL Transactions

State	Description	Entered via	Exits via
<b>01 - Proposed</b>	The transaction has been proposed by the Transferring registry. On entering this state, the ITL validates the transaction. The validation results in either <i>E02</i> or <i>E03</i> : <ul style="list-style-type: none"> <li>• valid (<i>E02</i>), the transaction transitions to <b>02 - Checked (No Discrepancy)</b>, or</li> <li>• invalid (<i>E03</i>), the transaction transitions to <b>03 - Checked (Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E01</i>: Idle</li> </ul>	<ul style="list-style-type: none"> <li>• <i>E02</i>: <b>02 - Checked (No Discrepancy)</b></li> <li>• <i>E03</i>: <b>03 - Checked (Discrepancy)</b></li> </ul>
<b>02 - Checked (No Discrepancy)</b>	The transaction has been checked by the ITL, and found to be valid. On entering this state, the ITL proposes the transaction to the acquiring registry for acceptance. When the ITL receives the proposal result from the acquiring registry, one of two	<ul style="list-style-type: none"> <li>• <i>E02</i>: <b>01 - Proposed</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E04</i>: <b>08 - Accepted</b></li> <li>• <i>E05</i>: <b>06 - Rejected</b></li> </ul>

State	Description	Entered via	Exits via
	<p>events occur:</p> <ul style="list-style-type: none"> <li>• <i>E04</i>, the acquiring registry accepted the transaction, in which case the transactions transitions to <b>08 - Accepted</b>, or</li> <li>• <i>E05</i>, the acquiring registry rejected the transaction, in which case, the transaction transitions to <b>06 - Rejected</b></li> </ul>		
<b>03 - Checked (Discrepancy)</b>	<p>The transaction has been checked by the ITL, and found to be invalid.</p> <p>On entering this state, the ITL terminates the transaction.</p> <p>The result of completing termination processing is <i>E11</i>, that transitions the transaction to <b>05 - Terminated</b>.</p>	<ul style="list-style-type: none"> <li>• <i>E03</i>: <b>01 - Proposed</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E11</i>: <b>05 - Terminated</b></li> </ul>
<b>04 - Completed</b>	<p>The transaction has been completed. Units in the transaction are available for other transactions.</p> <p>On entering this state, the ITL notifies the registries to finalise the transaction on their systems.</p> <p>The ITL receives notification from the registries that they have finalised the transaction.</p> <p>Depending on which registry's notification is received first, the transaction transitions to different next states:</p> <ul style="list-style-type: none"> <li>• if the notification from the transferring registry is received first (<i>E07</i>), the transaction transitions to <b>12 - TReg completed</b>.</li> <li>• if the notification from the acquiring registry is received first (<i>E08</i>), the transaction transitions to <b>13 - AReg completed</b>.</li> </ul>	<ul style="list-style-type: none"> <li>• <i>E06</i>: <b>08 - Accepted</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E07</i>: <b>12 - TReg completed</b></li> <li>• <i>E07</i>: <b>13 - AReg completed</b></li> </ul>
<b>05 - Terminated</b>	<p>The transaction has been terminated due to discrepancies. Units in the transaction are available for other transactions.</p> <p>On entering this state, the ITL notifies the transferring registry to terminate the transaction.</p> <p>When the ITL receives notification from the transferring registry that it has terminated the transaction (<i>E13</i>), the transaction transitions to <b>16 - All terminated</b>.</p>	<ul style="list-style-type: none"> <li>• <i>E11</i>: <b>03 - Checked (Discrepancy)</b></li> <li>• <i>E12</i>: <b>05 - Rejected</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E13</i>: <b>16 - All terminated</b></li> </ul>
<b>06 - Rejected</b>	<p>The ITL has received notification from the acquiring registry that it has rejected the transaction due to discrepancies.</p> <p>On entering this state, the ITL terminates the transaction.</p> <p>The result of completing termination processing is <i>E12</i>, that transitions the transaction to <b>05 - Terminated</b>.</p>	<ul style="list-style-type: none"> <li>• <i>E05</i>: <b>02 - Checked (No Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E12</i>: <b>05 - Terminated</b></li> </ul>
<b>07 - Cancelled</b>	<p>The transaction has been cancelled by the ITL's 24 hour transaction cleanup (<i>E14</i>)</p>	<ul style="list-style-type: none"> <li>• <i>E14</i>: <b>01 - Proposed</b> <b>02 - Checked (No Discrepancy)</b> <b>03 - Checked (Discrepancy)</b> <b>06 - Rejected</b> <b>08 - Accepted</b></li> </ul>	N/A
<b>08 - Accepted</b>	<p>The ITL has received notification from the acquiring registry that it has accepted the transaction.</p>	<ul style="list-style-type: none"> <li>• <i>E04</i>: <b>02 - Checked (No Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E06</i>: <b>04 - Completed</b></li> </ul>

State	Description	Entered via	Exits via
	On entering this state, the ITL finalises the transaction. The result of completing finalisation processing is <i>E06</i> , that transitions the transaction to <b>04 - Completed</b> .		
<b>12 - TReg completed</b>	The ITL has received notification from the transferring registry that it has finalised the transaction. The ITL has not yet received notification from the acquiring registry that it has finalised the transaction When the ITL receives notification from the acquiring registry that it has finalised the transaction ( <i>E10</i> ), the transaction transitions to <b>14 - All completed</b> .	<ul style="list-style-type: none"> <li>• <i>E07</i>: <b>04 - Completed</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E10</i>: <b>14 - All completed</b></li> </ul>
<b>13 - AReg completed</b>	The ITL has received notification from the acquiring registry that it has finalised the transaction. The ITL has not yet received notification from the transferring registry that it has finalised the transaction When the ITL receives notification from the transferring registry that it has finalised the transaction ( <i>E09</i> ), the transaction transitions to <b>14 - All completed</b> .	<ul style="list-style-type: none"> <li>• <i>E08</i>: <b>04 - Completed</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E09</i>: <b>14 - All completed</b></li> </ul>
<b>14 - All completed</b>	The transaction has been finalised by the ITL, the transferring registry, and the acquiring registry.	<ul style="list-style-type: none"> <li>• <i>E10</i>: <b>11 - TReg completed</b></li> <li>• <i>E09</i>: <b>11 - AReg completed</b></li> </ul>	N/A
<b>16 - All terminated</b>	The ITL has received notification from the transferring registry that it has terminated the transaction. The transaction has been terminated by the ITL, transferring registry, and the acquiring registry (if required).	<ul style="list-style-type: none"> <li>• <i>E13</i>: <b>05 - Terminated</b></li> </ul>	N/A
<b>Cancelable</b>	This is a superset of states that include <ul style="list-style-type: none"> <li>• <b>01 - Proposed</b></li> <li>• <b>02 - Checked (No Discrepancy)</b>,</li> <li>• <b>03 - Checked (Discrepancy)</b>,</li> <li>• <b>06 - Rejected</b>, and</li> <li>• <b>08 - Accepted</b></li> </ul> When the transaction is in one of these states, and has not transitioned for 24 hours or more, it is cancelled by the ITL's 24 hour transaction clean-up ( <i>E14</i> )	<ul style="list-style-type: none"> <li>• <i>E01</i>: <b>Idle</b></li> <li>• <i>E02, E03</i>: <b>01 - Proposed</b></li> <li>• <i>E04, E05</i>: <b>02 - Checked (No Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E06</i>: <b>04 - Completed</b></li> <li>• <i>E11, E12</i>: <b>05 - Terminated</b></li> <li>• <i>E14</i>: <b>07 - Cancelled</b></li> </ul>

**Table 4.10: ITL State Transition Event Descriptions - Two Registry Non-STL Transactions**

Id	Type	Description	Source
<i>E01</i>	message	ITL receives a transaction <b>proposal</b> via <i>acceptProposal()</i> from the transferring registry	Transferring Registry
<i>E02</i>	validation	ITL <b>validates</b> the proposed transaction	N/A
<i>E03</i>	validation	ITL finds <b>discrepancies</b> in the proposed transaction	N/A
<i>E04</i>	message	ITL receives <i>acceptNotification()</i> from the acquiring registry confirming that the proposed transaction has been <b>accepted</b> .	Acquiring Registry
<i>E05</i>	message	ITL receives <i>acceptNotification()</i> from the acquiring registry <b>rejecting</b> the proposed transaction.	Acquiring Registry

Id	Type	Description	Source
E06	process complete	ITL <b>finalises</b> the transaction	N/A
E07	message	ITL receives <i>acceptNotification()</i> from the transferring registry notifying the <b>transferring registry has completed</b> the transaction. The acquiring registry has yet to notify the ITL that it has completed.	Transferring Registry
E08	message	ITL receives <i>acceptNotification()</i> from the acquiring registry notifying the <b>acquiring registry has completed</b> the transaction. The transferring registry has yet to notify the ITL that it has completed.	Acquiring Registry
E09	message	ITL receives <i>acceptNotification()</i> from the transferring registry notifying the <b>transferring registry has completed</b> the transaction. The acquiring registry has already notified the ITL it has completed.	Transferring Registry
E10	message	ITL receives <i>acceptNotification()</i> from the acquiring registry notifying the <b>acquiring registry has completed</b> the transaction. The transferring registry has already notified the ITL it has completed.	Acquiring Registry
E11 E12	process complete	ITL <b>terminates</b> the transaction	N/A
E13	message	ITL receives <i>acceptNotification()</i> from the transferring registry notifying the <b>transferring registry has terminated</b> the transaction.	Transferring Registry
E14	time	The proposed transaction has not transitioned for 24 hours, and is <b>cancelled</b> by the ITL's 24 hour transaction clean-up.	N/A

#### 4.1.14.2 Two Registry Transaction State Transitions, involving STLs

This state transition diagram, Figure 4.14, describes transactions that do not involve STLs. The descriptions of the state and events are contained in Table 4.11 and Table 4.12 below.

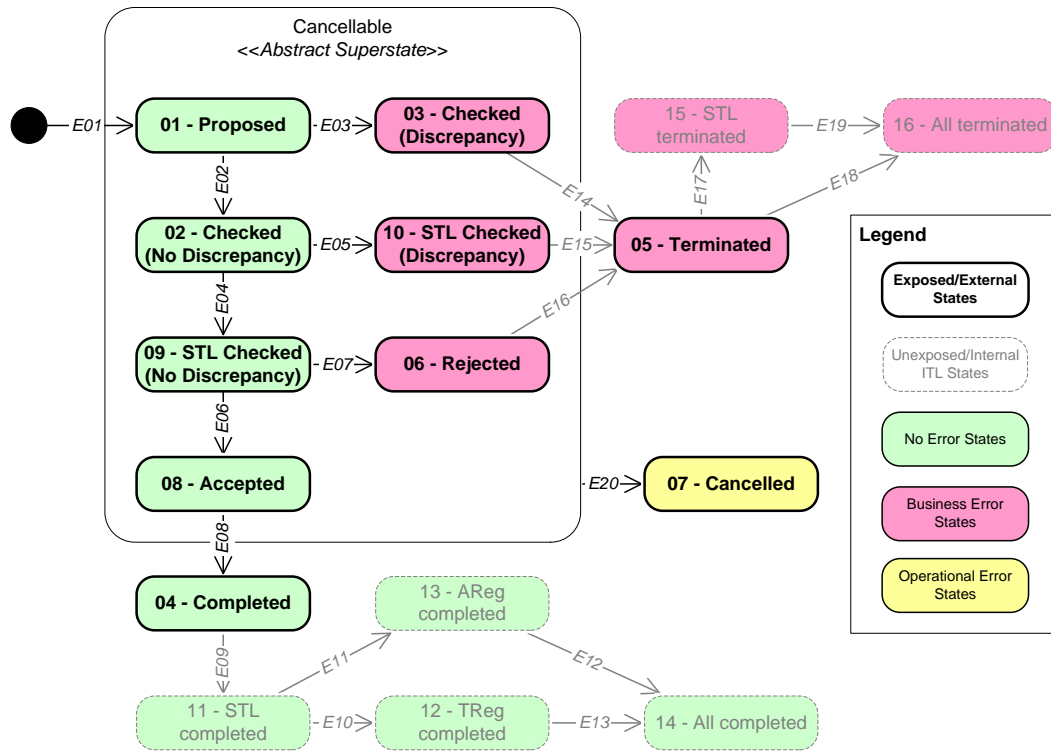


Figure 4.14: State Transition Diagram: Two Registry Transactions, not involving STLs

Table 4.11: State Descriptions - Two Registry STL Transactions

State	Description	Entered via	Exits via
01 - Proposed	The transaction has been proposed by the Transferring registry. On entering this state, the ITL validates the transaction. The validation results in either <i>E02</i> or <i>E03</i> : <ul style="list-style-type: none"> <li>valid (<i>E02</i>), the transaction transitions to <b>02 - Checked (No Discrepancy)</b>, or</li> <li>invalid (<i>E03</i>), the transaction transitions to <b>03 - Checked (Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li><i>E01</i>: Idle</li> </ul>	<ul style="list-style-type: none"> <li><i>E02</i>: <b>02 - Checked (No Discrepancy)</b></li> <li><i>E03</i>: <b>03 - Checked (Discrepancy)</b></li> </ul>
02 - Checked (No Discrepancy)	The transaction has been checked by the ITL, and found to be valid. On entering this state, the ITL proposes the transaction the STL for validation. When the ITL receives the validation result from the STL, one of two events occur: <ul style="list-style-type: none"> <li><i>E04</i>, the STL validated the transaction, in which case the transactions transitions to <b>09 - STL Checked (No Discrepancy)</b>, or</li> <li><i>E05</i>, the STL rejected the transaction, in which case, the transaction transitions to <b>10 - STL Checked (Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li><i>E02</i>: <b>01 - Proposed</b></li> </ul>	<ul style="list-style-type: none"> <li><i>E04</i>: <b>09 - STL Checked (No Discrepancy)</b></li> <li><i>E05</i>: <b>10 - STL Checked (Discrepancy)</b></li> </ul>
03 - Checked (Discrepancy)	The transaction has been checked by the ITL, and found to be invalid. On entering this state, the ITL terminates	<ul style="list-style-type: none"> <li><i>E03</i>: <b>01 - Proposed</b></li> </ul>	<ul style="list-style-type: none"> <li><i>E14</i>: <b>05 - Terminated</b></li> </ul>

State	Description	Entered via	Exits via
	the transaction. The result of completing termination processing is <i>E14</i> , that transitions the transaction to the <b>05 - Terminated</b> .		
<b>04 - Completed</b>	The transaction has been completed. Units in the transaction are available for other transactions. On entering this state, the ITL notifies the STL to finalise the transaction on its systems. When the ITL receives notification from the STL that it has finalised the transaction ( <i>E09</i> ), the transaction transitions to <b>11 - STL completed</b> .	<ul style="list-style-type: none"> <li>• <i>E06</i>: <b>09 – STL Checked (No Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E09</i>: <b>11 - STL completed</b></li> </ul>
<b>05 - Terminated</b>	The transaction has been terminated due to discrepancies. Units in the transaction are available for other transactions. If this state was entered via <ul style="list-style-type: none"> <li>• <i>E14</i> (from <b>03 - Checked Discrepancy</b>), the ITL notifies the transferring registry to terminate the transaction,</li> <li>• <i>E15</i> (from <b>10 - STL Checked (Discrepancy)</b>) or <i>E16</i> (from <b>06 - Rejected</b>), the ITL notifies the STL to terminate the transaction,</li> </ul> If the ITL received notification from the registry that it has terminated the transaction ( <i>E18</i> ), the transaction transitions to <b>16 - All terminated</b> . If the ITL receives notification from the STL that it has terminated the transaction ( <i>E17</i> ), the transaction transitions to <b>15 - STL terminated</b> .	<ul style="list-style-type: none"> <li>• <i>E14</i>: <b>03 - Checked (Discrepancy)</b></li> <li>• <i>E15</i>: <b>03 - STL Checked (Discrepancy)</b></li> <li>• <i>E16</i>: <b>05 - Terminated</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E17</i>: <b>15 - STL terminated</b></li> <li>• <i>E18</i>: <b>16 - All terminated</b></li> </ul>
<b>06 - Rejected</b>	The ITL has received notification from the acquiring registry that it has rejected the transaction due to discrepancies. On entering this state, the ITL terminates the transaction. The result of completing termination processing is <i>E16</i> , that transitions the transaction to <b>05 - Terminated</b> .	<ul style="list-style-type: none"> <li>• <i>E07</i>: <b>09 - STL Checked (No Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E16</i>: <b>05 - Terminated</b></li> </ul>
<b>07 - Cancelled</b>	The transaction has been cancelled by the ITL's 24 hour transaction cleanup ( <i>E14</i> )	<ul style="list-style-type: none"> <li>• <i>E20</i>: <b>01 - Proposed</b></li> <li><b>02 - Checked (No Discrepancy)</b></li> <li><b>03 - Checked (Discrepancy)</b></li> <li><b>06 - Rejected</b></li> <li><b>08 - Accepted</b></li> <li><b>09 - STL Checked (No Discrepancy)</b></li> <li><b>10 - STL Checked (Discrepancy)</b></li> </ul>	N/A
<b>08 - Accepted</b>	The ITL has received notification from the acquiring registry that it has accepted the transaction. On entering this state, the ITL finalises the transaction. The result of completing finalisation processing is <i>E08</i> , that transitions the transaction to <b>04 - Completed</b> .	<ul style="list-style-type: none"> <li>• <i>E06</i>: <b>09 - STL Checked (No Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E08</i>: <b>04 - Completed</b></li> </ul>
<b>09 - STL Checked (No Discrepancy)</b>	The ITL has received notification from the STL that the transaction is valid. On entering this state, the ITL proposes the	<ul style="list-style-type: none"> <li>• <i>E04</i>: <b>02 - Checked (No Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E06</i>: <b>08 - Accepted</b></li> <li>• <i>E07</i>:</li> </ul>

State	Description	Entered via	Exits via
	<p>transaction to the acquiring registry for acceptance.</p> <p>When the ITL receives the proposal result from the acquiring registry, one of two events occur:</p> <ul style="list-style-type: none"> <li>• <i>E06</i>, the acquiring registry accepted the transaction, in which case the transactions transitions to <b>08 - Accepted</b>, or</li> <li>• <i>E07</i>, the acquiring registry rejected the transaction, in which case, the transaction transitions to <b>06 - Rejected</b></li> </ul>		<b>06 - Rejected</b>
<b>10 - STL Checked (Discrepancy)</b>	<p>The ITL has received notification from the STL that the transaction is invalid.</p> <p>On entering this state, the ITL terminates the transaction.</p> <p>The result of completing termination processing is <i>E15</i>, that transitions the transaction to <b>05 - Terminated</b>.</p>	<ul style="list-style-type: none"> <li>• <i>E05</i>: <b>02 - Checked (No Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E15</i>: <b>05 - Terminated</b></li> </ul>
<b>11 - STL completed</b>	<p>The ITL has received notification from the STL that it has finalised the transaction.</p> <p>On entering this state, the ITL notifies the registries to finalise the transaction on their systems.</p> <p>The ITL receives notification from the registries that they have finalised the transaction.</p> <p>Depending on which registry's notification is received first, the transaction transitions to different next states:</p> <ul style="list-style-type: none"> <li>• if the notification from the transferring registry is received (<i>E10</i>), the transaction transitions to <b>12 - TReg completed</b>.</li> <li>• if the notification from the acquiring registry is received (<i>E11</i>), the transaction transitions to <b>13 - AReg completed</b>.</li> </ul>	<ul style="list-style-type: none"> <li>• <i>E09</i>: <b>04 - Completed</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E10</i>: <b>12 - TReg completed</b></li> <li>• <i>E11</i>: <b>13 - AReg completed</b></li> </ul>
<b>12 - TReg completed</b>	<p>The ITL has received notification from the transferring registry that it has finalised the transaction.</p> <p>The ITL has not yet received notification from the acquiring registry that it has finalised the transaction</p> <p>When the ITL receives notification from the acquiring registry that it has finalised the transaction (<i>E13</i>), the transaction transitions to <b>14 - All completed</b>.</p>	<ul style="list-style-type: none"> <li>• <i>E10</i>: <b>11 - STL completed</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E13</i>: <b>14 - All completed</b></li> </ul>
<b>13 - AReg completed</b>	<p>The ITL has received notification from the acquiring registry that it has finalised the transaction.</p> <p>The ITL has not yet received notification from the transferring registry that it has finalised the transaction</p> <p>When the ITL receives notification from the transferring registry that it has finalised the transaction (<i>E12</i>), the transaction transitions to <b>14 - All completed</b>.</p>	<ul style="list-style-type: none"> <li>• <i>E10</i>: <b>11 - STL completed</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E12</i>: <b>14 - All completed</b></li> </ul>
<b>14 - All completed</b>	<p>The transaction has been finalised by the ITL, the STL, and both registries</p> <p>The last registry has notified the ITL that it has finalised the transaction (<i>E12</i> or <i>E13</i>)</p>	<ul style="list-style-type: none"> <li>• <i>E13</i>: <b>12 - TReg completed</b></li> <li>• <i>E12</i>: <b>13 - AReg completed</b></li> </ul>	N/A

State	Description	Entered via	Exits via
<b>15 - STL terminated</b>	The ITL has received notification from the STL that it has terminated the transaction. On entering this state, the ITL notifies the transferring registry to terminate the transaction. When the ITL receives notification from the transferring registry that it has terminated the transaction ( <i>E19</i> ), the transaction transitions to <b>16 - All terminated</b> .	<ul style="list-style-type: none"> <li>• <i>E17</i>: <b>05 - Terminated</b></li> </ul>	<i>E19</i> : <b>16 - All terminated</b>
<b>16 - All terminated</b>	The transaction has been terminated by the ITL, the STL, and both registries. The transferring registry has notified the ITL that it has terminated the transaction ( <i>E18</i> , <i>E19</i> )	<ul style="list-style-type: none"> <li>• <i>E18</i>: <b>05 - Terminated</b></li> <li>• <i>E19</i>: <b>15 - STL Terminated</b></li> </ul>	N/A
<b>Cancelable</b>	This is a superset of states that include <ul style="list-style-type: none"> <li>• <b>01 - Proposed</b></li> <li>• <b>02 - Checked (No Discrepancy)</b>,</li> <li>• <b>03 - Checked (Discrepancy)</b>,</li> <li>• <b>06 - Rejected</b>,</li> <li>• <b>08 - Accepted</b>,</li> <li>• <b>09 - STL Checked (No Discrepancy)</b>, and</li> <li>• <b>10 - STL Checked (Discrepancy)</b></li> </ul> When the transaction is in one of these states, and has not transitioned for 24 hours or more, it is cancelled by the ITL's 24 hour transaction clean-up ( <i>E14</i> )	<ul style="list-style-type: none"> <li>• <i>E01</i>: <b>Idle</b></li> <li>• <i>E02</i>, <i>E03</i>: <b>01 - Proposed</b></li> <li>• <i>E04</i>, <i>E05</i>: <b>02 - Checked (No Discrepancy)</b></li> <li>• <i>E06</i>, <i>E07</i>: <b>09 - STL Checked (No Discrepancy)</b></li> </ul>	<ul style="list-style-type: none"> <li>• <i>E08</i>: <b>04 - Completed</b></li> <li>• <i>E14</i>, <i>E15</i>, <i>E16</i>: <b>05 - Terminated</b></li> <li>• <i>E20</i>: <b>07 - Cancelled</b></li> </ul>

**Table 4.12: ITL State Transition Event Descriptions - Two Registry STL Transactions**

Id	Type	Description	Source
<i>E01</i>	message	ITL receives a transaction <b>proposal</b> via <i>acceptProposal()</i> from the transferring registry	Transferring Registry
<i>E02</i>	validation	ITL <b>validates</b> the proposed transaction	N/A
<i>E03</i>	validation	ITL finds <b>discrepancies</b> in the proposed transaction	N/A
<i>E04</i>	message	ITL receives <i>acceptNotification()</i> from the STL confirming the proposed transaction has been <b>validated</b> by the CITL	STL
<i>E05</i>	message	ITL receives <i>acceptNotification()</i> from the STL notifying the STL has found <b>discrepancies</b> in the proposed transaction	STL
<i>E06</i>	message	ITL receives <i>acceptNotification()</i> from the acquiring registry confirming that the proposed transaction has been <b>accepted</b> .	Acquiring Registry
<i>E07</i>	message	ITL receives <i>acceptNotification()</i> from the acquiring registry <b>rejecting</b> the proposed transaction.	Acquiring Registry
<i>E08</i>	process complete	ITL <b>finalises</b> the transaction	N/A
<i>E09</i>	message	ITL receives <i>acceptNotification()</i> from the STL notifying the <b>STL has completed</b> the transaction in the CITL and the ITL can proceed to notify the registries to complete.	STL
<i>E10</i>	message	ITL receives <i>acceptNotification()</i> from the transferring registry notifying the <b>transferring registry has completed</b> the transaction. The acquiring registry has yet to notify the ITL that it has completed.	Transferring Registry
<i>E11</i>	message	ITL receives <i>acceptNotification()</i> from the acquiring registry notifying the <b>acquiring registry has completed</b> the transaction. The transferring registry has yet to notify the ITL that it has completed.	Acquiring Registry
<i>E12</i>	message	ITL receives <i>acceptNotification()</i> from the transferring registry notifying the <b>transferring registry has completed</b> the transaction. The acquiring registry has already notified the ITL it has completed.	Transferring Registry



Id	Type	Description	Source
E13	message	ITL receives <i>acceptNotification()</i> from the acquiring registry notifying the <b>acquiring registry has completed</b> the transaction. The transferring registry has already notified the ITL it has completed.	Acquiring Registry
E14 E15 E16	process complete	ITL <b>terminates</b> the transaction	N/A
E17	message	ITL receives <i>acceptNotification()</i> from STL notifying the <b>STL has terminated</b> the transaction.	STL
E18 E19	message	ITL receives <i>acceptNotification()</i> from the transferring registry notifying the <b>transferring registry has terminated</b> the transaction.	Transferring Registry
E20	time	The proposed transaction has not transitioned for 24 hours, and is <b>cancelled</b> by the ITL's 24 hour transaction clean-up.	N/A

## List of Functions for Transaction Data Exchange

### 4.1.15 Registry Web Services and Functions

In order to participate in data exchange with the ITL, registries must precisely implement Web services that the ITL can use to send it information. The following table shows the Web services methods registries are required to expose for unit transactions. Detailed technical information about the specifications for these Web service methods are in Annex B.

**Table 4.13: Registry Public Web Service Methods**

Public Web Service Method	Page
AcceptNotification	B-5
AcceptProposal	B-6

In addition to the above Web service methods that the registry must precisely implement so that they may be used by the ITL, the registry must have capabilities to build transactions, validate transactions, and log transactions. The following functions implement those responsibilities. Note that these functions are not exposed to the public, so they provide more flexibility in how they are implemented. Detailed technical information about the specifications for the Web service methods are in Annex B.

**Table 4.14: Registry Internal Functions**

Private Function	Page
Check_Version	B-7
Data_Integrity_Checks	B-8
Finalise_Transaction	B-9
Generate_Proposal	B-10
Preliminary_Checks	B-11
Update_Units	B-12
Validate_Proposal	B-13
Write_To_File	B-14
Write_To_Message_Log	B-15
Write_Transaction	B-16
Write_Transaction_Block	B-17
Write_Transaction_Status	B-18

### 4.1.16 ITL Web Services and Functions

Like the registries, the ITL must precisely implement public Web services called AcceptNotification and AcceptProposal to be used by registries in data exchange. Detailed

technical information about the specifications for these Web service methods are in Annexes I and K to this document. The ITL also contains extensive functionality for checking and logging data. Detailed technical information about the specifications for these Web service methods are in Annex K to this document.

## Transaction Checks

The ITL executes numerous checks on all transactions to assure the authenticity of a message, the format of the message, the sequence of the message, and the validity of the unit transaction. The following categories of checks are performed on the ITL. The list of specific checks and associated response codes is included in Annex E.

It is recommended that registries implement similar checks to reduce the number of discrepancies identified by the ITL.

### 4.1.17 Version and Authentication Checks

Version and authentication checks are performed within the Communications Hub as preliminary checks upon receipt of the HTTP SOAP request and do not involve any interaction with the ITL database. If these checks are passed, the message is placed in the message queue for processing. Failures due to authentication and poorly formed XML content are returned as HTTP SOAP errors. Failures due to transaction checks are returned in the ResponseObject in an HTTP SOAP response initiated by the ITL to the Originating Registry.

### 4.1.18 Message Viability Checks

Messages are placed in one of three different queues and are processed on a first-come-first-served basis. The time in which the message is added into the queue becomes the official time stamp in which the ITL acknowledges receipt of the message. However, should the ITL database be unavailable for an extended period of time due to hardware failure, messages remain in the queue until such time in which they can be processed. These checks determine whether the message from the queue is still viable and can be processed.

### 4.1.19 System Status Checks

After the message has been retrieved from the message queue and the location of the message file has been written to the message log, the ITL performs checks to determine if the systems (registries, STLs, and the ITL itself) involved in the transaction are identifiable and in a status that allows it to participate in the transaction.

System statuses are describe in Table 4.15.

The response codes returned to message senders when a system is in an incompatible status are described in Annex E.

**Table 4.15 - System Status Descriptions**

Code	Status	Applicable Systems	Impact on Transactions	Impact on Reconciliations	Other Messages, including ETS
0	Full Operation	All	ITL will send, accept and process all transaction messages with this system	Processed normally	Processed normally

Code	Status	Applicable Systems	Impact on Transactions	Impact on Reconciliations	Other Messages, including ETS
1	Stop Transactions	Registries	ITL will reject proposals and all other transaction messages involving this system and will send a response code to the message sender.	Processed normally	Processed normally
2	Not Operational	All	ITL will reject proposals and all other transaction messages involving this system and will send a response code to the message sender.	Not processed	Not sent
3	Closed	Registries	Used when a registry is not <i>live</i> with the ITL, e.g. prior to go-live or after it has shutdown. No messages will be sent to or accepted from the registry and transactions involving the registry will be terminated.	Not processed	Not sent
4	Suspend Proposals	Registries	ITL will queue transaction proposals from or to this system for later processing. Other transaction messages will be sent or accepted normally	Processed normally	Processed normally
5	Suspend All Transaction Processing	Registries	ITL will queue for later processing all transaction proposals and messages from or to this system.	Processed normally	Processed normally
6	Stop Proposals	Registries	ITL will reject proposals involving this system and the proposer will receive a response code. Other transaction messages will be processed normally	Processed normally	Processed normally
7	Stop Proposals and Suspend Transaction Processing	Registries	ITL will reject proposals involving this system and the proposer will receive a response code. Other transaction messages will be queued for later processing.	Processed normally	Processed normally

#### 4.1.20 Data Integrity Checks for Transactions

This category of checks is performed by the ITL's `Data_Integrity_Checks` function to identify whether incoming messages contain data that do not meet basic data integrity checks. If any data in a message fail these checks, the message is returned to the sender with an appropriate response code. The message is not logged in the ITL's Transaction Log table and is not processed further. All data integrity checks are critical checks; if they result in failure, no further checks are processed.

#### 4.1.21 Message Sequence Checks for Transactions from Registries

After the data in the message have been checked, the ITL performs checks to ensure that the message received has been submitted in the proper sequence, including whether process status is consistent and appropriate.

This includes checking the special *out-of-sequence* case: if the incoming message is a transaction proposal (acceptProposal), and the transaction id is the same as a transaction that is already in the ITL.

When *out-of-sequence* messages are received, the ITL **will not** respond to these messages with another asynchronous message. The messages are logged in the ITL's message log for auditing and troubleshooting purposes.

#### **4.1.22 General Transaction Checks**

The ITL performs this category of checks for all transaction messages involving unit blocks.

#### **4.1.23 Transaction-specific Checks**

The ITL performs this category of checks on all Kyoto transactions for the specified Transaction Types. The checks include checks performed by the ITL to verify limitations on registry transactions. These include maintaining and monitoring the Commitment Period Reserve and limits on issuance, both by national registries and the CDM Registry.

## 5 Reconciliation Process

### Reconciliation Process Flow

The data on unit holdings in registries and the ITL are reconciled on a periodic basis on the basis of a data snapshot at a specified time. The snapshot taken must treat proposed transactions (in any status prior to "Completed") as if the transaction had not yet occurred. All unit type and account types for unit blocks held by the registry must be totaled for purposes of reconciliation as if they had not been changed or transferred by any ongoing transactions. This approach is necessary to ensure consistency of totals and unit blocks with the ITL, which will not commit changes in ownership or unit block types (or other attributes) until the message with the transaction status of "Completed" is received from the Initiating Registry.

It is recommended that registries delay committing database transactions for proposed transactions and sending the messages for "Completed" transactions for a short period of time surrounding the reconciliation snapshot date and time. The amount of time recommended will vary based on message processing time and is within the discretion of the Registry Manager. The ITL will not change the processing of messages to avoid possible inconsistency. A prolonged period of registry non-operation or suspension of transactions for reconciliation purposes is not foreseen.

A reconciliation action is completed when no inconsistencies are discovered or when any discovered inconsistencies have been resolved. The reconciliation process is implemented in phases in which different types of data are requested:

- Confirm Reconciliation
- Phase 1 – Validate Totals
- Phase 2 – Validate Unit Blocks
- Phase 3 – Review Audit Logs

Procedures for the use of this reconciliation process are to be agreed among the administrators of the ITL and registries. These procedures will address, for example, the scheduling of reconciliation phases and approaches to manual intervention to correct inconsistencies. They will include the possibility of directly initiating later reconciliation phases without first passing the earlier phases, or choosing to continue with the later phases even where the earlier phases did not identify any inconsistencies.

#### Confirm Reconciliation

1. Prior to the agreed upon snapshot time, the ITL Administrator opens a reconciliation action for the applicable registry and records the status as "Confirmed" (0).
2. The ITL calls the InitiateReconciliation Web service on the registry. This message contains the designated snapshot time and acts as confirmation of the snapshot date and time previously agreed to by the registry.
3. If the registry is part of a supplementary program, the ITL also calls the InitiateReconciliation Web service on the STL. This message contains the designated snapshot time previously agreed upon.
4. At the designated time the ITL and the registry create a snapshot upon which the following data analysis relies.

#### Phase 1 – Validate Totals

1. Upon completion of its snapshot, the ITL requests unit holding totals by account type, Commitment Period and unit type from the registry. To do so the ITL calls the ProvideTotals Web service method on the registry.

2. The registry receives the request from the ITL, compiles the totals, and sends the totals to the ITL by calling the ReceiveTotals web service on the ITL.
3. The ITL performs preliminary checks on the message and adds the message to the processing queue. When the ITL removes the message from the queue it performs registry validation, reconciliation data integrity and message sequence checks. Failure results in rejection of message without data recording or further processing.
4. If all the checks are passed, the ITL compares the totals sent by the registry with its own records and determines a result.
5. The ITL records the new status of the reconciliation action.
  - a. If the status is "Validated" (2), the ITL checks if the Party is in a supplementary program. If it is, the ITL initiates STL Reconciliation processing. If the Party is not in a supplementary program, the ITL sends notification to the registry that the reconciliation completed successfully. The ITL also removes the freeze flag from any units that remain flagged from a previous reconciliation action at that registry.
  - b. If the new status is "Inconsistent Totals" (3):
    - i. If the registry is not part of a supplementary program the ITL requests the registry to send unit block details (as described in Phase 2)
    - ii. If the registry is part of a supplementary program, the ITL once again requests unit holding totals from the registry, but this time, by account type, commitment period, unit type *and account identifier* by calling the ProvideTotals web service method on the registry with the parameter *byAccountFlag* set to 1 (i.e. back to step 1, but calling ProvideTotals with *byAccountFlag* set to 1)
    - iii. If the "Inconsistent Totals" status was a result of a call to ProvideTotals with *byAccountFlag* set to 1 (step 5.b.ii), the ITL request the registry to send unit block details (as described in Phase 2)

#### Phase 2 – Validate Unit Blocks

1. The ITL calls the ProvideUnitBlocks Web service method on the registry to request that the registry send unit blocks. This request may be limited to unit blocks for a specific unit type, account type, Commitment Period combination that failed the totals check.
2. The registry sends its unit block inventory to the ITL by calling the ReceiveUnitBlocks Web service method.
3. ITL performs preliminary checks on the message and adds the message to the processing queue. When the ITL removes the message from the queue it performs registry validation, reconciliation data integrity and message sequence checks. Failure results in rejection of message without data recording or further processing.
4. If all the checks are passed, the ITL compares each unit block sent by the registry against the ITL records. If blocks do not match, they are marked as inconsistent.
5. If no inconsistent blocks are found and Phase 2 is not the starting phase, a manual intervention is triggered to explain why the totals check in the first phase of reconciliation failed. If inconsistent blocks are found, the ITL requests the registry to send a transaction history since the last reconciliation for each inconsistent block.

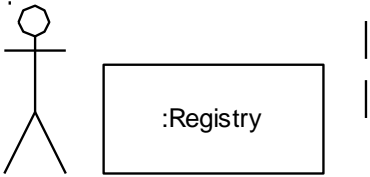






#### Phase 3 – Review Audit Logs

1. The ITL calls the ProvideAuditTrail Web service method on the registry to request the transaction history for each inconsistent block.

2. The registry sends the transaction history to the ITL by calling the ReceiveAuditTrail Web service method.
3. The ITL performs preliminary checks on the message and adds the message to the processing queue. When the ITL removes the message from the queue it performs registry validation, reconciliation data integrity and message sequence checks. Failure results in rejection of message without data recording or further processing.
4. If all the checks are passed the ITL writes the transaction history to a flat file and stores the location in the ITL's Message Log table.
5. The ITL and Registry Administrators research and correct the cause of the inconsistency through manual intervention. Once corrected, a new reconciliation is immediately initiated by the ITL Administrator.

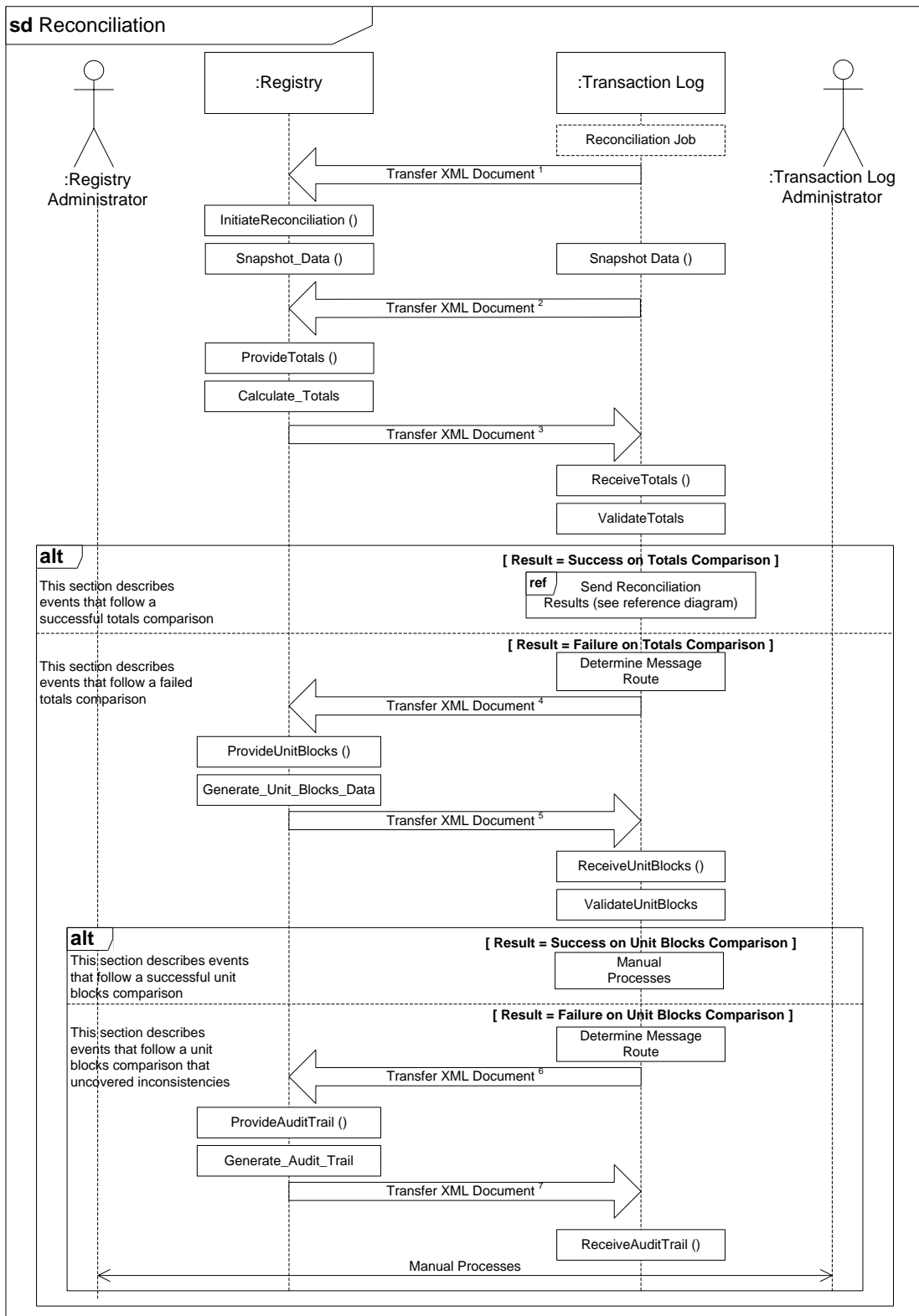


**Table 5.1: Notation used in Reconciliation Behaviour Diagrams**

Diagram Element	Description
<p>Actors &amp; swim lanes</p> 	<p>At the top of each diagram the participants in the process are represented by a word preceded with a colon (:). Actions involving a participant are presented in the "swim lane" which is directly underneath the participant's icon or box, and represented by a dashed vertical line.</p>
	<p>This symbol indicates that the diagram is a sequence diagram. The symbol is followed by the name of the process.</p>
	<p>This symbol indicates that there is a secondary sub-diagram for the component which provides additional detail of the functionality.</p>
	<p>This symbol indicates that the process supports alternative outcomes in the prior step. Within an alternative, there may be a second alternative scenario, equivalent to programs which contain nested "if...then" statements. In the issuance process, for example, the issuance is either accepted (Result = Success) or a discrepancy is identified (Result = Failed). If successful, the registry can either confirm the issuance or terminate the issuance.</p>
	<p>This symbol indicates that the process within the box will only be executed if a certain condition is met.</p>
	<p>This symbol indicates that the process in the box is repeated a number of times. For example in the Time Synchronization process, the processes within the box are executed once for each registry that interacts with the ITL.</p>
	<p>This symbol represents a message containing an XML document, its transfer and the "acknowledgement" of its receipt. The message is "sent" from one component and "received" by another component, as indicated in footnote (x).</p>
<p>Boxes with dotted outlines</p>	<p>These boxes represent a component or area of functionality necessary to the process, but which does not have specifically defined input or output parameters used for messaging. These components could be defined and implemented by developers in many different ways.</p>
<p>Boxes with solid outlines</p>	<p>These boxes represent a component which performs a specific task necessary to the process. These components either receive or produce the information which is used for messaging.</p>

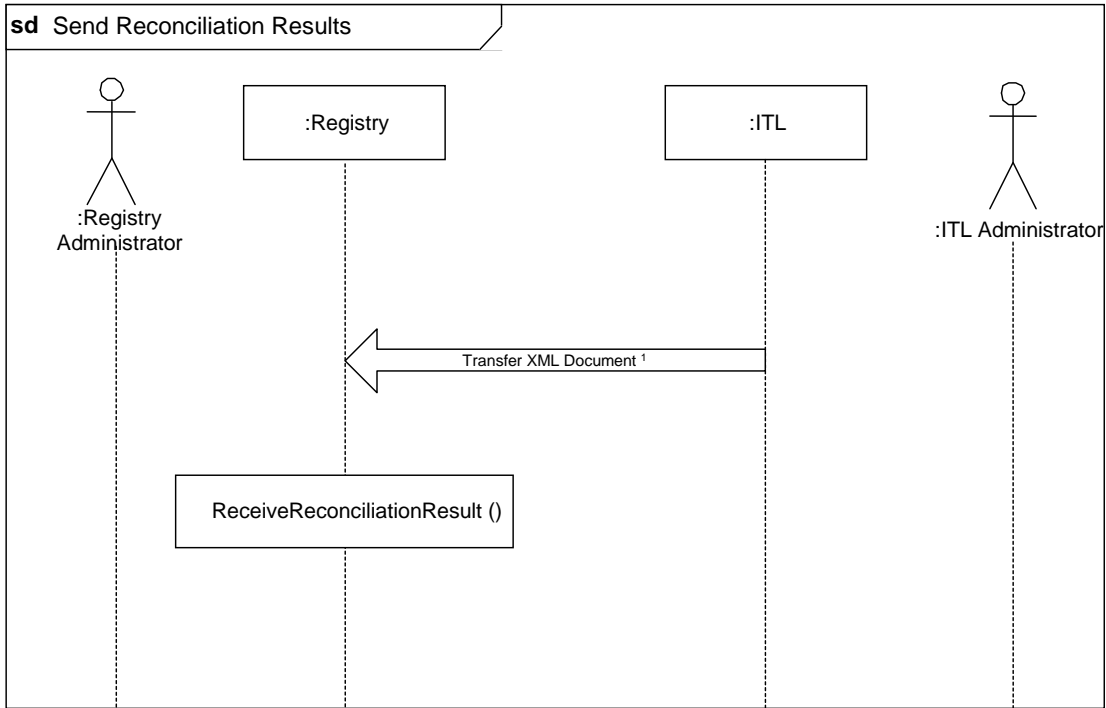
## 5.1.1 Reconciliation Behaviour Diagrams

Figure 5.1: Reconciliation Behaviour Diagram



1. The ITL creates an XML document and sends it to the InitiateReconciliation Web service on the registry.
2. The ITL creates an XML document and sends it to the ProvideTotals Web service on the registry.
3. The registry sends an XML document to the ReceiveTotals Web service on the ITL.
4. The ITL sends an XML document to the ProvideUnitBlocks Web service on the registry.
5. The registry sends an XML document to the ReceiveUnitBlocks Web service on the ITL.
6. The ITL sends an XML document to the ProvideAuditTrail Web service on the registry.
7. The registry sends an XML document to the ReceiveAuditTrail Web service on the ITL.

**Figure 5.2: Send Reconciliation Results Behaviour Diagram**



1. The UpdateReconciliation function on the ITL sends an XML document to the ReceiveReconciliationResult Web service on the registry to inform the registry that Reconciliation is complete.

## Reconciliation Stage Tables

The Stage Table represents the sequence of events in terms of the "stage" and its relation to the "status" of a reconciliation action. The stage defines where in the process of information exchange a particular message or evaluation occurs. A stage ends and a new stage begins when a message has been successfully transmitted and received by either a registry or the ITL or when the last step of a process occurs. The order in which each defined stage occurs may vary based on the specific process and based on the results of the ITL validation process. The numbers assigned to stages should not be used as an indicator of acceptable stage sequences.

**Table 5.2: Confirm Reconciliation**

Confirm Reconciliation	Stage Name	Stage Ends With	Reconciliation Status	Sent To	Generated From
Supplementary Program, Confirm Snapshot Time with Registry and STL	Confirm Registry	Message 1	"Confirmed"	Registry	ITL
	Confirm STL	Message 2	"Confirmed"	STL	ITL

**Table 5.3: Reconciliation Phase 1 - Validate Account Totals**

Phase 1	Stage Name	Stage Ends With	Reconciliation Status	Sent To	Generated From
Supplementary Program, Validated Totals at ITL and STL	Confirm Registry	Message 1	"Confirmed"	Registry	ITL
	Confirm STL	Message 2	"Confirmed"	STL	ITL
	Request	Message 3	"Initiated"	Registry	ITL
	Totals Sent	Message 4	"Initiated"	ITL	Registry
	Totals Evaluated	Message 5	"Validated"	Registry	ITL
	Totals By Account Sent	Message 6	"Validated"	ITL	Registry
	Totals By Account Sent	Message 7	"Validated"	STL	ITL
	Totals Evaluated by STL	Message 8	"STL Validated"	ITL	STL
	Totals Evaluated by STL	Message 9	"STL Validated"	Registry	ITL
	Reconciliation Complete	No message	"STL Validated"	--	--

**Table 5.4: Reconciliation Phase 2 - Validate Unit Blocks**

Phase 2	Stage Name	Stage Ends With	Reconciliation Status	Sent To	Generated From
Supplementary Program, Inconsistent Totals at ITL	Confirm Registry	Message 1	"Confirmed"	Registry	ITL
	Confirm STL	Message 2	"Confirmed"	STL	ITL
	Request	Message 3	"Initiated"	Registry	ITL
	Totals Sent	Message 4	"Initiated"	ITL	Registry
	Totals Evaluated	Message 5	"Totals Inconsistent"	Registry	ITL
	Unit Blocks Sent	Message 6	"Totals Inconsistent"	ITL	Registry
	Unit Blocks Evaluated	Message 7	"Unit Blocks Inconsistent"	Registry	ITL
	Audit Trail Sent	Message 8	"Unit Blocks Inconsistent"	ITL	Registry
	Manual Intervention	Message 9	"Complete with Manual Intervention"	Registry	ITL
	Reconciliation Complete	No Message	"Complete with Manual Intervention"	--	--
	New reconciliation action will be initiated by ITL				

**Table 5.5: Reconciliation Phase 3 - Review Audit Logs**

Phase 3	Stage Name	Stage Ends With	Reconciliation Status	Sent To	Generated From
<b>Supplementary Program, Validated Totals at ITL, Inconsistent Totals at STL</b>	Confirm Registry	Message 1	"Confirmed"	Registry	ITL
	Confirm STL	Message 2	"Confirmed"	STL	ITL
	Request	Message 3	"Initiated"	Registry	ITL
	Totals Sent	Message 4	"Initiated"	ITL	Registry
	Totals Evaluated	Message 5	"Validated"	Registry	ITL
	Totals By Account Sent	Message 6	"Validated"	ITL	Registry
	Totals By Account Sent Relay	Message 7	"Validated"	STL	ITL
	Totals Evaluated By STL	Message 8	"STL Totals Inconsistent"	ITL	STL
	Totals Evaluated By STL Relay	Message 9	"STL Totals Inconsistent"	Registry	ITL
	Unit Blocks Sent to STL	Message 10	"STL Totals Inconsistent"	ITL	Registry
	Unit Blocks Sent to STL relay	Message 11	"STL Totals Inconsistent"	STL	ITL
	STL Unit Blocks Inconsistent	Message 12	"STL Unit Blocks Inconsistent"	ITL	STL
	STL Unit Blocks Inconsistent Relay	Message 13	"STL Unit Blocks Inconsistent"	Registry	ITL
	Audit Trail Sent to STL	Message 14	"STL Unit Blocks Inconsistent"	ITL	Registry
	Audit Trail Sent to STL Relay	Message 15	"STL Unit Blocks Inconsistent"	STL	ITL
	Manual Intervention	Message 16	"STL Complete with Manual Intervention"	ITL	STL
	Manual Intervention	Message 17	"STL Complete with Manual Intervention"	Registry	ITL
	Reconciliation Complete	No Message	"STL Complete with Manual Intervention"	--	--
	New reconciliation action will be requested by STL				

## List of Functions for Reconciliation Process

### 5.1.2 Registry Web Services and Functions

In order to participate in reconciliation with the ITL, registries must precisely implement Web services that the ITL can use to send it information. The following table shows the Web services methods registries are required to expose for reconciliation. Detailed technical information about the specifications for these Web service methods are in Annex C.

**Table 5.6: Registry Public Web Service Methods**

Public Web Service Method	Figure
InitiateReconciliation	C5
ProvideAuditTrail	C6
ProvideTotals	C7
ProvideUnitBlocks	C8
ReceiveReconciliationResult	C9

In addition to the above Web service methods that the registry must precisely implement so that they may be used by the ITL, the registry must have additional capabilities to build transactions, validate transactions, and log transactions. The following functions implement those responsibilities. Note that these functions are not exposed to the public, so they provide more flexibility in how they are implemented.

**Table 5.7: Registry Internal Functions**

Private Function	Figure
Close_Reconciliation_Action	C2
Snapshot_Data	C10
Write_To_Reconciliation_Log	C11
Write_To_Reconciliation_Status	C12

### 5.1.3 ITL Web Services and Functions

Like the registries, the ITL must precisely implement public Web services called ReceiveTotals, ReceiveUnitBlocks, and ReceiveAuditTrail to be used by registries in data exchange. Detailed technical information about the specifications for these Web services can be found in the ITL Technical Specification.

The ITL also contains functionality for recording data. Detailed technical information about the specifications for these functions can be found in the ITL Technical Specification.

## Reconciliation Checks and Responses

The ITL executes numerous checks on all reconciliation messages to assure authenticity, format, and sequence of message. The following categories of checks are performed on the ITL. The list of specific checks and associated response codes is included in Annex E.

### 5.1.4 Version and Authentication Checks for Reconciliation

Preliminary checks, including version and authentication checks, are performed upon receipt of the HTTP SOAP request from a registry and do not involve any interaction with the ITL database. If these checks are passed, the message is placed in the message queue for processing. Failures due to authentication and poorly formed XML content are returned as HTTP SOAP fault errors. Failures due to any reconciliation check are returned in the ResponseObject in an HTTP SOAP response.

### 5.1.5 Registry Validation Checks for Reconciliation

When the message has been retrieved from the message queue and recorded in the message log, checks are performed to determine if the registries involved in the reconciliation action are identifiable and eligible to participate.

### 5.1.6 Data Integrity Checks for Reconciliation

This category of checks is performed by the ITL to identify whether incoming messages contain data not meeting basic data integrity checks. If any data in a message fail these checks, the message is returned to the sender with an appropriate response code. The message is not logged and is not processed further. Data integrity checks are critical checks and if they result in failure, no further checks are processed.

Note that as part of reconciliation, transactions and unit blocks are passed into the ITL, but those items are minimally checked by the data integrity checks. If there is a problem with the format of a transaction or a unit block, the reconciliation process will identify and log those items as the source of an inconsistency.

### 5.1.7 Message Sequence Checks for Reconciliation Messages Received from Registries

After the data in the message have been checked, the ITL performs checks to ensure that the message received has been submitted in the proper sequence, including whether process status is consistent and appropriate.

When *out-of-sequence* messages are received, the ITL **will not** respond to these messages with another asynchronous message. The messages are logged in the ITL's message log for auditing and troubleshooting purposes.

### 5.1.8 Other Reconciliation Checks and Messages

The ITL performs this category of checks to compare a registry's unit holding records with the ITL unit holding records.

## 6 ITL Administrative Functions

The ITL has six types of administrative Web service functions that involve transmitting messages to registries. These are for ensuring the integrity of the transactions and the wider accounting of units under the Kyoto Protocol. Some messages are periodic while others are initiated by administrators of the ITL or registries:

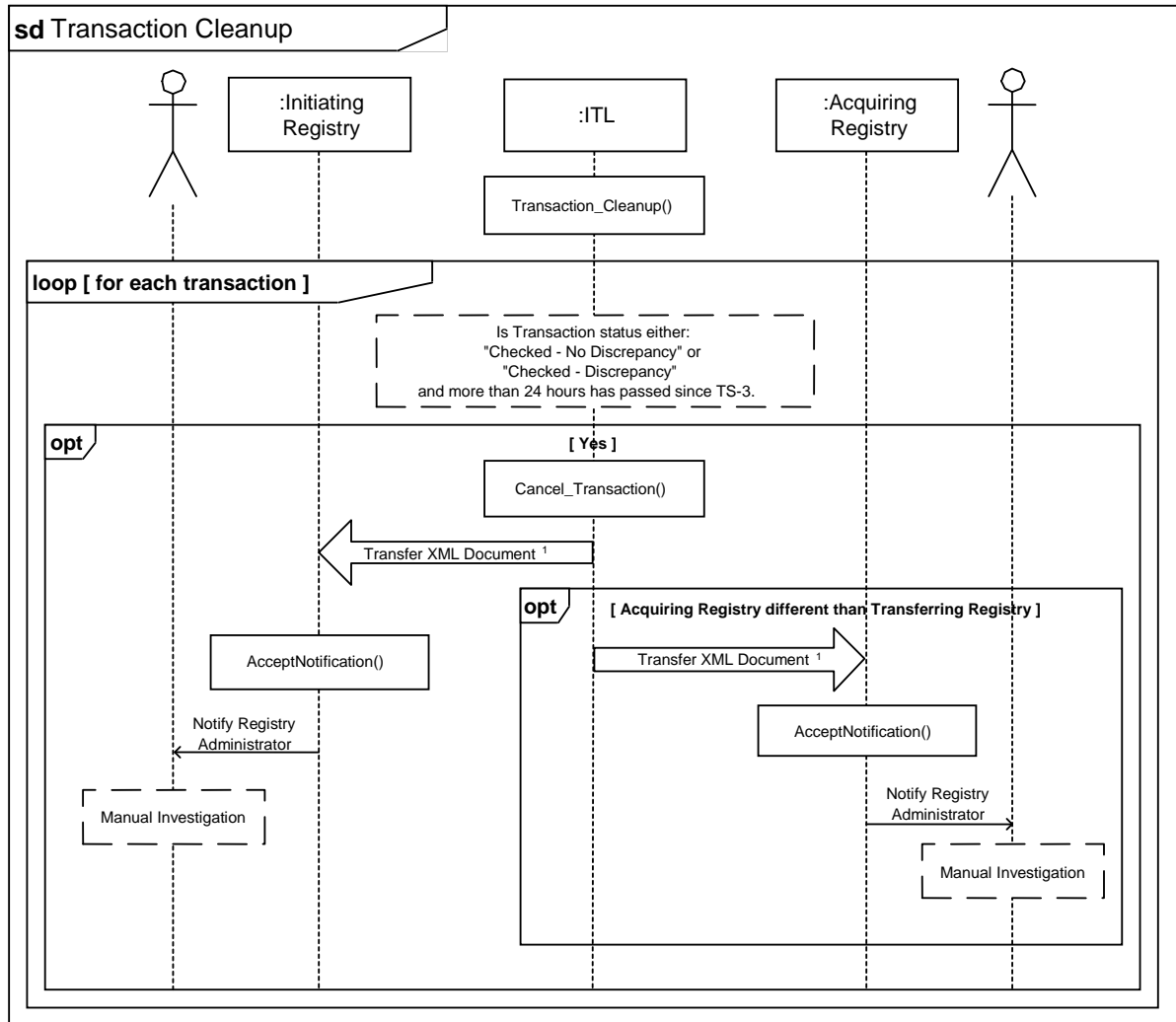
- Transaction Clean-up. This function cancels transactions that are not completed within the allowable timeframe and notifies registries of this action using AcceptNotification Web service.
- Notifications. This function on the ITL sends notifications to the AcceptITLNotice Web service on registries to notify them of actions which they need to undertake. The messages require registries to submit transactions.
- General Messages. The ITL may send general messages to a registry through the AcceptMessage Web service.
- Transaction Status Information. The registry may at any time submit a request to the GetTransactionStatus Web service on the ITL Communications Hub to provide the status of a specific transaction. This is a synchronous communication and the ITL provides the current status of the transaction as an immediate response.
- Time Synchronization. The ITL may at any time submit a request to the ProvideTime Web service on a registry to provide the time. This is a synchronous communication and the registry must provide the time as an immediate response.
- Heartbeat connection health monitoring by STLs. The STLs may use the AcceptMessage Web service to send heartbeat requests to the ITL to monitor the *health* of their connection to the ITL.

### 24-Hour Transaction Clean-up

In order to maintain data integrity, the ITL identifies transactions in progress for which a message has not been received within 24 hours. This check shall be performed once an hour. The ITL cancels these transactions. After the transaction is cancelled, the unit status is modified such that they are available to be involved in another transaction and a notification is sent to the registries involved in the transaction. The System Administrators of the registries should review the notification, investigate the reason for the lack of communication, and reinstate the transaction as a new transaction, if appropriate. The clean-up process uses the AcceptNotification Web service to send the message about a cancelled transaction to the registry. For such cancelled transactions, the response code returned for aged messages is 8002 (Transaction Clean-up).



Figure 6.1: Transaction Clean-up Diagram

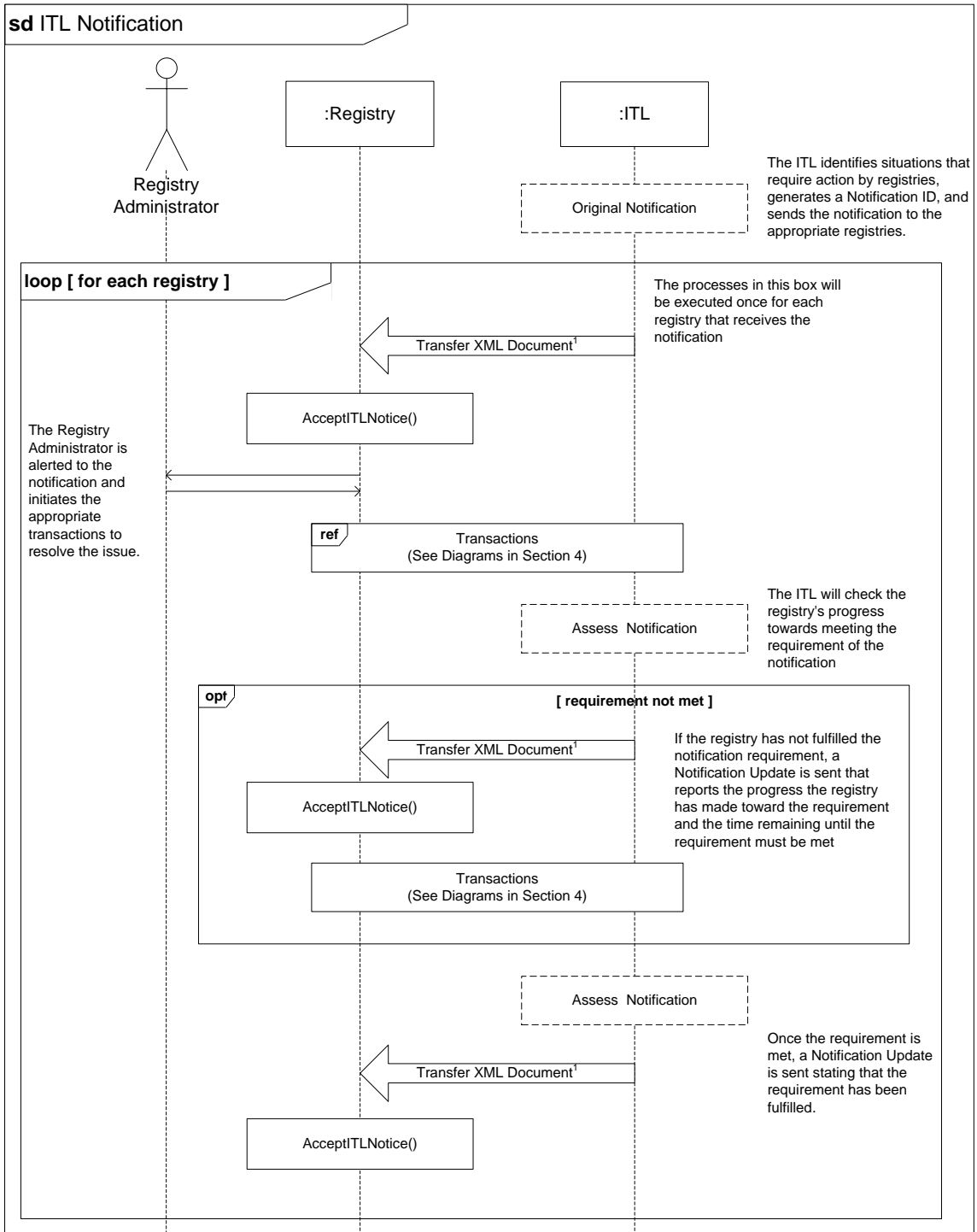


1. The CancelTransaction function on the Transaction Log creates and sends an XML document to the AcceptNotification Web service on the registry to inform the registry that the transaction has been cancelled.

## Notifications

The ITL performs these administrative functions upon initiation by the ITL Administrator to evaluate data and inform the registries of specific actions required. Each of these functions may result in notification to one or more registries regarding actions that must be taken by a registry. Each of these notifications are associated with a Notification Type Code defined in Annex G and may be repeated by the ITL as reminders of the required action. All notifications use the AcceptITLNotice Web service.

Figure 6.2: ITL Notification



1. The function prepares and sends an XML document to the AcceptITLNotice Web service on the registry.

### **6.1.1 Net Source Cancellation**

In the case that the review and Compliance Committee procedures under the Kyoto Protocol find that a specific LULUCF activity of a Party has resulted in a net source of emissions, the ITL will notify the Party of the quantity of units it is required to cancel within 30 days, and the associated LULUCF code as part of a net source cancellation action. The notification will also specify the Commitment Period of the units to be cancelled. This quantity of units must be cancelled into a Net Source Cancellation Account (Account Type Code 210). The registry will initiate cancellation transactions, providing reference to the identifier of the notification sent by the ITL so the ITL can track when the registry has completed the required cancellation.

### **6.1.2 Non-compliance Cancellation**

In the case that the Compliance Committee determines that a Party is in non-compliance with its emissions target under Article 3 of the Kyoto Protocol, the ITL will notify the Party of the quantity of units valid for the subsequent Commitment Period that it is required to cancel within 30 days as part of a non-compliance cancellation action. The notification will specify the Commitment Period for which the units must be cancelled. This quantity of units must be cancelled into the Non-compliance Cancellation Account (Account Type Code 220) for the commitment period. The registry will initiate cancellation transactions, providing reference to the identifier of the notification sent by the ITL so the ITL can track when the registry has completed the required cancellation.

### **6.1.3 Impending tCER or ICER Expiry**

The ITL will notify each national registry of the unit blocks of any tCERs held in retirement and tCER replacement accounts or ICERs held in a retirement account that are to expire within 30 days.. The notification indicates that the specified tCERs or ICERs are to be replaced before their expiry dates. The registry will initiate replacement transactions that reference the Notification ID sent by the ITL.

### **6.1.4 Reversal of Storage for CDM Project**

At the request of the CDM Executive Board in the case that a reversal of storage of greenhouse gases has occurred at a CDM Project, the ITL will temporarily suspend transfers of all ICERs generated by the Project (except to cancellation or replacement accounts). The ITL will then calculate how many units each registry must replace, on the basis of their holdings (excluding cancelled or previously replaced units) of the affected ICERs, and notify each affected registry of the requirement to replace this quantity of ICERs within 30 days.

The registry will then initiate replacement or cancellation transactions, providing reference to the identifier of the notification sent by the ITL so the ITL can track when the registry has completed the required replacement or cancellation. Units being cancelled must be cancelled into the Mandatory Cancellation Account (Account Type Code 250). Once the required replacement or cancellation has been completed, the ITL will restore the eligibility of the ICERs to be transferred.

### **6.1.5 Non-submission of Certification Report for CDM Project**

At the request of the CDM Executive Board in the case that the participants in a CDM Project have not submitted a certification report for the Project, the ITL will make ICERs generated by the Project ineligible for transfer (except to replacement and cancellation accounts). The ITL will notify each affected registry that these ICERs must be replaced or cancelled within 30 days. The registry will then initiate replacement or cancellation transactions, providing reference to the identifier of the notification sent by the ITL so the ITL can track when the registry has completed the required replacement or cancellation. Units being cancelled must be cancelled into the Mandatory Cancellation Account (Account Type Code 250).

### **6.1.6 Excess Issuance for CDM Project**

In the case that the CDM Executive Board requires a designated operational entity (DOE) to transfer units to a cancellation account, within 30 days, as a result of excess CERs, tCERs or ICERs having been issued for a CDM Project, it shall inform the DOE of this requirement and provide it with a Notification ID. The ITL will notify registries of the required cancellation to be undertaken by the DOE, using the same Notification ID provided to the DOE by the CDM Executive Board. The units must be cancelled into the Excess Issuance Cancellation Account (Account Type Code 240) at the CDM Registry. The entity will then initiate transactions, via registries, providing reference to the Notification ID so the ITL can track when the required cancellation has been completed.

### **6.1.7 Commitment Period Reserve**

Where an upward revision of a Party's CPR level raises it above the registry's current holdings of units, or where the cancellation or replacement of units within the registry reduces unit holdings below the CPR level, the ITL will notify the Party of the quantity of units by which it is required to increase its unit holdings within 30 days. The relevant Commitment Period will be specified in the notification. The registry will acquire sufficient units, with the relevant Applicable Commitment Period Identifier, from other registries to meet this requirement. Since transactions are submitted by the transferring, not acquiring, registry, these transactions will not reference any Notification ID. The ITL will check each registry's unit holdings against its CPR level on a periodic basis and send the notification when the CPR level is not maintained.

### **6.1.8 Unit Carry-over**

After the end of the true-up period and after the Compliance Committee has completed its consideration of all information reviewed under Article 8 of the Kyoto Protocol, the ITL notifies each registry of:

- The number of units which the registry may carry-over within 30 days, and the Commitment Period to which they may be carried-over.

The registry will then initiate carry-over transactions, up to the limit specified in the notification, within 30 days. For all carry-over transactions, the transaction must contain the Notification ID.

### **6.1.9 Expiry Date Change**

The ITL will notify registries when an expiry date change is required for tCERs or ICERs. The notification will be sent to registries holding any affected units. In the case of tCERs, the notification will reference the Original Commitment Period Identifier of the affected units. In the case of ICERs, the notification will reference the Project Number associated with these units. In both cases, the notification will include the new expiry date. The registry will initiate expiry date change transactions within 30 days, providing reference to the notification identifier sent by the ITL so the ITL can track when the registry has completed the required transaction.

### **6.1.10 Notification Update**

After an initial notification has been sent, the ITL may send an additional notification when a registry fulfills its obligation or to update a registry's progress towards meeting the requirement. The Notification ID for this message will be the same as the original, but the notification type will indicate that this is a notification update. The message content will contain remarks that reference the original notification, indicate whether or not the requirement has been met, indicate the number of days left to fulfill the requirement, and update the number of units the registry must address. The notification update is provided for informational purposes only, and the registry does not need to respond to it.

### 6.1.11 EU15 Commitment Period Reserve

Where an upward revision of the EU15 CPR level raises it above the EU15's current holdings of units, or where the cancellation or replacement of units within an EU15 registry reduces unit holdings below the EU15 CPR level, the ITL will notify all the EU15 registries of the quantity of units by which it is required to increase the EU15 unit holdings within 30 days. The relevant Commitment Period will be specified in the notification. On or more of the EU15 registries will acquire sufficient units, with the relevant Applicable Commitment Period Identifier, from other registries to meet this requirement. Since transactions are submitted by the transferring, not the acquiring, registry, these transactions will not reference any Notification ID. The ITL will check each EU15 registry's unit holdings against the EU15 CPR level on a periodic basis and send the notification when the EU15 CPR level is not maintained.

The AcceptITLNotice Web service defines the complete set of data elements available for notifications. Not all data elements are provided for each notification type, and they are shown as optional in the ITLNoticeRequest. Figure 6.3 specifies which of these optional data elements are provided by the ITL for notifications of each of the notification types described above. Optional ITLNoticeRequest data elements not shown for the notification type in this table are not provided.

**Table 6.1: ITL Notification Data Elements**

Notification Type	Notification Data Element	Notification Data Element Requirement
Net Source Cancellation (Type 1)	targetValue	Required. Specifies the quantity of units that must be cancelled.
	LULUCFActivity	Required. Specifies the LULUCF activity resulting in a net source of emissions.
	commitPeriod	Required. Specifies the Commitment Period of units to be cancelled.
	actionDueDate	Required. Specifies the date by which the units must be cancelled.
Non-compliance Cancellation (Type 2)	targetValue	Required. Specifies the quantity of units that must be cancelled.
	commitPeriod	Required. Specifies the Commitment Period for which the units must be cancelled.
	actionDueDate	Required. Specifies the date by which the units must be cancelled.
Impending tCER or ICER Expiry (Type 3)	actionDueDate	Required. Specifies the date by which the units must be replaced.
	unitBlockIdentifiers	Required. Specifies the unit blocks of any tCERs held in retirement and tCER replacement accounts or ICERs held in a retirement account that are to expire within 30 days.
Reversal of Storage for CDM Project (Type 4)	projectNumber	Required. Specifies the Project number of the Project that has experienced the reversal of storage.

Notification Type	Notification Data Element	Notification Data Element Requirement
	targetValue	Required. Specifies the quantity of ICERs that must be cancelled or replaced.
	actionDueDate	Required. Specifies the date by which the units must be cancelled or replaced.
Non-submission of Certification Report for CDM Project (Type 5)	projectNumber	Required. Specifies the Project number of the Project for which the certification report has not been submitted.
	targetValue	Required. Specifies the total number of units from the specified Project that must be cancelled or replaced.
	actionDueDate	Required. Specifies the date by which the units must be cancelled or replaced.
Excess Issuance for CDM Project (Type 6)	projectNumber	Required. Specifies the Project number of the Project for which there has been excess issuance.
	unitType	Required. Specifies the type of units issued for the affected Project.
	targetValue	Required. Specifies the total number of units that must be cancelled to the Excess Issuance Cancellation Account (Account Type 240) at the CDM Registry.
	actionDueDate	Required. Specifies the date by which the units must be cancelled.
Commitment Period Reserve (Type 7)	targetValue	Required. Specifies the quantity of units which must be acquired to satisfy the Commitment Period Reserve requirement.
	commitPeriod	Required. Specifies the relevant Commitment Period.
	actionDueDate	Required. Specifies the date by which the registry must increase its unit holdings by the specified amount.
Unit Carry-over (Type 8)	targetValue	Required. Specifies the quantity of units which may be carried over the subsequent commitment period.
	commitPeriod	Required. Specifies the Commitment Period to which the indicated number of units may be carried-over.
	actionDueDate	Required. Specifies the date by which units must be carried over.
Expiry Date Change (Type 9)	projectNumber	Required for ICERs. Not provided for tCERs. (All tCERs for a given Commitment Period must have the same expiry date.)
	unitType	Required. Must be tCERs (6) or ICERs (7).

Notification Type	Notification Data Element	Notification Data Element Requirement
	targetValue	Required. Specifies the number of units which must have their expiry date changed.
	targetDate	Required. Specifies the new expiry date for the units.
	commitPeriod	Required for tCERs. Not provided for ICERs. Specifies the Original Commitment Period Identifier of the affected units.
	actionDueDate	Required. Specifies the date by which affected units must have their expiry date changed.
Notification Update (Type 10)	projectNumber	Matching the original notification.
	unitType	Matching the original notification.
	targetValue	Matching the original notification. Specifies the number of units that remain to be acted upon. This value will be 0 if the notification requirement has been fully met.
	targetDate	Matching the original notification.
	LULUCFActivity	Matching the original notification.
	commitPeriod	Matching the original notification.
	actionDueDate	Matching the original notification.
EU15 Commitment Period Reserve (Type 11)	targetValue .	Required. Specifies the quantity of units which must be acquired to satisfy the Commitment Period Reserve requirement
	commitPeriod	Required. Specifies the relevant Commitment Period.
	actionDueDate	Required. Specifies the date by which the EU15 registries must increase their total unit holdings by the specified amount

Implementation note: The only transaction types that can reduce commitment period holdings below the CPR or EU15 CPR are: External Transfer (3), Cancellation (4) and Replacement (6). The CPR and EU15 CPR checks prevent external transfers from doing so. The cancellation and replacement transactions can do so, but trigger a CPR or EU15 CPR notification immediately. This notification can also be generated by a periodic background process and also some external events, such as changes to the CPR.

## General Messages

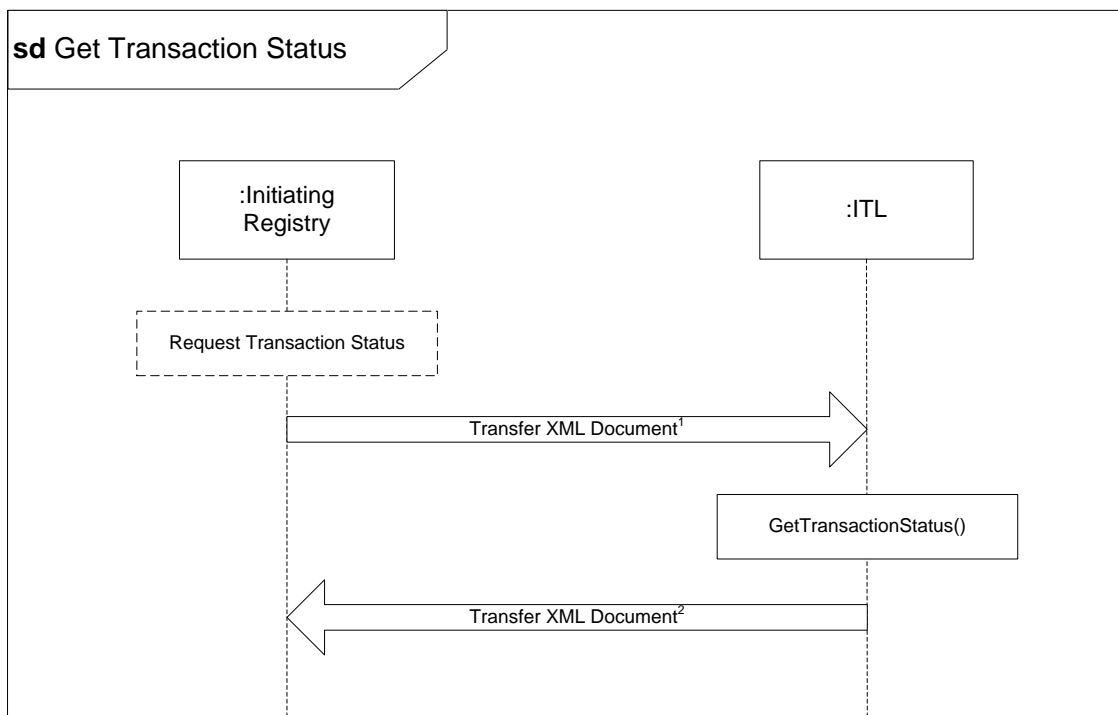
The AcceptMessage Web service at a registry may be used to deliver general messages to the Registry Administrator. These messages could involve planned ITL maintenance periods, change management, time synchronization problems, or other operational issues and plans. This communication channel offers a secure alternative to email communication.

## Transaction Status Service

The ITL provides a public Web service to return the current status of a transaction at the ITL. This service may be used by registries to query the status of a transaction for which verification has not yet been received.

Registries may call the GetTransactionStatus Web service method on the ITL with a specified transaction number, and the most recent transaction status will be returned immediately to the registry. If the transaction ID requested is not found, the function will return failure along with response code 8001 (Transaction Not Found).

Figure 6.3: Get Transaction Status Diagram



1. The Initiating Registry prepares and sends a request to the GetTransactionStatus function on the Transaction Log.
2. The GetTransactionStatus prepares and sends a response to the calling function on the Initiating Registry with the status of the specified transaction.

## Time Synchronization

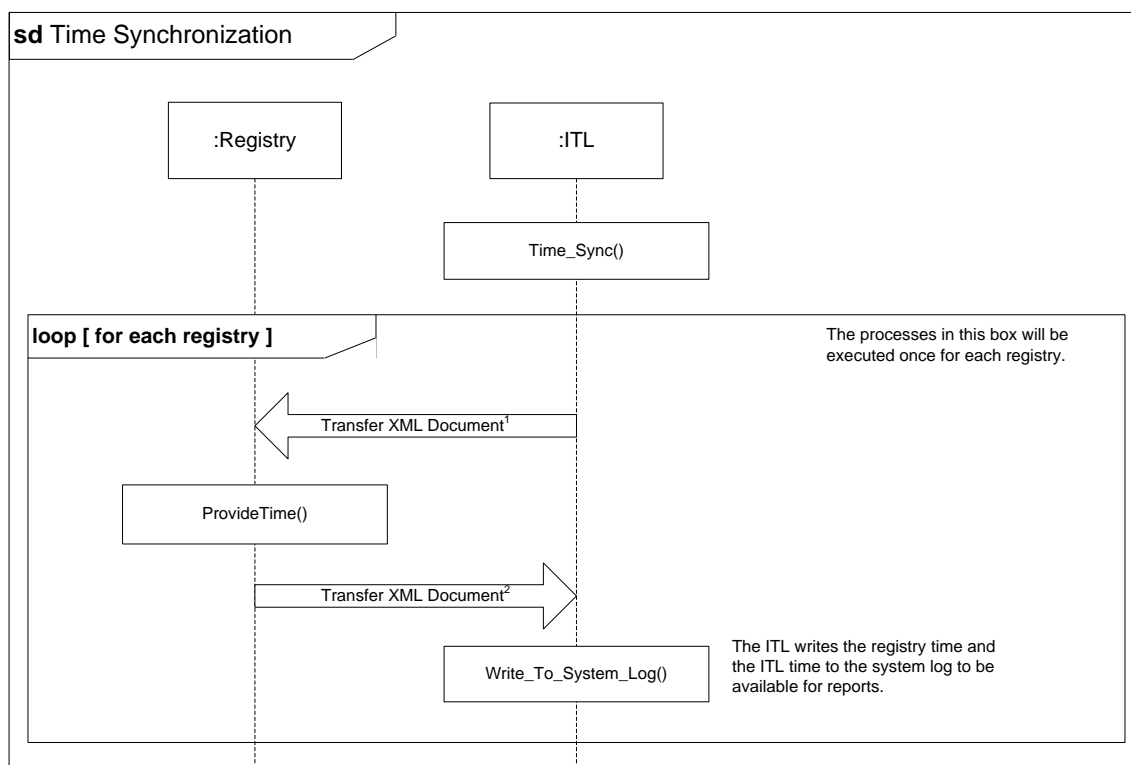
In order to maintain consistent system time between the registries and the ITL, the ITL checks the system time of each registry on a periodic basis. If the time is found to be unsynchronized by more than a specified amount, a message is sent to the Registry Administrator of that registry. In order to accommodate this function, each registry must make available a ProvideTime function which is used by the ITL to retrieve the current time of the registry.

Registries must implement the ProvideTime public Web service method for the ITL to call. The ITL will compare the time this function returns with the official system time. Detailed specifications for the ProvideTime method are in Annex D.

The ITL will log the time synchronization result and contact the Registry Administrator using a manual process or through a general message if a time problem is identified.



**Figure 6.4: Time Synchronization Diagram**



1. The function Time\_Sync prepares and sends a request to the ProvideTime Web service on the registry.
2. The ProvideTime function returns a response to the ITL with the current system time on the registry.

## Heartbeat connection health monitoring by STLs

Due to the criticality of connections between STLs and the ITL, STLs may monitor the health of their connection to the ITL by using specially formatted *acceptMessage* web service requests.

When a STL sends an *acceptMessage* request with the format  
 'HEARTBEATREQ' + *R*

The ITL will respond with an *acceptMessage* request with the format  
 'HEARTBEATREP' + *R* + '#' + *dateTime\_received* + '#'

Where

*R* is a random number generated by the STL,  
 with a maximum value of 1,000,000,000,000,000

*dateTime\_received* is the date and time the ITL received the message from the STL  
 in the format 'dd/mm/yyyy hh:mm:ss'

The frequency of the heartbeat requests is agreed between the STL and the ITL (a frequency of 5 minutes is recommended).

The time the ITL has to respond to requests before an "unhealthy" connection is suspected is also agreed between the STL and the ITL (a value of 15 minutes is recommended).

## 7 Data Logging Specifications

To support the need for the Transaction Log and registries to maintain accurate and consistent information, and to provide tools for use in the reconciliation process to resolve inconsistencies, five types of data logs shall be maintained by the registries and the ITL:

- A transaction log (including both transaction summary and detailed unit holdings);
- A reconciliation history log;
- A notification log;
- An internal audit log; and
- A message archive.

These logs are required to support auditing functionality, both internal and external. The reconciliation process constitutes one type of external audit of a registry.

All data in these data logs shall be maintained until, at minimum, the end of the third Commitment Period after the applicable Commitment Period of the associated units. In the case of ICERs, all related data shall be maintained until, at minimum, the end of the second Commitment Period after the latest crediting period of the associated units. Data older than one year may be archived to a secure location outside of the registry or Transaction Log, as long as it can be retrieved or accessed within a 48 hour period should an inconsistency or question arise.

General messages, such as those received through the Accept Message Web service, should be maintained by the registry, but no specific logs are required or recommended.

### Transaction Log

The Transaction Log contains a record of each proposed transaction sent to the ITL. Each record contains a summary of the transaction content and the subsequent outcome of the transaction. Registries will be required to provide Transaction Log data to the ITL involving specific units if an inconsistency is found for specific units during a reconciliation process.

In general, it is recommended that the logging of a transaction message sent to the ITL occur after the receipt of a SOAP response indicating that the message was successfully transmitted and received.

The information in Figure 7.1 shall be maintained in the Transaction Log. A specific data model for these data is not required.

**Table 7.1: Transaction Log Attributes**

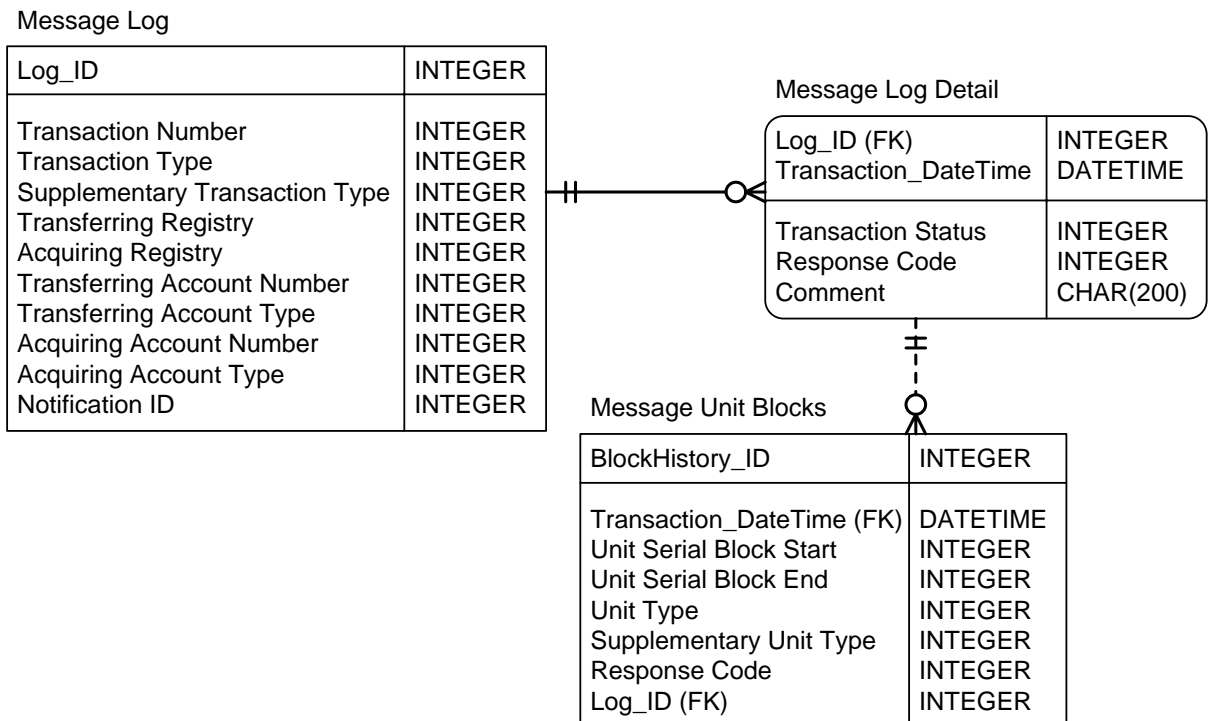
Attribute	Notes
Transaction Identifier	
Transaction Type	
Supplementary Transaction Type	Required for registries subject to a supplementary program. This would be null for non-EU registries.
Transferring Account Type	
Transferring Account	
Acquiring Registry Identifier	
Acquiring Account	
Acquiring Account Type	

Attribute	Notes
Notification ID	
Transaction Status	Contained in child table.
Transaction Status DateTime	Contained in child table.
Unit Block(s)	Contained in child table.
Response Code(s)	Contained in child table along with unit block data.

To support this information, three tables in parent-child relationships are necessary:

- A parent table containing the transaction identifier, related attributes and status information;
- A child table identifying the various statuses a transaction may be processed through; and
- A second child table identifying serial blocks and response code results. The diagram in Figure 7.2 below contains an example entity-relationship model of these tables.

**Figure 7.1: Transaction Log Entity Relationship Diagram**



## Reconciliation History Log

The Reconciliation Log contains a record of each reconciliation action conducted by the ITL for a registry. As described in Section 5, each Reconciliation action consists of multiple steps or sub-processes. This Reconciliation Log contains one or more records for each step in a Reconciliation action.

The Reconciliation process is initiated and driven by messages from the ITL to a registry. The registry shall log each request and its response in its Reconciliation Log. The ITL shall maintain a parallel Reconciliation Log containing all requests, responses received, and results sent to a registry. Although information in the Reconciliation Log are not shared directly as part of the Reconciliation itself, access to this information by the Registry Administrator may be necessary to identify the manual intervention needed in order to resolve inconsistencies.

In general, it is recommended that the logging of a transaction message sent to the ITL occur after the receipt of a SOAP response indicating that the message was successfully transmitted and received.

The information in Figure 7.3 shall be maintained in the Reconciliation Log. A specific data model for these data is not required.

**Table 7.2: Reconciliation History Log Attributes**

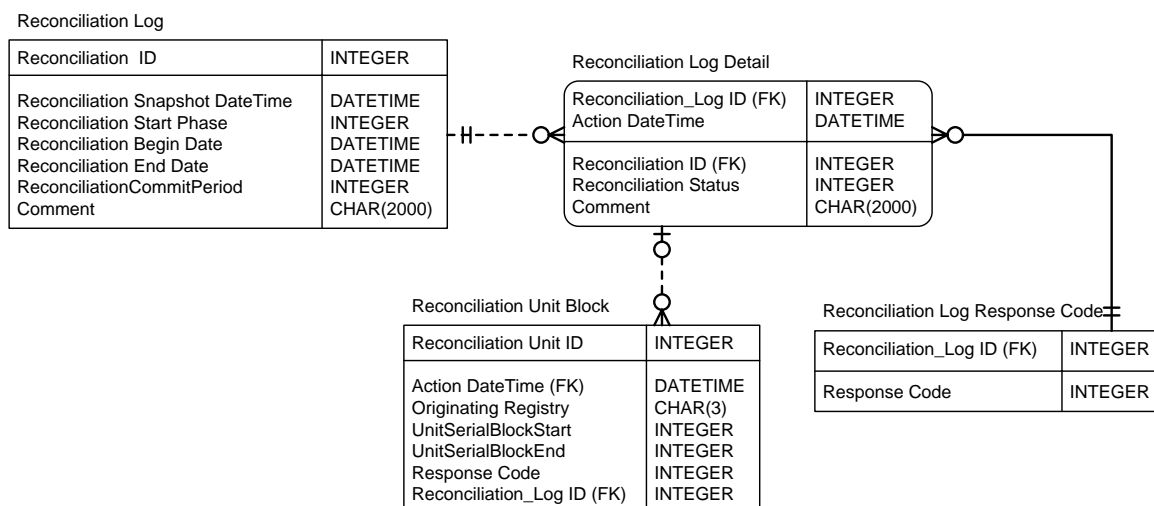
Attribute	Notes
Reconciliation ID	Unique identifier for a reconciliation action as requested by the ITL
Reconciliation Begin Date	
Reconciliation End Date	
Reconciliation Snapshot DateTime	
Reconciliation Start Phase	
Response Code	Contained in child table.
Unit Blocks	Contained in child table.
Reconciliation Comment	Information recorded by the Registry Manager regarding corrective actions for manual intervention.
Reconciliation Status	Contained in child table.
Reconciliation Status Log DateTime	Contained in child table.

To support this information, three tables in parent-child relationships are necessary:

- A parent table containing the reconciliation number, date and time reconciliation was initiated, ended, phase requested, and the date and time the snapshot of current holdings was requested;
- A child table identifying changes in the status of the reconciliation action;
- A child table identifying the response codes returned by the ITL and associated with the change of a status; and
- A child table to the specific activity containing the unit blocks that were identified as inconsistent and by the ITL corresponding response code information.

The diagram in Figure 7.4 below contains an example entity-relationship model of this information.

**Figure 7.2: Reconciliation History Log Entity Relationship Diagram**



## Notification Log

Each registry and the ITL shall also maintain a log of notifications generated by the ITL and sent to a registry via the AcceptITLNotice Web service. These notifications inform the registry regarding specific actions that should be taken relating to units. See Section 6.

The Notification Log at the registry shall contain the attributes in Figure 7.5. A specific data model for these data is not required.

**Table 7.3: Notification Log Attributes**

Attribute	Notes
Notification ID	As generated by the ITL and sent to the registry.
Notification Status	
Notification Type	
Notification Date	
Total Units	If appropriate. For example, notifications relating to Reversal of Storage.
Project Number	If appropriate.
Notification Text or Message Location	To store a complete copy of the notification content.

## Internal Audit Log

Each registry and the ITL shall also maintain an internal log of changes to data which are critical to the transaction or reconciliation process. The scope and design of this functionality is the responsibility of the Registry Administrator. The internal audit log shall capture information on internal and external transactions, including in particular the user ID and date/time of all recorded transactions. Information contained in this log is not shared directly with the ITL. It is required to provide additional information for use by the Registry Administrator for manual intervention when an inconsistency is discovered in the reconciliation process.

The internal audit log shall contain the attributes in Figure 7.6. A specific data model for these data is not required.

**Table 7.4: Internal Log Attributes**

Attribute	Notes
Activity Type	For example, insert, delete, update, login attempt.
Activity DateTime	
Entity Affected	For example, table name.
Field Modified	Attribute in table that was updated.
Old Value	
New Value	
User ID	Person who executed change if not performed through Web service.
Source of Activity	Identifies the server or workstation activity was submitted on.

## Message Archive

Each registry and the ITL are required to store a copy of messages sent and received, in their entirety, as stand alone files. These files provide additional information for use by the registry or the ITL Administrator when an inconsistency is discovered which relates to a messaging problem which cannot be resolved through the use of the transaction history or internal audit logs.

The location and the medium for this storage are at the discretion of the registry or the ITL Administrator. The naming convention of the files must enable an authorized user to retrieve the file for a specific transaction or reconciliation. It is recommended that the files be stored in compressed formats using the following naming convention:

aa\_bb-#####-cc.zip

Where:

aa = Registry country code per ISO3166, and "ITL" for Transaction Log messages, and "CDM" for CDM Registry messages

bb-#####-## = Transaction identifier

and cc = sequential number generator

For example, a file sent from the ITL about a proposed German transaction would be named:

ITL\_DE\_152\_1.zip

## **8 Change Management Specifications**

### **Objectives**

The Technical Specifications for Data Exchange provide a stable, agreed-upon platform for the development and deployment of the communications modules built for registries and the ITL. It is expected, however, that changes to the messages and to the criteria to ensure data quality and accuracy may be necessary over time. It is a requirement of the Technical Specifications for Data Exchange that changes provide backward compatibility to the extent possible. It is the goal of these requirements to allow existing functionality to remain valid when new requirements or changes are necessary.

Anticipating these needs, this specification establishes a technical architecture that allows adjustments within the specification without imposing significant additional development costs to registries or the ITL. A change management process, supported by Registry Systems' Administrators, to determine when and how this will be managed, shall also be established.

### **Technical Specifications**

To minimize the impact of changes and to manage the process of ensuring that all participants implement required changes within a necessary timeframe, registries and the ITL must conform to the following technical specification:

#### **8.1.1 Version Definition**

The Technical Specifications shall be assigned a version number, managed through the ITL. A version number consists of two elements, a major version number and a minor version number. Major version numbers will change infrequently and reflect a fundamental change in the DES. It is likely that a major version change would require significant coding changes to be undertaken. A minor version number indicates small changes that can be implemented completely within the existing messages structure.

The ITL will maintain a list of the one or more versions that each ITL release supports. The ITL Administrator will inform Registry Systems administrators when particular versions of the DES will no longer be supported by future releases of the ITL.

Within each XML message sent to or received from the ITL, the major and minor version numbers are checked to ensure message compatibility between the ITL and Registry Systems. A registry that has an incompatible version number will have its requests rejected and receive a response indicating that the version of the Data Exchange Standards is out-of-date.

### **ITL Web Portal**

The ITL will provide an extranet website that will post information on version status as well as allow registered users to review upcoming functional changes, time for implementation, and the technical specifications for these changes. Users who have valid user accounts and passwords through the VPN will have access to this site. The site will maintain a history of all version changes and patches released from the site. It will be managed by the ITL Administrator.

#### **8.1.2 Web Service Modifications**

Changes to Web services or subsequent functions that require new parameters constitute a major version change. All registries will have a specified period of time to comply with the new requirements. Detailed specifications will be provided with sample testing procedures for the registries to test the new components against the ITL.

### **8.1.3 Support Table Content Modification**

Changes to data content in the form of new response codes or support tables are considered minor version changes. These data will be available in XML format for download from the ITL website. Included in these tables are codes identifying which response codes are new, have been modified, or have been retired. Registries must refresh their tables with current support table data as needed.



## 9 Registry Initialization Specifications

Initialization is the process of bringing a registry system on-line, allowing it to fully participate with the ITL in a trading scheme. Prior to a registry participating in message exchange with the ITL, the registry must comply with a series of initialization requirements and procedures. These tasks ensure that the registry meets both the functional and non-functional requirements of the Data Exchange Standards and will be able to converse consistently with the ITL. The registry will not be able to participate in any transactions until all initialization tasks are complete.

### Staff Identification and Planning

The first step to initialize a registry is to identify those individuals responsible for the registry and its operation, including the Registry Manager. The Registry Manager, or a person assigned by the Party, must submit a schedule and plan for initialization to the ITL Manager. The schedule and plan, should detail, in writing, the timing projected for each major initialization task, including projected start and end dates. This is necessary so that the ITL Managers can ensure that the appropriate level of support and assistance is available during this period. Since it is anticipated that multiple registries will be in the process of testing and initialization simultaneously, the ITL Manager will assign a primary staff person to work with each registry during this period. It will be important for the ITL and registry staff to develop a working relationship and excellent communication.

Although no specific format is required, it is recommended that the schedule and plan address the following areas, which comprise an initialization checklist:

#### Registry Checklist

- Assign Registry Manager and support staff.
- Define initialization schedule with projected milestone completion dates. Initialization should be completed within a 2-month time period.
- Submit Registry Documentation to ITL Manager.
- Enable VPN access to and from ITL.
- Obtain and test registry digital signature.
- Perform tests and assess results received back from the ITL Manager.
- Set up production environment.
- Verify time synchronization with ITL.
- Provide government account information for Cancellation, Replacement, and Retirement Accounts for the first Commitment Period.

#### ITL Checklist

- Assign ITL Manager and primary staff to work with Registry Manager.
- Review registry schedule and set up ITL schedule.
- Review registry documentation.
- Enable VPN access to and from registry.
- Set up digital signature and participate in security and authentication tests.
- Assist with registry tests.
- Participate in tests and prepare analysis of test results.
- Receive account and contact data from registry.
- Set up production data for registry.
- Process and quality assure government account information.

## Documentation

The first task of a registry is to provide documentation of their registry system. This documentation is needed to show that non-functional requirements of the DES are met and that the registry will be operated in a manner consistent with excellent operating practices. These requirements ensure the national registry has an adequate plan for addressing operational and security requirements of the application.

### 9.1.1 Database and Application Backup

A database and application backup plan should be submitted outlining a detailed backup plan for the production database and software. It is recommended that database backups be performed at a minimum frequency of daily.

Specific elements of the plan include:

- Identification of personnel responsible for backup (include a primary individual and an alternate, or a staffing plan);
- Identification of specific backup schedule and procedures (i.e., backup at 7 p.m. each evening from terminal X by User ID Y);
- Identification of backup media and its location;
- Identification of the number of backup generations planned;
- Definition of strategy to monitor performance of backup tasks, including notification of backup failures, log review, spot checks, audit, management reporting, etc.;
- Identification of scope or content of backup procedures (i.e., database, application software, server logs, etc.); and
- Identification of backup hardware and software.

### 9.1.2 Disaster Recovery Plan

A disaster recovery plan designed to ensure business continuity in the event of catastrophic failure or disruption of the host environment should be submitted in conjunction with the backup procedures. The primary objective of a disaster recovery plan is to enable an organization to survive a disaster and to reestablish normal business operations as quickly as possible. In order to survive a catastrophic event, the organization must assure that critical operations can resume normal processing within a reasonable time frame. A contingency plan should be laid out in the event that the primary facility cannot perform required daily operations. In order for this plan to be effective, periodic testing and evaluation should be performed to ensure validity and viability.

Specific elements of the plan include:

- Identification of an off-site facility with adequate disk space/storage and availability to serve as an emergency hosting environment;
- Definition of specific minimum hardware and software requirements to host the registry on a temporary basis;
- Definition of roles and responsibilities for primary and alternate personnel at the off-site location;
- Definition of roll-back procedures to step back to the latest backup. This may include obtaining daily transactions from the ITL that were not included in the last backup;

- Notify all appropriate Parties that a contingency plan is in effect (i.e., ITL, other registries or users);
- Identification of off-site location of documentation and procedure manuals, as well as any paper-based forms, necessary to deploy under a Disaster Recovery scenario;
- Definition of periodic testing strategy to demonstrate readiness to implement disaster recovery plan; and
- Definition of expectation for time frame in which registry could begin operation following a disaster. The time frame would depend on the volume of transactions, cost and other factors and is not expected to be the same for each registry.

### 9.1.3 Security Plan

A security plan is defined in order to protect the application and data from unrestricted and unsolicited use. Secure access to the data should be provided at multiple intervals to insure redundancy of protection. For Web security, the following three primary areas should be addressed:

1. **Server security.** The Web and/or database server should be secured not only by user ID and password but also physically to prevent unauthorized access to the data and application. As with most dynamic connectors to databases, a connection with full access must be granted to the Web server because various queries will need to access different tables or views to construct the HTML from the query. To prevent unauthorized use of these open data connections, the servers should be physically secured. In addition, security can be assigned at the table level on a database.
2. **User-authentication security.** This level of security insures no unauthorized access to information in the registry. This is accomplished by requiring unique user IDs and passwords that are regularly maintained by a Systems Administrator.
3. **Session security.** This level of security insures that data are not intercepted as they are broadcast over the Internet. This is accomplished by encrypting data passed to and from the registry.

Specific elements of the plan include:

- Definition of rules and responsibilities for security, recognizing that actions by persons are the most significant contributing factor to the success or failure of security planning;
- Determine physical access to the Web and/or Database server;
- Assign a network and database administrator and alternates, user ID, passwords, and specific responsibilities;
- Activate audit trails recording activities at the server, database and data levels;
- Employ encryption of data transferred to and from the registry;
- Require frequent changes to password, restricting replication over a period of time;
- Require passwords of specific length with a specific number of alpha and numeric characters. For example, 6 digit passwords with a minimum of 2 numbers; and
- Delete all unused User ids and passwords immediately and remove inactive user IDs from the database on a regular basis.

### 9.1.4 Application Logging Documentation

To demonstrate conformance with Section 7 of the Data Exchange Standards, the Registry Manager is asked to provide a summary of the registry capability to maintain database logs and activity logs.

- Database Logging. Database administrators are required to implement transaction logging where logs for files can be periodically shipped to a remote server or alternate site. For Oracle databases, this is the equivalent of archive logging; for MS SQL Server this might be implemented with log shipping.
- Activity logging. Activity logging should be utilized to track unauthorized attempts to log on to the server as well as general usage.

The documentation should include:

- Definition of regular backup and archival of transaction logs;
- Definition of hardware utilized to store logs; and
- Assignment of personnel to review activity logs on a regular basis.

#### **9.1.5 Time Validation Plan**

For the successful data exchange, a registry must define and follow specific procedures to validate server time on a periodic basis.

The plan should include:

- Schedule for periodic time validation;
- Identification of time server to provide validation;
- Assignment of personnel to perform or monitor time validation;
- Maintenance of documentation of time validations and any time adjustments resulting;
- Definition of tolerance for time validation discrepancies; and
- Definition of process for adjusting time.

#### **9.1.6 Version Change Management**

A clear migration path should exist to upgrade from version to version of registry software and database schemas. When a new version is released it must go through the testing sequence to insure that it is operable. This invokes preparing a testing environment and a test plan and a migration path to move the code and database schema to production assuming it has passed the testing sequence.

The Change Management Plan should include:

- Deployment Strategy;
- Test Plan;
- Notification strategy; and
- Data management/loading plan.

#### **9.1.7 Test Plan and Test Report**

The test plan ensures that a registry has performed basic testing and is capable of participating in the tests outlined in Annex H which are required of a registry prior to being authorized to submit production transactions to the ITL. The test plan describes the various levels and types of testing that will be done throughout development.

A test plan should be outlined that steps through the basic system tasks to ensure no changes made in the test environment will affect day-to-day processing. This should cover System Administrator functionality, as well as all user-level testing. All test cases should be documented and archived for proof of concept and documentation purposes. A migration plan should be clearly outlined to move the test code, schema, or data to the production environment with minimal impact to the overall system (choose times of least traffic).

The test plan should include:

- Description of overall test strategy, testing procedures and documentation;
- Identification of Test criteria;
- Identification of Testing tools;
- Assignment of personnel to perform testing of the software, both on the initial release and for an upgrade in hardware or software;
- Description of test environment and management of that environment to ensure that results replicate the results expected in a production environment;
- Evidence that the plan provides for systematic testing in logical order of all module; subsystem, and system requirements against a well-defined set of test cases;
- A method for documenting the performance of all tests in a test log; a method for identifying and reporting any anomalies or errors; and a procedure for tracking problems from detection to resolution;
- Plans for creating the test environment, including all needed software and hardware purchases, are consistent with the application development schedule; and
- Evidence that regression testing is a fundamental element of the plan.

### **9.1.8 Operational Plan**

The operational plan ensures that the registry has appropriately planned and staffed the operational requirements of the registry, so that the ITL Administrator can foresee that the registry will continue to maintain effective operation once the registry has been approved to operate in a production mode with the ITL. The operational plan will address many of the requirements for the initial approval, but will provide a demonstration that the initial standards and requirements will be addressed on an ongoing basis.

The operational plan should include:

- Definition of operational logs and record keeping;
- Staffing and management; including training;
- Security management plan;
- Ongoing performance evaluations and assessments;
- Data management strategy, including archiving and data quality assessment;
- Modernization and technology assessment strategy; and
- Technical support plan.

### **Initialization Tests**

Once the ITL Administrator is satisfied these requirements have been met, the national registry can begin to establish electronic communication with the ITL. These procedures are performed in stages and are described below. Detailed requirements and processes for these tests are defined in Annex H.

**Table 9.1: Table of Initialization Tests**

Test	Test Type	Who Initiates the Test	Description of Test
Communication Initialization	Required	Registry and ITL	Installation, initialization and test of the VPN.
ITL Extranet Login	Required	Registry	Creation and verification of ITL extranet account and password.
Registry Transaction Web Services	Required	Registry	Registry tests ITL Web services.
ITL Transaction Web Services	Required	ITL	ITL tests the registry Web services.
Query Services	Recommended	Registry	Test of querying capabilities.
Registry Reconciliation Web Services	Required	ITL	Web service test from ITL requesting reconciliation data.
ITL Reconciliation Web Services	Required	Registry	Web service test in which registry submits reconciliation data.
Data Request	Required	ITL	Web service initiated by the ITL requesting data from a registry.
Data Identifier Initialization	Recommended	Registry	Download and import of lookup table data from ITL extranet website.

## Communication Initialization

A registry must be able to demonstrate that a secure communication channel can be established to and from the ITL Communications Hub. The Registry Manager may elect for the ITL Manager to remotely administer the VPN. This requires the ITL Manager to assist or direct the installation and configuration of the VPN at the registry network. The test to validate connectivity can be performed at the time of installation. If the Registry Manager elects to install and configure the VPN, then an appointment shall be negotiated with the ITL Manager to test VPN connectivity. The test must demonstrate that the ITL and registry are able to connect to and send transmissions to and from each other. The IP address for both the registry VPN and ITL VPN shall be recorded and documented as valid and trusted connections to each other. This test shall be conducted and completed within a single business week. The ITL Manager shall notify the Registry Manager whether the test result was accepted or rejected as incomplete. The results of the Communication Initialization Test are recorded in the ITL database.

The tests conducted are as follows:

- Registry must test for Internet access;
- Registry records IP address of ITL for site-to-site configuration of the VPN. ITL will supply the configuration specifications for the VPN;
- Registry pings ITL IP address to validate VPN hardware can see ITL VPN;
- ITL records IP address of registry;
- Registry acquires digital certificate from Third Party Certificate Authority and install appropriate files;

- ITL is sent public key of certificate either from Certificate Authority or from the registry; and
- Authentication test of Digital Certificate is initiated by the sending and receiving of public keys.

## **Access to ITL Website**

There are two websites that support distribution of data from the ITL database, one which is public and one which requires security to access.

### **9.1.9 Public ITL Website**

Access to the ITL public website for the purposes of querying unit transparent data does not require an account or password. The Registry Manager is responsible for checking the site to review content and alert the ITL Manager of any discrepancies in data. This test is for the benefit of the registry and does not require confirmation from the ITL. This test can be performed at any time.

### **9.1.10 Access to ITL Extranet**

The ITL extranet hosts information regarding change management files or patches as well as XML datasets of all response codes and key identifier tables. Access to this site requires a login and password. The Registry Manager must request an account and password for access to this site. This test is for the benefit of the registry and does not require confirmation from the ITL. This test can be performed at any time.

## **Web Services Testing**

The ITL Manager will host both a test and production environment for registries to test Web services independent of production data. Testing of Web services includes registries' testing Web services against the ITL and the ITL Manager testing the registries' Web services.

All registries must first test all Web services through the ITL test environment. These tests shall confirm that all Web services have met functional specifications. The Registry Manager shall negotiate a timeframe for conducting these tests with the ITL Manager. The ITL Manager shall notify the Registry Manager as to whether the test results were accepted or rejected as incomplete. The results of the Web Services Test are recorded in the ITL database. These tests shall include:

- Registry Transaction Web Service Tests;
- ITL Transaction Web Service Tests;
- Query Web Service Tests;
- Registry Reconciliation Web Services; and
- ITL Reconciliation Web Services.

## **Request for Other Data**

The ITL requires that a registry provide information to the ITL for possible distribution to the various ITL websites. These data include information regarding account and representative information as well as general queries for time synchronization. The ITL Manager shall negotiate a time frame for performing these requests with a Registry Manager. These tests shall be conducted and completed within a single business day. The ITL Manager shall notify the Registry Manager regarding whether the test results were accepted or rejected as incomplete. The results of the Data Request Test are recorded in the ITL database. The following Web services will be tested for the submission of data. For additional information on the content and methodology for Web service testing, see Annex H.

## Data Identifier Initialization

Registries are expected to load all response code data as well as data for all key identifier lookup tables into their systems. The following table identifies the data that the registry needs to download from the ITL extranet website for integration into their systems. This initialization is for the benefit of the registry and does not require confirmation from the ITL. This data initialization can be performed at any time. The datasets are listed below.

**Table 9.2: Look-up Table Initialization**

Data Set	Description
Account Type Code	Account Descriptions
Registry Code	ISO-1066 Country Codes and Identifiers for Registries
Unit Type Code	Identifies type of unit
Supplementary Unit Type Code	Identifies additional unit type codes for an STL
Transaction Type Code	Identifies the type of transaction
Transaction Status Code	Identifies the status of a transaction
Response Catalog	Lists of all possible response code and descriptors
Supplementary Transaction Code	Identifies additional Transaction Type codes for an STL
Notification Type Code	Identifies types of notifications

## Full System Test

Once each individual component of the registry-ITL communication system has been tested, the registry should initiate a series of transactions and allow them to continue to completion without manual intervention. These transactions will be documented in the test plan and will be representative of the different types of communications between the registry and the ITL. Once each transaction has ended with a predictable result, the registry is certified to use the production environment.

## Reconciliation Services and Schedule

Initially, requests for reconciliation data shall be requested daily and after a timeframe in which no errors have been reported over a sample period of time involving numerous transactions the ITL Manager and Registry Manager may negotiate a less frequent time frame for reconciliation requests. The Registry Manager may also negotiate the time of day in which it does not present an undo burden to provide reconciliation data requests to the ITL. Failure to provide reconciliation data when requested can cause the suspension of transaction privileges.

## Government Account Information

Prior to receiving authorization to operate in production mode with the ITL, the registry will provide to the ITL information about the account identifiers and account types for all government accounts used by the registry for holding units for cancellation, retirement and replacement purposes. The registry will provide this information to the ITL Administrator in a data file in the comma delimited format as defined in Figure 9.3.



**Table 9.3: Government Account File Specifications**

<b>Column #</b>	<b>Data Element</b>	<b>Data Attributes</b>	<b>Description</b>
1	Registry Code	Alphanumeric (3)	Per Annex F
2	Account Identifier	Numeric	Per Annex F
3	Commitment Period	Numeric	Per Annex G
4	Account Type Code	Numeric	Per Annex G
5	Account Status	Numeric	1 = Open 2 = Closed
6	Action	Numeric	1 = Add 2 = Update 3 = Delete (or archive)

This format will be used on an ongoing basis to update these records, which are used to verify that the Commitment Period and account type are accurate when transactions are submitted to cancel, retire or replace units.

For registries which will initialize for the ITL after first initializing with the STL, the STL will provide the required government account data to the ITL for this purpose.

*[This page intentionally left blank.]*

---

**DATA EXCHANGE STANDARDS FOR  
REGISTRY SYSTEMS UNDER THE KYOTO PROTOCOL  
TECHNICAL SPECIFICATIONS (Version 1.1.10)**

**ANNEXES**

**24 April 2013**

*[This page intentionally left blank.]*

## Table of Contents

<b>Annex A: Glossary of Terms</b> .....	A-1
<b>Annex B: Web Services and Functions for Transaction Processing</b> .....	B-1
1. Introduction.....	B-1
2. Objects and Structures.....	B-1
2.1 Result Identifier.....	B-1
2.2 Transaction Object.....	B-1
2.3 TransactionUnitBlock Object .....	B-2
2.4 UnitBlock_Identity Object.....	B-2
2.5 Check Response Object.....	B-2
2.6 Evaluation Result Object .....	B-2
3. Specified Functions .....	B-3
<b>Annex C: Web Services and Functions for Reconciliation</b> .....	C-1
1. Introduction.....	C-1
2. Objects and Structures.....	C-1
2.1 Reconciliation Object .....	C-1
2.2 Totals Object.....	C-1
3. Functions and Web Services .....	C-2
<b>Annex D: Web Services for Administrative Processes</b> .....	D-1
1. Introduction.....	D-1
2. Objects and Structures.....	D-1
<b>Annex E: List of Checks and Response Codes for Message Processing</b> .....	E-1
1. Purpose .....	E-1
2. Summary of Response Specifications .....	E-1
3. Message Validity Response Codes .....	E-4
3. System Status Checks .....	E-6
4. ITL Transaction Response Codes.....	E-8
5. Acquiring Registry Transaction Response Codes .....	E-42
6. ITL Reconciliation Response Codes .....	E-42
7. CITL Checks and Messages .....	E-43
8. ITL Administrative Response Codes.....	E-43
9. Obsolete Response Codes .....	E-44
<b>Annex F: Definition of Identifiers</b> .....	F-1
1. Introduction.....	F-1
2. General Rules for XML Formats .....	F-1
3. Recommended Display and Report Formats.....	F-1
4. Serial Numbers.....	F-1
5. Account Numbers.....	F-3
6. Transaction Numbers .....	F-3
7. Reconciliation Numbers .....	F-4
8. Project Numbers.....	F-4
9. Notification Numbers.....	F-5
<b>Annex G: List of Codes</b> .....	G-1

<b>Annex I: Messaging Service Specifications</b> .....	I-1
1. Introduction.....	I-1
2. SOAP Message Exchange.....	I-1
3. Standards and Requirements .....	I-1
3.1 Versioning.....	I-1
3.2 Encoding Styles .....	I-1
3.3 SOAP Interoperability .....	I-1
3.4 SOAP-Specific Port .....	I-2
4. SOAP Message Specifications .....	I-2
4.1 SOAP Envelope Element.....	I-2
4.2 The Header Element.....	I-2
4.3 The Body Element .....	I-2
4.3.1 Namespaces .....	I-2
5. WSDL Requirements.....	I-2
 <b>Annex K: Description Language (WSDL) Documentation</b> .....	 K-1
1. Introduction.....	K-1
2. Summaries of Web Service Operations .....	K-2
2.1 AcceptMessage .....	K-2
2.2 AcceptNotification .....	K-2
2.3 AcceptTLNotice .....	K-3
2.4 AcceptProposal.....	K-4
2.5 ProvideTime.....	K-5
2.6 InitiateReconciliation.....	K-5
2.7 ReceiveReconciliationResult .....	K-5
2.8 ProvideAuditTrail .....	K-6
2.9 ProvideTotals .....	K-6
2.10 ProvideUnitBlocks.....	K-7
2.11 GetTransactionStatus .....	K-7
2.12 ReceiveTotals .....	K-8
2.13 ReceiveUnitBlocks.....	K-8
2.14 ReceiveAuditTrail.....	K-9
3. Registry WSDL.....	K-10
4. Transaction Log WSDL .....	K-13
5. Common WSDL .....	K-16
 <b>Annex L: WSDL Examples and Instructions</b> .....	 L-1
1. Using the Web Services and WSDLs for Specific Transactions .....	L-1
1.1 Principles .....	L-1
1.2 Transaction Rules.....	L-1
1.3 Transaction Element Descriptions.....	L-3
1.4 Transaction Unit Block Element Descriptions.....	L-4
1.5 General Notes about each Kyoto Protocol Transaction Type .....	L-5
2. Transaction Examples.....	L-6

2.1	AcceptProposal.....	L-7
2.2	AcceptNotification .....	L-30
3.	Notification Examples.....	L-33
3.1	Net Source Cancellation (Type 1) .....	L-33
3.2	Non-compliance Cancellation (Type 2) .....	L-36
3.3	Impending tCER or ICER Expiry (Type 3) .....	L-40
3.4	Reversal of Storage for CDM Project (Type 4).....	L-43
3.5	Non-submission of Certification Report for CDM Project (Type 5).....	L-48
3.6	Excess Issuance for CDM Project (Type 6).....	L-51
3.7	Commitment Period Reserve Change (Type 7) .....	L-52
3.8	Unit Carry-over (Type 8) .....	L-54
3.9	Expiry Date Change (Type 9) .....	L-57
<b>Annex M: EU-ETS Supplementary Scheme Web Service Documentation.....</b>		<b>M-1</b>
1.	Introduction.....	M-1
2.	Account Management Web Service.....	M-2
2.1	Account Management web service operations.....	M-3
2.1.1	CreateAccountRequest .....	M-3
2.1.2	UpdateAccountRequest .....	M-5
2.1.3	CloseAccountRequest .....	M-6
2.1.4	UpdateVerifiedEmissionsRequest .....	M-6
2.1.5	ReceiveAccountOperationOutcomeRequest .....	M-7
2.2	Account Management WSDLs.....	M-8
2.2.1	Account Management Registry WSDL Description .....	M-8
2.2.2	Account Management Transaction Log WSDL Description .....	M-9
2.3	Account Management Processes - transaction examples .....	M-15
2.4	Account Management Processes.....	M-15
3.	Generic Web Service .....	M-19
3.1	Generic Web Service Process.....	M-19
3.1.1	Behaviour diagram.....	M-19
3.1.2	Logging .....	M-20
3.1.3	Constraints on operations support by the Generic Web Service .....	M-20
3.2	Generic Web Service message handling .....	M-20
3.2.1	Error handling and response codes .....	M-20
3.2.2	Message life time .....	M-21
3.2.3	Recovery from problems .....	M-21
3.3	Generic Web Service operations .....	M-22
3.3.1	Generic Web Service message elements.....	M-22
3.4	Generic Web Service WSDLs.....	M-23
3.6	Generic Web Service examples .....	M-23

## Annex A Glossary of Terms

**Figure A1: Glossary of Terms**

Term	Definition
AAU	Assigned amount units. These are tradable units derived from an Annex I Party's emissions target under the Kyoto Protocol. They may be counted by Annex I Parties towards compliance with their emissions target and are equal to one tonne of carbon dioxide equivalent gases.
Account	An account is used to partition a registry and can hold units. There are three accounts types: holding account, cancellation account and retirement account.
Accuracy	Condition in which information is not modified randomly by the software system.
Acknowledgement	An acknowledgement is the communication that is returned by a Web service (located at either the Transaction Log or at a registry) that a message has been successfully received. The acknowledgement occurs before the message is evaluated in any way other than format checks and minimum version requirements.
Administrator	A role to configure and maintain a software system. Configuration can range from system set-up to amending data and parameters within the system.
Annex I Party	A Party to the UNFCCC listed in Annex I to the UNFCCC. These are industrialized countries, including those with economies in transition.
Article	An Article of the Kyoto Protocol.
Attribute	Identifier for a piece of information.
Audit	Checking of recorded data.
Authentication	The process to confirm the identity of a user.
Authorization	The process to verify a permission to do something.
Cancellation	Cancellation is the action taken by the ITL for a proposed transaction when no response has been received from a registry within 24 hours.
CER	Certified Emission Reduction unit. These are tradable units generated by Projects that reduce emissions in Non-Annex I Parties under the CDM. They may be counted by Annex I Parties towards compliance with their emissions target and are equal to one tonne of carbon dioxide equivalent gases.
CDM	Clean Development Mechanism under Article 12 of the Kyoto Protocol. Projects in Non-Annex I Parties under the CDM result in reduced emissions, or enhanced removals. These Projects generate credits in the form of CERs, tCERs or ICERs.
CDM Executive Board	The board supervising the CDM. It is serviced by the Secretariat.
CDM Project	A Project under the Clean Development Mechanism under Article 12 of the Kyoto Protocol.
CDM Registry	The registry established by the CDM Executive Board on behalf of Non-Annex I Parties hosting CDM Projects. It is to ensure the accurate accounting of transactions of CERs, tCERs, and ICERs by those Parties. Its administrator is the Secretariat.

(cont.)



**Figure A1: Glossary of Terms (cont.)**

<b>Term</b>	<b>Definition</b>
Certificate Authority	Provides a Digital Certificate for site-to-site authentication to positively identify an organization and encrypt data communications between the organization and other certificate holders.
Commitment Period	A specified period in which an Annex I Party is to show compliance with its emissions target. The first Commitment Period for the Kyoto Protocol is from 2008 to 2012.
Communications Hub	The central communications component integrated in the ITL, through which all registries, the ITL, and any STLs communicate.
Component	A component is a group of programming functions that perform related tasks.
Conference of the Parties (COP)	The supreme decision-making body under the UNFCCC. Attended by delegations from all state Parties to the UNFCCC. The COP generally meets once a year.
Crediting Period	The period for which emission reductions or enhancement in removal of greenhouse gases from the atmosphere from a CDM or Joint Implementation Project are monitored and verified.
Customisation	Configuration of systems toward specific user needs within certain boundaries.
Denial of Service Attack	A very high number of requests in very short period aimed at a software system with the goal of achieving an overload and crash of that software system.
Digital Certificate	Provided by the Certificate Authority to ensure authentication of documents.
Discrepancy	A discrepancy is a finding by the ITL that a proposed transaction does not conform to agreed transaction rules.
Downtime	The time in which a software system is not available for use.
Emissions Trading	The trading of units which may count towards compliance by Annex I Parties with their emissions targets. Emissions trading is provided for under Article 17 of the Kyoto Protocol. Domestic (e.g., UK) and regional (e.g., EU) emissions trading schemes are also being established under the umbrella of Article 17 emissions trading under the Kyoto Protocol.
Encryption	A way of protecting data from unauthorized access.
Entities	Legal entities authorized by a government to participate in emissions trading or joint implementation Projects. Private and/or public entities involved in the CDM. Such entities may be from public, private or non-governmental sectors.
ERU	Emission reduction units. These are tradable units generated by Joint Implementation Projects in Annex I Parties. They may be counted by Annex I Parties towards compliance with their emissions target and are equal to one tonne of carbon dioxide equivalent gases.
Exchange Mechanism	System for exchanging data.
Finalization	Finalization is the action taken by a registry to complete a transaction which has been validated by the Transaction Log.
Function	A specific section of programming code within a component which performs a specific task.

(cont.)

**Figure A1: Glossary of Terms (cont.)**

Term	Definition
Functional Requirement	Requirement for which the quality test result is binary (e.g., yes/no or right/wrong).
GUI	Graphical User Interface.
Inconsistency	An inconsistency is a finding by the ITL that the unit information provided to the registry as part of a data reconciliation process differs from the information retained by the ITL.
Integrity	Assurance that data cannot be modified by any Party not authorized to do so.
International Transaction Log (ITL)	An electronic database established by the Secretariat to monitor the validity of transactions between registries under the Kyoto Protocol.
Invalidation	An invalidation is a finding by the Transaction Log that a message does not conform to the messaging requirements (including data formats, identifiers, etc.) in these Technical Specifications.
Joint Implementation Project	A Project under Joint Implementation under Article 6 of the Kyoto Protocol.
Kyoto Protocol	Allied agreement to the UNFCCC containing emission reduction targets for Annex I Parties.
ICER	Long-term Certified Emission Reduction unit. These are tradable units generated by Projects that enhance removals of greenhouse gases from the atmosphere in Non-Annex I Parties under the CDM. ICERs expire at the end of the crediting period of the Project (though these crediting periods may be renewed such that the Project may continue for up to 60 years). ICERs may be counted by Annex I Parties towards compliance with their emissions target and are equal to one tonne of carbon dioxide equivalent gases.
Logging	Functionality of a software system that stores information on the system for auditing and tracking.
Major Version Number	A major version number is the number assigned to the Technical Specifications for the Data Exchange Standards for purposes of identifying a specific set of technical requirements. The major version number changes only when a change in the Technical Specifications requires programming changes in a registry.
Minor Version Number	A minor version number is the number assigned to the Technical Specifications for the Data Exchange Standards for the purposes of identifying a specific set of technical requirements. The minor version number changes do not require programming changes within registries. These changes may involve response code table updates, for example.
Message	A message is a communication between the ITL and a registry or STL. It includes all data exchanged, including transaction and reconciliation data, requests for logs and responses. Messages are transported through HTTP SOAP requests.
National Registry	A registry established by an Annex I Party.

(cont.)

**Figure A1: Glossary of Terms (cont.)**

Term	Definition
Net Source	A Net Source is an activity which emits more greenhouse gases than it absorbs over a given period. A Net Source Cancellation is a transaction specific to the case where a LULUCF activity under Article 3.3 or 3.4 of the Kyoto Protocol, which would generally result in RMU issuance through its net absorption of greenhouse gases, is found to be a Net Source.
Non-Annex I Party	A Party to the UNFCCC which is not listed in Annex I to the UNFCCC. These are developing countries.
Non-functional Requirement	Requirement for which the quality test result is measure or a score (e.g., from 1-10 or high/medium/low)
Notification	A notification is a communication to a registry from the ITL about a required or recommended action involving unit transactions.
Party	A state that has ratified the Kyoto Protocol.
Process	The business area or category of interaction between registries and the ITL. The primary processes are unit issuance, unit conversion, external transfers, internal transfers (including cancellation replacement, and retirement), unit carry-over, expiry date change, and reconciliation. In addition, there is an ITLadministration process which addresses the need to manage message exchange failures, reference data, and manual intervention relating to reconciliation processes.
Protocol	Formal rules describing how to transmit data.
Reconciliation	The process by which data from different registry systems are compared and inconsistencies are resolved.
Recovery	The complete re-installation and reconfiguration of data or a software system.
Registry	A software system for the accounting of transactions in AAUs, RMUs, ERUs, CERs, tCERs, and ICERs. Includes national registries and the CDM registry.
Registry System	Generic term for national registries, the CDM Registry and Transaction Logs.
Removal	Removals of greenhouse gases from the atmosphere through LULUCF activities. Such removals may lead to the generation of RMUs, tCERs or ICERs. They are the "opposite" of emissions of greenhouse gasses.
Response	A response is the data sent following the processing of a proposed transaction. Typically the response includes the transaction ID, an indicator that the proposed transaction was successful or unsuccessful, and, if unsuccessful, the response code(s) providing the reason for the failure.
Reversal of Storage	A Reversal of Storage refers to a case in which an afforestation or reforestation activity under a CDM Project is found to be a Net Source. Where this CDM Project has previously generated the issuance of ICERs through its net absorption of greenhouse gases, a Reversal of Storage would require replacement of ICERs equal to the quantity of the Reversal of Storage.
RMU	Removal units. These are tradable units generated on the basis of removals of greenhouse gases from the atmosphere through LULUCF activities under Articles 3.3 and 3.4 of the Kyoto Protocol. They may be counted by Annex I Parties towards compliance with their emissions target and are equal to one tonne of carbon dioxide equivalent gases.

(cont.)

**Figure A1: Glossary of Terms (cont.)**

Term	Definition
Robust	A characteristic of a software system that describes the extent to which it is protected from loss of service or data integrity.
Role	A role is a set of permissions for functions that a person is allowed to perform. A role may be assigned to a user (person) or a group.
Scalability	The ability of a software system to handle higher workload than initially planned without modifying the program code.
Secretariat	Secretariat to the UNFCCC.
SOAP	Simple Object Access Protocol.
Stage	The stage of a transaction or reconciliation defines where in the process of data exchange a particular message or evaluation occurs. A stage ends and a new stage begins when a message has been successfully transmitted and occurred by either a registry or the ITL or when the last step of a process occurs.
Status	The transaction and reconciliation status describe the current state of the review process. As each process moves through the defined stages, the status will be updated to reflect the result of the ITL (or an external registry) evaluation. A transaction status might move from proposed to checked (no discrepancy), and to final. A reconciliation status might change from initiated to ITL validated.
Supplementary Program	An emissions tracking program for GHG emissions which operates as a complementary program to the Kyoto Protocol and shares communications with Kyoto Protocol participants through ITL Communications Hub.
Supplementary Transaction Log (STL)	An electronic database established for a Supplementary Program to monitor the validity of transactions between registries under that program.
tCER	Temporary Certified Emission Reduction unit. These are tradable units generated by Projects that enhance removals of greenhouse gases from the atmosphere in Non-Annex I Parties under the CDM. tCERs expire at the end of the Commitment Period subsequent to the Commitment Period within which they were issued. tCERs may be counted by Annex I Parties towards compliance with their emissions target and are equal to one tonne of carbon dioxide equivalent gases.
Termination	Termination is the action taken by a registry to end a proposed transaction which has been determined to be invalid, for which a discrepancy has been identified, for which the allowable response time has lapsed, or which it no longer wishes to process.
Transaction	The term transaction is used to describe a unique operation on a unit or block of units. A transaction is comprised of a series of actions related to a specific process. Each "transaction" is processed in stages and results in the return of a message to the registry identifying subsequent data on the transaction. A resubmission of the same information, following a transaction failure, is a new transaction.
True-up Period	The period from the end of the Commitment Period (2012) until 100 days after the completion of the Kyoto Protocol reviews of emissions information relating to the Commitment Period. Transfers of units may continue to take place during this period. The true-up period may therefore last until some time in 2015.

(cont.)

**Figure A1: Glossary of Terms (cont.)**

Term	Definition
UML	Unified Modeling Language. Notation standard for describing software systems.
Unavailable Status	Units which are involved in transactions that have been proposed and received by the ITL, and are waiting for a response from either another registry (for an external transfer) or the proposing registry, are "unavailable" for other transfer. These units are flagged as unavailable. Similarly, a unit involved in an inconsistency is marked as unavailable until the inconsistency is resolved. An ICER generated by a CDM Project for which a reversal of storage has occurred, or for which a certification report has not been submitted, is also marked as unavailable.
UNFCCC	United Nations Framework Convention on Climate Change. This is the framework treaty to which the Kyoto Protocol is allied.
Unit	Generic term for AAUs, RMUs, ERUs, CERs, tCERs, and ICERs.
Universal Time	Equivalent to Greenwich Mean Time (24-hour clock).
User	A person (human being) who interacts with a system.
User Acceptance Test	A test performed by a user of the system against a set of predefined test cases.
User Interface	The interface used by a person to interact with an application.
Virus	A software program that harms software systems or other software programs.
VPN	Virtual Private Network.
Web Service	A Web service is a group of operations that perform communication tasks to and from a registry and the ITL. Separate Web services are defined for different processes.
WSDL	Web Service Description Language.
XML	Extensible Markup Language. Standard used for structured data storage.

*[This page intentionally left blank.]*

## **Annex B**

### **Web Services and Functions for Transaction Processing**

#### **1. Introduction**

This annex contains a specification for the Web services and functions the registries must implement in order to establish communication with the ITL for transaction processing. Each Web service operation and suggested function is detailed in order to specify how the registry passes information to and from the ITL. Accordingly, the data elements that are passed into and out of a Function accessible through Web services are defined.

Some of these operations and functions are exposed to the public through Web services and others are functions that operate internally to the registry. In general, the input and output parameters have not been specified for functions that operate internally to a system. The design of the Web services public functions, including how they store and process data, may vary from registry to registry.

Only Web services and operations that are required for a registry are listed in this annex. For a complete list of the corresponding and interacting Web service operations and functions performed by the ITL, reference the ITL Technical Specifications document, Annex E.

#### **2. Objects and Structures**

The following sections explain the conventions and structured data storage utilised in the specifications for each function.

##### **2.1 Result Identifier**

Each function returns a Result Identifier. This value will indicate whether the function succeeded or failed. A failure could be the result of a business decision (a failed check) or the result of an unanticipated exception error (a run time error). The convention used here is that zero (0) indicates failure and one (1) indicates success.

When the ITL receives a http response of a web service request with the ResultIdentifier = 0, (indicating a failure of the request), the ITL does not automatically re-attempt the web service request. It is recommended that registry systems and STLs should follow this practice, and not re-attempt a web service request for which a ResultIdentifier=0 is returned in the http response.

##### **2.2 Transaction Object**

The Transaction Object is used to store all the elements involved in a transaction required for processing. The Transaction Object is a parent class to the TransactionUnitBlock object. The structure of the Transaction Object is as follows:

Structure	TransactionObject
	TransactionIdentifier As String
	TransactionType As Integer
	SuppTransactionType As Integer (optional)
	TransferringRegistryIdentifier As String
	TransferringRegistryAccountType As Integer (optional)
	TransferringRegistryAccountIdentifier As Long (optional)
	AcquiringRegistryIdentifier As String
	AcquiringRegistryAccountType As Integer (optional)
	AcquiringRegistryAccountIdentifier As Long (optional)
	NotificationIdentifier As Integer (optional)
	TransactionBlocks (array) As TransactionUnitBlockObject

This object contains a SuppTransactionType element which is used by the Supplementary Transaction Log for the European Commission GHG Trading Program.

## 2.3 TransactionUnitBlock Object

The functions below reference a TransactionUnitBlock Object to obtain the unit blocks associated with a transaction. Often, an array of TransactionUnitBlock Objects is called for because more than one block can be associated with a transaction. The structure of the TransactionUnitBlock Object is as follows:

Structure      TransactionUnitBlockObject  
                  UnitSerialBlockStart As Long  
                  UnitSerialBlockEnd As Long  
                  OriginatingRegistryIdentifier As String  
                  UnitType As Integer  
                  SuppUnitType As Integer (optional)  
                  OriginalCommitPeriod As Integer  
                  ApplicableCommitPeriod As Integer  
                  LULUCFActivity As Integer (optional)  
                  ProjectIdentifier As Integer (optional)  
                  Track As Integer (optional)  
                  BlockRole As String (optional)  
                  TransferringRegistryAccountType As Integer (optional)  
                  TransferringRegistryAccountIdentifier As Long (optional)  
                  AcquiringRegistryAccountType As Integer (optional)  
                  AcquiringRegistryAccountIdentifier As Long (optional)  
                  YearInCommitmentPeriod As Integer (optional)  
                  InstallationIdentifier As Long (optional)  
                  ExpiryDate As DateTime (optional)

This object contains a SuppUnitType, Year in Commitment Period, and Installation Identifiers which are used by the Supplementary Transaction Log for the European Commission's GHG trading program.

## 2.4 UnitBlock\_Identity Object

The UnitBlock\_Identity Object contains the attributes that uniquely identify a unit block within the ITL and is generally returned as part of a notification to a Registry in conjunction with a CheckResponse Object.

Structure      UnitBlock\_IdentityObject  
                  UnitSerialBlockStart As Long  
                  UnitSerialBlockEnd As Long  
                  OriginatingRegistryIdentifier As String

## 2.5 Check Response Object

The CheckResponse Object is used to collect all checks that are applicable to a process. As each check is evaluated, the result identifier is modified to indicate a success or failure for the check. When the CheckResponse Object is returned, it only includes those response codes that have failed.

Structure      CheckResponseObject  
                  ResultIdentifier As Integer  
                  ResponseCode As Integer

## 2.6 Evaluation Result Object

This object stores the specific CheckResponse Object along with the affected Unit Blocks.

Structure      EvaluationResultObject  
                  CheckResponseObject  
                  UnitBlocks (array) As UnitBlock\_Identity Object



## 10 3. Specified Functions

The following functions and Web service operations will be implemented to carry out the duties of a registry. They are listed in alphabetical order.

**Figure B1: AcceptNotification (Web Service)**

<b>Purpose</b>
<p>This is the HTTP SOAP request that registries and the ITL use to send notifications regarding transactions in process.</p> <p>This Web service is hosted on both the ITL and the registry. See Annex K for specifications on the WSDL requirements.</p>
<b>Inputs</b>
TransactionObject, EvaluationResultObject
<b>Process</b>
<p>This Web service performs preliminary checks on the content of the XML message before processing.</p> <p><b>Clarification for systems developers:</b> When this web service request is invoked <i>out-of-sequence</i>, systems <b>should not</b> respond to these requests by attempting to notify the requester using an asynchronous message. The requests should be logged in the registry's message log for auditing and troubleshooting purposes.</p>
<b>Outputs</b>
Result_Identifier
<b>Recommended Functions</b>
Data_Integrity_Check

**Figure B2: AcceptProposal (Web Service)**

<b>Purpose</b>
This is the HTTP SOAP request that the ITL uses to receive proposals from registries. Registries also must implement this service to receive proposals for external transfers from the ITL.
<b>Inputs</b>
TransactionObject
<b>Process</b>
<p>This Web service should perform preliminary checks on the content of the XML message before logging the message and writing contents of the message to a file.</p> <p><b>Clarification for systems developers:</b> When this web service request is invoked <i>out-of-sequence</i>, systems <b>should not</b> respond to these requests by attempting to notify the requester using an asynchronous message. The requests should be logged in the registry's message log for auditing and troubleshooting purposes.</p>
<b>Outputs</b>
Result_Identifier
<b>Recommended Functions</b>
Data_Integrity_Check Write_To_File Write_To_Message_Log

**Figure B3: Check\_Version (Function)**

<b>Purpose</b>
This function compares the major and minor version number in the HTTP SOAP request. These values are checked against the current version of the DES. Major version numbers must match or any forthcoming transactions to the ITL will be rejected. Messages with an outdated minor version are still processed, although a warning is issued. This is an optional function for registries.
<b>Inputs</b>
MajorVersion, MinorVersion
<b>Process</b>
Compare MajorVersion. If not a match, take corrective actions to update Registry system. This requires compliance with change management processes as defined on the ITL intranet website.  Compare MinorVersion. If not a match, then be prepared to update system with minor patches or enhancements as recommended on the ITL intranet website.
<b>Outputs</b>
CheckResponseObject

**Figure B4: Data\_Integrity\_Check (Function)**

<b>Purpose</b>
This function will check the data in the TransactionObject to ensure that it meets the minimum requirements to begin processing the proposed transaction. This is an optional function for registries.
<b>Inputs</b>
TransactionObject
<b>Process</b>
For the following processes, ensure that the incoming message meets the following criteria. See Annex E for a list of appropriate response codes to return for data integrity errors.
<b>Outputs</b>
CheckResponseObject TransactionObject

**Figure B5: Finalise\_Transaction (Function)**

<b>Purpose</b>
When a transaction is complete, this function is called to update the unit holding records of the registry.
<b>Inputs</b>
TransactionObject
<b>Process</b>
<p>This function evaluates the transaction type to determine the data which must be updated.</p> <p>If the transaction type = 1 (Issuance) insert records representing unit blocks and accounts, as appropriate.</p> <p>If the transaction type = 2, 7 or 8 (Conversion, Carry-over, or Expiry Date Change), the units are free to be used in a trade. Registries will commit the change of unit type and Project ID (for Conversion), the applicable Commitment Period (for Carry-over), or the expiry date (for Expiry Date Change).</p> <p>If the transaction type = 3, 4, 5 (External, Cancellation, Retirement) or (10 and the supplementary transaction types = 52, 53, 55, 01, 02, 41), the units are free to be used in a trade. Registries will commit the transfer ownership of the units to the acquiring Registry and accounts.</p> <p>If the transaction type = 6 (Replacement), the units are free to be used in a trade. Registries will commit the transfer to the replacement account and record the relationship between the two types of blocks.</p>

**Figure B6: Generate\_Proposal (Function)**

<b>Purpose</b>
This function creates the TransactionObject and corresponding TransactionUnitBlock for submission of a proposed transaction to the ITL. This function invokes AcceptProposal() on the ITL.
<b>Inputs</b>
<b>Outputs</b>
TransactionObject, TransactionUnitBlock
<b>Call(s)</b>
Write_Transaction Write_Transaction_Block Write_Transaction_Status

**Figure B7: Preliminary\_Checks (Function)**

<b>Purpose</b>
The registry calls this function when a message is received through one of its Web services. This function will authenticate the sender of the message and check the version number. If the message contains a transaction, it will call another function to write the message to a file. If there are no problems, the message is ready to be processed.
<b>Inputs</b>
TransactionObject, ReconciliationObject
<b>Process</b>
If this message contains a transaction, call Write_To_File to record the contents of the message.
<b>Outputs</b>
Result_Identifier
<b>Call(s)</b>
Check_Version Write_To_File

**Figure B8: Update\_Units (Function)**

<b>Purpose</b>
This function will record the transaction as final and update other records relating to the transaction, as appropriate.
<b>Inputs</b>
TransactionObject, TransactionUnitBlockObject
<b>Process</b>
<p>If the transaction type is External (3), the transaction is recorded as final and the units are removed from the appropriate accounts.</p> <p>If transaction type is Carry-over (7), then the Applicable Commitment Period for the unit is updated to the current Commitment Period.</p> <p>If the transaction type is Conversion (2), and the unit was not previously (2) RMU, then the Unit Type is updated to (3) for ERU and the Project ID is updated.</p> <p>If the transaction type is Conversion (2), and the previous unit type was (2) RMU, the Unit Type is updated to (4) for ERU converted from RMU and the Project ID is updated.</p> <p>If the transaction type is Cancellation (4) or Retirement (5), then transaction is recorded as final and the units moved to the appropriate account(s).</p> <p>If the transaction type is Expiry Date Change (8), update the expiry date.</p> <p>If the transaction type is Internal/Supplementary (10-51), (10-52) or (10-55), or (4-3) then the Supplementary Unit Type is updated to the appropriate code for the new Supplementary Transaction Type Code.</p>
<b>Outputs</b>
Result_Identifier



**Figure B9: Validate\_Proposal (Function)**

<b>Purpose</b>
This function is used only by acquiring registries to validate external transfers. This function could call other functions on the registry to perform additional checks.
<b>Inputs</b>
TransactionObject, TransactionUnitBlockObject
<b>Process</b>
Incoming proposals are evaluated and processed. Data integrity checks should be applied to proposed transaction. Log and record the transaction, applicable blocks and the transaction status. If the proposal is accepted or rejected, log the new transaction status and call the ITL AcceptNotification Web service, submitting the TransactionObject and TransactionUnitBlockObject with the transaction status and any applicable response codes.
<b>Calls</b>
Data_Integrity_Check Write_To_Message_Log Write_Transaction Write_Transaction_Block Write_Transaction_Status AcceptNotification

**Figure B10: Write\_To\_File (Function)**

<b>Purpose</b>
This function will create a text file, write the contents of the HTTP SOAP Request to the file, then add the file to a master Zip file.
<b>Inputs</b>
Web_Service_URL, Transaction_Type, XML Message Content, File_name
<b>Table(s)</b>
This function does not interact with the database.
<b>Process</b>
Retrieve the to and from elements in the contents of the SOAP request. Retrieve the Registry Code and Transaction Identifier values. Generate the file name by concatenating these two values along with a random number. Write contents of XML body to text file and store in the Zip file. If the file fails to write to file, return HTTP SOAP response error.
<b>Outputs</b>
Result_Identifier
<b>Call(s)</b>

**Figure B11: Write\_To\_Message\_Log (Function)**

<b>Purpose</b>
This function will append a record to the Message Log. The Message Log table records the history of all message exchange. The contents of any HTTP SOAP Request received involving a transaction are parsed and constructed as text files that are then stored in a larger Zip file. The Message Log tracks the time when the file was created, the name of the file and the name of the Zip file in which it has been compressed.
<b>Inputs</b>
Filename, Master_File_Name, Registry_Code, Reconciliation ID, Transaction ID, Web service
<b>Table(s)</b>
Message Log
<b>Process</b>
Append to Message_Log, Registry_Code, File_Name, Master_File_Name, SystemTime( ) Reconciliation ID or Transaction ID, Web service to Registry_Code, File_Name, File_Path, Submission_Date, Recon_ID, Transaction_ID, Web_Service.
<b>Outputs</b>
Result_Identifier
<b>Call(s)</b>

**Figure B12: Write\_Transaction (Function)**

<b>Purpose</b>
This function will insert a record into the Transaction Log table.
<b>Inputs</b>
TransactionObject
<b>Table(s)</b>
Transaction Log
<b>Process</b>
Append to Transaction Log the elements in the TransactionObject.
<b>Outputs</b>
Result_Identifier
<b>Call(s)</b>
Write_Transaction_Block

**Figure B13: Write\_Transaction\_Block (Function)**

<b>Purpose</b>
This function will insert a record into the Transaction Block table.
<b>Inputs</b>
TransactionObject, TransactionUnitBlockObject
<b>Table(s)</b>
Transaction_Block
<b>Process</b>
For each instance of the TransactionUnitBlockObject array associated with the TransactionObject, insert a record into the Transaction Block table.
<b>Outputs</b>
Result_Identifier
<b>Call(s)</b>

**Figure B14: Write\_Transaction\_Status (Function)**

<b>Purpose</b>
This function will insert a record into the Transaction Log History table.
<b>Inputs</b>
TransactionObject
<b>Table(s)</b>
Transaction Log History
<b>Process</b>
Append a record to the Transaction Log History table with Transaction ID, system time stamp and current transaction status from TransactionObject.
<b>Outputs</b>
Transaction Status ID, unique identifier for new Transaction Status History record, Result_Identifier
<b>Call(s)</b>

## Annex C

### Web Services and Functions for Reconciliation

#### 1. Introduction

This annex contains a specification for the Web services and functions the registries must implement in order to establish communication with the ITL for reconciliation. Each Web service operation and suggested function is detailed in order to specify how the registry passes information to and from the ITL. Accordingly, the data elements that are passed into and out of a function accessible through Web services are defined.

Some of these operations and functions are exposed to the public through Web services and others are functions that operate internally to the registry. In general, the input and output parameters have not been specified for functions that operate internally to a system. The design of the Web services public functions, including how they store and process data, may vary from registry to registry.

Only Web services and operations that are required for a registry are listed in this annex. For a complete list of the corresponding and interacting Web service operations and functions performed by the ITL, reference the ITL Technical Specifications document, Annex E.

#### 2. Objects and Structures

The following sections explain the conventions and structured data storage utilised in the specifications for each function. Also consult the TransactionObject, TransactionUnitBlockObject, UnitBlockIdentityObject, and CheckResponseObject defined in Annex B.

##### 2.1 Reconciliation Object

The Reconciliation Object is used to describe a reconciliation action and its current status.

Structure	ReconciliationObject
	ReconciliationIdentifier As String
	ReconciliationBeginDate As DateTime
	ReconciliationEndDate As DateTime
	ReconciliationSnapshotDateTime As DateTime
	ReconciliationStatusCode As Integer
	ReconciliationStatusDateTime As DateTime
	ReconciliationStartPhaseCode As Integer
	ReconciliationCommitPeriod As Integer
	ReconciliationComment As String
	ReconciliationResponseCodes (Array) As Integer

##### 2.2 Totals Object

The Totals Object is used by the reconciliation process to store the number of units held by a registry, account type, or account.

Structure	TotalsObject
	ReconciliationIdentifier As String
	ReconciliationSnapshotDate As DateTime
	HoldingRegistry As String
	AccountType As Integer
	AccountCommitPeriod As Integer
	AccountIdentifier As Integer
	UnitType As Integer
	SupplementaryUnitType As Integer
	UnitCount As Integer

### 3. Functions and Web Services

**Figure C1: Calculate\_Totals (Function)**

<b>Purpose</b>
<p>This function will calculate the total number of units held at the registry in response to a request from the ITL. The totals will be calculated by querying the snapshot data that was stored at the agreed upon date and time for the reconciliation.</p> <p>The ITL may request totals only for units held by a specific account type or unit type. If no specific requests are made, the totals should be calculated for each distinct combination of account type and unit type.</p>
<b>Inputs</b>
ReconciliationObject, Holding Account Type, Unit Type
<b>Process</b>
<p>If no limiting criteria were passed into the ProvideTotals service,</p> <p style="padding-left: 40px;">Calculate the number of units held grouped by account type, Account Commitment Period, and unit type. Below is an example of an SQL query to calculate the totals.</p> <pre style="padding-left: 80px;"> Select Account Type, Account Commitment Period, Unit Type, Sum(end_block - start_block + 1) as Unit Totals From Reconciliation Snapshot Data Where Reconciliation ID = current reconciliation action ID Group By Account Type, Account Commitment Period, and Unit Type </pre> <p>If limiting criteria were passed into the ProvideTotals service,</p> <p style="padding-left: 40px;">Calculate the number of units held by the specified account type, Account Commitment Period, or unit type. Below is an example of an SQL query to calculate these totals.</p> <pre style="padding-left: 80px;"> Select Account Type, Account Commitment Period, Unit Type, Sum(end_block - start_block + 1) as Unit Totals From Reconciliation Snapshot Data Where Reconciliation ID = current reconciliation ID and Account Type = account type specified by ITL and Unit Type = unit type specified by ITL Group By Account Type, Account Commitment Period, and Unit Type </pre> <p>Store the totals returned in an array of the TotalObject, and call the ReceiveTotals Web service method on the ITL.</p>
<b>Outputs</b>
TotalsObject array
<b>Call(s)</b>



**Figure C2: Close\_Reconciliation\_Action (Function)**

<b>Purpose</b>
This function will update the registry's Reconciliation Log with the end date of a reconciliation action.
<b>Inputs</b>
ReconciliationObject
<b>Process</b>
Update Reconciliation Log tables so that Reconciliation End Date = date and time when ITL confirms reconciliation compliance. This information is received from the ReceiveReconciliationResult Web service operation.
<b>Outputs</b>
Result_Identifier
<b>Call(s)</b>

**Figure C3: Generate\_Audit\_Trail (Function)**

<b>Purpose</b>
<p>This function identifies past transactions involving unit blocks that are in question. In order to identify the source of an inconsistency, the ITL will request data for all the transactions that occurred within a specified time period for the unit blocks in question. This function will accumulate a list of transactions by querying the transaction log for the specified time period and send the audit trail to the ITL.</p>
<b>Inputs</b>
<p>ReconciliationObject, UnitBlock_IdentityObject, Begin Transaction Time, End Transaction Time</p>
<b>Process</b>
<p>Query the Transaction Log and the Transaction Block tables and return all available information for the specified units and time period. Below is an outline of an SQL query to generate the audit trail.</p> <p style="padding-left: 40px;">Select Transaction Number, Transaction Date, Transaction Type, Transferring Registry, Acquiring Registry, Transferring Account, Acquiring Account Originating Registry, Start Block, End Block, Unit Type, Applicable Commit Period, Original Commit Period, Project ID, LULUCF Activity Code, Track ID, Expiry Date</p> <p style="padding-left: 40px;">From Transaction Log, Transaction Block Where Transaction Date &gt;= requested begin transaction time and Transaction Date &lt;= requested end transaction time and Originating Registry = requested originating registry and ( (Start Block &gt;= requested start block and Start Block &lt;= requested end block) or (End Block &gt;= requested start block and End Block &lt;= requested end block) or (Start Block &lt; requested start block and End Block &gt; requested end block) )</p> <p>Store the audit trail to a TransactionObject array, and call the ReceiveAuditTrail Web service method on the ITL.</p>
<b>Outputs</b>
<p>UnitBlockObject array</p>
<b>Call(s)</b>

**Figure C4: Generate\_Unit\_Blocks (Function)**

<b>Purpose</b>
<p>This function identifies unit blocks requested by the ITL for comparison during reconciliation. This function will accumulate a list of unit blocks by querying the snapshot unit block data that was stored at the requested date and time for the reconciliation.</p> <p>The ITL may request only the unit blocks of a specific Unit Type and/or held by a specific Account Type be sent. If no criteria are specified, all unit blocks should be sent to the ITL.</p>
<b>Inputs</b>
ReconciliationObject, Holding Account Type, Unit Type
<b>Process</b>
<p>If no limiting criteria were passed to the ProvideUnitBlocks service,</p> <p style="padding-left: 40px;">Generate a complete list of unit blocks. Below is an outline of an SQL query to generate the list of unit blocks.</p> <p style="padding-left: 80px;">Select Account Type, Unit Type, Originating Registry Code, Applicable Commitment Period, Start Block, End Block From Reconciliation Snapshot Data Where Reconciliation ID = current reconciliation action ID</p> <p>If limiting criteria were passed to the ProvideUnitBlocks service,</p> <p style="padding-left: 40px;">Generate a list of unit blocks that meet the specified criteria. Below is an outline of an SQL query to generate the list of unit blocks.</p> <p style="padding-left: 80px;">Select Account Type, Unit Type, Originating Registry Code, Applicable Commitment Period, Start Block, End Block From Reconciliation Snapshot Data Where Reconciliation ID = current reconciliation action ID and Account Type = account type specified by ITL and Unit Type = unit type specified by ITL and Account Commit Period = Account Commit Period specified by ITL</p> <p>Store the Unit Blocks identified in an array of the UnitBlockObject, and call the ReceiveUnitBlocks Web service method on the ITL.</p>
<b>Outputs</b>
TransactionObject array
<b>Call(s)</b>

**Figure C5: InitiateReconciliation (Web Service)**

<b>Purpose</b>
This is the HTTP SOAP request that the ITL uses to confirm an upcoming reconciliation action.
<b>Inputs</b>
ReconciliationObject
<b>Process</b>
<p>This service receives the reconciliation identifier and snapshot date and time from the ITL. This time will have already been agreed upon by the ITL and registry administrators. Upon receipt of this message the registry should record the reconciliation identifier with a status of 0 ("Confirmed") and schedule the snapshot (if it has not already done so). There is no response to the ITL at this point.</p> <p>See Section 5 for detailed information on the reconciliation process.</p>
<b>Outputs</b>
CheckResponseObject
<b>Call(s)</b>
Write_To_Reconciliation_Log Write_To_Reconciliation_Status

**Figure C6: ProvideAuditTrail (Web Service)**

<b>Purpose</b>
This is the HTTP SOAP request that the ITL uses to request audit trail data from a registry.
<b>Inputs</b>
ReconciliationObject
<b>Process</b>
<p>This function receives a request from the ITL to provide a transaction history for a specified set of units within a specified timeframe. The requested units may be explicitly listed, or they may be requested for a specified unit type or account type of the owner. If a cancellation or retirement account type is specified, a specific Commitment Period for the account may also be specified.</p> <p>Call Write_To_Reconciliation_Unit_Block to record as inconsistent any unit blocks that have been listed.</p> <p>If this request follows a request to provide unit blocks, call Write_To_Reconciliation_Status to record the new status as "Unit Blocks Inconsistent."</p> <p>If this is the first request for information by the ITL during this reconciliation action (i.e. the ITL skipped the request for totals and units blocks), update the Reconciliation Log to indicate the start phase for this action is Phase 3. Call Write_To_Reconciliation_Status to record the new status as "Initiated."</p> <p>See Section 5 for detailed information on the reconciliation process.</p>
<b>Outputs</b>
CheckResponseObject
<b>Call(s)</b>
<p>Write_To_Reconciliation_Status  Write_To_Reconciliation_Unit_Block</p>

**Figure C7: ProvideTotals (Web Service)**

<b>Purpose</b>
This is the HTTP SOAP request that the ITL uses to initiate a request for total number of units held by each registry.
<b>Inputs</b>
ReconciliationObject
<b>Process</b>
<p>This service receives a request for the total number of units held by each registry. It will be called by the ITL upon completion of the data snapshot for the reconciliation action. The request may be for a specified account type, unit type, or a combination of the two. When the registry is part of a supplementary program, the request may also include a specified account identifier. If a cancellation or retirement account type is specified, a specific Commitment Period for the account may also be specified. If no limiting criteria are passed, the request is for all totals, grouped by account type and unit type.</p> <p>Call Write_To_Reconciliation_Log to record the start phase of this reconciliation action as Phase 1.</p> <p>Call Write_To_Reconciliation_Status to record the latest status as "Initiated."</p> <p>See Section 5 for detailed information on the reconciliation process.</p>
<b>Outputs</b>
CheckResponseObject
<b>Call(s)</b>
<p>Write_To_Reconciliation_Log</p> <p>Write_To_Reconciliation_Status</p>

**Figure C8: ProvideUnitBlocks (Web Service)**

<b>Purpose</b>
This is the HTTP SOAP request that the ITL uses to initiate a request for the unit blocks held at a registry.
<b>Inputs</b>
ReconciliationObject
<b>Process</b>
<p>This service receives a request for the unit blocks held at a registry. The request may be for a specified account type, unit type, or a combination of the two. When the registry is part of a supplementary program, request may also include a specified account identifier. If a cancellation or retirement account type is specified, a specific Commitment Period for the account may also be specified. If no limiting criteria are passed, the request is for all unit blocks.</p> <p>If this request follows a prior request to provide total counts, call Write_To_Reconciliation_Status to record the new status as "Totals Inconsistent."</p> <p>If this is the first request for information during this reconciliation action (i.e. the ITL skipped the request for totals), update the Reconciliation Log to indicate the start phase for this action is Phase 2. Call Write_To_Reconciliation_Status to record the new status as "Initiated."</p> <p>See Section 5 for detailed information on the reconciliation process.</p>
<b>Outputs</b>
CheckResponseObject
<b>Call(s)</b>
<p>Write_To_Reconciliation_Status Write_To_Reconciliation_Log</p>

**Figure C9: ReceiveReconciliationResult (Web Service)**

<b>Purpose</b>
This is the HTTP SOAP request that the ITL uses to inform the registry that it has finished processing a reconciliation action.
<b>Inputs</b>
ReconciliationObject
<b>Process</b>
This Web service receives the result of the ITL's reconciliation analysis. Registries will record the results of the analysis in their Reconciliation Logs. The ITL will call this service when there is an abnormal end to the ITL reconciliation analysis. The reason for the abnormal end will be described by the array of response codes. The ITL will also call this service when the reconciliation action is successfully completed.
<b>Outputs</b>
CheckResponseObject
<b>Call(s)</b>
Close_Reconciliation_Action Write_To_Reconciliation_Log Write_To_Reconciliation_Status



**Figure C10: Snapshot\_Data (Function)**

<b>Purpose</b>
This function will take a "snapshot" of the data at the agreed upon time. The reconciliation action will be based on this data snapshot.
<b>Inputs</b>
ReconciliationObject
<b>Process</b>
<p>At the specified time, create a "snapshot" of the unit block data. The fields needed in the snapshot for each unit block include:</p> <ul style="list-style-type: none"><li>• Holding Account Type Code</li><li>• Holding Account Commitment Period</li><li>• Unit Type Code</li><li>• Originating Registry Code</li><li>• Applicable Commitment Period</li><li>• Start Block Number</li><li>• End Block Number</li></ul> <p>The Reconciliation ID should also be stored. These data should be preserved at least until the reconciliation action has closed.</p>
<b>Outputs</b>
Result_Identifier
<b>Call(s)</b>

**Figure C11: Write\_To\_Reconciliation\_Log (Function)**

<b>Purpose</b>
This function inserts a new record into the Reconciliation Log table.
<b>Inputs</b>
ReconciliationObject
<b>Table(s)</b>
Reconciliation Log
<b>Process</b>
Append to Reconciliation Log table as follows: <ul style="list-style-type: none"><li>• Recon_ID = input reconciliation ID</li><li>• Recon_Action BeginDateTime = input reconciliation begin DateTime</li><li>• Recon_Log_Comment = input comment</li><li>• Recon_Action EndDateTime = input reconciliation end DateTime</li><li>• Recon_Snapshot_DateTime = input reconciliation snapshot DateTime</li><li>• Recon_Start Phase_Code = input reconciliation starting phase</li><li>• Recon_CommitPeriod = input reconciliation Commitment Period</li></ul>
<b>Outputs</b>
Result_Identifier
<b>Call(s)</b>

**Figure C12: Write\_To\_Reconciliation\_Status (Function)**

<b>Purpose</b>
This function inserts a new record into the Reconciliation_Status_History table.
<b>Inputs</b>
ReconciliationObject
<b>Table(s)</b>
Reconciliation Status History
<b>Process</b>
<p>Append to Reconciliation Status History table as follows:</p> <ul style="list-style-type: none"> <li>• Recon_ID = input reconciliation ID</li> <li>• Recon_Status_Code = input reconciliation status code</li> <li>• Recon_Comment = input comment</li> <li>• Recon_Log_DateTime = system time</li> </ul> <p>For each response code (if any) associated with this reconciliation status, append to the Reconciliation_Log_Response_Codes as follows:</p> <ul style="list-style-type: none"> <li>• response code = input response code</li> </ul>
<b>Outputs</b>
Result_Identifier ReconciliationObject
<b>Call(s)</b>

**Figure C13: Write\_To\_Reconciliation\_Unit\_Block (Function)**

<b>Purpose</b>
This function will add a record to the Reconciliation_Unit_Block table to indicate that there is an inconsistency between the registry and the ITL concerning the ownership of that unit block.
<b>Inputs</b>
ReconciliationObject, UnitBlockIdentityObject
<b>Table(s)</b>
Reconciliation Status History
<b>Process</b>
<p>For each Unit Block append to the Reconciliation_Unit_Block table as follows:</p> <p>ReconciliationID = input reconciliation ID  ActionDateTime = system DateTime  OriginatingRegistry = input originating registry  UnitSerialBlockStart = input block start  UnitSerialBlockEnd = input block end  ResponseCode = input response code</p>
<b>Outputs</b>
Result_Identifier
<b>Call(s)</b>

## **Annex D**

### **Web Services for Administrative Processes**

#### **1. Introduction**

This annex contains a specification for the Web services and functions the registries must implement in order to establish communication with the ITL for administrative processes. Each Web service operation is detailed in order to specify how the registry passes information to and from the ITL.

The design of the Web services public functions, including how they store and process data, may vary from registry to registry.

Only Web services and operations that are required for a registry are listed in this annex. For a complete list of the corresponding and interacting Web service operations and functions performed by the ITL, reference the ITL Technical Specifications document, Annex E.

#### **2. Objects and Structures**

The Web services defined for administrative processes make extensive use of the TransactionObject and the TransactionUnitBlockObject. See Annex B for a definition of these objects.

**Figure D1: AcceptITLNotice (Web Service)**

<b>Purpose</b>
<p>This HTTP SOAP service receives incoming messages forwarded from the ITL. This Web service may be used to receive notifications from the ITL. The ITL will call this Web service to inform the registry about issues identified by administrative and clean-up processes on the ITL. This Web service will be used by the following processes:</p> <ol style="list-style-type: none"> <li>1. Inform the registry when cancellation of units is required as part of a net source cancellation action.</li> <li>2. Inform the registry when cancellation of units is needed as a result of the Party being found by the Compliance Committee to be in non-compliance with its emissions target for the previous Commitment Period.</li> <li>3. Inform the registry of tCERs held in retirement and tCER replacement accounts or ICERs held in a retirement account that are about to expire and must be replaced.</li> <li>4. Inform the registry of the need to replace or cancel units due to a Reversal of Storage at a Project.</li> <li>5. Inform the registry of ICERs that may not be traded due to a lack of certification report for the Project associated with the ICERs.</li> <li>6. Inform the registry that the CDM Registry issued too many CERs for a Project and that it will be necessary for the operational entity responsible for that Project to cancel or replace some units.</li> <li>7. Inform the registry of changes in the Commitment Period Reserve (CPR) or changes in unit holdings which result in holdings below the Party's CPR level.</li> <li>8. Inform the registry of Outstanding Units at the end of the Commitment Period.</li> <li>9. Inform the registry of expiry date changes required for tCERs or ICERs.</li> <li>10. Inform the registry of the status of a prior notification.</li> <li>11. Inform the registry of changes in the EU15 Commitment Period Reserve (EU15 CPR) or changes in unit holdings which result in the EU15 holdings below the EU15 CPR level.</li> </ol>
<b>Inputs</b>
<p>From, To, MajorVersion, MinorVersion, MessageContent, MessageDateTime, NotificationStatus, NotificationType, NotificationID, ProjectID, UnitType, TargetValue, TargetDate, LULUCFActivity, CommitPeriod, ActionDueDate, TransactionUnitBlockObjectArray</p>
<b>Process</b>
<p>Upon receipt of an incoming message, the message will be logged and the system administrator notified.</p>
<b>Outputs</b>
<p>Result_Identifier</p>
<b>Call(s)</b>

**Figure D2: AcceptMessage (Web Service)**

<b>Purpose</b>
This HTTP SOAP service receives incoming messages forwarded from the ITL. This Web service may be used to receive messages from the ITL or another registry. The Content input element is flexible and can contain a large amount of text. This allows the function to act as a generic service for registries to send messages to each other.
<b>Inputs</b>
From, To, MajorVersion, MinorVersion, MessageContent, MessageDateTime
<b>Process</b>
Upon receipt of an incoming text message from the AcceptMessage Web service, the system administrator shall be alerted of its arrival.
<b>Outputs</b>
Result_Identifier
<b>Call(s)</b>

**Figure D3: GetTransactionStatus (Web Service)**

<b>Purpose</b>
This is the HTTP SOAP request registries use to retrieve the current status of a transaction at the ITL.
<b>Inputs</b>
Transaction ID
<b>Process</b>
This function will query the ITL database and return the latest status of a specified transaction. It will also return the date and time the status was last updated.
<b>Outputs</b>
TransactionStatus TransactionStatusUpdateDateTime Transaction Response Codes
<b>Call(s)</b>



**Figure D4: ProvideTime (Web Service)**

<b>Purpose</b>
This is the HTTP SOAP request that the ITL uses to request the current time from a registry. The request provides the ITL time as an informational service to assist in recognizing time synchronization problems.
<b>Inputs</b>
From, To, MajorVersion, MinorVersion, ITL SystemTime
<b>Process</b>
The ITL will periodically request a registry to return the current time in order to ensure that the system time is in synchronization with the ITL.
<b>Outputs</b>
SystemTime Result_Identifier ResponseCodes
<b>Call(s)</b>

*[This page intentionally left blank.]*

## Annex E

### List of Checks and Response Codes for Message Processing

#### Index of Checks and Response Codes in this Annex:

Table E 1: Check Categories .....	E-2
Table E 2: Version and Authentication Checks .....	E-4
Table E 3: Message Viability Checks .....	E-5
Table E 4: System Status Checks .....	E-6
Table E 5: ITL Transaction Data Integrity Checks .....	E-8
Table E 6: Message Sequence for Transaction Messages .....	E-13
Table E 7: General Transaction Checks .....	E-14
Table E 8: Registry Messages .....	E-42
Table E 9: ITL Reconciliation Responses .....	E-42
Table E 10: ITL Administrative Responses .....	E-43
Table E 11: Obsolete Response Codes .....	E-44

## 1. Purpose

This annex describes the response codes that can be returned to registries and STLs during processing and the reasons for them being returned. These response codes arise from checks performed by the ITL, STLs and acquiring registries.

## 2. Summary of Response Specifications

The ITL sends responses to registries to report the result of message processing. Responses can be “simple” with only two pieces of data to report, or “complex” containing large record sets of data. Simple responses, which check for a minimal version control test, merely send back an indication that the message was received successfully and that the message is queued for processing. Complex responses are messages that are sent after a transaction or reconciliation action has been evaluated by the ITL. If the transaction or the reconciliation action was unsuccessful, the ITL sends a message containing response codes providing the reason for the failure.

Each registry shall have a response lookup table and the capability to receive and import updates to this table to support change management specifications described in Section 8. Registries shall use programming techniques which facilitate updates to codes in these tables without programming changes whenever possible.

A registry may not change the number associated with a response or the meaning of the response description. Each registry may elect to translate the response description into a different language for display or reporting purposes only. A registry may request that a response be added to the system by submitting a written request to the ITL administrator (or through an alternative process).

Response codes are assigned to the following range of numbers by category. Registries must be implemented in a manner such that receipt of a message with a response code not in the response lookup tables can still be processed according to the message flows in the data exchange specification. The ITL Administrator should be informed of the code so that the reason for it can be investigated. The purpose of this requirement is to ensure that a response code not explicitly specified in Annex E should not present a technical problem to the receiving registry in terms of following the correct message flow.

**Table E 1: Check Categories**

<b>Category</b>	<b>Response Code Range</b>	<b>Category Description</b>	<b>Action Upon Failure</b>
Version and Authentication	1000 - 1299	Checks to authenticate sender and to validate version of DES during preliminary processing.	Message returned with response codes or HTTP Soap Error. Message not placed into message queue (unless only a minor version inconsistency is identified).
Message Viability	1300 - 1399	Checks to determine whether the message is viable when processed from the queue.	Message returned with response codes. Message not logged in the Transaction Log table.
System Status Validation	1500 - 1599	Checks to validate the status of systems (registry, STL, ITL) during queue processing.	Message returned with response codes. Message not logged in the Transaction Log table.
Transaction Data Integrity	2000 - 2999	Basic checks of data content including numeric ranges and validity of codes during queue processing.	Message returned with response codes. Message not logged in the Transaction Log table.
Message Sequence for Transaction Messages	3000 - 3999	Checks to validate message order and transaction status.	Message returned with response codes. Message not logged in the Transaction Log table. The messages using these codes have been discontinued and must not be used
General Transaction Checks	4000 - 4999	Checks applicable to all transactions involving unit blocks. These checks are applicable to all transactions, except for Issuance.	Message returned with response codes and transaction status. Message logged in the Transaction Log table.
Transaction-specific Checks	5000 - 5899	Kyoto Protocol transaction checks specific to designated transaction types.	Message returned with response codes and transaction status. Message logged in the Transaction Log table.
Acquiring Registry Messages	5900 - 5999	Response codes generated by registries.	Message returned with response codes and transaction status. Message logged in the Transaction Log table.
Reconciliation Data Integrity	6200 - 6299	Basic checks for data content in reconciliation messages.	Message returned with response codes. Message not logged in Reconciliation Log table.
Reconciliation Message Sequence	6300 - 6399	Checks to validate message order and reconciliation status.	Message returned with response codes. Message not logged in ITL Reconciliation Log table.
Other Reconciliation Checks and Messages	6400 - 6500	Basic reconciliation checks.	Message returned with response codes and transaction status. Message logged in ITL Reconciliation

Category	Response Code Range	Category Description	Action Upon Failure
			Log table.
EUEUTL Checks and Messages	7000 - 7999	EUEUTL checks enforcing EU ETS policy. Non EU ETS registries may receive these response codes if transacting with EU ETS registries.	Message returned with response codes and transaction status. Message logged in the Transaction Log table.
ITL Administrative Response Codes	8000 - 8999	Checks and responses related to administrative functions.	Message returned with response codes.

### 3. Message Validity Response Codes

The following responses are used in checks for well-formedness, DES version compatibility, and registry validation. These response codes apply to all data exchange processes.

**Table E 2: Version and Authentication Checks**

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
SOAP error	Certificate Check	Version and Authentication	Certificate must be recognized.	All		
SOAP error	SOAP Identifier	Version and Authentication	Initiating Registry must be consistent with sender of SOAP message.	All		
SOAP error	WSDL Check	Version and Authentication	Message must conform to WSDL.	All		
1031	Major Version	Version and Authentication	Major Version number in transaction message must match current Major Version number for DES.	All	<p>Changes in major version numbers denote major changes to the DES requiring registry reprogramming; these are not backwards compatible with the earlier major DES versions.</p> <p>This check is performed as soon as the ITL receives a message and, if failed, the response code is returned synchronously by the ITL to the registry.</p>	24/CP.8, annex, paragraph 9

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
1032	Minor Version	Version and Authentication	Minor Version number in transaction message should match current Minor Version number for DES.	All	<p>Changes in minor version numbers denote less substantial changes to the DES that may be taken into account by a registry without complex reprogramming; minor DES versions are backwards compatible with earlier minor versions, such that a registry and the ITL with different minor versions can still communicate.</p> <p>This check is performed as soon as the ITL receives a message and, if failed, the response code is returned synchronously by the ITL to the registry.</p>	24/CP.8, annex, paragraph 9

**Table E 3: Message Viability Checks**

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
1301	Message Age	Message Viability	Message must be processed within 24 hours of submission.	All	Transactions for which messages are held in the ITL queues for more than 24 hours will be cancelled by the ITL	24/CP.8, annex, paragraph 12

Table E 4: System Status Checks

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
1501	Initiating Registry	System Status Validation	Initiating Registry must be listed in Registry table.	All	Registry table refers to a table in the ITL database	24/CP.8, annex, paragraph 20 (b)
1502	Acquiring Registry	System Status Validation	Acquiring Registry must be listed in the Registry table	All	Registry table refers to a table in the ITL database	24/CP.8, annex, paragraph 20 (b), RSNM-35 Invalid Acquiring Registry Response Code
1503	Initiating Registry Transactions Status	System Status Validation	Initiating Registry status must allow it to send messages of type <i>acceptProposal</i> or <i>acceptNotification</i> .	All	Registry status must not be <i>Stop Transactions, Not Operational, or Closed</i>	24/CP.8, annex, paragraph 19
1504	Acquiring Registry Transactions Status	System Status Validation	Acquiring Registry status must allow it to process messages of type <i>acceptProposal</i> .	All	Registry status must not be <i>Stop Proposals, Stop Proposals and Suspend Transactions, Stop Transactions, Not Operational, or Closed</i>	24/CP.8, annex, paragraph 19
1505	ITL Operational Status	System Status Validation	The ITL must be in "Full Operational" status to accept messages.	All	Messages are not queued. The response code is provided in the synchronous http response to the web service request	RSN-20 Registry Statuses



Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
1510	Registry Reconciliation Status	System Status Validation	Registry status must allow reconciliation actions to be conducted. (The ITL will maintain the current status of each registry. In this case, the ITL must recognize that the registry is available for reconciliation.)	All	Registry status must not be <i>Not Operational</i> , or <i>Closed</i>	24/CP.8, annex, paragraph 19
1511	Initiating Registry or STL Authentication	System Status Validation	The identity of the Registry or STL in the "from" element of the message must match the identity of the sender of the message.	All	This check is performed as soon as the ITL receives a message and, if failed, the response code is returned synchronously by the ITL to the registry.	
1512	Sending Registry and Initiating Registry Match	System Status Validation	The sender of the proposal message is not the transferring registry.	All		
1516	Initiating Registry Proposal Status	System Status Validation	Initiating Registry status must allow it to send messages of type <i>acceptProposal</i> .	All	Registry status must not be <i>Stop Proposals</i> , or <i>Stop Proposals and Suspend Transaction Processing</i>	RSNCM-20 Registry Statuses
1517	Kyoto Protocol Messaging Suspended	System Status Validation	The status of the registry allows it to send an <i>acceptMessage</i> message.	All	Registry status must not be <i>Closed</i> .	RSNCM-20 Registry Statuses
1518	EUTL Transaction Status	System Status Validation	The status of the EUTL must allow the processing of <i>acceptProposal</i> and <i>acceptNotification</i> messages to or from EU-ETS registries	All	EUTL status must not be <i>Not Operational</i> .	RSNCM-20 Registry Statuses
1519	EUTL Reconciliation Status	System Status Validation	The status of the EUTL must allow the processing of reconciliation related messages.	All	EUTL status must not be <i>Not Operational</i> .	RSNCM-20 Registry Statuses

#### 4. ITL Transaction Response Codes

The following responses are used by the ITL to support transaction processing.

**Table E 5: ITL Transaction Data Integrity Checks**

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
2001	Transaction Mask	Data Integrity	Transaction ID must be comprised of a valid registry code followed by numeric values.	All	Checks the format of the identifier given to the transaction by the registry	13/CMP.1, annex, paragraph 41 (a) 24/CP.8, annex, paragraph 17
2002	Transaction Type Code	Data Integrity	Transaction type Code must be valid.	All	Checks that the transaction type is one recognized by the DES; other checks ensure that other message attributes meet other transaction requirements	24/CP.8, annex, paragraphs 6, 17
2003	Supplementary Transaction Type Code	Data Integrity	Supplementary Transaction Type Code must be valid.	All	Supplementary trading schemes (e.g. EU greenhouse gas emissions trading scheme – EU ETS) identify some transactions using supplementary transaction type codes	

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
2004	Transaction Status Code	Data Integrity	Transaction status code must be valid.	All	These codes show the position of the message within a defined transaction sequence; the transaction status given must be recognized by the DES	24/CP.8, annex, paragraphs 6, 7
2006	Account Type Code	Data Integrity	Account Type Code must be valid.	All	Checks that the account type code is one recognized by the DES	24/CP.8, annex, paragraph 16
2007	Initiating Account Identifier	Data Integrity	Initiating Account Identifier must be greater than zero.	All	Initiating account identifier must be greater than zero and must not exceed 999,999,999,999,999	24/CP.8, annex, paragraph 16
2008	Acquiring Account Identifier	Data Integrity	Acquiring Account Identifier must be greater than zero.	All	Acquiring account identifier must be greater than zero and must not exceed 999,999,999,999,999	24/CP.8, annex, paragraph 16
2010	Originating Registry	Data Integrity	The Originating Registry of all unit blocks must be valid.	All	Originating country codes are part of the serial number of each unit and indicate the party for which the unit was issued; they remain unchanged, even with transfer to another registry	24/CP.8, annex, paragraph 14
2011	Unit Type Code	Data Integrity	Unit Type Code must be valid.	All	Unit type codes are part of the serial number of each unit and indicate whether the unit is an AAU, RMUs, ERU (from an AAU), ERU (from an RMU), CER, tCER or ICER	24/CP.8, annex, paragraph 14

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
2012	Supplementary Unit Type Code	Data Integrity	Supplementary Unit Type Code must be valid.	All	This code gives information on units assigned specific attributes under supplementary trading schemes; the ITL passes this info to STLs	
2013	Unit Serial Block	Data Integrity	Unit Serial block start and Unit Serial block end must be present.	All	Serial numbers are packaged in blocks to reduce data volume (in both data storage and data transmission); each block is defined by its start and end serial numbers; all other components of the serial numbers in between the start and end are identical	24/CP.8, annex, paragraphs 7 (d), 14
2014	Unit Serial Range	Data Integrity	Unit Serial block end must be greater than or equal to the Unit Serial block start.	All		24/CP.8, annex, paragraphs 7 (d), 14
2015	LULUCF Activity Code	Data Integrity	RMUs, ERUs converted from RMUs, tCERs and ICERs must have a valid LULUCF activity code.	All	There are seven LULUCF codes: afforestation and reforestation (a combined code), deforestation, forest management, cropland management, grazing land management, revegetation and wetland drainage and rewetting; ; these codes conform to the way in which Parties are to report data on LULUCF activities	24/CP.8, annex, paragraph 14 and 2/CMP.7, annex, paragraphs 6

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
2016	No LULUCF Activity Code	Data Integrity	AAUs, ERUs converted from AAUs and CERs must not have a LULUCF activity code.	All		24/CP.8, annex, paragraph 14
2017	Project ID	Data Integrity	ERUs, CERs, tCERs, and ICERs must have a valid Project ID.	All	CDM project identifiers are generated by the CDM Executive Board; JI project identifiers are generated by the Party (under track 1) or the Article 6 Supervisory Committee (under track 2)	24/CP.8, annex, paragraph 14
2018	No Project ID	Data Integrity	AAUs or RMUs must not have a Project ID.	All		24/CP.8, annex, paragraph 14
2019	ERU Track Code	Data Integrity	ERUs must have a valid track code.	All	Track 1 codes indicate that ERUs are generated from JI projects verified through national procedures; track 2 codes indicate that ERUs are generated from JI projects verified through procedures under the Article 6 Supervisory Committee	13/CMP.1, annex, paragraph 29 24/CP.8, annex, paragraph 14
2020	No Track Code	Data Integrity	AAUs, RMUs, CERs, tCERs, and ICERs must not have a track code.	All		13/CMP.1, annex, paragraph 29 24/CP.8, annex, paragraph 14

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
2021	Expiry Date	Data Integrity	tCERs and ICERs must have an Expiry Date.	All	The expiry date indicates the date by which tCERs and ICERs must be replaced with other units	
2022	No Expiry Date	Data Integrity	AAUs, RMUs, ERUs and CERs must not have an Expiry Date.	All		5/CMP.1, annex, paragraphs 42, 46
2023	Only a transaction proposal can have a status of Proposed	Data Integrity	A notification cannot have the status Proposed.	All		
2024	Overlapping Unit Blocks	Data Integrity	The unit blocks of a transaction must not overlap each other.	All		
2025	Notification ID	Data Integrity	The notification ID does not refer to an existing incomplete ITL notice sent to the registry.	All		
2026	Acquiring and Transferring Registry Consistency	Data Integrity	For all transactions, except for external transfers, the Initiating and Acquiring Registries must be the same.	All		
2027	Number of Unit Blocks in External Transfer	Data Integrity	The number of unit blocks in proposed external transfer exceeds the maximum number of 3000.	All	The maximum number of 3000 is the value as of 30 Sept 2010. This may be changed in the future, based on operational experience.	RSNCM-12
2028	Number of Unit Blocks in Transaction	Data Integrity	The number of unit blocks in proposed transaction exceeds the maximum number of 3000.	All	The maximum number of 3000 is the value as of 30 Sept 2010. This may be changed in the future, based on operational experience.	RSNCM-12

**Table E 6: Message Sequence for Transaction Messages**

As of DES version 1.1.2 (Annex E version 1.1.5), messages using *out-of-sequence* response codes have been discontinued and should not be used.

Although the checks are still performed in the ITL, the ITL has been modified so as not to respond to *out-of-sequence* messages.

The codes are still recorded in the ITL logs, and may be used for problem analysis purposes. The codes may be found in §9 “Obsolete Response Codes” in this annex.

Registries should also not respond to *out-of-sequence* messages.

**Table E 7: General Transaction Checks**

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
4002	Prior Record of Units	General Transaction	Units identified in the transaction must already exist in the ITL.	All	Serial numbers for units involved in transactions (except issuance transactions, to which the general transaction checks do not apply) are checked against records already stored in the ITL; this check builds upon previous checks that ensure that each component of the serial number is provided in a valid format	13/CMP.1, annex, paragraphs 39, 40
4003	Registry Holds Units	General Transaction	Units identified in the transaction must be held by Initiating Registry.	All	A registry cannot initiate a transaction involving units it does not hold	13/CMP.1, annex, paragraphs 39, 40
4004	Unit Block Attributes	General Transaction	All attributes of all unit blocks must be consistent with ITL unit block attributes except where attributes are changed by the current transaction.	All	The ITL checks all serial number components to ensure consistency with ITL records; the only differences are where the transaction involves changes to components (e.g. changing the unit type code from AAU to ERU during conversion), for which registries provide the modified unit attributes	13/CMP.1, annex, paragraphs 27, 36



Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
4005	Single Applicable Commitment Period	General Transaction	All unit blocks in transaction must be for a single Applicable Commitment Period.	All	This is to avoid technical complexity; actions involving units of multiple commitment periods must be separated into multiple transactions	
4007	Acquiring and Transferring Registries for External Transactions	General Transaction	For external transfers, the Initiating and Acquiring Registries must be different.	All	For external transfers, the Initiating and acquiring registries must be different	13/CMP.1, annex, paragraph 40
4008	Units Have ITL Inconsistencies	General Transaction	Units identified in the transaction must not have inconsistencies identified through reconciliation with the ITL.	All	Inconsistencies between ITL and registry data for particular units, discovered through the reconciliation process, must be resolved before the unit may be transacted	24/CP.8, annex, paragraph 26
4010	Units are Unavailable	General Transaction	Units identified in the transaction must not be involved in another transaction.	All	The data exchange standards give priority to completing transactions in the order that they are initiated	24/CP.8, annex, paragraph 13
4011	Units are Cancelled	General Transaction	Cancelled units must not be subject to further transactions.	All		13/CMP.1, annex, paragraph 35
4012	Units are Retired	General Transaction	Retired units must not be subject to further transactions.	All		13/CMP.1, annex, paragraph 35

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
4013	Units are Expired	General Transaction	Expired tCERs and ICERs must not be subject to further transactions, except cancellation to a mandatory cancellation account	All	The exception is to allow tCERs and ICERs to be cleaned out of holding accounts after their expiry	5/CMP.1, annex, paragraphs 42, 46, 53
4014	Units Previously Used in Replacement	General Transaction	Units previously used to replace tCERs or ICERs must not be subject to further transactions.	All	Such units would be in replacement accounts; replacement functions similarly to cancellation, in that units used in replacement transactions may not be subsequently transferred out of the replacement account	5/CMP.1, annex, paragraph 35
4015	ICER Transaction Ineligibility	General Transaction	ICERs must not be transferred to a holding or retirement account where the CDM Executive Board has notified a replacement requirement for the associated Project.	All	The CDM Executive Board informs the ITL of the total quantity of ICERs from a project which requires replacement; the ITL notifies each registry of the quantity it is required to replace and blocks all ICERs from the project concerned from being transferred to any account other than replacement or cancellation accounts; in the paragraph 49 case, ICERs remaining in holding accounts after sufficient units are replaced are unblocked and are again available for transactions	5/CMP.1, annex, paragraphs 49, 50

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
4016	Transaction Must Contain Unit Block Information	General Transaction	A transaction proposal must contain at least one unit block.	All		
4017	Transferring Account Type and Identifier Consistency	General Transaction	A transferring account identifier must not match a government account identifier on record at the ITL unless the transferring account type also matches the account type on record.	All		
4018	Acquiring Account Type and Identifier Consistency	General Transaction	The acquiring account identifier and type must be consistent with the government account information supplied to the ITL Administrator.	All		
4019	Internal and External Transfers to cancellation, retirement, replacement accounts	General Transaction	Only cancellation, retirement, replacement and expiry date change transactions can have cancellation, retirement or replacement accounts as acquiring accounts.	All	It is not allowed to use internal transfers (transaction type 10) or external transfers (transaction type 03) to transfer units to cancellation, retirement or replacement accounts, with the exception of cancellations by means of external transfers to the excess issuance account of the CDM registry.	
4021	Notifications over-fulfillment	General Transaction	The sum of the units in the proposed transaction added to the sum of the units previously transferred to fulfil this notification cannot exceed the number of units required for this notification	All		

**Figure E8: Transaction specific Checks**

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5001	National Registry Issuance	Transaction-specific	AAUs and RMUs must be issued by a national registry.	Issuance	The CDM registry cannot issue AAUs or RMUs	13/CMP.1, annex, paragraph 39
5002	No ERU Issuance	Transaction-specific	ERUs cannot be issued.	Issuance	ERUs can only be generated through the conversion of AAUs and RMUs; the DES refer to this transaction as “conversion”, while the CMP decisions refer to “issuance” of ERUs. The check name and check description refer to the DES terminology for issuance.	13/CMP.1, annex, paragraph 29
5003	CDM Registry Issuance	Transaction-specific	CERs, tCERs, and ICERs must be issued by the CDM Registry.	Issuance	National registries cannot issue CERs, tCERs or ICERs	5/CMP.1, annex, paragraph 37  13/CMP.1, annex, paragraph 39
5004	Single Issuance Unit Type	Transaction-specific	A transaction must not issue more than one Unit Type.	Issuance	This is to avoid technical complexity; issuance of multiple unit types must be separated into multiple transactions	

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5005	Single Issuance Commitment Period	Transaction-specific	The Original Commitment Period must be the same for all units issued by the transaction.	Issuance	This is to avoid technical complexity; issuance of units for multiple commitment periods must be separated into multiple transactions	
5006	Consistent Applicable Commitment Period	Transaction-specific	The Applicable Commitment Period must be the same as the Original Commitment Period for all units issued by the transaction.	Issuance	Prior to any carry-over for the unit, the applicable CP identifier must be the same as the original CP identifier	
5007	Issued Serial Numbers	Transaction-specific	Serial numbers for proposed issuance must not already exist in the ITL.	Issuance	Serial numbers for units to be issued must not have been used previously	3/CMP.1, annex, appendix D, paragraph 7 13/CMP.1, annex, paragraphs 24, 27, 29
5008	AAU Issuance Quantity	Transaction-specific	The quantity of AAUs issued must not exceed allowed quantity for the Commitment Period.	Issuance	The allowed quantity of AAU issuance is set by assigned amount pursuant to Article 3.7 and 3.8 of the Kyoto Protocol, as recorded in the C&A database; AAUs may be issued in multiple transactions over time but must not exceed the allowed quantity	13/CMP.1, annex, paragraph 23

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5009	RMU Issuance Quantity	Transaction-specific	The quantity of RMUs issued for each LULUCF Activity Type must not exceed the allowed quantity for that LULUCF Activity Type and Commitment Period.	Issuance	The allowed quantities of RMU issuance, by LULUCF activity, are set by rules under Article 3.3 and 3.4, taking account of the limits established by decision 11/CP.7; the ITL requires that the allowed quantities, by LULUCF activity and as confirmed by the Article 8 and compliance processes, be stored in the C&A database so that the ITL can access it from there; the common reporting format for reporting these activities separates data by LULUCF activity; RMUs may be issued in multiple transactions over time but must not exceed the allowed quantity for each LULUCF activity type	13/CMP.1, annex, paragraphs 25, 26, 28
5010	CDM Issuance Unit Type	Transaction-specific	The type of units to be issued for each CDM Project must be consistent with the Project activity.	Issuance	ITL requires information from the CDM Executive Board on whether the project activity is a LULUCF activity, in which case a tCER or ICER must be issued	5/CMP.1, annex, paragraph 38
5011	Consistency of Unit Type Issued for a LULUCF CDM Project	Transaction-specific	Choice of unit type must be consistent with previous issuance of tCERs and ICERs for the Project.	Issuance	A unit type may be selected (between tCERs or ICERs) for the first issuance from a project activity; this unit type must be maintained thereafter	5/CMP.1, annex, paragraph 39

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5012	CDM Issuance Quantity	Transaction-specific	CER, tCER, or ICER issuance for each CDM Project must not exceed quantity specified by CDM Executive Board.	Issuance	The ITL needs to receive copies of the issuance instructions from the CDM Executive Board stating the allowed level of issuance; units for a single project and crediting period may be issued in multiple transactions over time but must not exceed the allowed quantities of CERs or tCERs/ICERs	3/CMP.1, annex, appendix D, paragraph 6
5013	CDM LULUCF Activity Code	Transaction-specific	The LULUCF Activity Code of CERs, tCERs, or ICERs proposed for issuance must be consistent with the project activity.	Issuance	This information is included in the serial numbers of tCERs and ICERs; the ITL requires information on the project activity from the CDM Executive Board prior to the first issuance of units for the project, in order to check whether a LULUCF code is necessary	24/CP.8, annex, paragraph 14  5/CMP.1, annex, paragraph 38
5014	CDM Project ID	Transaction-specific	A valid CDM Project ID must be present for the issuance of all CERs, tCERs, and ICERs.	Issuance	This information is included in the serial numbers of all CERs, tCERs and ICERs; the ITL requires the project identifier from the CDM Executive Board prior to the first issuance of units for the project	24/CP.8, annex, paragraph 14

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5015	tCER Expiry Date	Transaction-specific	Expiry Date for tCERs must be consistent with the end date of the Commitment Period subsequent to the Original Commitment Period of the tCER.	Issuance	This information is included in the serial numbers of all tCERs; as this date is not yet known, the issuing registry and the ITL initially assumes a common date (31 December 2017) for the issuance process; if necessary, this expiry date may be changed later using the expiry date change transaction	5/CMP.1, annex, paragraph 42
5016	ICER Expiry Date	Transaction-specific	Expiry date for ICERs must be consistent with the End Date of the Crediting Period for the Project specified by the CDM Executive Board.	Issuance	This information is included in the serial numbers of all ICERs; the ITL requires this information from the CDM Executive Board prior to the first issuance of units for the project and must subsequently be informed of any renewals in crediting periods; if necessary, this expiry date may be changed later using the expiry date change transaction	5/CMP.1, annex, paragraph 46
5017	Issuance of AAUs and RMUs to Holding Accounts	Transaction-specific	The Acquiring Account for an issuance transaction involving AAUs or RMUs must be a holding account.	Issuance		



Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5018	RMU Issuance Limit for Forest Management	Transaction-specific	The quantity of RMUs issued for LULUCF Activity Type 3 (forest management) must not exceed the Party's Forest Management cap, plus net cancellations for activity types 1 and 2 up to a limit of 165,000,000 units.	Issuance		<a href="#">Decision 13/CMP.1, annex, paragraph 40</a>
5019	National Registry Originating Registry Code	Transaction-specific	A national registry cannot issue units on behalf of another national registry.	Issuance		
5020	CDM Originating Registry Code	Transaction-specific	The originating country of CDM issued units must be the country hosting the CDM project referenced by the units.	Issuance		
5051	National Registry Conversion	Transaction-specific	The Initiating Registry converting AAUs or RMUs must be a national registry.	Conversion	<a href="#">The CDM registry cannot convert units to ERUs</a>	<a href="#">13/CMP.1, annex, paragraph 29</a>
5052	Holding Account Conversion	Transaction-specific	The Initiating Account for a conversion transaction must be a holding account.	Conversion	<a href="#">Conversion occurs within a holding account; the transfer of ERUs to other accounts occurs after unit conversion, as a separate transaction; units held in cancellation, retirement, replacement or the pending account of the CDM registry cannot be converted to ERUs</a>	

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5053	Conversion Eligibility (Track 1)	Transaction-specific	If the unit is a Track 1 ERU, the Party of the Initiating Registry must be determined to meet eligibility criteria 1 through 6.	Conversion	All eligibility criteria must be met by the Annex B Party before its registry may convert units to ERUs with a track 1 code in their serial numbers; the ITL draws upon up-to-date records in the C&A database concerning each Party's fulfilment of the criteria	9/CMP.1, annex, paragraphs 21-23
5054	Conversion Eligibility (Track 2)	Transaction-specific	If the unit is a Track 2 ERU, the Party of the Initiating Registry must be determined to meet eligibility criteria 1, 2, and 4.	Conversion	A smaller set of eligibility criteria must be met by the Annex B Party before its registry may convert units to ERUs with a track 2 code in their serial numbers; the ITL draws upon up-to-date records in the C&A database concerning each Party's fulfilment of the criteria	9/CMP.1, annex, paragraphs 21-24
5056	Conversion Unit Type	Transaction-specific	Units for conversion must be AAUs or RMUs.	Conversion	CERs, tCERs and ICERs cannot be converted to ERUs	13/CMP.1, annex, paragraph 29
5057	Single Conversion Unit Type	Transaction-specific	A transaction must not convert more than one unit type.	Conversion	This is to avoid technical complexity; conversion from multiple unit types must be done through multiple transactions	
5058	Conversion by Issuing Registry	Transaction-specific	Units for conversion must have been issued by Initiating Registry.	Conversion	A Party cannot convert AAUs or RMUs issued by other Parties and since acquired via an external transaction	13/CMP.1, annex, paragraph 29

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5059	Project ID	Transaction-specific	A valid JI Project ID must be present for the conversion of all ERUs.	Conversion	This information is included in the serial numbers of all ERUs; the ITL requires the project identifier from the Party (track 1) or the Article 6 Supervisory Committee (track 2) before first conversion for the project	13/CMP.1, annex, paragraph 29
5060	JI Conversion Unit Type	Transaction-specific	The type of units to be converted to ERUs for each JI Project must be consistent with Project activity.	Conversion	AAUs must be converted for emission-reducing JI activities; RMUs must be converted for LULUCF JI activities, on the basis that (a) LULUCF JI projects must conform to definitions, accounting, modalities and guidelines under Article 3.3 and 3.4 and (b) ERUs converted from RMUs cannot be carried over; the ITL requires information on the project activity from the Party (track 1) or the Article 6 Supervisory Committee (track 2) prior to the first conversion of units for the project	9/CMP.1, COP/MOP draft decision, paragraph 4 16/CMP.1, annex, paragraph 11 13/CMP.1, annex, paragraph 15

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5061	Track 2 ERU Conversion Quantity	Transaction-specific	Track 2 ERU Conversion for each Track 2 JI Project must not exceed the quantity specified by the Joint Implementation Supervisory Committee.	Conversion	The ITL needs to receive copies of the issuance instructions from the Article 6 Supervisory Committee (track 2) stating the allowed level of issuance; units for a single project and crediting period may be converted in multiple transactions but must not exceed the allowed quantity	9/CMP.1, annex, paragraphs 24, 39
5062	Project track code	Transaction-specific	The unit block track code must match the JI project track code.	Conversion		

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5101	General Transferring Registry Eligibility for External Transfers	Transaction-specific	The Party of an initiating national registry must be determined to meet eligibility criteria 1 through 6, except for the first external transfer of a track 2 ERU which the Registry has converted or for transfers to the Excess Issuance Cancellation Account at the CDM Registry.	External	All eligibility criteria must be met by the Annex B Party before its registry may conduct external transfers to other registries; the ITL draws upon up-to-date records in the C&A database concerning each Party's fulfilment of the criteria; this check is not applied where the external transfer is a first external transfer of track 2 ERUs which the Registry has converted, as this initial distribution of track 2 ERUs to project participants in other Parties is allowed under special eligibility conditions in the JI track 2 case (this does not allow subsequent emission trading of track 2 ERUs); this exception does not itself allow such a transfer to proceed, but allows such a transfer to progress without check failure to check 89, which tests if the JI track 2 eligibility criteria are met; this check does not apply to external transfers initiated by the CDM registry, as this would be distribution of CERs, tCERs or ICERs to project participants under Article 12 (rather than emissions trading under Article 17)	9/CMP.1, paragraph 24  11/CMP.1, annex, paragraph 2

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5102	ERU Track 2 Transferring Registry Eligibility for External Transfers	Transaction-specific	If the transaction is the first external transfer of a track 2 ERU which the Registry has converted, the Party of the initiating national registry must be determined to meet eligibility criteria 1, 2, and 4.	External	This check builds upon the exception provided by the previous check; the initial distribution of track 2 ERUs to project participants in other Parties is allowed as long as the smaller set of eligibility conditions in the JI track 2 case are met by the Annex B Party; the ITL draws upon up-to-date records in the C&A database concerning each Party's fulfilment of the criteria	9/CMP.1, annex, paragraph 24
5103	Acquiring Registry Eligibility for External Transfers	Transaction-specific	The Party of an acquiring national registry must be determined to meet eligibility criteria 1 through 6, except for transfers initiated by the CDM Registry or for transfers to the Excess Issuance Cancellation Account at the CDM Registry.	External	As with initiating registries, all eligibility criteria must be met by an Annex B Party before its registry may receive external transfers from other registries (this applies also to track 2 ERUs); the ITL draws upon up-to-date records in the C&A database concerning each Party's fulfilment of the criteria; this check is not applied where the transfer comes from the CDM registry, as this would be the distribution of CERs, tCERs or ICERs to project participants under Article 12 (rather than emissions trading under Article 17)	9/CMP.1, annex, paragraph 21 3/CMP.1, annex, paragraph 66 11/CMP.1, annex, paragraph 2

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5104	Commitment Period Reserve	Transaction-specific	The total quantity of all units held in a national registry, which may be used for compliance for the applicable Commitment Period of a transaction, must not fall below the CPR level for the Party for that Commitment Period, except where the transaction is a first transfer of Track 2 ERUs converted by the registry or a transfer to the Excess Issuance Account in the CDM Registry.	External	External transfers must not cause the unit holdings in a registry to fall below the CPR level of the Annex B Party; the first transfers of track 2 ERUs are not subject to this CPR level check CPR level: The CPR level, as reported by each Party and confirmed under the review and compliance procedures, is recorded in the C&A database; this CPR level is further reduced by any first external transfers of track 2 ERUs which the Registry has converted (since the Article 6 Supervisory Committee ensures that generation of such ERUs is associated with reductions in emissions, such that their transfer does not affect the compliance situation of the Party); this reduction is shown in the check description column by adding this amount to the holdings side of the equation) Holdings level: Only units holdings which have not expired or been cancelled or replaced may count towards holdings for CPR purposes; adjustments ensure that only these holdings are taken into account. The commitment period reserve is also reduced by the required cancellations and replacements not carried out within the 30 days timeframe.	11/CMP.1, annex, paragraphs 8-10

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5105	External Transfers to CDM Registry	Transaction-specific	CDM Registry can only receive external transfers to Cancellation accounts for compensating excess issuance of CERs, tCERs and ICERs.	External	The CDM registry cannot acquire units, except where an operational entity has been required by the CDM Executive Board to cancel units to compensate excess CER, tCER or ICER issuance	3/CMP.1, annex, paragraph 22 3/CMP.1, annex, appendix D, paragraph 3
5106	Suspension from making external transfers	Transaction-specific	The Party of an initiating national registry must not have been suspended from making external transfers as a result of not meeting its emission target for the previous Commitment Period.	External	This refers to one of the consequences for an Annex B Party being found not to be in compliance with its Article 3.1 emissions target	24/CP.7, annex, section XV, paragraph 5 (c)
5107	Cancellation Account Commitment Period (external transfer)	Transaction-specific	Any unit blocks cancelled by means of an external transfer to the Excess Issuance Cancellation Account in the CDM registry must have the same Applicable Commitment Period as the Cancellation Account.	External		
5108	Notification ID for tCER and ICER Cancellations to Excess Issuance Cancellation Accounts (external transfer)	Transaction-specific	tCERs and ICERs may only be transferred to the Excess Issuance Cancellation Account in the CDM Registry in the case that excess tCER and ICER issuance is being compensated pursuant to an Excess Issuance Notification.	External		



Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5109	Project ID for Excess Issuance Cancellation	Transaction-specific	For transfers to the Excess Issuance Cancellation Account at the CDM Registry, the Project ID for the units to be cancelled must be consistent with the Project ID contained in the Excess Issuance Notification.	External		
5110	Correct Expiry Date for tCERs and ICERs	Transaction-specific	If there exists an outstanding expiry date change notification (Type 9) affecting the tCERs or ICERs to be transferred, the units' expiry date must match the target expiry date specified in the notification.	External		
5111	EU15 Commitment Period Reserve	Transaction-specific	The total quantity of all units held by EU15 registries, which may be used for compliance for the applicable Commitment Period of a transaction, must not fall below the CPR level for the EU15 for that Commitment Period, except where the transaction is a first transfer to a nonEU15 registry of Track 2 ERUs converted by the registry or a transfer to the Excess Issuance Account in the CDM Registry.	External	Note that external transfers between EU15 registries are not subject to this check.	* Added through a change request referring to the "Agreement Between the European Community and its Member States under Article 4 of the Kyoto Protocol", see FCCC/CP/2002/2
5112	Condition on external transfer of CER, ICER and tCER valid of the second commitment period	Transaction-specific	A party that doesn't have a quantified emission limitation and reduction commitment for the second commitment period is not eligible to transfer or acquire CER, ICER, tCER in that commitment period	External	This check doesn't apply to the forwarding by the CDM registry, to other units types, to other commitment periods or to the cancellation of units to the CDM registry	1/CMP.8 (Doha Amendment), paragraph 13

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5113	AAU, RMU, and ERU Cancellation to Administrative Cancellation Accounts	Transaction-specific	AAUs, RMUs, and ERUs cannot be transferred to administrative cancellation accounts.	Cancellation	This check only applies to national registries.	
5151	National Registry Cancellation	Transaction-specific	Cancellation to Net Source, Non-Compliance, and Voluntary Cancellation Accounts must take place in a national registry.	Cancellation	These cancellation accounts do not exist in the CDM registry	13/CMP.1, annex, paragraphs 21, 32, 33, 37
5152	CDM Registry Cancellation	Transaction-specific	Cancellation to Excess Issuance and Administrative Cancellation Account must not take place in a national registry.	Cancellation	These cancellation accounts do not exist in national registries	3/CMP.1, annex, appendix D, paragraph 3 and 8/CMP.7 paragraph 28
5153	Cancellation Accounts	Transaction-specific	The Acquiring Account for a cancellation transaction must be a cancellation account.	Cancellation	Holding, retirement, replacement and pending accounts cannot acquire units in cancellation transactions	3/CMP.1, annex, appendix D, paragraph 3 13/CMP.1, annex, paragraphs 21, 32, 33, 37 5/CMP.1, annex, appendix D, paragraph 3
5154	Cancellation Account Identifier	Transaction-specific	Account identifiers must be provided for acquiring accounts in cancellation transactions.	Cancellation	The ITL normally checks only the account type; in the case of cancellation, the ITL checks that the account identifier matches an account specified for this purpose by the registry	24/CP.8, annex, paragraph 7

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5155	Cancellation Account Commitment Period (cancellation)	Transaction-specific	The unit blocks to be cancelled must have the same Applicable Commitment Period as the Cancellation Account.	Cancellation	Required to enable the ITL to monitor cancellations against requirements for the commitment period	13/CMP.1, annex, paragraph 21
5156	tCER and ICER Cancellation to Net Source and Non-compliance Cancellation Accounts	Transaction-specific	tCERs and ICERs cannot be transferred to net source cancellation accounts or non-compliance cancellation accounts.	Cancellation		5/CMP.1, annex, paragraph 52
5157	Notification ID for tCER and ICER Cancellations to Excess Issuance Cancellation Accounts (cancellation via External Transfer)	Transaction-specific	tCERs may only be transferred to the excess issuance cancellation account in the CDM registry in the case that excess tCER issuance is being compensated pursuant to an excess issuance notification. ICERs may only be transferred to the excess issuance cancellation account in the CDM registry in the case where excess ICER issuance is being compensated pursuant to an excess issuance cancellation notification referencing the same project as the ICERs being cancelled.	Cancellation	tCERs may be transferred to excess issuance cancellation accounts to compensate excess tCERs issuance; ICERs may be transferred to excess issuance cancellation accounts to compensate excess ICERs issuance; the ITL needs to be informed of such compensation requirements, as set by the CDM Executive Board	5/CMP.1, annex, paragraph 52
5158	Notification ID for Net Source Cancellations	Transaction-specific	Units may only be transferred to a net source cancellation account if a notification has been received from the ITL and this ID is reported in the transaction.	Cancellation	Required to enable the ITL to monitor cancellation against notifications	

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5159	Notification ID for Non-compliance Cancellations	Transaction-specific	Units may only be transferred to a non-compliance cancellation account if a notification has been received from the ITL and this ID is reported in the transaction.	Cancellation	Required to enable the ITL to monitor cancellation against notifications	
5160	Cancellation Notification for Reversal of Storage	Transaction-specific	A valid Notification ID must be provided for cancellation upon reversal of storage for a CDM project.	Cancellation		
5161	Cancellation Notification for Lack of Certification Report	Transaction-specific	A valid Notification ID must be provided for cancellation upon a non-submission of Certification Report for a CDM project.	Cancellation		
5166	Incorrect commitment period	General Transaction	The units used in a net source cancellation transaction for any LULUCF type must have an applicable commitment period that match the commitment period of the notification	Cancellation		
5201	National Registry Replacement	Transaction-specific	The Initiating Registry replacing units must be a national registry.	Replacement	CDM registry cannot replace units; tCERs and ICERs in the CDM registry which require replacement are instead transferred to a mandatory cancellation account within the CDM registry	5/CMP.1, annex, paragraphs 44, 48, 49, 50 and appendix D paragraph 3
5202	tCER Replacement Accounts	Transaction-specific	The Acquiring Account for a replacement transaction involving tCERs must be a tCER replacement account.	Replacement	The acquiring account for a replacement transaction involving tCERs must be a tCER replacement account	5/CMP.1, annex, paragraph 43

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5203	ICER Replacement Accounts	Transaction-specific	The Acquiring Account for a replacement transaction involving ICERs must be an ICER replacement account.	Replacement	The acquiring account for a replacement transaction involving ICERs must be a ICER replacement account	5/CMP.1, annex, paragraph 47
5204	Replacement Account Identifier	Transaction-specific	Account identifiers must be provided for acquiring accounts in replacement transactions.	Replacement	The ITL normally checks only the account type; in the replacement case, the ITL checks that the account identifier matches an account specified for this purpose by the registry	24/CP.8, annex, paragraph 7
5205	Replacement Account Commitment Period	Transaction-specific	The Unit Blocks used for replacement must have the same Applicable Commitment as the Replacement Account.	Replacement	Required to enable the ITL to monitor replacements against requirements for the commitment period	13/CMP.1, annex, paragraph 21
5206	Unit Type to be Replaced	Transaction-specific	Units to be replaced must be tCERs or ICERs.	Replacement	AAUs, RMUs, ERUs and CERs cannot be replaced; these can only be used to replace tCERs and ICERs	5/CMP.1, annex, paragraphs 44, 48, 49, 50
5207	Multiple Replacement	Transaction-specific	A unit may be replaced only once.	Replacement	As part of the replacement transaction, the ITL flags its records of tCERs and ICERs as having been replaced; only one replacement is needed to compensate the extra emission made possible by retiring a tCER or ICER; units which have already been replaced are not replaced again upon expiry	

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5209	Quantity of Replacement Units	Transaction-specific	The quantity of units replaced must equal the quantity of replacing units.	Replacement		5/CMP.1, annex, paragraphs 44, 48-, 49,50
5211	Location of Replaced tCERs	Transaction-specific	tCERs to be replaced must be held in a Retirement account or a tCER Replacement account.	Replacement	tCERs in holding or cancellation accounts do not need to be replaced upon expiry (as they have not resulted in higher emissions being allowed)	5/CMP.1, annex, paragraph 44
5212	Location of Replaced ICERs	Transaction-specific	ICERs to be replaced must be held in Retirement or Holding accounts.	Replacement	Requirements to replace ICERs need to be fulfilled through replacing units which have already been retired or are on a holding account.	5/CMP.1, annex, paragraphs 48, 49, 50
5213	ICER Replacement Units (upon Expiry)	Transaction-specific	ICER Replacement accounts (upon expiry) cannot acquire tCERs or ICERs.	Replacement		5/CMP.1, annex, paragraph 48
5214	tCER Replacement Units (upon Expiry)	Transaction-specific	tCER replacement accounts (for unit expiry) cannot acquire ICERs.	Replacement		5/CMP.1, annex, paragraph 44
5215	ICER Replacement Units (upon Reversal of Storage or Lack of Certification Report)	Transaction-specific	ICER Replacement accounts (for Reversal of Storage) may not acquire tCERs and may not acquire ICERs with a Project Number other than that specified in the replacement notification.	Replacement		5/CMP.1, annex, paragraphs 49, 50

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5216	Replacement Notification upon tCER Expiry	Transaction-specific	If provided, the Replacement Notification ID must be valid and must be for replacement upon tCER expiry.	Replacement	Replacement may occur before the ITL sends a notification to the registry indicating that 30 days remain before expiry; however, if a notification ID is submitted, it should be of the right type	5/CMP.1, annex, paragraphs 54, 55
5217	Replacement Notification upon ICER Expiry	Transaction-specific	If provided, the Replacement Notification ID must be valid and must be for replacement upon ICER expiry.	Replacement	Replacement may occur before the ITL sends a notification to the registry indicating that 30 days remain before expiry; however, if a notification ID is submitted, it should be of the right type	5/CMP.1, annex, paragraphs 54, 55
5218	Replacement Notification for Reversal of Storage	Transaction-specific	A valid Notification ID must be provided for replacement upon reversal of storage for a CDM project.	Replacement	Required to enable the ITL to monitor replacement against notifications	5/CMP.1, annex, paragraphs 49, 54
5219	Replacement Notification for Lack of Certification Report	Transaction-specific	A valid Notification ID must be provided for replacement upon a non-submission of Certification Report for a CDM project.	Replacement	Required to enable the ITL to monitor replacement against notifications	5/CMP.1, annex, paragraphs 50, 54
5220	Project ID for ICERs Replacement (upon Reversal of Storage or lack of Certification Report)	Transaction-specific	For ICER replacement transactions upon Reversal of Storage or lack of a Certification Report, the Project ID for the ICERs to be replaced must be consistent with the Project ID contained in the replacement notification.	Replacement	Required to enable the ITL to monitor replacement against notifications	5/CMP.1, annex, paragraphs 49, 50, 54

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5251	National Registry Retirement	Transaction-specific	The Initiating Registry retiring units must be a national registry.	Retirement	The CDM registry cannot retire units	13/CMP.1, annex, paragraphs 13, 34
5252	Retirement Account	Transaction-specific	The Acquiring Account for a retirement transaction must be a retirement account.	Retirement		13/CMP.1, annex, paragraph 34
5253	Retirement Account Identifier	Transaction-specific	Account identifiers must be provided for acquiring accounts in retirement transactions.	Retirement	The ITL normally checks only the account type; in the case of retirement, the ITL checks that the account identifier matches an account specified for this purpose by the registry	24/CP.8, annex, paragraph 7  13/CMP.1, annex, paragraph 22
5254	Retirement Account Commitment Period	Transaction-specific	The Unit Blocks retired must have the same Applicable Commitment as the Retirement Account.	Retirement	Required to enable the ITL to monitor retirement requirements for the commitment period	13/CMP.1, annex, paragraph 21(f)
5255	CER, tCER and ICER Retirement Eligibility	Transaction-specific	The Party of the Initiating Registry must be determined to meet eligibility criteria 1 through 6.	Retirement	All eligibility criteria must be met by the Annex B Party before its registry may conduct retirements of CERs, tCERs or ICERs; the ITL draws upon records in the C&A database concerning each Party's fulfilment of the criteria	3/CMP.1, annex, paragraph 31  13/CMP.1, annex, paragraph 42(d)



Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5256	tCER and ICER Retirement Limit	Transaction-specific	tCER and ICER retirement must not exceed allowed quantity.	Retirement	The allowed quantity of tCER and ICER retirement is based on the level of allowed AAUs issuance by the Annex B Party for the commitment period; the ITL draws this quantity from the C&A database	5/CMP.1, annex, paragraph 51  13/CMP.1, annex, paragraph 31  16/CMP.1, annex, paragraph 14
5257	Eligibility to retire CERs, ICERs and tCERs for the second commitment period	Transaction-specific	Only parties with second commitment period quantified emission limitation and reduction commitment and for which the Doha amendment to the art.3 para 9 of the Kyoto Protocol has entered into force are authorized to retire CER, ICER, tCER	Retirement	This check applies to CER, ICER and tCER valid for any commitment period. This check doesn't apply to other units	1/CMP.8 (Doha Amendment), paragraph 14
5301	National Registry Carry-over	Transaction-specific	The Initiating Registry carrying over units must be a national registry.	Carry-over	The CDM registry cannot carry over units	13/CMP.1, annex, paragraphs 15, 36
5302	Holding Account Carry-Over	Transaction-specific	The Initiating Account for a carry-over transaction must be a holding account.	Carry-over	Units cannot be carried over if they are cancelled, used in replacement, or retired	
5303	Subsequent Commitment Period	Transaction-specific	Units may only be carried-over to the commitment period referred to in the carry-over notification.	Carry-over	It is not allowed to "skip" commitment periods in carry-over	13/CMP.1, annex, paragraphs 15, 36

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5304	Units Available for Carry-over	Transaction-specific	The quantity of units carried-over must not exceed the limit for carry-over established by the Compliance Committee for the Party and reported to the registry in the Unit Carry-over notification.	Carry-over	Only AAUs, ERUs (converted from AAUs) and CERs may be carried over to the next commitment period; these quantities must take account of the level of carry-over allowed for ERUs and CERs; the quantities of units available for carry-over, determined through the reporting, review and compliance procedures, are to be stored in the C&A database and contained in the final compilation and reporting report for each Annex B Party	13/CMP.1, annex, paragraphs 15, 62 (c)
5305	RMU Carry-over	Transaction-specific	RMUs may not be carried over.	Carry-over		13/CMP.1, annex, paragraph 16
5306	ERU (from RMUs) Carry-over	Transaction-specific	ERUs converted from RMUs may not be carried over.	Carry-over		13/CMP.1, annex, paragraph 15 (a)
5307	ICER or tCER Carry-over	Transaction-specific	tCERs or ICERs may not be carried over.	Carry-over		5/CMP.1, annex, paragraphs 41-45
5310	Notification ID for Carry-over	Transaction-specific	Units may only be carried over if a notification has been received from the ITL and this ID is reported in the transaction.	Carry-over	Required to enable the ITL to monitor carry-over against notifications	

Response Code	Check Name	Check Category	Check Description	Transaction Type	Additional Notes	Reference to decision
5312	ERU Carry-over Quantity	Transaction-specific	The quantity of ERUs (which have not been converted from RMUs) carried-over by the Party must not exceed 2.5% of that Party's allowed quantity of AAU issuance for the Commitment Period.	Carry-over		<a href="#">13/CMP.1, annex, paragraph 15 (a)</a>
5313	CER Carry-over Quantity	Transaction-specific	The quantity of CERs carried-over by the Party must not exceed 2.5% of that Party's allowed quantity of AAU issuance for the Commitment Period.	Carry-over		<a href="#">13/CMP.1, annex, paragraph 15 (b)</a>
5453	Notification ID for Expiry Date Change	Transaction-specific	An Expiry Date Change may only take place if it is pursuant to a valid Expiry Date Change notification (Type 9) from the ITL.	Expiry Date Change		
5454	Units for Expiry Date Change Referenced by Notification	Transaction-specific	The units for Expiry Date change must be the units referenced in the ITL Expiry Date Change notification (Type 9).	Expiry Date Change		
5455	Expiry date must match the one of the notification	Transaction-specific	One or more of the units for Expiry Date Change have an expiry date that do not match the expiry date set by the notification the transaction refers to. This message cannot be processed.	Expiry Date Change		

## 5. Acquiring Registry Transaction Response Codes

The following codes are generated by the recipient of an External Transfer.

**Table E 8: Registry Messages**

Response Code	Response Description
5902	Acquiring account does not exist.
5903	Acquiring account is not eligible to receive units.
5904	Transaction inconsistent with Party policy.
5905	Transaction rejected by account holder.
5906	Account has been closed.

## 6. ITL Reconciliation Response Codes

The following codes may be generated by the ITL when processing reconciliation messages.

**Table E 9: ITL Reconciliation Responses**

Response Code	Check Name	Check Description
6201	Reconciliation Identifier	Reconciliation Identifier must be greater than zero.
6202	Reconciliation Mask	Reconciliation ID must be comprised of a valid registry code followed by numeric values.
6203	Reconciliation Status Validity	Reconciliation status must be a value between 1 and 11.
6205	Account Type Validity	Account Type must be valid.
6206	Unit Type Validity	Unit Type Code must be valid.
6207	Supplementary Unit Type Validity	Supplementary Unit Type Code must be valid.
6301	Reconciliation ID Does Not Exist	Reconciliation ID must exist in the Reconciliation Log table.
6311	Reconciliation ID Sent by STL Does Not Exist	Reconciliation ID sent by the STL must already exist in the ITL.
6312	Reconciliation Status Not Valid	Reconciliation status sent by the STL must be one of certain enumerated statuses.

Response Code	Check Name	Check Description
6313	Reconciliation Status of "STL Totals Inconsistent" is Out of Sequence	If the incoming reconciliation status is "STL Totals Inconsistent," the previously recorded status at the ITL must be "Validated."
6314	Reconciliation Status of "STL Unit Blocks Inconsistent" Out of Sequence	If the incoming reconciliation status is "STL Unit Blocks Inconsistent," the previously recorded status at the ITL must be "STL Totals Inconsistent."
6315	Reconciliation status of "STL Validated" is out of sequence.	If the incoming reconciliation status is "STL Validated," the previously recorded status at the ITL must be "Validated," "STL Totals Inconsistent," or "STL Unit Blocks Inconsistent."
6316	Reconciliation Status of "STL Complete with Manual Intervention" is Out of Sequence	If the incoming reconciliation status is "STL Complete with Manual Intervention," the previously recorded status at the ITL must be "STL Totals Inconsistent," or "STL Unit Blocks Inconsistent."
6420	Account Type/Unit Type Unit Blocks	The registry and ITL unit blocks for a specified account type commitment period and unit type must be consistent.
6430	Account Type/Unit Type Unit Blocks Unexpected Consistency	If the totals have failed in the previous stage, the Unit Block compare by account type, commitment period, and unit type must also fail.
6600	Successful Reconciliation of Totals	The reconciliation has been completed with a successful reconciliation of unit totals.

## 7. EUTL Checks and Messages

The EUTL checks enforce EU ETS policy and the response codes and descriptions are described in Table XII-1: Community Independent Transaction Log Response Codes in the COMMISSION REGULATION (EC) No 2216/2004 of 21 December 2004 as amended by COMMISSION REGULATION (EC) No 916/2007 of 31 July 2007. Registries that are not members of the EU ETS may receive these response codes if transacting with members of the EU ETS. The response codes will always be in the range 7000 to 7999.

## 8. ITL Administrative Response Codes

Table E 10: ITL Administrative Responses

Response Code	Check Name	Check Description
8001	Transaction Not Found	The transaction number for which a status was requested must be present in the ITL Transaction Log.
8002	Transaction Clean-up	For ongoing transactions, a message must be received within 24 hours of the last transaction.

## 9. Obsolete Response Codes

The table below describes the checks that are no longer used and the version of Annex E in which they become obsolete.

**Table E 11: Obsolete Response Codes**

Response Code	Check Name	Check Description	Comments	Version in which it became obsolete
1513	Allowed Acquiring Registry Action	Only the acquiring registry can accept or reject an external transaction	This check has become redundant, as its function is subsumed by checks 3023 and 3025	1.1.8
1514	Allowed Initiating Registry Action	Only the initiating registry can complete or terminate its transaction.	This check is no longer applicable with the new message flow introduced in DES 1.1.6.	1.1.8
1515	Transaction Status Not Compatible with an STL	An STL cannot check transactions which were not submitted to it	Incorrect initial categorisation. Should have been a message sequence check. Moved to 3503 in this table.	1.1.8
2005	Transaction Status DateTime	Data Integrity	Cannot be implemented given the current data format	1.1.3
2009	Notification Type Code	Notification Type Code must be valid.	Notification Type Code is not passed in any message so this check cannot be performed.	1.1.001
3001	Transaction ID Not Unique	Transaction ID for proposed transactions must not already exist in the ITL.	<p>Transactions, when proposed by a registry in the first message of a transaction sequence, must use an identifier which has never been used</p> <p>24/CP.8, annex, paragraph 17</p>	1.1.5

Response Code	Check Name	Check Description	Comments	Version in which it became obsolete
3002	Prior Record of Transaction ID from Registry/STL	Transaction ID for ongoing transactions must already exist in the ITL.	If the message is not the first in a transaction sequence, it cannot be a transaction proposal : the transaction ID must already be known to the ITL  24/CP.8, annex, paragraphs 6, 7, 17	1.1.5
3003	Transaction Status Out of Sequence for Prior Completed Status	Previous completed transactions cannot be completed again.	The fact that a transaction has been completed is indicated by a "completed" status for the transaction  24/CP.8, annex, paragraphs 6, 7	1.1.5  No longer applicable after ITL 2.x (DES 1.1.6)
3004	Transaction Status Out of Sequence for Prior Rejected Status	Previously rejected transactions cannot be completed.	The fact that a transaction has been rejected by the acquiring registry is indicated by a "rejected" status for the transaction (only for external transfers)  24/CP.8, annex, paragraphs 6, 7	1.1.5  No longer applicable after ITL 2.x (DES 1.1.6)
3005	Transaction Status Out of Sequence for Prior ITL Discrepancy Status	Transactions for which an ITL discrepancy has been previously identified cannot be completed.	This condition is shown by an "Checked discrepancy" status for the transaction  24/CP.8, annex, paragraphs 6, 7	1.1.5  No longer applicable after ITL 2.x (DES 1.1.6)
3006	Transaction Status Out of Sequence for Prior STL Discrepancy Status	Transactions for which an STL discrepancy has been previously identified cannot be completed.	This condition is shown by an "STL Checked - discrepancy" status for the transaction	1.1.5  No longer applicable after ITL 2.x (DES 1.1.6)

Response Code	Check Name	Check Description	Comments	Version in which it became obsolete
3007	Transaction Status Out of Sequence for Prior Terminated Status	Previously terminated transactions cannot be completed.	This condition is shown by a “terminated” status for the transaction; it refers to a transaction terminated by the initiating registry  24/CP.8, annex, paragraphs 6, 7	1.1.5  No longer applicable after ITL 2.x (DES 1.1.6)
3008	Transaction Status Out of Sequence for Prior Cancelled Status	Previously cancelled transactions cannot be completed.	This condition is shown by a “cancelled” status for the transaction; it refers to a transaction cancelled by the ITL (e.g. after no response to a message was received after 24 hours)  24/CP.8, annex, paragraphs 6, 7	1.1.5  No longer applicable after ITL 2.x (DES 1.1.6)
3009	Transaction Status Out of Sequence for Prior Accepted Status	Previously accepted external transactions cannot be terminated.	Once the ITL has found no discrepancy and the acquiring Party has accepted the proposed transfer, the transaction can no longer be terminated by the initiating registry  24/CP.8, annex, paragraphs 6, 7	1.1.5  No longer applicable after ITL 2.x (DES 1.1.6)
3010	Transaction Status Out of Sequence for Accepted or Rejected Status	Transaction status of Accepted or Rejected is not valid for non-external transactions.	The status codes of “accepted” and “rejected” are only applicable to external transfers; they may only be applied by an acquiring registry  24/CP.8, annex, paragraphs 6, 7	1.1.5  No longer applicable after ITL 2.x (DES 1.1.6)



Response Code	Check Name	Check Description	Comments	Version in which it became obsolete
3011	Transaction Status Not Compatible with Initiating Registry	Transaction status from Initiating Registry must indicate status of Proposed, Completed, or Terminated.	Only the status codes of “proposed”, “completed” or “terminated” may be applied by an initiating registry; the other statuses are applied by acquiring registries, the ITL or an STL  24/CP.8, annex, paragraphs 6, 7	1.1.5  No longer applicable after ITL 2.x (DES 1.1.6)
3012	Transaction Status Not Compatible with Acquiring Registry	Transaction status from Acquiring Registry must indicate status of Rejected or Accepted.	Only the status codes of “accepted” or “rejected” may be applied by an acquiring registry; the others are applied by initiating registries, the ITL or an STL  24/CP.8, annex, paragraphs 6, 7	1.1.5  No longer applicable after ITL 2.x (DES 1.1.6)
3013	Transaction Status Out of Sequence for Prior Completed, Cancelled, or Terminated Status	Previously Completed, Cancelled, or Terminated transactions cannot have their status changed by subsequent notifications.		1.1.5  No longer applicable after ITL 2.x (DES 1.1.6)
3014	Transaction Status (accepted) Out of Sequence for Prior Rejected Status	Previously rejected transactions cannot be accepted.		1.1.5  No longer applicable after ITL 2.x (DES 1.1.6)
3015	Transaction Status (accepted) Out of Sequence for Prior ITL Discrepancy Status	Transactions for which an ITL discrepancy has been previously identified cannot be accepted or rejected.		1.1.5

Response Code	Check Name	Check Description	Comments	Version in which it became obsolete
3016	Transaction Status (accepted) Out of Sequence for Prior STL Discrepancy Status	Only transactions that have been validated by STLs as without discrepancy, and yet to be completed, terminated, or cancelled, may be accepted or rejected by the acquiring registry		1.1.5  No longer applicable after ITL 2.x (DES 1.1.6)
3017	Transaction Status Out of Sequence for STL checking	A transaction can only be checked for discrepancy by an STL once.		1.1.5  No longer applicable after ITL 2.x (DES 1.1.6)
3019	Transaction Status (Completed) Out of Sequence for STL checking	A transaction involving an STL registry cannot be completed if it was not approved by the STL.		1.1.5  No longer applicable after ITL 2.x (DES 1.1.6)
3020	Non-involved Registry	Only the transaction's transferring and acquiring registries can send and <i>acceptNotification</i> message related to the transaction		
3021	Non-involved STL	An STL can only validate transactions for which the registries are its members.	For example, the EUTL can only validate transactions involving ETS registries	
3022	Invalid Transaction Status	The transaction status contained in the <i>acceptNotification</i> is illegal for the incoming message.		
3023	Invalid Transaction Status for Transferring Registry	The transaction status contained in the <i>acceptNotification</i> message is illegal when coming from the transferring registry.		
3024	Invalid Transaction Status for Acquiring Registry	The transaction status contained in the <i>acceptNotification</i> message is illegal when coming from the acquiring registry.		

Response Code	Check Name	Check Description	Comments	Version in which it became obsolete
3025	Invalid Transaction Status for STL	The transaction status contained in the <i>acceptNotification</i> message is illegal when coming from the STL.		
3026	Accepted/rejected Status Only for External Transfers	AcceptNotification messages from registries containing a status of <i>accepted</i> or <i>rejected</i> is invalid for single registry transactions.		
3027	Out of Sequence Completed Message From Transferring Registry	The current transaction status in the ITL is inconsistent with an <i>acceptNotification</i> message from the transferring registry containing a status of <i>completed</i> .		
3028	Out of Sequence Terminated Message From Transferring Registry	The current transaction status in the ITL is inconsistent with an <i>acceptNotification</i> message from the transferring registry containing a status of <i>terminated</i> .		
3029	Out of Sequence Accepted/Rejected Message	The current transaction status in the ITL is inconsistent with an <i>acceptNotification</i> message containing a status of <i>accepted</i> or <i>rejected</i> .		
3030	Out of Sequence STL Check (discrepancy/no discrepancy) Message From STL	The current transaction status in the ITL is inconsistent with an <i>acceptNotification</i> message from the STL containing a status of <i>STL Check (discrepancy/no discrepancy)</i> .		
3031	Out of Sequence Completed Message From Acquiring Registry	The current transaction status in the ITL is inconsistent with an <i>acceptNotification</i> message from the acquiring registry containing a status of <i>completed</i> .		

Response Code	Check Name	Check Description	Comments	Version in which it became obsolete
3032	Out of Sequence Completed Message From STL	The current transaction status in the ITL is inconsistent with an <i>acceptNotification</i> message from the STL containing a status of <i>completed</i> .		
3033	Out of Sequence Terminated Message from STL	The current transaction status in the ITL is inconsistent with an <i>acceptNotification</i> message from the STL containing a status of <i>terminated</i> .		
3501	Transaction Status Not Compatible with an STL	Transaction status from STL must indicate status of Discrepancy or No Discrepancy.	Only the status codes of “discrepancy” or “no discrepancy” may be applied by an STL; the others are applied by a registry or the ITL  24/CP.8, annex, paragraphs 6, 7	1.1.5
3502	Prior record of Transaction ID from STL	Transaction ID for ongoing transactions must exist in ITL.	Any messages received from an STL cannot be the first in a transaction sequence; the transaction ID must already be known to the ITL  24/CP.8, annex, paragraphs 7 (e), 17  This code is redundant, and is covered by response code 3002.	1.1.5
3503	Transaction Status Not Compatible with an STL	An STL cannot check transactions which were not submitted to it.	Originally, this was the incorrectly categorised response code 1515. It has not been recategorised, and renumbered to 3503.	1.1.8

<b>Response Code</b>	<b>Check Name</b>	<b>Check Description</b>	<b>Comments</b>	<b>Version in which it became obsolete</b>
4001	Applicable Commitment Period	Applicable Commitment Period must correspond to the current or next Commitment Period (including their true-up periods).	The check is superfluous given the timing of actions is governed by events external to the ITL, such as review processes. Removal of the check does not reflect a change in policy.	1.1.001
4006	Acquiring and Transferring Registry Consistency	For all transactions except for external transfers, the Initiating and Acquiring Registries must be the same.	Replaced by 2026 to include Issuance transactions.	1.1.001
4009	Units Have STL Inconsistencies	General Transaction	The unit status codes indicating an inconsistency flagged by a STL during reconciliation has been discarded rendering this response code unimplementable.	1.1.3
5208	Single Replacement Registry	Transaction-specific	Redundant with Check 4003	1.1.3
5210	One-To-Many Replacement Units	Transaction-specific	A transaction cannot contain many-to-many relationships between replaced and replacing blocks.	1.1.6
6204	Reconciliation Snapshot DateTime	Reconciliation snapshot must be a date between 01-OCT-2004 and the current date plus 30 days.	Messages sent to the ITL don't contain the snapshot date time - there is no possibility to return this response code to a registry	1.1.3
6302	Reconciliation Status Not Valid	Reconciliation status sent by registry must be valid.	No reconciliation status is sent by the registry – the check can't be implemented by the ITL	1.1.3

<b>Response Code</b>	<b>Check Name</b>	<b>Check Description</b>	<b>Comments</b>	<b>Version in which it became obsolete</b>
6303	Reconciliation Status Out of Sequence	Incoming reconciliation status must be the same as the reconciliation status recorded by the ITL.	No reconciliation status is sent by the registry – the check can't be implemented by the ITL	1.1.3
6304	Consistent Reconciliation Snapshot DateTime	The registry reconciliation snapshot DateTime must be consistent with the ITL Reconciliation Snapshot DateTime.	Not applicable: incoming reconciliation messages do not have a snapshot date time	1.1.3
6410	Account Type/Unit Type Totals	The totals for account types, commitment period, and unit types must be consistent.	No reconciliation status is sent by the registry – the check can't be implemented by the ITL	1.1.3
6440	Snapshot DateTime Validity	The DateTime for reconciliation action proposed by the STL must be in the future.	As the ITL is responsible for scheduling snapshots, these are always in the future and there is no possibility to return this response code to a registry	1.1.3
6450	Ongoing Reconciliation	A reconciliation action cannot be initiated at the registry because there is already an ongoing action.	There is no possibility for a registry to return this response code to the ITL.	1.1.3

## **Annex F Definition of Identifiers**

### **1. Introduction**

This annex provides information on the required structure of identification numbers for the core entities associated with data exchange.

### **2. General Rules for XML Formats**

Identifiers shall be transmitted in XML format as an element tag comprised of attributes (components of the identifier). For numeric elements, leading zeros should not be included as place holders for a number shorter than the maximum length. For components which do not apply (such as Project ID for an AAU), an attribute is not required.

### **3. Recommended Display and Report Formats**

For display and reporting, it is recommended that each element of the identifier be separated by a single dash "-" and no spaces. For example, a transaction number would be represented as follows: NZ-132-1. Leading zeros would not be displayed. Please note that the separating dash is not included in the XML structure for identifiers and should not be stored data. Consistent with the requirements below, all serial numbers shall be stored as elements.

### **4. Serial Numbers**

The serial number of a unit shall be unique throughout all registries and the ITL. Serial numbers are defined only by registries.

Whenever possible, a set of units shall be transmitted as a unit block defined by the starting block number and the ending block number. Within a unit block, every element of the serial number must be identical except for the unique number element. Number elements within a block must be complete and consecutive.

When necessary to perform a transaction, track, record, or otherwise characterize a unit or unit block, registries or the ITL shall create multiple unit blocks from a single unit block.

Although each unit is identified uniquely by its originating registry code and its assigned unique number and this identification should be used by registries, communication about unit(s) will conform to the Unit Block definition requirements below. For a single unit the start and end block elements contain the same value.

**Figure F1: Serial Number Identifiers**

Identifier	Display Order	Identifier Required for the Following Unit Types	Data Type	Length	Range or Codes
Originating Registry	1	All	Alphanumeric	3	Two-letter country codes in ISO3166, as of 01 January 2005
Unit Type	2	All	Numeric	2	1 = AAU 2 = RMU 3 = ERU converted from AAU 4 = ERU converted from RMU 5 = CER 6 = tCER 7 = ICER
Supplement-ary Unit Type	3		Numeric	2	Blank for Kyoto-only Units, or as defined by STL
Unit Serial Block Start	4	All	Numeric	15	Unique numeric values assigned by registry from 1 - 999,999,999,999,999
Unit Serial Block End	5	All	Numeric	15	Unique numeric values assigned by registry from 1 - 999,999,999,999,999
Original Commitment Period	6	All	Numeric	2	1 - 99
Applicable Commitment Period	7	All	Numeric	2	1 - 99
LULUCF Activity	8	RMU, ERU (converted from an RMU), tCER, ICER	Numeric	3	1 = Afforestation and reforestation 2 = Deforestation 3 = Forest management 4 = Cropland management 5 = Grazing land management 6 = Revegetation 7 = Wetland drainage and rewetting
Project Identifier	9	ERU, CER, ICER, tCER	Numeric	7	Numeric value assigned by registry for Project, unique throughout the registries system (i.e. across all national registries). The Project Number is the combination of the Originating Registry and the Project Identifier.
Track	10	ERU	Numeric	2	1 or 2
Expiry Date	11	ICER, tCER	Date		Expiry Date for tCERs or ICERs



## 5. Account Numbers

The account number shall be unique throughout all registries. An account number for an account that is deactivated or deleted cannot be reused.

Account numbers are created and defined only by registries.

Holding accounts do not have an applicable Commitment Period.

**Figure F2: Account Number Identifiers**

Element	Display	Data Type	Length	Range or Codes
Registry Identifier	1	Alphanumeric	3	Two-letter country codes in ISO3166, as of 01 January, 2005, or CDM for accounts in the CDM Registry, or EU for accounts in the European Community Registry.
Account Type	2	Numeric	3	100 = Holding Account 110 = Pending Account 120 = Operator Holding Account 121 = Person Holding Account 210 = Net Source Cancellation Account 220 = Non-compliance Cancellation Account 230 = Voluntary Cancellation Account 240 = Excess Issuance Cancellation Account 250 = Mandatory Cancellation 260 = Administrative Cancellation 300 = Retirement Account 411 = tCER Replacement Account for Expiry 421 = ICER Replacement Account for Expiry 422 = ICER Replacement Account for Reversal of Storage 423 = ICER Replacement Account for Non-submission of Certification Report
Account Identifier	3	Numeric	15	Unique numeric values assigned by registry from 1 - 999,999,999,999,999
Applicable Commitment Period	4	Numeric	2	0 for all holding accounts 1 - 99 for retirement and cancellation accounts

## 6. Transaction Numbers

The transaction number shall be unique within the registries and within the ITL.

A transaction number for a transaction that is terminated or cancelled cannot be reused. Resubmission of a transfer for which a transaction has been terminated or cancelled shall be assigned a new, unique transaction number.

Transaction numbers are defined only by registries.

**Figure F3: Transaction Numbers**

Identifier	Display	Data Type	Length	Range or Codes
Registry Identifier	1	Alphanumeric	3	Two-letter country codes in ISO3166, as of 01 January, 2005 using EU for the European Community Registry, and CDM for transfers from the pending account in the CDM Registry.
Transaction Identifier	2	Numeric	15	Unique numeric values assigned by registry from 1 - 999,999,999,999,999

## 7. Reconciliation Numbers

The reconciliation number shall be unique throughout all registries and the ITL.

Reconciliation numbers are defined only by the ITL once a reconciliation is confirmed by the ITL for a registry. The reconciliation ends when the ITL determines there are no discrepancies or when a manual intervention to correct inconsistencies is complete. If a resubmission of information is needed because of a data format problem, the same reconciliation number shall be used.

**Figure F4: Reconciliation Numbers**

Identifier	Display Order	Data Type	Length	Range or Codes
Registry Identifier	1	Alphanumeric	3	Two-letter country codes in ISO3166, as of 01 January, 2005 using EU for the European Community Registry and CDM for CDM Registry reconciliation.
Reconciliation Identifier	2	Numeric	15	Unique numeric values assigned by Transaction Log from 1 - 999,999,999,999,999

## 8. Project Numbers

The Project Number shall be unique for all Projects subject to Kyoto Protocol requirements.

Project numbers are defined only by registries or the CDM Executive Board (in cooperation with the CDM Registry Administrator).

**Figure F5: Project Numbers**

Identifier	Display Order	Data Type	Length	Range or Codes
Party Identifier	1	Alphanumeric	3	Two-letter country codes in ISO3166, as of 01 January, 2005
Project Identifier	2	Numeric	7	Numeric value assigned by a registry for a Project. The Project Identifier is unique for all Project Numbers with a given Party Identifier.

## 9. Notification Numbers

The Notification number shall be unique throughout the ITL and Registry Systems.

Notification numbers are defined only by the ITL.

Notification numbers are not registry or Party specific and are numeric values, maximum length 12.

*[This page intentionally left blank.]*

## Annex G List of Codes

Annex G defines the codes for all support tables used in these technical specifications. These codes define the acceptable values for elements defined in Annex F or otherwise required for data exchange.

**Figure G1: List of Tables**

Table	Description
Account Type Code	Identifies the type of unit accounts in a registry.
Commitment Period Code	Identifies the Commitment Period.
LULUCF Activity Code	Identifies Land Use, Land Use Change, and Forestry categories.
Notification Status Code	Identifies status of notification request.
Notification Type Code	Identifies notification type codes.
Party Type Code	Defines the role of the Party.
Reconciliation Status Code	Identifies the status of a reconciliation process.
Transaction Status Code	Identifies the status of a transaction.
Transaction Type Code	Identifies the type of transaction.
Unit Type Code	Identifies the type of unit.

**Figure G2: Account Type Code**

<b>Code</b>	<b>Description</b>
100	Holding Account
110	Pending Account
120	Operator Holding Account
121	Person Holding Account
210	Net Source Cancellation Account (Type 1) (national registries only)
220	Non-compliance Cancellation Account (Type 2) (national registries only)
230	Voluntary Cancellation Account (Type 3) (national registries only)
240	Excess Issuance Cancellation Account (Type 4) (CDM Registry only)
250	Mandatory Cancellation Account (Type 5)
260	Administrative Cancellation Account (Type 6) (CDM Registry only)
300	Retirement Account
411	tCER Replacement Account for Expiry (Type 1)
421	ICER Replacement Account for Expiry (Type 1)
422	ICER Replacement Account for Reversal of Storage (Type 2)
423	ICER Replacement Account for Non-submission of Certification Report (Type 3)

**Figure G3: Commitment Period Code**

<b>Code</b>	<b>Description</b>
0	Supplementary Program Commitment Period (2005-2007)
1	First Commitment Period
2	Second Commitment Period
3	Third Commitment Period
4	Fourth Commitment Period

**Figure G4: LULUCF Activity Codes**

<b>Code</b>	<b>Description</b>
1	Afforestation and reforestation
2	Deforestation
3	Forest management
4	Cropland management
5	Grazing land management
6	Revegetation
7	Wetland drainage and rewetting

**Figure G5: Notification Status Code**

<b>Code</b>	<b>Description</b>
1	Initial
2	Incomplete
3	Complete

**Figure G6: Notification Type Codes**

<b>Code</b>	<b>Description</b>
1	Net Source Cancellation
2	Non-compliance Cancellation
3	Impending Expiry of tCER or ICER
4	Reversal of Storage for CDM Project
5	Non-submission of Certification Report for CDM Project
6	Excess Issuance for CDM Project
7	Commitment Period Reserve
8	Unit Carry-over
9	Expiry Date Change
10	Notification Update
11	EU15 Commitment Period Reserve

**Figure G7: Party Type Codes**

<b>Code</b>	<b>Description</b>
1	Initiating Registry
2	Acquiring Registry

**Figure G8: Reconciliation Status Code**

<b>Code</b>	<b>Description</b>
0	Confirmed
1	Initiated
2	Validated
3	ITL Totals Inconsistent
4	ITL Unit Blocks Inconsistent
5	ITL Completed
6	ITL Completed with Manual Intervention
7	ITL Start Request Denied
8	STL Totals Inconsistent
9	STL Unit Blocks Inconsistent
10	STL Validated
11	STL Completed with Manual Intervention
98	ITL Totals by Account Type Inconsistent
99	ITL Manual Intervention Required



**Figure G9: Transaction Status Code**

<b>Code</b>	<b>Description</b>
1	Proposed
2	Checked (No Discrepancy)
3	Checked (Discrepancy)
4	Completed
5	Terminated
6	Rejected
7	Cancelled
8	Accepted
9	STL Checked (No Discrepancy)
10	STL Checked (Discrepancy)
11	STL Completed
12	TREG Completed
13	AREG Completed
14	ALL Completed
15	STL Terminated
16	ALL Terminated

**Figure G10: Transaction Type Code**

<b>Code</b>	<b>Description</b>
1	Issuance - Initial creation of a unit
2	Conversion - Transformation of a unit to create an ERU
3	External - External Transfer of unit between registries
4	Cancellation - Internal transfer of unit
5	Retirement - Internal transfer of unit
6	Replacement - Replacement of tCER or ICER
7	Carry-over - Change of validity to subsequent Commitment Period
8	Expiry Date Change
10	Internal transfer of a unit/supplementary program transaction

**Figure G11: Unit Type Code**

<b>Code</b>	<b>Description</b>
0	Non-Kyoto Unit
1	AAU - Assigned Amount Unit
2	RMU - Removal Unit
3	ERU - Emission Reduction Unit (converted from an AAU)
4	ERU - Emission Reduction Unit (converted from an RMU)
5	CER - Certified Emission Reduction Unit
6	tCER - Temporary CER
7	ICER - Long-term CER

# Annex H

## Registry Initialization - Functional Test Plan

### Table of Contents

	<u>Page</u>
<b>1. Introduction.....</b>	<b>1</b>
1.1 PURPOSE .....	1
1.2 GUIDING PRINCIPLES AND ASSUMPTIONS .....	1
1.3 PREREQUISITES .....	7
1.3.1 Connectivity .....	7
1.3.2 Documentation Review .....	8
<b>2. Technical Information .....</b>	<b>9</b>
2.1 WEB SERVICE DEFINITIONS .....	9
2.2 CODES .....	9
2.3 SEED DATA REQUIREMENTS .....	9
2.3.1 Accounts .....	9
2.3.2 Registry Identifiers .....	10
2.3.2.1 National Registry Test Cases	10
2.3.2.2 CDM Registry Test Cases	10
2.3.3 Project Identifiers .....	10
2.3.4 Commitment Period Reserve .....	11
2.3.5 Issuance, Conversion and Retirement Limits .....	11
2.3.6 Units Holdings in Third-party Simulators .....	12
<b>3. Discrete Transaction Tests .....</b>	<b>14</b>
3.0 GENERAL INFORMATION .....	14
3.0.1 Confirmation Reconciliation .....	14
3.0.2 Test Parameters .....	14
3.0.3 STL Checks .....	14
3.0.4 Commitment Period .....	14
3.1 ISSUANCE .....	15
3.1.1 Components Tested.....	15
3.1.2 Test Steps .....	15
3.1.3 Test Cases .....	16
3.2 CONVERSION .....	17
3.2.1 Components Tested.....	17
3.2.2 Test Steps .....	17
3.2.3 Test Cases .....	18
3.3 EXTERNAL TRANSFER .....	19
3.3.1 Components Tested.....	19
3.3.2 Test Steps .....	19
3.3.3 Test Cases .....	20
3.4 CANCELLATION .....	22
3.4.1 Components Tested.....	22
3.4.2 Test Steps .....	22
3.4.3 Test Cases .....	22
3.5 RETIREMENT.....	25
3.5.1 Components Tested.....	25
3.5.2 Test Steps .....	26
3.5.3 Test Cases .....	26
3.6 REPLACEMENT.....	28
3.6.1 Components Tested.....	28
3.6.2 Test Steps .....	29
3.6.3 Test Cases .....	30
3.7 CARRY-OVER.....	34
3.7.1 Components Tested.....	34
3.7.2 Test Steps .....	34

3.7.3	Test Cases .....	35
3.8	EXPIRY DATE CHANGE .....	37
3.8.1	Components Tested.....	37
3.8.2	Test Steps.....	38
3.8.3	Test Cases.....	38
<b>4.</b>	<b>Test Cases (National Registry) .....</b>	<b>41</b>
4.0	GENERAL INFORMATION .....	41
4.0.1	Confirmation Reconciliation .....	41
4.0.2	Test Parameters .....	41
4.0.3	STL Checks .....	41
4.0.4	Commitment Period .....	41
4.1	ISSUANCE TRANSACTION TESTS .....	42
4.1.1	Components Tested.....	42
4.1.2	Test Steps.....	42
4.1.3	Test Cases.....	43
4.2	CONVERSION TRANSACTION TESTS .....	45
4.2.1	Components Tested.....	45
4.2.2	Test Steps.....	46
4.2.3	Test Cases.....	46
4.3	EXTERNAL TRANSFER TRANSACTION TESTS.....	48
4.3.1	Components Tested.....	48
4.3.2	Test Steps.....	49
4.3.3	Test Cases.....	50
4.4	CANCELLATION TRANSACTION TESTS.....	61
4.4.1	Components Tested.....	61
4.4.2	Test Steps.....	61
4.4.3	Test Cases.....	62
4.5	EXPIRY DATE CHANGE TRANSACTION TESTS .....	66
4.5.1	Components Tested.....	66
4.5.2	Test Steps.....	66
4.5.3	Test Cases.....	67
4.6	RETIREMENT TRANSACTION TESTS .....	71
4.6.1	Components Tested.....	71
4.6.2	Test Steps.....	71
4.6.3	Test Cases.....	72
4.7	REPLACEMENT TRANSACTION TESTS .....	76
4.7.1	Components Tested.....	76
4.7.2	Test Steps.....	76
4.7.3	Test Cases.....	77
4.8	ADDITIONAL TESTS .....	85
4.8.1	Components Tested.....	85
4.8.2	Test Steps.....	86
4.8.3	Test Cases.....	87
<b>5.</b>	<b>EU ETS Member Registry Test Cases .....</b>	<b>97</b>
5.0	GENERAL INFORMATION .....	97
5.0.1	Test Sequence .....	97
5.0.2	Applicable Commitment Period .....	97
5.1	ACCOUNT MANAGEMENT WEB SERVICES .....	97
5.1.1	Components Tested.....	97
5.1.2	Test Steps.....	97
5.1.3	Test Cases.....	97
5.2	INTERNAL TRANSFER TRANSACTION TESTS.....	100
5.2.1	Components Tested.....	100
5.2.2	Test Steps.....	100
5.2.3	Test Cases.....	100
<b>6.</b>	<b>Test Cases (CDM Registry) .....</b>	<b>108</b>
6.0	GENERAL INFORMATION .....	108
6.0.1	Confirmation Reconciliation .....	108

6.0.2	Test Parameters .....	108
6.0.3	Commitment Period .....	108
6.1	ISSUANCE TRANSACTION TESTS .....	108
6.1.1	Components Tested.....	108
6.1.2	Test Steps.....	109
6.1.3	Test Cases.....	109
6.2	INTERNAL TRANSFER TRANSACTION TESTS.....	114
6.2.1	Components Tested.....	114
6.2.2	Test Steps.....	114
6.2.3	Test Cases.....	115
6.3	EXTERNAL TRANSFER TRANSACTION TESTS.....	126
6.3.1	Components Tested.....	126
6.3.2	Test Steps.....	126
6.3.3	Test Cases.....	128
6.4	CANCELLATION TRANSACTION TESTS.....	135
6.4.1	Components Tested.....	135
6.4.2	Test Steps.....	135
6.4.3	Test Cases.....	137
6.5	EXPIRY DATE CHANGE TRANSACTION TESTS .....	143
6.5.1	Components Tested.....	143
6.5.2	Test Steps.....	143
6.5.3	Test Cases.....	144
6.6	ADDITIONAL TESTS .....	147
6.6.1	Components Tested.....	147
6.6.2	Test Steps.....	147
6.6.3	Test Cases.....	148
<b>7.</b>	<b>Annex H Testing for CP2 .....</b>	<b>153</b>
<b>7.1</b>	<b>Introduction.....</b>	<b>154</b>
7.1.1	PURPOSE.....	154
7.1.2	GUIDING PRINCIPLES AND ASSUMPTIONS.....	154
7.1.2.1	Principles and Scope .....	154
7.1.2.2	Assumptions .....	154
7.1.2.3	Test Coverage .....	155
7.1.3	PREREQUISITES .....	156
<b>7.2.</b>	<b>CP2 Seed Data, Conventions and Execution .....</b>	<b>157</b>
7.2.1	SEED DATA SPECIFICATIONS.....	157
7.2.1.1	Registry Accounts.....	157
7.2.1.2	Registry and ITL Limits and CPR .....	157
7.2.1.3	Third-Party Registries .....	158
7.2.1.4	Jl and CDM Projects .....	160
7.2.1.5	ITL Configuration Settings .....	162
7.2.2	TEST SCENARIO AND TEST CASE CONVENTIONS.....	162
7.2.2.1	Registry ZZ .....	162
7.2.2.2	Test Scenarios and Test Cases.....	162
7.2.3	TEST EXECUTION .....	163
7.2.3.1	General .....	163
7.2.3.2	Reconciliation Tests.....	163
<b>7.3.</b>	<b>Test Scenarios .....</b>	<b>164</b>
7.3.1	SCENARIO 1 – ISSUANCE AND EXTERNAL TRANSFER OF AAUS AND RMUS .....	164
7.3.1.1	Overview .....	164
7.3.1.2	Assumptions .....	164
7.3.1.3	Test Cases.....	164
7.3.2	SCENARIO 2 – CONVERSION .....	168
7.3.2.1	Overview .....	168
7.3.2.2	Assumptions .....	168
7.3.2.3	Test Cases.....	168
7.3.3	SCENARIO 3 – CANCELLATION UNRELATED TO CDM PROJECTS .....	171
7.3.3.1	Overview .....	171

7.3.3.2	Assumptions .....	171
7.3.3.3	Test Cases .....	171
7.3.4	SCENARIO 4 – CANCELLATION RELATED TO CDM PROJECTS .....	174
3.4.1	Overview .....	174
7.3.4.2	Assumptions .....	174
7.3.4.3	Test Cases .....	174
7.3.5	SCENARIO 5 – CDM NOTIFICATIONS .....	177
7.3.5.1	Overview .....	177
7.3.5.2	Assumptions .....	177
7.3.5.3	Test Cases .....	177
7.3.6	SCENARIO 6 – END OF CP2 .....	181
7.3.6.1	Overview .....	181
7.3.6.2	Assumptions .....	181
7.3.6.3	Test Cases .....	181
7.3.7	SCENARIO 7 –COMPLEX RECONCILIATION.....	185
7.3.7.1	Overview .....	185
7.3.7.2	Assumptions .....	185
7.3.7.3	Test Cases .....	185
7.3.8	SCENARIO 8 – NON-FUNCTIONAL AND EXTRAORDINARY CASES .....	188
7.3.8.1	Overview .....	188
7.3.8.2	Assumptions .....	188
7.3.8.3	Test Cases .....	188

## **11 1. Introduction**

### **1.1 Purpose**

This annex provides the specification for the functional testing component of registry initialization. A registry must successfully complete these test cases prior to being authorized to submit transactions to the ITL in a production environment. The ITL Administrator, drawing on operational experience, may require additional or modified tests in order to verify a registry is able to interoperate with the ITL as required by the DES. This annex specifies each test to be conducted, evaluated, and recorded by the ITL and registry.

The successful completion of this initialization functional test plan verifies that a registry's implementation of the Data Exchange Standards is sufficient to allow interoperation with the ITL.

Specifications for testing for Commitment Period 2 (CP2) are fully contained in Section 7 of this document.

### **1.2 Guiding Principles and Assumptions**

Initialization is defined between a registry and the ITL. Testing between the ITL and EUTL will occur independently of Annex H initialization. The test cases specified in this annex evaluate the performance of the registry and its ability to interact with the ITL. Third-party registries and required STLs will be simulated and administrated by the initialization coordinator. Functional testing for initialization focuses on one registry at a time to isolate sources of variation.

The following functional components of the registry will be tested to ensure the Data Exchange Standards have been implemented properly and the registry will perform as expected in a production environment:

- Every Web service defined by the DES will be tested;
- Every Kyoto transaction will be tested;
- Every transaction flow described in behavior diagrams contained in the DES technical specification will be tested;
- A registry's ability to receive and act upon notifications received from the ITL will be tested for every notification type defined by the DES (Section 6); and
- A registry's ability to respond to and act upon ITL administrative requests will be tested.

**Figure 1.1  
Functional Component-National Registry Test Mapping**

<b>ITL Web Service</b>	AcceptMessage	801
	AcceptProposal	100, 101, 102, 200, 201, 300, 301, 304, 305, 306, 307, 308, 309, 310, 400, 401, 500, 501, 600, 700, 701, 702, 703, 704, 800, 802, 803, 804, 805, 806, 893, 1200, 1201, 1202, 1292
	AcceptNotification	100, 101, 102, 200, 201, 300, 301, 303, 304, 305, 306, 307, 308, 309, 310, 400, 401, 500, 501, 600, 700, 701, 702, 703, 704, 800, 802, 803, 804, 805, 806, 893, 1200, 1201, 1292
	GetTransactionStatus	308
	ReceiveReconciliationResult	893, 1292
	ProvideAuditTrail	893, 1292
	ProvideTotals	1291, 1292
	ProvideUnitBlocks	1292, 1292
	ReceiveAuditTrail	893, 1292
	ReceiveTotals	191, 291, 391, 491, 591, 691, 791, 891, 893, 1291, 1292
	ReceiveUnitBlocks	191, 291, 391, 491, 591, 691, 791, 891, 893, 1291, 1292
<b>Registry Web Services</b>	ProvideTime	190, 290, 390, 490, 590, 690, 790, 890, 1290
	AcceptMessage	800
	AcceptProposal	300, 301, 302, 304, 305, 310, 704, 894, 1292
	AcceptNotification	100, 101, 102, 200, 201, 303, 306, 307, 3009, 400, 401, 500, 501, 600, 700, 701, 702, 703, 800, 802, 803, 804, 805, 806, 1200, 1201
	AcceptITLNotification	303, 400, 401, 500, 501, 700, 701, 702, 703, 704, 800, 805
	InitiateReconciliation	191, 291, 391, 491, 591, 691, 791, 891, 892, 1291, 1292

(cont.)



<b>Registry Web Services (cont.)</b>	ReceiveReconciliationResult	191, 291, 391, 491, 591, 691, 791, 891, 892, 1291, 1292
	ProvideAuditTrail	892, 1292
	ProvideTotals	191, 291, 391, 491, 591, 691, 791, 891, 892, 1291, 1292
	ProvideUnitBlocks	191, 291, 391, 491, 591, 691, 791, 891, 892, 1291, 1292
<b>EUTL-Related (Account Management) Web Services</b>	ReceiveAccountOperation OutcomeRequest	1104
<i>On Registry, called by ITL</i>	CreateAccountRequest	1100
	ReceiveAccountOperation Outcome	1100, 1101, 1102, 1103
<i>On ITL, called by registry or EUTL</i>	UpdateAccountRequest	1101
	CloseAccountRequest	1102
	UpdateVerifiedEmissions Request	1103
<b>DES Behavior Diagrams</b>	Figure 4.5: Single Registry Transaction Behavior Diagram	100, 101, 102, 200, 201, 400, 401, 500, 501, 600, 700, 701, 702, 703, 800, 802, 803, 804, 805, 806, 1200, 1201, 1202
	Figure 4.6: Discrepancy Notification Sequence Diagram	102, 1201
	Figure 4.7: Terminate Transaction Sequence Diagram	102, 1201
	Figure 4.9: External Transfer Behavior Diagram	300, 301, 302, 303, 304, 305, 306, 307, 308, 309, 310, 704, 892, 1292
	Figure 4.10: Discrepancy Notification Sequence Diagram	
	Figure 4.11: Terminate External Transaction Sequence Diagram	307, 309
	Figure 5.1: Reconciliation Behavior Diagram	191, 291, 391, 491, 591, 691, 791, 891, 892, 1291, 1292
	Figure 5.2: Send Reconciliation Results Behavior Diagram	191, 291, 391, 491, 591, 691, 791, 891, 892, 1291, 1292
	Figure 6.1: Transaction Clean-up Diagram	309
	Figure 6.2: ITL Notification	303, 400, 500, 501, 700, 701, 702, 703, 704, 800, 806
	Figure 6.3: Get Transaction Status Diagram	308

(cont.)

<b>DES Behavior Diagrams (cont.)</b>	Figure 6.4: Time Synchronization Diagram	190, 290, 390, 490, 590, 690, 790, 890, 1290
<b>Notification Type</b>	1 - Net Source Cancellation	400
	2 - Non-compliance Cancellation	805
	3 - Impending Expiry of tCER or ICER	700
	4 - Reversal of Storage for CDM Project	701
	5 - Non-submission of Certification Report for CDM Project	702
	6 - Excess Issuance for CDM Project	303
	7 - Commitment Period Reserve	704
	8 - Unit Carry-over	800
	9 - Expiry Date Change	500, 501
	10 - Notification Update	703

**Figure 1.1  
Functional Component-CDM Registry Test Mapping**

<b>ITL Web Service</b>	AcceptMessage	2600
	AcceptProposal	2100, 2101, 2102, 2103, 2104, 2200, 2201, 2202, 2204, 2205, 2206, 2207, 2208, 2209, 2300, 2301, 2302, 2304, 2305, 2400, 2401, 2402, 2403, 2500, 2501, 2601, 2602, 2603, 2604, 2692
	AcceptNotification	2100, 2101, 2102, 2103, 2104, 2200, 2201, 2202, 2203, 2204, 2205, 2206, 2207, 2208, 2209, 2300, 2301, 2303, 2304, 2305, 2400, 2401, 2402, 2403, 2500, 2501, 2601, 2602, 2603, 2604, 2692
	GetTransactionStatus	2302
	ReceiveReconciliationResult	n/a
	ProvideAuditTrail	n/a
	ProvideTotals	n/a
	ProvideUnitBlocks	n/a
	ReceiveAuditTrail	2692
	ReceiveTotals	2191, 2291, 2391, 2491, 2591, 2691, 2692
	ReceiveUnitBlocks	2191, 2291, 2391, 2491, 2591, 2691, 2692
<b>Registry Web Services</b>	ProvideTime	2190, 2290, 2390, 2490, 2590, 2690
	AcceptMessage	2600
	AcceptProposal	2304, 2305, 2692
	AcceptNotification	2100, 2101, 2102, 2103, 2104, 2200, 2201, 2202, 2203, 2204, 2205, 2206, 2207, 2208, 2209, 2300, 2301, 2303, 2305, 2400, 2401, 2402, 2403, 2500, 2501, 2601, 2602, 2603, 2604
	AcceptITLNotification	2400, 2401, 2402, 2403, 2500, 2501
	InitiateReconciliation	2191, 2192, 2291, 2292, 2391, 2392, 2491, 2492, 2591, 2592, 2691, 2692, 2692
	ReceiveReconciliationResult	2191, 2291, 2391, 2491, 2591, 2691, 2692
	ProvideAuditTrail	2692
	ProvideTotals	2191, 2291, 2391, 2491, 2591, 2691, 2692
	ProvideUnitBlocks	2191, 2291, 2391, 2491, 2591, 2691, 2692

(cont.)

<b>DES Behavior Diagrams</b>	Figure 4.5: Single Registry Transaction Behavior Diagram	2100, 2101, 2102, 2103, 2104, 2200, 2201, 2202, 2203, 2204, 2205, 2206, 2207, 2208, 2209, 2400, 2401, 2402, 2403, 2500, 2501, 2601, 2602, 2603, 2604
	Figure 4.6: Discrepancy Notification Sequence Diagram	
	Figure 4.7: Terminate Transaction Sequence Diagram	2305
	Figure 4.9: External Transfer Behavior Diagram	2300, 2301, 2302, 2303, 2304, 2305, 2306, 2692
	Figure 4.10: Discrepancy Notification Sequence Diagram	2305
	Figure 4.11: Terminate External Transaction Sequence Diagram	2300, 2303, 2305, 2306
	Figure 5.1: Reconciliation Behavior Diagram	2191, 2291, 2391, 2491, 2591, 2691, 2692
	Figure 5.2: Send Reconciliation Results Behavior Diagram	2191, 2291, 2391, 2491, 2591, 2691, 2692
	Figure 6.1: Transaction Clean-up Diagram	2303
	Figure 6.2: ITL Notification	2400, 2401, 2402, 2403, 2500, 2501
	Figure 6.3: Get Transaction Status Diagram	2302
	Figure 6.4: Time Synchronization Diagram	2190, 2290, 2390, 2490, 2590, 2690
	<b>Notification Type</b>	1 - Net Source Cancellation
2 - Non-compliance Cancellation		n/a
3 - Impending Expiry of tCER or ICER		n/a
4 - Reversal of Storage for CDM Project		2400
5 - Non-submission of Certification Report for CDM Project		2402
6 - Excess Issuance for CDM Project		2304, 2401
7 - Commitment Period Reserve		n/a
8 - Unit Carry-over		n/a
9 - Expiry Date Change		2500, 2501
10 - Notification Update		2403

The test cases described in this document are split into two sections. Test cases included in Section 3 describe discrete transaction tests that ensure a national registry is capable of engaging in each required transaction type. Test cases in Section 4 are grouped around business case scenarios to reflect a possible sequential flow of Kyoto transaction types in a commitment period. Later tests may be dependent on the successful completion of transactions in preceding tests; therefore, the national registry must proceed with testing in the sequential order described in Section 4 of Annex H.

Following each group of tests, a registry will be expected to reconcile successfully with the ITL. Every group of tests will end with reconciliation including a comparison of data at the totals and, if requested by the ITL, at the unit block level. The expected account and unit holdings are presented as the expected results for each reconciliation test. If a registry fails a reconciliation test by not meeting the expected results specified, the reconciliation will continue through to the audit trail phase. The registry must resolve the underlying cause of the reconciliation failure, clear its logs, and restart testing from the first test case. Within the test cases, specific serial block numbers are specified only for units transferred into the registry from simulated third-party registries as a national registry may have different conventions within the bounds of the DES for assigning unit serial numbers.

Prior to each confirmation reconciliation, the ITL will send the registry a ProvideTime request to ensure time synchronization.

A registry must pass all relevant test cases to satisfy the requirements of the functional testing component of initialization. An individual test case will be considered to be passed by a registry when all of the following criteria are met with the exception of provisions in Appendix 1:

- The expected results specified within the test case are achieved in full;
- All messages for a given transaction sequence as specified in the expected results for each test case must be sent/received by the registry (i.e., finalizing a transaction in the database but not sending a required final message constitutes a test case failure);
- No sequence errors have been encountered;
- No unexpected check failures have been encountered; and
- The transaction specified by the test case is verified by the confirmation reconciliation that takes place after each group of tests.

The registry will begin initialization with a clean database so that all logs will reflect only messages and transactions required by test cases. The registry will submit an electronic version of its logs in human-readable format at the conclusion of testing. Registry logs will be reviewed at the initialization coordinator's discretion to ensure logging specifications are implemented sufficiently and to serve as documentation of initialization.

The registry will not be required to submit erroneous data for the purpose of negative testing.

Test cases involving notifications from the ITL requiring the registry to take action will not specify an action due date 30 days in the future as will typically be the case in a production environment. For these test cases, the date specified by the ITL in the notification will have no bearing on notification follow-up. The registry should act immediately.

The time frame to complete functional testing is less than 2 days.

## **1.3 Prerequisites**

The registry must conclude all internal testing it intends to conduct before entering the functional testing component of initialization comprising Annex H.

### **1.3.1 Connectivity**

VPN connectivity must be established between the registry and the ITL initialization environment.

The registry and the ITL must successfully connect using SSL and all tests required to test network connectivity must be successfully executed by the designated network specialists.

Basic Web service communication must be established. Thus, the registry is able to both:

- Send a message using the AcceptMessage Web service to the ITL; and
- Receive and respond to a ProvideTime request from the ITL.

### **1.3.2 Documentation Review**

The registry must provide documentation as described in Section 9 of the DES to demonstrate the implementation of non-functional requirements specified in the DES. Before beginning the Annex H test cases, the registry must submit and receive approval of the required documentation described in Section 9.2.7 of the DES.

Testing as part of Annex H is to be conducted on the testing environment described in the documentation required by Section 9.2.7.

## 12 2. Technical Information

### 2.1 Web Service Definitions

Refer to Annex K of the Data Exchange Standards for the definitions of the Web services. All Web services described in Annex K will be employed by the test cases described in this annex.

### 2.2 Codes

Refer to Annex G of the Data Exchange Standards for a description of the codes employed throughout the test cases described in this annex.

### 2.3 Seed Data Requirements

#### 2.3.1 Accounts

The initializing registry must create the following accounts for both commitment periods 1 and 2 (with the exception of holding accounts, which do not have an applicable commitment period). If a registry is unable to use the identifiers described here, it may substitute alternative identifiers. If a registry cannot use a specific account for a specific transaction detailed as part of a test in Annex H, the registry may use another valid account as long as it is within the bounds of the DES. Any alternative identifiers must be communicated to the initialization coordinator prior to commencing this functional test plan.

**Figure 2.1:  
National Registry Accounts**

Code	Account Type	CP	ID
100	Holding Account	0	1
120	Operator Holding Account (EU registry only)	0	3
121	Person Holding Account (EU registry only)	0	4
210	Net Source Cancellation Account (national registry only)	1	5
220	Non-compliance Cancellation Account (national registry only)	1	6
220	Non-compliance Cancellation Account (national registry only)	2	16
230	Voluntary Cancellation Account (national registry only)	1	7
250	Mandatory Cancellation Account	1	9
300	Retirement Account	1	10
300	Retirement Account	2	19
411	tCER Replacement Account for Expiry	1	11
421	ICER Replacement Account for Expiry	1	12
422	ICER Replacement Account for Reversal in Carbon Storage	1	13
423	ICER Replacement Account for Failure to Submit Certification Report	1	14

**Figure 2.2:  
CDM Registry Accounts**

<b>Code</b>	<b>Account Type</b>	<b>CP</b>	<b>ID</b>
100	Holding Account	0	1
100	Holding Account	0	2
100	Holding Account	0	3
100	Holding Account	0	4
100	Holding Account	0	13
110	Pending Account	0	5
240	Excess Issuance Cancellation Account (CDM Registry only)	1	6
240	Excess Issuance Cancellation Account (CDM Registry only)	2	16
250	Mandatory Cancellation Account	1	7

## **2.3.2 Registry Identifiers**

### **12.1.1.1 2.3.2.1 National Registry Test Cases**

The registry codes specified in each test are not always existing registry codes. Registry code ZZ is a placeholder for the initializing registry's code (ZZ should be replaced with NZ if the New Zealand registry is initializing). Registry code YY refers to the registry code assigned to the third-party registry simulator. Registry code BO is used to refer to a non-Annex I registry (Bolivia) serving as a Project host Party. The registry codes for the CDM Registry, ITL, and EUTL remain CDM, ITL, and CTL, respectively.

### **12.1.1.2 2.3.2.2 CDM Registry Test Cases**

The registry codes specified in each test case for initializing the CDM registry are existing registry codes.

## **2.3.3 Project Identifiers**

Tests in initialization involve fictitious CDM and JI Project identifiers. If registry design requires advance Project set-up, the following Projects will be used. The required structures for transmission and display of identifiers defined in Annex F apply for data transmission throughout the test cases.



**Figure 2.3:  
National Registry Project Identifiers**

Project Type	Display Project Number	Transmitted Project Number		Crediting Period End Date	Comment
		Party Code	Project ID		
CDM	BO-329	BO	329	16/1/2028	ICERs
CDM	BO-1983	BO	1983	29/3/2015	ICERs
JI	ZZ-221	ZZ	221		Track 1, converting AAUs.
JI	ZZ-365	ZZ	365		Track 1, converting RMUs.
CDM	BO-12	BO	12	31/12/2017	tCERs
CDM	BO-1	BO	1		CERs
CDM	BO-812	BO	812	29/3/2017	ICER

**Figure 2.4:  
CDM Registry Project Identifiers**

Project Type	Display Project Number	Transmitted Project Number		Crediting Period End Date	Comment
		Party Code	Project ID		
CDM	BR-3	BR	3	16/1/2028	ICERs
CDM	BR-2345	BR	2345	16/1/2028	ICERs
CDM	BR-6	BR	6	31/12/2017	tCERs
CDM	IN-4	IN	4		CERs
CDM	IN-2	IN	2	31/12/2017	tCERs

#### **2.3.4 Commitment Period Reserve**

The ITL will set the Commitment Period Reserve (CPR) for registry ZZ at 50 units for Section 3 and at 6200 units for Sections 4 & 5. If the registry has built internal checks to prevent transactions that would violate its CPR, the registry should set its CPR below 50 units to allow all tests required during initialization. Test ID 704 will explicitly test the registry's ability to respond to a CPR notification.

#### **2.3.5 Issuance, Conversion and Retirement Limits**

The ITL maintains data specifying issuance and conversion limits for a national registry. National registries are not required to but may implement internal system checks to prevent submitting a transaction proposal that would trigger a failure response code from the ITL. For the purposes of initialization functional testing, the ITL and registry must reflect different limits because tests check the registry's ability to receive a failure code. The same limits are applicable for each commitment period.

**Figure 2.5:  
Issuance, Conversion and Retirement Limits**

<b>Limit Type</b>	<b>Registry Limit</b>	<b>ITL Limit</b>	<b>Comment</b>
AAU Issuance	95,100 AAUs	5,001 AAUs	Used in Test ID 102
RMU Issuance	2,000 RMUs	1,001 RMUs	For LULUCF Category 1: Afforestation and reforestation. LULUCF Categories 2-6 are not used.
Conversion Limit	300 ERUs	201 ERUs	Limit for ERUs converted from both AAUs and RMUs
CDM Project Limits	15,000,000 units	15,000,000 units	Limit applies to each CDM project used in this annex.
ICER & tCER Retirement Limit	951 ICERs & tCERs	951 ICERs & tCERs	Allows retirement of all ICERs and tCERs in Test ID 600

### 2.3.6 Units Holdings in Third-party Simulators

The ITL will reflect the unit holdings required for transactions with third-party registry simulators required for testing. The following tables are for use by the ITL in constructing seed data.

**Figure 2.6:  
Registry Simulator Units Holdings for National Registry Test Cases**

Test ID	Registry	Account Type/ID	Unit Type	Serial Range	Project Number	LULUCF Activity	Expiry Date
Replacement-2 & 302	CDM	110/5	ICER (7)	1501-1700	BO-1983	1	2015-03-29
300	CDM	110/5	tCER (6)	9001-9200	BO-12	1	2017-12-31
310	YY	100/8888	AAU (1)	1-100			
Expiry Data Change-1 & 301	CDM	110/5	ICER (7)	251-450	BO-329	1	2028-01-16
304	CDM	110/5	CER (5)	3001-3150	BO-1		
305	CDM	110/5	ICER (7)	4001-4200	BO-812	1	2017-03-29
704	YY	100/8888	AAU (1)	1001-2000			
893 & 1292	YY	100/8888	AAU (1)	101-200			

**Figure 2.7:  
Registry Simulator Units Holdings for CDM Registry Test Cases**

Test ID	Registry	Account Type/ID	Unit Type	Serial Range	Commitment Period
2304	GB	100/8888	AAU (1)	1-5000	1
2692	GB	100/8888	AAU (1)	8001-8100	2

## **13 3. Discrete Transaction Tests**

### **3.0 General Information**

This section defines a set of discrete transaction tests designed to test the registry's ability to perform a single transaction type without dependencies on preceding tests. Since all transaction types, with the exception of issuance, depend on the presence of existing units, an issuance or external transfer of units must precede each of these discrete transaction test cases. These tests are organized by a transaction-specific test group, defined as all the tests included under each transaction subsection: 3.1 Issuance, 3.2 Conversion, 3.3 External Transfer, 3.4 Cancellation, 3.5 Retirement, 3.6 Replacement, 3.7 Carry-over, and 3.8 Expiry Date Change.

All national registries must successfully execute all tests included in Section 3 to proceed to Section 4. The transaction-specific tests included in this section need not be executed in sequence.

The registry must begin Section 3 with a database clear of unit holdings and transaction records, but must reflect the required accounts specified in Figure 2.1: National Registry Accounts, and limits specified in Figure 2.5: Issuance, Conversion, and Retirement Limits.

#### **3.0.1 Confirmation Reconciliation**

After each transaction-specific test group, the ITL will request that the registry provide its current totals by account and unit type, and subsequently its current unit block holdings. No optional parameters will be specified in the totals request. The unit blocks specified as part of the unit block request will include all blocks once generated as testing progresses. The confirmation reconciliation will monitor the registry's progress through initialization.

Since the discrete tests in Section 3 need not be completed in sequence, the reconciliation results may vary. The Expected Results for reconciliation tests are presented as the change in unit holdings resulting from the discrete transaction test immediately preceding the reconciliation. For example, if the preceding transaction test requires the registry to issue 100 AAUs, the Expected Results section of the following reconciliation test will state the change in registry holdings as "+100" which should be understood to mean an additional 100 AAUs from the last successful reconciliation result. If a registry attempts a transaction test and fails the ensuing reconciliation test due to a problem with the transaction, then the registry and ITL should agree on a manual intervention to ensure consistency in reconciliation tests following later discrete transaction tests. The ITL will call the registry's ProvideTime Web service before each confirmation reconciliation to ensure time synchronization.

#### **3.0.2 Test Parameters**

Data elements required in the DES for WSDLs and Web service communication are to be included during transactions and notifications encountered during test scenarios. However, only the data elements unique to the test are specified in Annex H for each test as entries in two sections: Relevant Notification Parameters and Relevant Transaction Parameters. The information provided under each test description is sufficient for the registry to generate messages with all required data elements; elements not specified are obvious or will be provided in preceding notifications. Notification identifiers have not been specified, as the registry must verify receipt of notifications by proposing required transactions specifying the notification identifier provided by the ITL.

#### **3.0.3 STL Checks**

If the initializing registry is an EU registry, the EUTL simulator will provide a "STL Checked, No discrepancy" response for all transactions in Section 3.

#### **3.0.4 Commitment Period**

All tests cases are assumed to take place during Commitment Period 1 unless otherwise stated.

### 3.1 Issuance

#### 3.1.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptNotification ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	Single Registry Transaction Behavior Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	None

#### 3.1.2 Test Steps

Steps:

1. The registry will propose an issuance transaction (Transaction Type 1).
2. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.
3. The ITL should call the registry's AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
4. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
5. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.

### 3.1.3 Test Cases

<b>Test ID</b>	Issuance-1
<b>Description</b>	
Issue 100 AAUs to Holding Account	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Acquiring Registry Code	ZZ
Acquiring Account Type	100 (Holding Account)
Acquiring Account ID	1
Originating Registry Code	ZZ
Unit Type	1
Unit Quantity	100
<b>Expected Result</b>	
100 AAUs successfully issued to Holding Account #1 of registry ZZ.	
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).	
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).	

<b>Test ID</b>	Issuance-2
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following change in totals:	
accountType accountCommitPeriod unitType unitCount	100 0 1 +100
If requested by the ITL, the registry should return the following change in unit block holdings:	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include an additional 100 units ZZ 1 100 1

## 3.2 Conversion

### 3.2.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptNotification ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	Single Registry Transaction Behavior Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	None

### 3.2.2 Test Steps

1. The registry will propose a conversion transaction (Transaction Type 2). Note that the unit type specified should be the resulting unit type, not the current unit type.
2. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.
3. The ITL should call the registry's AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
4. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
5. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.

### 3.3.3 Test Cases

<b>Test ID</b>	Conversion-1	
<b>Description</b>		
Issue 100 AAUs to Holding Account		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	ZZ	
Acquiring Registry Code	ZZ	
Acquiring Account Type	100 (Holding Account)	
Acquiring Account ID	1	
Originating Registry Code	ZZ	
Unit Type	1	
Unit Quantity	100	
<b>Expected Result</b>		
100 AAUs successfully issued to Holding Account #1 of registry ZZ.		
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).		
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).		

<b>Test ID</b>	Conversion-2	
<b>Description</b>		
Convert 100 AAUs in Holding Account #1 into ERUs.		
<b>Relevant Parameters</b>		
Transferring Registry Code	ZZ	
Transferring Account Type	100	
Acquiring Account ID	1	
Acquiring Registry Code	ZZ	
Project ID	221	
Originating Registry Code	ZZ	
Unit Type	3	
Unit Quantity	100	
Track	1	
<b>Expected Result</b>		
100 AAUs should be successfully converted into ERUs and remain in holding account #1.		
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).		
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).		



<b>Test ID</b>	Conversion-3	
<b>Description</b>		
ITL Request Totals		
<b>Relevant Parameters</b>		
Reconciliation Identifier	ITL to determine	
Reconciliation Snapshot DateTime	Now ()	
<b>Expected Result</b>		
The registry should return the following change in totals:		
accountType	100	
accountCommitPeriod	0	
unitType	3	
unitCount	+100	
If requested by the ITL, the registry should return the following change in unit block holdings:		
unitSerialBlockStart	Determined by registry to include an additional 100	
unitSerialBlockEnd	units	
originatingRegistryCode	ZZ	
unitType	3	
AccountType	100	
ApplicableCommitmentPeriod	1	

### 3.3 External Transfer

#### 3.3.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptNotification ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	External Registry Transaction Behavior Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	None

#### 3.3.2 Test Steps

1. The registry will propose an External transfer transaction (Transaction Type 3).
2. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.
3. The ITL then calls the simulated third-party registry (YY) AcceptProposal Web service and YY should call the ITL's AcceptNotification Web service after processing the

proposal. For the purposes of testing, the simulated registry will return messages to the ITL, which will proceed to step 4.

4. The ITL should call the registry's (ZZ) AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
5. On receiving this notification, the registry (ZZ) will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
6. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.

### 3.3.3 Test Cases

<b>Test ID</b>	External Transfer-1	
<b>Description</b>		
Issue 100 AAUs to Holding Account		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	ZZ	
Acquiring Registry Code	ZZ	
Acquiring Account Type	100 (Holding Account)	
Acquiring Account ID	1	
Originating Registry Code	ZZ	
Unit Type	1	
Unit Quantity	100	
<b>Expected Result</b>		
100 AAUs successfully issued to Holding Account #1 of registry ZZ.		
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).		
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).		

<b>Test ID</b>	External Transfer-2
<b>Description</b>	
Transfer 50 AAUs to an external third-party (holding account of country YY). YY agrees to the transaction.	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	YY
Acquiring Account Type	100
Acquiring Account ID	8888
Originating Registry Code	ZZ
Unit Type	1
Unit Quantity	50
<b>Expected Result</b>	
50 AAUs from Holding Account #1 should be successfully transferred to the holding account of external registry YY, holding account 8888.	
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy). (Communications with YY are hidden from ZZ).	
The ITL will then send a notification to registry with Transaction Status Code of 8 (Accepted).	
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).	

<b>Test ID</b>	External Transfer-3
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following change in totals:	
accountType accountCommitPeriod unitType unitCount	100 0 1 +50
If requested by the ITL, the registry should return the following change in unit block holdings:	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include an additional 50 units ZZ 3 100 1

### 3.4 Cancellation

#### 3.4.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptNotification AcceptITLNotice ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	ITL Notification Single Registry Transaction Behavior Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	Net Source Cancellation (Type 1)

#### 3.4.2 Test Steps

1. The registry will receive a notification.
2. The registry will propose a Cancellation transaction (Transaction Type 4).
3. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.
4. The ITL should call the registry's AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
5. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
6. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.
7. The registry will receive a notification update indicating the requirement has been fulfilled.

#### 3.4.3 Test Cases

<b>Test ID</b>	Cancellation-1	
<b>Description</b>		
Issue 100 AAUs to Holding Account		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	ZZ	
Acquiring Registry Code	ZZ	
Acquiring Account Type	100 (Holding Account)	
Acquiring Account ID	1	
Originating Registry Code	ZZ	
Unit Type	1	
Unit Quantity	100	
<b>Expected Result</b>		
<p>100 AAUs successfully issued to Holding Account #1 of registry ZZ.</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p>		

<b>Test ID</b>	Cancellation-2
<b>Description</b>	
Registry will receive a notification (Type 1) of net source cancellation with instructions to cancel 50 AAUs.	
<b>Relevant Notification Parameters</b>	
Message Content	The Compliance Committee has found that LULUCF activities have resulted in a net source of emissions. AAUs must be cancelled to offset these emissions during initialization.
Notification Type	1
Target Value	50
LULUCF Activity	2
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Acquiring Registry Code	ZZ
Acquiring Account Type	210 (Net Source Cancellation Account)
Acquiring Account ID	5
Unit Type	1
Unit Quantity	50
Originating Registry	ZZ
<b>Expected Result</b>	
<p>50 AAUs originating from ZZ should be cancelled (transferred to cancellation account type 210) successfully.</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p> <p>The registry will log the notification update (requirement fulfilled).</p>	

<b>Test ID</b>	Cancellation-3	
<b>Description</b>		
ITL Request Totals		
<b>Relevant Parameters</b>		
Reconciliation Identifier	ITL to determine	
Reconciliation Snapshot DateTime	Now ()	
<b>Expected Result</b>		
The registry should return the following change in totals:		
accountType accountCommitPeriod unitType unitCount	100 0 1 +50	
accountType accountCommitPeriod unitType unitCount	210 1 1 +50	
If requested by the ITL, the registry should return the following change in unit block holdings:		
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include an additional 50 units ZZ 1 100 1	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include an additional 50 units ZZ 1 210 1	

### 3.5 Retirement

#### 3.5.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptNotification ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	Single Registry Transaction Behavior Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	None

### 3.5.2 Test Steps

1. The registry will propose a Retirement transaction (Transaction Type 5).
2. The registry will submit the proposal to the ITL's AcceptProposal, and verify that it receives a response indicating that the proposal was received.
3. The ITL should call the registry's AcceptNotification method to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
4. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
5. The registry will then call the ITL's AcceptNotification method to notify the ITL of its result.

### 3.5.3 Test Cases

<b>Test ID</b>	Retirement-1
<b>Description</b>	
Issue 100 AAUs to Holding Account	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Acquiring Registry Code	ZZ
Acquiring Account Type	100 (Holding Account)
Acquiring Account ID	1
Originating Registry Code	ZZ
Unit Type	1
Unit Quantity	100
<b>Expected Result</b>	
100 AAUs successfully issued to Holding Account #1 of registry ZZ.  The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).  The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).	



<b>Test ID</b>	Retirement-2	
<b>Description</b>		
Retire 50 units		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	ZZ	
Transferring Account Type	100	
Transferring Account ID	1	
Acquiring Registry Code	ZZ	
Acquiring Account Type	300	
Acquiring Account ID	10	
Originating Registry Code	ZZ	
Unit Type	1	
Unit Quantity	50	
<b>Expected Result</b>		
<p>The registry will propose a retirement transaction (Type 5).</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p> <p>50 AAUs will be transferred to the retirement account (Type 300).</p>		

<b>Test ID</b>	Retirement-3	
<b>Description</b>		
ITL Request Totals		
<b>Relevant Parameters</b>		
Reconciliation Identifier	ITL to determine	
Reconciliation Snapshot DateTime	Now ()	
<b>Expected Result</b>		
The registry should return the following change in totals:		
accountType accountCommitPeriod unitType unitCount	100 0 1 +50	
accountType accountCommitPeriod unitType unitCount	300 1 1 +50	
If requested by the ITL, the registry should return the following change in unit block holdings:		
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include an additional 50 units ZZ 1 100 1	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include an additional 50 units ZZ 1 300 1	

### 3.6 Replacement

#### 3.6.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptProposal AcceptNotification AcceptITLNotice ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	ITL Notification Single Registry Transaction Behavior Diagram External Transfer Behavior Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	Non-submission of Certification Report for CDM Project (Type 5)

### 3.6.2 Test Steps

1. The ITL will notify the registry of Non-submission of Certification Report for a CDM Project. This message is sent to the registry's AcceptITLNotice method. The message will inform the registry that it must replace ICERs and will specify a notification identifier. This notification will have a status of 1.
2. The registry will propose a Replacement transaction (Transaction Type 6) as described in each test case.
3. The registry will submit the proposal to the ITL's AcceptProposal, and verify that it receives a response indicating that the proposal was received.
4. The ITL should call the registry's AcceptNotification method to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
5. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
6. The registry will then call the ITL's AcceptNotification method to notify the ITL of its result.
7. The ITL will notify the registry via AcceptITLNotice that they have satisfied this notification. This notice will reference the original notification identifier and a status of 3.

### 3.6.3 Test Cases

<b>Test ID</b>	Replacement-1	
<b>Description</b>		
Issue 200 AAUs to Holding Account		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	ZZ	
Acquiring Registry Code	ZZ	
Acquiring Account Type	100 (Holding Account)	
Acquiring Account ID	1	
Originating Registry Code	ZZ	
Unit Type	1	
Unit Quantity	200	
<b>Expected Result</b>		
<p>200 AAUs successfully issued to Holding Account #1 of registry ZZ.</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p>		

<b>Test ID</b>	Replacement-2	
<b>Description</b>		
The CDM registry transfers 200 ICERs from Project 1983.		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	CDM	
Transferring Account Type	110	
Transferring Account ID	5	
Acquiring Registry Code	ZZ	
Acquiring Account Type	100	
Acquiring Account ID	1	
Originating Registry Code	BO	
Unit Serial Block Start	1501	
Unit Serial Block End	1700	
LULUCF Activity	1	
Project ID	1983	
Expiry Date	2015-03-29	
Unit Type	7	
<b>Expected Result</b>		
<p>The CDM registry transfers 200 ICERs with Project ID 1983 to the Holding Account in ZZ.</p> <p>After receiving the AcceptProposal call, the Acquiring Registry Code (ZZ) sends an acknowledgement that it has received the proposal. ZZ approves the transaction and finalizes it in the database.</p> <p>The acquiring registry's notification to the ITL should include a Transaction Status Code of 8 (Accepted).</p>		

<b>Test ID</b>	Replacement-3
<b>Description</b>	
Registry receives Non-submission of Certification Report for CDM Project notification (Type 5). The registry is directed to replace the affected 200 ICERs, which should be in the holding account, with 200 AAUs from the holding account. The registry will fulfill the notification instruction.	
<b>Relevant Notification Parameters</b>	
messageContent	The CDM Executive Board has not received a Certification Report, and it has requested that all ICERs generated by the specified CDM be replaced. For the purposes of initialization, replace the ICERs with AAUs.
notificationType	5
notificationStatus	1
projectNumber	BO-1983
originatingRegistryCode	BO
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	ZZ
Acquiring Account Type	423
Acquiring Account ID	14
Originating Registry Code	ZZ
Unit Type	1
Unit Quantity	200
First Block -- Replacing units: start	To be determined by registry
First Block -- Replacing units: end	To be determined by registry
Last Block -- Units to be replaced: start	1501
Last Block -- Units to be replaced: end	1700
originatingPartyCode	BO
<b>Expected Result</b>	
<p>The registry will propose a replacement transaction (Type 6).</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p> <p>200 AAUs from the holding account should be transferred to the ICER Replacement Account for Failure to Submit Certification Report (Type 423).</p>	

<b>Test ID</b>	Replacement-4	
<b>Description</b>		
ITL Request Totals		
<b>Relevant Parameters</b>		
Reconciliation Identifier	ITL to determine	
Reconciliation Snapshot DateTime	Now ()	
<b>Expected Result</b>		
The registry should return the following change in totals:		
accountType accountCommitPeriod unitType unitCount	100 0 7 +200	
accountType accountCommitPeriod unitType unitCount	423 1 1 +200	
If requested by the ITL, the registry should return the following change in unit block holdings:		
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1501 1700 ZZ 1 100 1	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include an additional 200 units issued in test Replacement-1 ZZ 1 423 1	

## 3.7 Carry-over

### 3.7.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptITLNotice AcceptNotification ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	ITL Notification Single Registry Transaction Behavior Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	Unit Carry-over (Type 8)

### 3.7.2 Test Steps

1. The ITL will notify the registry that it has units eligible for carry-over to commitment period 2.
2. The registry will propose a carry-over transaction (Transaction Type 7).
3. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.
4. The ITL should call the registry's AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
5. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
6. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.



### 3.7.3 Test Cases

<b>Test ID</b>	Carry-over-1	
<b>Description</b>		
Issue 100 AAUs to Holding Account		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	ZZ	
Acquiring Registry Code	ZZ	
Acquiring Account Type	100 (Holding Account)	
Acquiring Account ID	1	
Originating Registry Code	ZZ	
Unit Type	1	
Unit Quantity	100	
<b>Expected Result</b>		
100 AAUs successfully issued to Holding Account #1 of registry ZZ.		
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).		
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).		

<b>Test ID</b>	Carry-over-2
<b>Description</b>	
Registry receives a Unit Carry-over notification (Type 8) with directions to carry-over 50 AAUs. The registry will propose a Carry-over transaction to fulfill the notification direction.	
<b>Relevant Notification Parameters</b>	
messageContent	Due to the end of Commitment Period, the units specified within this message may be carried-over to the subsequent Commitment Period. For the purposes of initialization, carry-over 50 AAUs.
notificationType	8
notificationStatus	1
targetValue	50
commitPeriod	2
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	ZZ
Originating Registry Code	ZZ
Unit Type	1
Unit Total	50
originatingRegistryCode	ZZ
originalCommitPeriod	1
applicableCommitPeriod	2
<b>Expected Result</b>	
<p>The registry will propose a carry-over transaction (Type 7).</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p> <p>50 AAUs should now have applicable commitment period = 2.</p>	

<b>Test ID</b>	Carry-over-3
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following change in totals:	
accountType accountCommitPeriod unitType unitCount	100 0 1 +100
If requested by the ITL, the registry should return the following change in unit block holdings:	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include an additional 50 units issued in test Carry-over-1 ZZ 1 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include an additional 50 units issued in test Carry-over-1 ZZ 1 100 2

### 3.8 Expiry Date Change

#### 3.8.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptProposal AcceptNotification AcceptITLNotice ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	ITL Notification Single Registry Transaction Behavior Diagram External Registry Transaction Behavior Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	Expiry Date Change (Type 9)

### 3.8.2 Test Steps

1. The ITL will send the registry a notification (Notification Type 9) through its AcceptITLNotice Web service alerting it that certain units require an expiry date change.
2. The registry will propose an Expiry Date Change transaction (Transaction Type 8).
3. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.
4. The ITL should call the registry's AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
5. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
6. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.

### 3.8.3 Test Cases

<b>Test ID</b>	Expiry Date Change-1	
<b>Description</b>		
The CDM registry transfers 200 ICERs to the registry.		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	CDM	
Transferring Account Type	110	
Transferring Account ID	5	
Acquiring Registry Code	ZZ	
Acquiring Account Type	100	
Acquiring Account ID	1	
Originating Registry Code	BO	
LULUCF Activity	1	
Project ID	329	
Unit Serial Block Start	251	
Unit Serial Block End	450	
Expiry Date	2028-01-16	
Unit Type	7	
<b>Expected Result</b>		
The CDM registry transfers 200 ICERs to the Holding Account in ZZ.		
After receiving the AcceptProposal call, the Acquiring Registry Code (ZZ) sends an acknowledgement that it has received the proposal. ZZ approves the transaction and finalizes it in the database.		
The acquiring registry's notification to the ITL should include a Transaction Status Code of 8 (Accepted).		

<b>Test ID</b>	Expiry Date Change-2
<b>Description</b>	
The ITL sends an Expiry Date Change notification (Type 9) to change the expiry date on all ICERs associated with CDM Project BO-329.	
<b>Relevant Notification Parameters</b>	
messageContent	Due to a change in the crediting period for the relevant CDM Project, the units specified within this message must have their expiry date changed.
notificationType	9
notificationStatus	1
projectNumber	BO-329
unitType	7
targetValue	200
targetDate	2029-11-25
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Acquiring Registry Code	ZZ
Acquiring Account Type	100
Acquiring Account ID	1
Originating Registry Code	BO
Unit Serial Block Start	251
Unit Serial Block End	450
LULUCF Activity	1
Project ID	329
Expiry Date	2029-11-25
Unit Type	7
<b>Expected Result</b>	
The registry will change the expiry date of all ICERs originating from Project BO-329 in one or more transactions. The transaction proposal will include the notification identifier.	

<b>Test ID</b>	Expiry Date Change-3	
<b>Description</b>		
ITL Request Totals		
<b>Relevant Parameters</b>		
Reconciliation Identifier	ITL to determine	
Reconciliation Snapshot DateTime	Now ()	
<b>Expected Result</b>		
The registry should return the following change in totals:		
accountType	100	
accountCommitPeriod	0	
unitType	7	
unitCount	+200	
If requested by the ITL, the registry should return the following change in unit block holdings:		
unitSerialBlockStart	251	
unitSerialBlockEnd	450	
originatingRegistryCode	BO	
unitType	7	
AccountType	100	
ApplicableCommitmentPeriod	1	

## **14 4. Test Cases (National Registry)**

### **4.0 General Information**

The registry must begin Section 4 with a database clear of unit holdings and transaction records, but must reflect the required accounts specified in Figure 2.1: National Registry Accounts, and limits specified in Figure 2.5: Issuance, Conversion, and Retirement Limits.

All national registries must proceed through all tests included in Section 4 in sequential order with the exception that registries participating in the EU Emissions Trading Scheme will bypass tests 892 and 893 in favor of similar tests in Section 5.

#### **4.0.1 Confirmation Reconciliation**

After each test group, the ITL will request that the registry provide its current totals by account and unit type, and subsequently its current unit block holdings. No optional parameters will be specified in the totals request. The unit blocks specified as part of the unit block request will include all blocks once generated as testing progresses. The confirmation reconciliation will monitor the registry's progress through initialization by ensuring the registry reflects the state of the data specified in the reconciliation test at the end of each test series. The ITL will call the registry's ProvideTime Web service before each confirmation reconciliation.

#### **4.0.2 Test Parameters**

Data elements required in the DES for WSDLs and Web service communication are to be included during transactions and notifications encountered during test scenarios. However, only the data elements unique to the test are specified in Annex H for each test as entries in two sections: Relevant Notification Parameters and Relevant Transaction Parameters. The information provided under each test description is sufficient for the registry to generate messages with all required data elements; elements not specified are obvious or will be provided in preceding notifications. Notification identifiers have not been specified, as the registry must verify receipt of notifications by proposing required transactions specifying the notification identifier provided by the ITL.

#### **4.0.3 STL Checks**

If the initializing registry is an EU registry, the EUTL simulator will provide a "STL Checked, No discrepancy" response for all transactions in Section 4. Upon passing the tests in Section 4, general for all registries, EUTL registries proceed to tests in Section 5 to evaluate performance for EUTL specific Web services and transactions.

#### **4.0.4 Commitment Period**

All tests cases are assumed to take place during Commitment Period 1 unless otherwise stated.

## 4.1 Issuance Transaction Tests

### 4.1.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptNotification ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	Single Registry Transaction Behavior Diagram Discrepancy Notification Sequence Diagram Terminate Transaction Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	None

### 4.1.2 Test Steps

Steps:

1. The registry will propose an issuance transaction (Transaction Type 1).
2. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.
3. The ITL should call the registry's AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
4. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
5. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.



### 4.1.3 Test Cases

<b>Test ID</b>	100
<b>Description</b>	
Issue 5000 AAUs to Holding Account	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Acquiring Registry Code	ZZ
Acquiring Account Type	100 (Holding Account)
Acquiring Account ID	1
Originating Registry Code	ZZ
Unit Type	1
Unit Quantity	5000
<b>Expected Result</b>	
5000 AAUs successfully issued to Holding Account #1 of registry ZZ.	
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).	
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).	

<b>Test ID</b>	101
<b>Description</b>	
Issue 1000 RMUs to Holding Account of registry ZZ	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Acquiring Registry Code	ZZ
Acquiring Account Type	100
Acquiring Account ID	1
Originating Registry Code	ZZ
Unit Type	2
LULUCF Activity	1
Unit Quantity	1000
<b>Expected Result</b>	
1000 RMUs successfully issued to Holding Account #1.	
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).	
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).	

<b>Test ID</b>	102
<b>Description</b>	
Issue 1000 AAUs to Holding Account to surpass AAU limit reflected by ITL.	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Acquiring Registry Code	ZZ
Acquiring Account Type	100 (Holding Account)
Acquiring Account ID	1
Originating Registry Code	ZZ
Unit Type	1
Unit Quantity	1000
<b>Expected Result</b>	
<p>The ITL's notification to the registry should include Response Code 5008: "The quantity of AAUs issued must not exceed the allowed quantity for the Commitment Period."</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 3 (Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 5 (Terminated).</p>	

<b>Test ID</b>	190
<b>Description</b>	
ITL will initiate Time Synchronization by calling the ProvideTime Web service.	
<b>Steps</b>	
The ITL will call the registry's ProvideTime Web service.	
<b>Expected Result</b>	
The time provided by the registry should be consistent with the time reflected by the ITL.	

<b>Test ID</b>	191
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following totals:	
accountType accountCommitPeriod unitType unitCount	100 0 1 5000
accountType accountCommitPeriod unitType unitCount	100 0 2 1000
If requested by the ITL, the registry should return the following unit blocks:	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 5000 units ZZ 1 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 1000 units ZZ 2 100 1

## 4.2 Conversion Transaction Tests

### 4.2.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptNotification ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	Single Registry Transaction Behavior Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	None

#### 4.2.2 Test Steps

1. The registry will propose a conversion transaction (Transaction Type 2). Note that the unit type specified should be the resulting unit type, not the current unit type.
2. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.
3. The ITL should call the registry's AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
4. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
5. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.

#### 4.2.3 Test Cases

<b>Test ID</b>	200
<b>Description</b>	
Convert 100 AAUs in Holding Account #1 into ERUs.	
<b>Relevant Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Acquiring Account ID	1
Acquiring Registry Code	ZZ
Project ID	221
Originating Registry Code	ZZ
Unit Type	3
Unit Quantity	100
Track	1
<b>Expected Result</b>	
100 AAUs should be successfully converted into ERUs and remain in holding account #1.  The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).  The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).	

<b>Test ID</b>	201
<b>Description</b>	
Convert 100 RMUs in Holding Account #1 into ERUs.	
<b>Relevant Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Acquiring Account ID	1
Acquiring Registry Code	ZZ
Project ID	365
Originating Registry Code	ZZ
Unit Type	4
Unit Quantity	100
Track	1
<b>Expected Result</b>	
100 RMUs should be successfully converted into ERUs and remain in holding account #1.	
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).	
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).	

<b>Test ID</b>	290
<b>Description</b>	
ITL will initiate Time Synchronization by calling the ProvideTime Web service.	
<b>Steps</b>	
The ITL will call the registry's ProvideTime Web service.	
<b>Expected Result</b>	
The time provided by the registry should be consistent with the time reflected by the ITL.	

<b>Test ID</b>	291
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following totals:	
accountType accountCommitPeriod unitType unitCount	100 0 1 4900
accountType accountCommitPeriod unitType unitCount	100 0 2 900
accountType accountCommitPeriod unitType unitCount	100 0 3 100
accountType accountCommitPeriod unitType unitCount	100 0 4 100
If requested by the ITL, the registry should return the following unit blocks:	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 4900 units ZZ 1 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 900 units ZZ 2 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from TEST ID 100 ZZ 3 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 4 100 1

### 4.3 External Transfer Transaction Tests

#### 4.3.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptProposal AcceptNotification AcceptITLNotice ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	24 Hour Transaction Cleanup Time Synchronization
<b>DES Behavior Diagrams</b>	External Registry Transaction Behavior Diagram Discrepancy Notification Sequence Diagram Terminate External Transaction Diagram Transaction Clean-up Diagram ITL Notification Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	Excess Issuance for CDM Project (Type 6)

#### 4.3.2 Test Steps

When the registry is the transferring registry:

1. The registry will propose an External transfer transaction (Transaction Type 3).
2. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.
3. The ITL then calls the simulated third-party registry (YY) AcceptProposal Web service and YY should call the ITL's AcceptNotification Web service after processing the proposal. For the purposes of testing, the simulated registry will return messages to the ITL, which will proceed to step 4.
4. The ITL should call the registry's (ZZ) AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
5. On receiving this notification, the registry (ZZ) will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
6. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.

When the registry is the acquiring registry:

1. For purposes of testing, the simulated third-party registry (YY) will generate proposals and pass them through the ITL to the registry being tested.
2. The ITL will call the registry's AcceptProposal Web service and the registry should call the ITL's AcceptNotification Web service after processing the proposal.

3. The ITL would notify the third-party that the proposal was accepted, and finalize the transaction upon receiving confirmation that the third-party had finalized the transaction.

#### 4.3.3 Test Cases

<b>Test ID</b>	300	
<b>Description</b>		
The CDM registry transfers 200 tCERs to the registry.		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	CDM	
Transferring Account Type	110	
Transferring Account ID	5	
Acquiring Registry Code	ZZ	
Acquiring Account Type	100	
Acquiring Account ID	1	
Originating Registry Code	BO	
LULUCF Activity	1	
Project ID	12	
Unit Serial Block Start	9001	
Unit Serial Block End	9200	
Expiry Date	2017-12-31	
Unit Type	6	
<b>Expected Result</b>		
The CDM registry transfers 200 tCERs to the Holding Account in ZZ.		
After receiving the AcceptProposal call, the Acquiring Registry Code (ZZ) sends an acknowledgement that it has received the proposal. ZZ approves the transaction and finalizes it in the database.		
The acquiring registry's notification to the ITL should include a Transaction Status Code of 8 (Accepted).		



<b>Test ID</b>	301	
<b>Description</b>		
The CDM registry transfers 200 ICERs from Project 329.		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	CDM	
Transferring Account Type	110	
Transferring Account ID	5	
Acquiring Registry Code	ZZ	
Acquiring Account Type	100	
Acquiring Account ID	1	
Originating Registry Code	BO	
Unit Serial Block Start	251	
Unit Serial Block End	450	
LULUCF Activity	1	
Project ID	329	
Expiry Date	2028-01-16	
Unit Type	7	
<b>Expected Result</b>		
<p>The CDM registry transfers 200 ICERs with Project ID 329 to the Holding Account in ZZ.</p> <p>After receiving the AcceptProposal call, the Acquiring Registry Code (ZZ) sends an acknowledgement that it has received the proposal. ZZ approves the transaction and finalizes it in the database.</p> <p>The acquiring registry's notification to the ITL should include a Transaction Status Code of 8 (Accepted).</p>		

<b>Test ID</b>	302	
<b>Description</b>		
The CDM registry transfers 200 ICERs from Project 1983.		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	CDM	
Transferring Account Type	110	
Transferring Account ID	5	
Acquiring Registry Code	ZZ	
Acquiring Account Type	100	
Acquiring Account ID	1	
Originating Registry Code	BO	
Unit Serial Block Start	1501	
Unit Serial Block End	1700	
LULUCF Activity	1	
Project ID	1983	
Expiry Date	2015-03-29	
Unit Type	7	
<b>Expected Result</b>		
<p>The CDM registry transfers 200 ICERs with Project ID 1983 to the Holding Account in ZZ.</p> <p>After receiving the AcceptProposal call, the Acquiring Registry Code (ZZ) sends an acknowledgement that it has received the proposal. ZZ approves the transaction and finalizes it in the database.</p> <p>The acquiring registry's notification to the ITL should include a Transaction Status Code of 8 (Accepted).</p>		

<b>Test ID</b>	303
<b>Description</b>	
The registry receives a notification (Type 6) of excess issuance for CDM Project (Project ID 329). The registry proposes an External Transfer transaction to transfer 50 ICERs generated under Project BO-329 from its Holding Account #1 to the CDM registry.	
<b>Relevant Notification Parameters</b>	
Message Content	The CDM Executive Board has determined an excess number of units have been distributed for CDM Project BO-329. Move the number of units indicated to the Excess Issuance Cancellation Account.
Notification Type	6
Project Number	BO-329
Target Value	50
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	CDM
Acquiring Account Type	240
Acquiring Account ID	6
Unit Type	7
Unit Serial Block Start	251
Unit Serial Block End	300
Originating Registry	BO
Project ID	329
<b>Expected Result</b>	
<p>The registry transfers 50 ICERs (from Project BO-329) from its holding account to the CDM registry's Excess Issuance Cancellation Account (Type 240).</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The ITL will then send a notification to registry with Transaction Status Code of 8 (Accepted). The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p>	

<b>Test ID</b>	304	
<b>Description</b>		
The CDM registry transfers 150 CERs from Project 1.		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	CDM	
Transferring Account Type	110	
Transferring Account ID	5	
Acquiring Registry Code	ZZ	
Acquiring Account Type	100	
Acquiring Account ID	1	
Originating Registry Code	BO	
Unit Serial Block Start	3001	
Unit Serial Block End	3150	
Project ID	1	
Unit Type	5	
<b>Expected Result</b>		
The CDM registry transfers 150 CERs with Project ID 1 to the Holding Account in ZZ.		
After receiving the AcceptProposal call, the Acquiring Registry Code (ZZ) sends an acknowledgement that it has received the proposal. ZZ approves the transaction and finalizes it in the database.		
The acquiring registry's notification to the ITL should include a Transaction Status Code of 8 (Accepted).		

<b>Test ID</b>	305
<b>Description</b>	
The CDM registry transfers 200 ICERs from Project 812.	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	CDM
Transferring Account Type	110
Transferring Account ID	5
Acquiring Registry Code	ZZ
Acquiring Account Type	100
Acquiring Account ID	1
Originating Registry Code	BO
Unit Serial Block Start	4001
Unit Serial Block End	4200
LULUCF Activity	1
Project ID	812
Expiry Date	2017-03-29
Unit Type	7
<b>Expected Result</b>	
The CDM registry transfers 200 ICERs with Project ID 812 to the Holding Account in ZZ.	
After receiving the AcceptProposal call, the Acquiring Registry Code (ZZ) sends an acknowledgement that it has received the proposal. ZZ approves the transaction and finalizes it in the database.	
The acquiring registry's notification to the ITL should include a Transaction Status Code of 8 (Accepted).	

<b>Test ID</b>	306
<b>Description</b>	
Transfer 100 AAUs to an external third-party (holding account of country YY). YY agrees to the transaction.	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	YY
Acquiring Account Type	100
Acquiring Account ID	8888
Originating Registry Code	ZZ
Unit Type	1
Unit Quantity	100

(cont.)

<b>Expected Result (cont.)</b>
<p>100 AAUs from Holding Account #1 should be successfully transferred to the holding account of external registry YY, holding account 8888.</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy). (Communications with YY are hidden from ZZ).</p> <p>The ITL will then send a notification to registry with Transaction Status Code of 8 (Accepted).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p>

<b>Test ID</b>	307
<b>Description</b>	
Transfer 100 of Holding Account's AAUs to an external third-party holding account of country YY. YY rejects the transaction because it does not recognize the specified account.	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	2
Acquiring Registry Code	YY
Acquiring Account Type	100
Acquiring Account ID	9999
Originating Registry Code	ZZ
Unit Type	1
Unit Quantity	100
<b>Expected Result</b>	
<p>This time, after the registry proposes the transaction, the ITL will send notification indicating that registry YY has rejected the proposal. The notification will have transaction status 6 (Rejected) and response code 5902: Acquiring account does not exist</p> <p>The registry should terminate the transaction, sending notification to the ITL that it has been done. The notification will have transaction status 5 (Terminated).</p>	

<b>Test ID</b>	308
<b>Description</b>	
Transfer 100 RMUs to an external third-party (holding account of country YY). Initializing registry ZZ then immediately calls the GetTransactionStatus Web service with the transaction number it specified.	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	YY
Acquiring Account Type	100
Acquiring Account ID	8888
Originating Registry Code	ZZ
Unit Type	2
Unit Quantity	100
<b>Expected Result</b>	
The registry will send a transaction proposal and then call the GetTransactionStatus Web service. The returned transaction status will be 2 - Checked, No Discrepancy. The registry has not finalized the proposed transaction.	

<b>Test ID</b>	309
<b>Description</b>	
YY does not respond to the external transfer of 100 RMUs in Test ID 308 and the 24-Hour Transaction Clean-up is triggered (for the purposes of initialization, in less than 24 hours from initiating the transaction).	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	YY
Acquiring Account Type	100
Acquiring Account ID	8888
Originating Registry Code	ZZ
Unit Type	2
Unit Quantity	100
<b>Expected Result</b>	
The ITL will send notification indicating that registry YY has not responded with-in 24 hours. The notification will have transaction status 7 (Cancelled) and response code 8802: Transaction Clean-up.  The registry should cancel the transaction, sending notification to the ITL that it has been done. The notification will have transaction status 7 (Cancelled).	

<b>Test ID</b>	310	
<b>Description</b>		
YY proposes a transaction transferring 100 AAUs originating from YY to the ZZ registry.		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	YY	
Transferring Account Type	100	
Transferring Account ID	8888	
Acquiring Registry Code	ZZ	
Acquiring Account Type	100	
Acquiring Account ID	1	
Originating Registry Code	YY	
Unit Serial Block Start	1	
Unit Serial Block End	100	
Unit Type	1	
<b>Expected Result</b>		
YY's AAUs should be successfully transferred to national registry ZZ.		
After receiving the AcceptProposal call, the Acquiring Registry Code (ZZ) should send an acknowledgement that it has received the proposal. ZZ approves the transaction and finalizes it in the database.		
The acquiring registry's notification to the ITL should include a Transaction Status Code of 8 (Accepted).		

<b>Test ID</b>	390	
<b>Description</b>		
ITL will initiate Time Synchronization by calling the ProvideTime Web service.		
<b>Steps</b>		
The ITL will call the registry's ProvideTime Web service.		
<b>Expected Result</b>		
The time provided by the registry should be consistent with the time reflected by the ITL.		



<b>Test ID</b>	391
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following totals:	
accountType accountCommitPeriod unitType unitCount	100 0 1 4900
accountType accountCommitPeriod unitType unitCount	100 0 2 900
accountType accountCommitPeriod unitType unitCount	100 0 3 100
accountType accountCommitPeriod unitType unitCount	100 0 4 100
accountType accountCommitPeriod unitType unitCount	100 0 6 200
accountType accountCommitPeriod unitType unitCount	100 0 7 550
accountType accountCommitPeriod unitType unitCount	100 0 5 150
If requested by the ITL, the registry should return the following unit blocks:	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 4800 units of the units generated in Test ID 100 ZZ 1 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 900 units of the units generated in Test ID 101 ZZ 2 100 1

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from TEST ID 100 ZZ 3 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 4 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1 100 YY 1 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	9001 9200 BO 6 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 150 units from Test ID 306 BO 7 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1501 1700 BO 7 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	3001 3150 BO 5 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	4001 4200 BO 812 100 1

## 4.4 Cancellation Transaction Tests

### 4.4.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptNotification AcceptITLNotice ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	ITL Notification Single Registry Transaction Behavior Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	Net Source Cancellation (Type 1)

### 4.4.2 Test Steps

1. The registry will receive a notification for Test ID 400, but will proceed directly to step 2 for Test ID 401.
2. The registry will propose a Cancellation transaction (Transaction Type 4).
3. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.
4. The ITL should call the registry's AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
5. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
6. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.
7. The registry will receive a notification update indicating the requirement has been fulfilled.

#### 4.4.3 Test Cases

<b>Test ID</b>	400
<b>Description</b>	
Registry will receive a notification (Type 1) of net source cancellation with instructions to cancel 100 AAUs.	
<b>Relevant Notification Parameters</b>	
Message Content	The Compliance Committee has found that LULUCF activities have resulted in a net source of emissions. AAUs must be cancelled to offset these emissions during initialization.
Notification Type	1
Target Value	100
LULUCF Activity	2
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Acquiring Registry Code	ZZ
Acquiring Account Type	210 (Net Source Cancellation Account)
Acquiring Account ID	5
Unit Type	1
Unit Quantity	100
Originating Registry	ZZ
<b>Expected Result</b>	
<p>100 AAUs originating from ZZ should be cancelled (transferred to cancellation account type 210) successfully.</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p> <p>The registry will log the notification update (requirement fulfilled).</p>	

<b>Test ID</b>	401
<b>Description</b>	
Holding Account of ZZ will attempt to cancel 100 AAUs into its voluntary cancellation account.	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Acquiring Registry Code	ZZ
Acquiring Account Type	230 (Voluntary Cancellation Account)
Acquiring Account ID	7
Originating Registry Code	ZZ
Unit Type	1
Unit Quantity	100
Originating Registry	ZZ
<b>Expected Result</b>	
100 AAUs originating from ZZ should be cancelled (transferred to cancellation account #7) successfully.	
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).	
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).	

<b>Test ID</b>	490
<b>Description</b>	
ITL will initiate Time Synchronization by calling the ProvideTime Web service.	
<b>Steps</b>	
The ITL will call the registry's ProvideTime Web service.	
<b>Expected Result</b>	
The time provided by the registry should be consistent with the time reflected by the ITL.	

<b>Test ID</b>	491
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following totals:	
accountType accountCommitPeriod unitType unitCount	100 0 1 4700
accountType accountCommitPeriod unitType unitCount	100 0 2 900
accountType accountCommitPeriod unitType unitCount	100 0 3 100
accountType accountCommitPeriod unitType unitCount	100 0 4 100
accountType accountCommitPeriod unitType unitCount	100 0 6 200
accountType accountCommitPeriod unitType unitCount	100 0 7 550
accountType accountCommitPeriod unitType unitCount	100 0 5 150
accountType accountCommitPeriod unitType unitCount	210 1 1 100
accountType accountCommitPeriod unitType unitCount	230 1 1 100
If requested by the ITL, the registry should return the following unit blocks:	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 4600 units of the units generated in Test ID 100 ZZ 1 100 1

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 900 units of the units generated in Test ID 101 ZZ 2 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from TEST ID 100 ZZ 3 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 4 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1 100 YY 1 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	9001 9200 BO 6 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 150 units from Test ID 306 BO 7 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1501 1700 BO 7 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	3001 3150 BO 5 100 1

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	4001 4200 BO 812 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 1 210 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 1 230 1

## 4.5 Expiry Date Change Transaction Tests

### 4.5.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptNotification AcceptITLNotice ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	ITL Notification Single Registry Transaction Behavior Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	Expiry Date Change (Type 9)

### 4.5.2 Test Steps

1. The ITL will send the registry a notification (Notification Type 9) through its AcceptITLNotice Web service alerting it that certain units require an expiry date change. In one case, the unit type specified is Type 6 (tCERs), which indicates that all tCERs must have their expiry date changed. In another case, the unit type specified is Type 7 (ICERs), and the Project Number of the Project for which the ICERs were issued is specified in the notification.
2. The registry will propose an Expiry Date Change transaction (Transaction Type 8).



3. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.
4. The ITL should call the registry's AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
5. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
6. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.

#### 4.5.3 Test Cases

<b>Test ID</b>	500
<b>Description</b>	
The ITL sends an Expiry Date Change notification (Type 9) to change the expiry date on all tCERs.	
<b>Relevant Notification Parameters</b>	
messageContent	Due to a change in the crediting period for the relevant CDM Project, the units specified within this message must have their expiry date changed.
notificationType	9
notificationStatus	1
unitType	6
targetValue	200
targetDate	2018-11-30
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Acquiring Registry Code	ZZ
Acquiring Account Type	100
Acquiring Account ID	1
Originating Registry Code	BO
LULUCF Activity	1
Project ID	12
Unit Serial Block Start	9001
Unit Serial Block End	9200
Expiry Date	2018-11-30
Unit Type	6
<b>Expected Result</b>	
The registry will change the expiry date of all tCERs in one or more transactions. The transaction proposal will include the notification identifier.	

<b>Test ID</b>	501
<b>Description</b>	
The ITL sends an Expiry Date Change notification (Type 9) to change the expiry date on all ICERs associated with CDM Project BO-329.	
<b>Relevant Notification Parameters</b>	
messageContent	Due to a change in the crediting period for the relevant CDM Project, the units specified within this message must have their expiry date changed.
notificationType	9
notificationStatus	1
projectNumber	BO-329
unitType	7
targetValue	150
targetDate	2029-11-25
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Acquiring Registry Code	ZZ
Acquiring Account Type	100
Acquiring Account ID	1
Originating Registry Code	BO
Unit Serial Block Start	301
Unit Serial Block End	450
LULUCF Activity	1
Project ID	329
Expiry Date	2029-11-25
Unit Type	7
<b>Expected Result</b>	
The registry will change the expiry date of all ICERs originating from Project BO-329 in one or more transactions. The transaction proposal will include the notification identifier.	

<b>Test ID</b>	590
<b>Description</b>	
ITL will initiate Time Synchronization by calling the ProvideTime Web service.	
<b>Steps</b>	
The ITL will call the registry's ProvideTime Web service.	
<b>Expected Result</b>	
The time provided by the registry should be consistent with the time reflected by the ITL.	

<b>Test ID</b>	591
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following totals:	
accountType accountCommitPeriod unitType unitCount	100 0 1 4700
accountType accountCommitPeriod unitType unitCount	100 0 2 900
accountType accountCommitPeriod unitType unitCount	100 0 3 100
accountType accountCommitPeriod unitType unitCount	100 0 4 100
accountType accountCommitPeriod unitType unitCount	100 0 6 200
accountType accountCommitPeriod unitType unitCount	100 0 7 550
accountType accountCommitPeriod unitType unitCount	100 0 5 150
accountType accountCommitPeriod unitType unitCount	210 1 1 100
accountType accountCommitPeriod unitType unitCount	230 1 1 100
If requested by the ITL, the registry should return the following unit blocks:	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 4600 units of the units generated in Test ID 100 ZZ 1 100 1

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 900 units of the units generated in Test ID 101 ZZ 2 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from TEST ID 100 ZZ 3 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 4 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1 100 YY 1 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	9001 9200 BO 6 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 150 units from Test ID 306 BO 7 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1501 1700 BO 7 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	3001 3150 BO 5 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	4001 4200 BO 812 100 1

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 1 210 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 1 230 1

## 4.6 Retirement Transaction Tests

### 4.6.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptNotification ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	Single Registry Transaction Behavior Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	None

### 4.6.2 Test Steps

1. The registry will propose a Retirement transaction (Transaction Type 5).
2. The registry will submit the proposal to the ITL's AcceptProposal, and verify that it receives a response indicating that the proposal was received.
3. The ITL should call the registry's AcceptNotification method to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
4. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
5. The registry will then call the ITL's AcceptNotification method to notify the ITL of its result.

### 4.6.3 Test Cases

<b>Test ID</b>	600
<b>Description</b>	
Retire all units save for 2000 AAUs originating from ZZ and 50 ERUs converted from AAUs.	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	ZZ
Acquiring Account Type	300
Acquiring Account ID	10
<b>Expected Result</b>	
<p>The registry will propose one or more retirement transaction(s) (Type 5).</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p> <p>All units in the holding account with the exception of 2000 AAUs originating from ZZ and 50 ERUs converted from AAUs will be transferred to the retirement account (Type 300).</p>	

<b>Test ID</b>	690
<b>Description</b>	
ITL will initiate Time Synchronization by calling the ProvideTime Web service.	
<b>Steps</b>	
The ITL will call the registry's ProvideTime Web service.	
<b>Expected Result</b>	
The time provided by the registry should be consistent with the time reflected by the ITL.	

<b>Test ID</b>	691
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following totals:	
accountType accountCommitPeriod unitType unitCount	100 0 1 2000
accountType accountCommitPeriod unitType unitCount	300 1 2 900
accountType accountCommitPeriod unitType unitCount	100 1 3 50
accountType accountCommitPeriod unitType unitCount	300 1 4 100
accountType accountCommitPeriod unitType unitCount	300 1 6 200
accountType accountCommitPeriod unitType unitCount	300 1 7 550
accountType accountCommitPeriod unitType unitCount	300 1 5 150
accountType accountCommitPeriod unitType unitCount	210 1 1 100
accountType accountCommitPeriod unitType unitCount	230 1 1 100
accountType accountCommitPeriod unitType unitCount	300 1 1 2700
accountType accountCommitPeriod unitType unitCount	300 1 3 50

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 2000 units of the units generated in Test ID 100 ZZ 1 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 900 units of the units generated in Test ID 101 ZZ 2 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 50 units from TEST ID 100 ZZ 3 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 4 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1 100 YY 1 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	9001 9200 BO 6 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 150 units from Test ID 306 BO 7 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1501 1700 BO 7 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	3001 3150 BO 5 300 1

(cont.)



<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	4001 4200 BO 812 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 1 210 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 1 230 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 2600 units of the units generated in Test ID 100 ZZ 1 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 50 units from TEST ID 100 ZZ 3 300 1

## 4.7 Replacement Transaction Tests

### 4.7.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptProposal AcceptNotification AcceptITLNotice ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	ITL Notification Single Registry Transaction Behavior Diagram External Transfer Behavior Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	Impending Expiry of tCER (Type 3) Reversal of Storage (Type 4) Non-submission of Certification Report for CDM Project (Type 5) Commitment Period Reserve (Type 7) Notification Update (Type 10)

### 4.7.2 Test Steps

1. The ITL will notify the registry of an impending expiry, reversal of storage, or a non-submission of certification report for a CDM Project. This message is sent to the registry's AcceptITLNotice method. The message will inform the registry that it must replace ICERs and will specify a notification identifier. This notification will have a status of 1.
2. The registry will propose a Replacement transaction (Transaction Type 6) as described in each test case.
3. The registry will submit the proposal to the ITL's AcceptProposal, and verify that it receives a response indicating that the proposal was received.
4. The ITL should call the registry's AcceptNotification method to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
5. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
6. The registry will then call the ITL's AcceptNotification method to notify the ITL of its result.
7. The ITL will notify the registry via AcceptITLNotice that they have satisfied this notification. This notice will reference the original notification identifier and a status of 3.

### 4.7.3 Test Cases

<b>Test ID</b>	700
<b>Description</b>	
Registry receives impending expiry notification for ICERs (Type 3). The registry will replace the 200 ICERs, which should be in the CP1 Retirement account with 200 AAUs from the holding account.	
<b>Relevant Notification Parameters</b>	
messageContent	ICERs held by this registry will expire within 30 days. The ICERs specified in this message must be replaced or cancelled before they expire.
notificationType	3
notificationStatus	1
projectNumber	BO-812
unitSerialBlockStart	4001
unitSerialBlockEnd	4200
originatingRegistryCode	BO
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	ZZ
Acquiring Account Type	421
Acquiring Account ID	12
Originating Registry Code	ZZ
Unit Type	1
Unit Quantity	200
First Block -- Replacing units: start	To be determined by registry
First Block -- Replacing units: end	To be determined by registry
Last Block -- Units to be replaced: start	4001
Last Block -- Units to be replaced: end	4200
originatingPartyCode	BO
<b>Expected Result</b>	
<p>The registry will propose a replacement transaction (Type 6).</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p> <p>200 AAUs from the holding account should be transferred to the tCER Replacement Account for Expiry (Type 411).</p>	

<b>Test ID</b>	701
<b>Description</b>	
Registry receives reversal of storage for CDM Project notification (Type 4). The registry will replace the affected 150 ICERs, which should be in the CP1 Retirement account with 150 AAUs from the holding account.	
<b>Relevant Notification Parameters</b>	
messageContent	A reversal of storage of greenhouse gas has occurred at a CDM Project and all transactions involving units associated with that Project are suspended until the specified number of units is replaced.
notificationType	4
notificationStatus	1
projectNumber	BO-329
targetValue	150
originatingRegistryCode	BO
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	ZZ
Acquiring Account Type	422
Acquiring Account ID	13
Originating Registry Code	ZZ
Unit Type	1
Unit Quantity	150
First Block -- Replacing units: start	To be determined by registry
First Block -- Replacing units: end	To be determined by registry
Last Block -- Units to be replaced: start	To be determined by registry
Last Block -- Units to be replaced: end	To be determined by registry
originatingPartyCode	BO
<b>Expected Result</b>	
<p>The registry will propose a replacement transaction (Type 6).</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p> <p>150 AAUs from the holding account should be transferred to the ICER Replacement Account for Reversal of Storage (Type 422).</p>	

<b>Test ID</b>	702
<b>Description</b>	
Registry receives Non-submission of Certification Report for CDM Project notification (Type 5). The registry is directed to replace the affected 200 ICERs, which should be in the CP1 Retirement account with 200 AAUs from the holding account. The registry will partially fulfill the notification direction by transferring only 120 AAUs into the replacement account.	
<b>Relevant Notification Parameters</b>	
messageContent	The CDM Executive Board has not received a Certification Report, and it has requested that all ICERs generated by the specified CDM be replaced. For the purposes of initialization, replace the ICERs with AAUs.
notificationType	5
notificationStatus	1
projectNumber	BO-1983
targetValue	200
originatingRegistryCode	BO
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	ZZ
Acquiring Account Type	423
Acquiring Account ID	14
Originating Registry Code	ZZ
Unit Type	1
Unit Quantity	120
First Block -- Replacing units: start	To be determined by registry
First Block -- Replacing units: end	To be determined by registry
Last Block -- Units to be replaced: start	To be determined by registry
Last Block -- Units to be replaced: end	To be determined by registry
originatingPartyCode	BO
<b>Expected Result</b>	
<p>The registry will propose a replacement transaction (Type 6).</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p> <p>120 AAUs from the holding account should be transferred to the ICER Replacement Account for Failure to Submit Certification Report (Type 423).</p>	

<b>Test ID</b>	703
<b>Description</b>	
Registry receives a notification update (Type 10) for the Non-submission of Certification Report for CDM Project notification in test 702. The registry will replace the remaining affected 80 ICERs, which should be in the CP1 Retirement account with 80 AAUs from the holding account.	
<b>Relevant Notification Parameters</b>	
messageContent	The CDM Executive Board has not received a Certification Report, and it has requested that all ICERs generated by the specified CDM be replaced.
notificationType	10
notificationStatus	2
projectNumber	BO-1983
targetValue	80
originatingRegistryCode	BO
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	ZZ
Acquiring Account Type	423
Acquiring Account ID	14
Originating Registry Code	ZZ
Unit Type	1
Unit Quantity	80
First Block -- Replacing units: start	To be determined by registry
First Block -- Replacing units: end	To be determined by registry
Last Block -- Units to be replaced: start	To be determined by registry
Last Block -- Units to be replaced: end	To be determined by registry
originatingPartyCode	BO
<b>Expected Result</b>	
<p>The registry will propose a replacement transaction (Type 6).</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p> <p>80 AAUs from the holding account should be transferred to the ICER Replacement Account for Failure to Submit Certification Report (Type 423).</p>	

<b>Test ID</b>	704	
<b>Description</b>		
ITL has sent two notifications (Type 7) that the registry's unit holdings are below the commitment period reserve and that it must acquire additional units. [The registry administrator will call the ITL operator to indicate the notification has been received by expliEUTLy stating the notification identifier.] Next, the third-party registry simulator will transfer the units required to fulfill the second notification and additional units to allow transactions in following tests.		
<b>Relevant Notification Parameters</b>		
messageContent	The account holdings for this registry are below the required CPR level. It is necessary for the registry to acquire the number of units specified in this message.	
notificationType	7	
notificationStatus	1	
targetValue - first notification	50	
targetValue - second notification	170	
<b>Relevant Test Parameters</b>		
Transferring Registry Code	YY	
Transferring Account Type	100	
Transferring Account ID	8888	
Acquiring Registry Code	ZZ	
Acquiring Account Type	100	
Acquiring Account ID	1	
Originating Registry Code	YY	
Unit Type	1	
Unit Serial Block Start	1001	
Unit Serial Block End	2000	
<b>Expected Result</b>		
<p>The registry will log the notification and call the ITL operator to report the notification identifier and receipt of an intelligible message.</p> <p>The third-party registry simulator will transfer 1000 AAUs, more than the 50 units required to meet the target value indicated in the notification.</p> <p>After receiving the AcceptProposal call, the Acquiring Registry Code (ZZ) sends an acknowledgement that it has received the proposal. ZZ approves the transaction and finalizes it in the database.</p> <p>The acquiring registry's notification to the ITL should include a Transaction Status Code of 8 (Accepted).</p>		

<b>Test ID</b>	790
<b>Description</b>	
ITL will initiate Time Synchronization by calling the ProvideTime Web service.	
<b>Steps</b>	
The ITL will call the registry's ProvideTime Web service.	
<b>Expected Result</b>	
The time provided by the registry should be consistent with the time reflected by the ITL.	

<b>Test ID</b>	791
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following totals:	
accountType accountCommitPeriod unitType unitCount	100 0 1 2450
accountType accountCommitPeriod unitType unitCount	300 1 2 900
accountType accountCommitPeriod unitType unitCount	100 0 3 50
accountType accountCommitPeriod unitType unitCount	300 1 4 100
accountType accountCommitPeriod unitType unitCount	300 1 6 200
accountType accountCommitPeriod unitType unitCount	300 1 7 550
accountType accountCommitPeriod unitType unitCount	300 1 5 150

(cont.)



<b>Expected Result (cont.)</b>	
The registry should return the following totals (cont.):	
accountType accountCommitPeriod unitType unitCount	210 1 1 100
accountType accountCommitPeriod unitType unitCount	230 1 1 100
accountType accountCommitPeriod unitType unitCount	300 1 1 2700
accountType accountCommitPeriod unitType unitCount	300 1 3 50
accountType accountCommitPeriod unitType unitCount	411 1 1 200
accountType accountCommitPeriod unitType unitCount	422 1 1 150
accountType accountCommitPeriod unitType unitCount	423 1 1 200
If requested by the ITL, the registry should return the following unit blocks:	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 1450 units of the units generated in Test ID 100 ZZ 1 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 900 units of the units generated in Test ID 101 ZZ 2 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 50 units from TEST ID 100 ZZ 3 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 4 300 1

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1 100 YY 1 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	9001 9200 BO 6 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 150 units from Test ID 306 BO 7 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1501 1700 BO 7 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	3001 3150 BO 5 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	4001 4200 BO 812 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 1 210 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 1 230 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 2600 units of the units generated in Test ID 100 ZZ 1 300 1

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 50 units from TEST ID 100 ZZ 3 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 200 units of the units generated in Test ID 100 ZZ 1 411 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 150 units of the units generated in Test ID 100 ZZ 1 422 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 200 units of the units generated in Test ID 100 ZZ 1 423 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1001 2000 YY 1 100 1

## 4.8 Additional Tests

This section contains tests to ensure the registry has correctly implemented the functional components of the Data Exchange Standards for the commitment period life cycle. The section includes a test to check implementation of the Web service to support the general message administrative function and continues with testing to evaluate registry performance in a subsequent commitment period.

### 4.8.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptNotification AcceptProposal AcceptITLNotice ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ProvideAuditTrail ReceiveReconciliationResult

(cont.)

<b>Administrative Functions</b>	Time Synchronization General Messages
<b>DES Behavior Diagrams</b>	Time Synchronization Diagram Single Registry Transaction Behavior Diagram External Transfer Behavior Diagram Reconciliation Behavior Diagram Send Reconciliation Results Behavior ITL Notification Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	Non-Compliance Notification (Type 2) Unit Carry-over (Type 8)

#### 4.8.2 Test Steps

1. The ITL will notify the registry that it has units eligible for carry-over to commitment period 2.
2. The registry will propose a carry-over transaction (Transaction Type 7).
3. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.
4. The ITL should call the registry's AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
5. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
6. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.
7. The ITL will send a general message to inform the registry that it is switching to commitment period 2.
8. The registry will switch to commitment period 2.
9. The registry will repeat the issuance tests as described for test IDs 802, 803, and 804.
10. The registry will then receive a notification to mark the start of the next test. The registry will proceed through the remaining tests, concluding with a full reconciliation through the audit trail phase for the unit blocks specified.
11. If an EU member registry is initializing, the registry will proceed to Section 5 testing once reconciliation is complete.

### 4.8.3 Test Cases

<b>Test ID</b>	800
<b>Description</b>	
Registry receives a Unit Carry-over notification (Type 8) with directions to carry-over 800 AAUs and 50 ERUs converted from AAUs in the Holding Account. The registry will propose one or more Carry-over transaction(s) to fulfill the notification direction.	
<b>Relevant Notification Parameters</b>	
messageContent	Due to the end of Commitment Period, the units specified within this message may be carried-over to the subsequent Commitment Period. For the purposes of initialization, carry-over 800 AAUs and 50 ERUs converted from AAUs.
notificationType	8
notificationStatus	1
targetValue	850
commitPeriod	2
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	ZZ
Originating Registry Code	ZZ
Unit Type	1
Unit Total	800
originatingRegistryCode	ZZ
Unit Type	3
Unit Total	50
originatingRegistryCode	ZZ
originalCommitPeriod	1
applicableCommitPeriod	2
<b>Expected Result</b>	
<p>The registry will propose one or more carry-over transaction(s) (Type 7).</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p> <p>800 AAUs and 50 ERUs converted from AAUs should now have applicable commitment period = 2.</p>	

<b>Test ID</b>	801
<b>Description</b>	
<p>The ITL will call the AcceptMessage Web service at the registry to deliver a general message to the registry administrator. The message will inform the registry administrator that the ITL is switching to the second commitment period. The registry will now function as if it were in the second commitment period. The registry administrator will call the ITL operator to indicate receipt of the message before proceeding to the next test.</p>	
<b>Expected Result</b>	
<p>The registry will call the ITL operator to indicate receipt of the message and report the notification identifier.</p>	

<b>Description</b>	
<p>Repeat the issuance tests 100, 101 and 102 in the second commitment period. The tests are the same with the exception that the original commitment period for the units is 2 and the serial blocks issued are unique.</p>	
<b>CP 1 Test ID</b>	<b>CP 2 Test ID</b>
100	802
101	803
102	804
<b>Expected Result</b>	
<p>The units issued should have an original commitment period = 2.</p>	

<b>Test ID</b>	805
<b>Description</b>	
Registry will receive a notification (Type 2) of non-compliance with instructions to cancel 100 AAUs. The registry will fulfill the request.	
<b>Relevant Notification Parameters</b>	
Message Content	The Compliance Committee has determined that the Party is in non-compliance with its emissions target under Articles 3.7 and 3.8 of the Kyoto Protocol. Units must be cancelled in order to bring the Party into compliance. For the purposes of initialization, cancel 100 AAUs with original commitment period =2 from the Holding Account.
Notification Type	2
Target Value	100
commitPeriod	2
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Acquiring Registry Code	ZZ
Acquiring Account Type	220 (Non-compliance Cancellation Account)
Acquiring Account ID	16
Unit Type	1
Unit Quantity	100
<b>Expected Result</b>	
<p>100 AAUs from the Holding Account should be cancelled (transferred to cancellation account type 220) successfully.</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p> <p>The registry will log the notification update (requirement fulfilled).</p>	

<b>Test ID</b>	806
<b>Description</b>	
Retire all units with applicable commitment period =2 save for 2000 AAUs with originating commitment period = 2.	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	ZZ
Acquiring Account Type	300
Acquiring Account ID	19
<b>Expected Result</b>	
<p>The registry will propose one or more retirement transaction(s) (Type 9).</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p> <p>All units in the holding account with commitment period =2 with the exception of 2000 AAUs with originating commitment period =2 will be transferred to the retirement account (Type 300) for commitment period 2.</p>	

<b>Test ID</b>	890
<b>Description</b>	
ITL will initiate Time Synchronization by calling the ProvideTime Web service.	
<b>Steps</b>	
The ITL will call the registry's ProvideTime Web service.	
<b>Expected Result</b>	
The time provided by the registry should be consistent with the time reflected by the ITL.	



<b>Test ID</b>	891
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following totals:	
accountType accountCommitPeriod unitType unitCount	100 0 1 3650
accountType accountCommitPeriod unitType unitCount	300 1 2 900
accountType accountCommitPeriod unitType unitCount	300 2 3 50
accountType accountCommitPeriod unitType unitCount	300 1 4 100
accountType accountCommitPeriod unitType unitCount	300 1 6 200
accountType accountCommitPeriod unitType unitCount	300 1 7 550
accountType accountCommitPeriod unitType unitCount	300 1 5 150
accountType accountCommitPeriod unitType unitCount	210 1 1 100
accountType accountCommitPeriod unitType unitCount	230 1 1 100
accountType accountCommitPeriod unitType unitCount	300 1 1 2700
accountType accountCommitPeriod unitType unitCount	300 1 3 50

(cont.)

<b>Expected Result (cont.)</b>	
The registry should return the following totals (cont.):	
accountType accountCommitPeriod unitType unitCount	411 1 1 200
accountType accountCommitPeriod unitType unitCount	422 1 1 150
accountType accountCommitPeriod unitType unitCount	423 1 1 200
accountType accountCommitPeriod unitType unitCount	220 2 1 100
accountType accountCommitPeriod unitType unitCount	300 2 1 3700
accountType accountCommitPeriod unitType unitCount	300 2 2 1000
If requested by the ITL, the registry should return the following unit blocks:	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 650 units of the units generated in Test ID 100 ZZ 1 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 900 units of the units generated in Test ID 101 ZZ 2 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 50 units from TEST ID 100 ZZ 3 300 2
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 4 300 1

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1 100 YY 1 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	9001 9200 BO 6 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 150 units from Test ID 306 BO 7 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1501 1700 BO 7 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	3001 3150 BO 5 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	4001 4200 BO 812 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 1 210 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 1 230 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 2600 units of the units generated in Test ID 100 ZZ 1 300 1

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 50 units from TEST ID 100 ZZ 3 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 200 units of the units generated in Test ID 100 ZZ 1 411 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 150 units of the units generated in Test ID 100 ZZ 1 422 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 200 units of the units generated in Test ID 100 ZZ 1 423 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1001 2000 YY 1 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 800 units of the units generated in Test ID 100 ZZ 1 300 2
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 2,000 units of the units generated in Test ID 802 ZZ 1 100 2
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units of the units generated in Test ID 802 ZZ 1 220 2
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 2900 units of the units generated in Test ID 802 ZZ 1 300 2

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 1000 units of the units generated in Test ID 803 ZZ 2 300 2

<b>Test ID</b>	892 [EU ETS registries bypass]
<b>Description</b>	
YY proposes a transaction transferring 100 AAUs originating from YY to the ZZ registry. Registry ZZ will accept the proposal and finalize the transaction in its database. However, registry YY does not complete the transaction before the ITL initializes a reconciliation. The registry will then be directed to roll-back the transaction as part of manual intervention.	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	YY
Transferring Account Type	100
Transferring Account ID	8888
Acquiring Registry Code	ZZ
Acquiring Account Type	100
Acquiring Account ID	1
Originating Registry Code	YY
Unit Serial Block Start	101
Unit Serial Block End	200
Unit Type	1
<b>Expected Result</b>	
<p>After receiving the AcceptProposal call, the acquiring registry (ZZ) should send an acknowledgement that it has received the proposal. ZZ approves the transaction and finalizes it in its database. The acquiring registry's (ZZ) notification to the ITL should include a Transaction Status Code of 8 (Accepted).</p> <p>The ITL initiates a reconciliation test immediately after receiving the ZZ's notification on the AcceptNotification Web service and, hypothetically, before the transferring registry (YY) finalizes the transaction and submits a notification with transaction status code of 4 (Completed). For the purposes of initialization testing, the transaction is cancelled without notifying the acquiring registry (ZZ).</p> <p>The reconciliation finds an inconsistency at the unit totals level and continues automatically through the audit trail phase, specifying the units involved in the transaction described in this test. The ITL operator will call the registry administrator with instructions to roll-back the transaction. The registry administrator will perform the manual intervention.</p> <p>The ITL will send notification that the reconciliation is "ITL Completed with Manual Intervention" (Status Code 6) and will begin a new reconciliation that should be successful. The new reconciliation must reach reconciliation status code 2: "Validated."</p> <p>The manual intervention should be logged.</p>	

<b>Test ID</b>	893 [EU ETS registries bypass]
<b>Description</b>	
The registry submits its transaction log, reconciliation history log, notification log, and message archive.	
<b>Expected Result</b>	
The ITL operator can verify randomly selected transactions resulting from Annex H testing. The logs will reflect only the registry activity described in Annex H and will conform with Data Logging Specifications described in Section 7 of the Data Exchange Standards.	

## 15 5. EU ETS Member Registry Test Cases

### 5.0 General Information

#### 5.0.1 Test Sequence

Members of the EU Emissions Trading Scheme must proceed through and pass each of the tests in the above test series (100-800) with the exception of Test IDs 892 and 893 before beginning the 1000 series tests. The 1000 series tests are in addition to the tests in Section 4.

#### 5.0.2 Applicable Commitment Period

The 1000 series of tests follow-on from the above tests and are conducted assuming an applicable commitment period = 2.

### 5.1 Account Management Web Services

#### 5.1.1 Components Tested

<b>ITL Web Services</b>	AcceptNotification
<b>Registry Web Services</b>	ReceiveAccountOperationOutcome
<b>STL Web Services</b>	CreateAccount UpdateAccount CloseAccount UpdateVerifiedEmissions ReceiveAccountOperationOutcome
<b>Administrative Functions</b>	None
<b>DES Behavior Diagrams</b>	None
<b>Notification Types</b>	None

#### 5.1.2 Test Steps

1. The registry will call the appropriate ITL Web service to reach the EUTL Web service with a proposal. The registry communicates with the EUTL via the ITL at all times.
2. The registry will submit the proposal to the ITL's Web services as specified in each test, and verify that it receives a response indicating that the proposal was received.
3. The ITL should call the registry's AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
4. Upon receiving EUTL notification via the ITL, the registry will finalize the transaction in its database or terminate the transaction depending on the notification sent from the EUTL.

#### 5.1.3 Test Cases

<b>Test ID</b>	1100	
<b>Description</b>		
The EU registry will call the CreateAccountRequest Web service to create a new Operator Holding Account (Type 120).		
<b>Relevant Message Parameters</b>		
correlationID	7	
accoutType	120	
accountIdentifier	3	
identifierInReg	ACCOUNT1	
installationIdentifier	9999999999999999 (Registry may specify)	
country	ZZ	
<b>Expected Result</b>		
<p>The registry will submit a well-formed message with all relevant data elements.</p> <p>The EUTL will respond that the create account request was accepted (ResultIdentifier = 1). The ITL will inform the registry of the result via the ReceiveAccountOperationOutcome Web service.</p> <p>The account should be able to acquire units.</p>		

<b>Test ID</b>	1101	
<b>Description</b>		
The registry will call the UpdateAccountRequest Web service to update the account contact persons address for the Operator Holding Account created in test 1100.		
<b>Relevant Message Parameters</b>		
correlationID	7	
accoutType	120	
accountIdentifier	3	
identifierInReg	ACCOUNT1	
installationIdentifier	9999999999999999 (Registry must specify same used in Test ID 1100)	
country	ZZ	
<b>Expected Result</b>		
<p>The registry submits a well-formed message with all required data elements.</p> <p>The EUTL rejects the Update Account Request (ResultIdentifier = 0) and includes an appropriate response code. The ITL will inform the registry of the result via the ReceiveAccountOperationOutcome Web service.</p> <p>The original address for the account contact person is not modified from Test ID 1100.</p>		



<b>Test ID</b>	1102
<b>Description</b>	
The registry calls the CloseAccountRequest to close the Operator Holding Account (Account Identifier 3) created in test 1100.	
<b>Relevant Message Parameters</b>	
CorrelationID	5
AccountIdentifier	3
<b>Expected Result</b>	
<p>The registry submits a well-formed message with all required data elements.</p> <p>The EUTL rejects the Update Account Request (ResultIdentifier = 0) and includes an appropriate response code. The ITL will inform the registry of the result via the ReceiveAccountOperationOutcome Web service.</p> <p>The Operator Holding Account remains open.</p>	

<b>Test ID</b>	1103
<b>Description</b>	
The registry calls the UpdateVerifiedEmissionsRequest Web service to report emissions for Installation1.	
<b>Relevant Message Parameters</b>	
CorrelationID	55
YearInCommitPeriod	2007
InstallationIdentifier	9999999999999999
VerifiedEmissions	1234
<b>Expected Result</b>	
<p>The registry submits a well-formed message with all required data elements.</p> <p>The EUTL accepts the Update Account Request (ResultIdentifier = 1). The ITL will inform the registry of the result via the ReceiveAccountOperationOutcome Web service.</p>	

## 5.2 Internal Transfer Transaction Tests

### 5.2.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptNotification AcceptProposal ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ProvideAuditTrail ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	Single Registry Transaction Behavior Diagram External Transfer Behavior Diagram Discrepancy Notification Sequence Diagram Terminate Transaction Sequence Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	None

### 5.2.2 Test Steps

1. The registry will propose an internal transfer transaction (Transaction Type 10).
2. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.
3. The ITL should call the registry's AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
4. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
5. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.

### 5.2.3 Test Cases

<b>Test ID</b>	1200
<b>Description</b>	
Transfer 1000 AAUs with applicable Commitment Period = 2 from Holding Account (Type 100, Account ID 1) into Operator Holding Account (Type 120, Account ID 3).	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	ZZ
Acquiring Account Type	120
Acquiring Account ID	3
Unit Type	1
Unit Quantity	1000
<b>Expected Result</b>	
1000 AAUs with applicable Commitment Period = 2 should be successfully transferred from holding account #1 to operator holding account #3.	
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).	
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).	

<b>Test ID</b>	1201
<b>Description</b>	
Transfer 300 AAUs with applicable Commitment Period = 2 from Holding Account (Type 100, Account ID 1) into Operator Holding Account (Type 120, Account ID 3). ITL approves transaction, STL rejects transaction.	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	ZZ
Acquiring Account Type	120
Acquiring Account ID	3
Unit Type	1
Unit Quantity	300
<b>Expected Result</b>	
300 AAUs should not be successfully transferred from holding account #1 to operator holding account #3.	
The ITL passes the STL notification to the registry including a Transaction Status Code of 10 (Checked, Discrepancy).	
The registry's notification to the ITL should include a Transaction Status Code of 7 (Cancelled).	

<b>Test ID</b>	1202
<b>Description</b>	
Transfer 400 AAUs with applicable Commitment Period = 2 from Holding Account (Type 100, Account ID 1) into Operator Holding Account (Type 120, Account ID 3). ITL approves transaction, but STL does not respond to proposal.	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	ZZ
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	ZZ
Acquiring Account Type	120
Acquiring Account ID	3
Unit Type	1
Unit Quantity	400
<b>Expected Result</b>	
400 AAUs should not be successfully transferred from holding account #1 to operator holding account #3.	
The ITL checks the transaction proposal and forwards it to the STL simulator. The STL simulator does not respond.	
The registry should not commit a database change and the 400 units should remain in the Holding Account (Type 100, Account ID 1).	

<b>Test ID</b>	1290
<b>Description</b>	
ITL will initiate Time Synchronization by calling the ProvideTime Web service.	
<b>Steps</b>	
The ITL will call the registry's ProvideTime Web service.	
<b>Expected Result</b>	
The time provided by the registry should be consistent with the time reflected by the ITL.	

<b>Test ID</b>	1291
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following totals:	
accountType	100
accountCommitPeriod	0
unitType	1

unitCount	2650
accountType accountCommitPeriod unitType unitCount	300 1 2 900
accountType accountCommitPeriod unitType unitCount	300 2 3 50
accountType accountCommitPeriod unitType unitCount	300 1 4 100
accountType accountCommitPeriod unitType unitCount	300 1 6 200
accountType accountCommitPeriod unitType unitCount	300 1 7 550
accountType accountCommitPeriod unitType unitCount	300 1 5 150

(cont.)

<b>Expected Result (cont.)</b>	
The registry should return the following totals (cont.):	
accountType accountCommitPeriod unitType unitCount	210 1 1 100
accountType accountCommitPeriod unitType unitCount	230 1 1 100
accountType accountCommitPeriod unitType unitCount	300 1 1 2700
accountType accountCommitPeriod unitType unitCount	300 1 3 50
accountType accountCommitPeriod unitType unitCount	411 1 1 200
accountType accountCommitPeriod unitType unitCount	422 1 1 150
accountType accountCommitPeriod unitType unitCount	423 1 1 200

accountType accountCommitPeriod unitType unitCount	220 2 1 100
accountType accountCommitPeriod unitType unitCount	300 2 1 3700
accountType accountCommitPeriod unitType unitCount	300 2 2 1000
accountType accountCommitPeriod unitType unitCount	120 0 1 1000
If requested by the ITL, the registry should return the following unit blocks:	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 650 units of the units generated in Test ID 100 ZZ 1 100 1

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 900 units of the units generated in Test ID 101 ZZ 2 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 50 units from TEST ID 100 ZZ 3 300 2
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 4 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1 100 YY 1 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	9001 9200 BO 6 300 1
unitSerialBlockStart	Determined by registry to include 150 units from

unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Test ID 306 BO 7 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1501 1700 BO 7 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	3001 3150 BO 5 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	4001 4200 BO 812 300 1

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 1 210 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units from Test ID 101 ZZ 1 230 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 2600 units of the units generated in Test ID 100 ZZ 1 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 50 units from TEST ID 100 ZZ 3 300 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 200 units of the units generated in Test ID 100 ZZ 1 411 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode	Determined by registry to include 150 units of the units generated in Test ID 100 ZZ

unitType AccountType ApplicableCommitmentPeriod	1 422 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 200 units of the units generated in Test ID 100 ZZ 1 423 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1001 2000 YY 1 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 800 units of the units generated in Test ID 100 ZZ 1 300 2

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 1,000 units of the units generated in Test ID 802 ZZ 1 100 2
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units of the units generated in Test ID 802 ZZ 1 220 2
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 2900 units of the units generated in Test ID 802 ZZ 1 300 2
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 1000 units of the units generated in Test ID 803 ZZ 2 300 2
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 1,000 units of the units generated in Test ID 802 ZZ 1 120 2



<b>Test ID</b>	1292
<b>Description</b>	
YY proposes a transaction transferring 100 AAUs originating from YY to the ZZ registry. Registry ZZ will accept the proposal and finalize the transaction in its database. However, registry YY does not complete the transaction before the ITL initializes a reconciliation. The registry will then be directed to roll-back the transaction as part of manual intervention.	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	YY
Transferring Account Type	100
Transferring Account ID	8888
Acquiring Registry Code	ZZ
Acquiring Account Type	100
Acquiring Account ID	1
Originating Registry Code	YY
Unit Serial Block Start	101
Unit Serial Block End	200
Unit Type	1
<b>Expected Result</b>	
<p>After receiving the AcceptProposal call, the acquiring registry (ZZ) should send an acknowledgement that it has received the proposal. ZZ approves the transaction and finalizes it in its database. The acquiring registry's notification to the ITL should include a Transaction Status Code of 8 (Accepted).</p> <p>The ITL initiates a reconciliation test immediately after receiving the ZZ's notification on the AcceptNotification Web service and, hypothetically, before the transferring registry (YY) finalizes the transaction and submits a notification with transaction status code of 4 (Completed). For the purposes of initialization testing, the transaction is cancelled without notifying the acquiring registry (ZZ).</p> <p>The reconciliation finds an inconsistency at the unit totals level and continues automatically through the audit trail phase, specifying the units involved in the transaction described in this test. The ITL operator will call the registry administrator with instructions to roll-back the transaction. The registry administrator will perform the manual intervention.</p> <p>The ITL will send notification that the reconciliation is "STL Completed with Manual Intervention" (Status Code 6) and begin a new reconciliation that should be successful. The new reconciliation must reach reconciliation status code 2: "Validated."</p> <p>The manual intervention should be logged.</p>	

<b>Test ID</b>	1293
<b>Description</b>	
The registry submits its transaction log, reconciliation history log, notification log, and message archive.	
<b>Expected Result</b>	
The ITL operator can verify randomly selected transactions resulting from Annex H testing. The logs will reflect only the registry activity described in Annex H and will conform with Data Logging Specifications described in Section 7 of the Data Exchange Standards.	

## 16 6. Test Cases (CDM Registry)

### 6.0 General Information

The registry must begin Section 6 with a database clear of unit holdings and transaction records, but must reflect the required accounts specified in Figure 2.2: CDM Registry Accounts, and limits specified in Figure 2.5: Issuance, Conversion, and Retirement Limits.

#### 6.0.1 Confirmation Reconciliation

After each test group, the ITL will request that the registry provide its current totals by account and unit type, and subsequently its current unit block holdings. No optional parameters will be specified in the totals request. The unit blocks specified as part of the unit block request will include all blocks once generated as testing progresses. The confirmation reconciliation will monitor the registry's progress through initialization by ensuring the registry reflects the state of the data specified in the reconciliation test at the end of each test series. The ITL will call the registry's ProvideTime Web service before each confirmation reconciliation.

#### 6.0.2 Test Parameters

Data elements required in the DES for WSDLs and Web service communication are to be included during transactions and notifications encountered during test scenarios. However, only the data elements unique to the test are specified in Annex H for each test as entries in two sections: Relevant Notification Parameters and Relevant Transaction Parameters. The information provided under each test description is sufficient for the registry to generate messages with all required data elements; elements not specified are obvious or will be provided in preceding notifications. Notification identifiers have not been specified, as the registry must verify receipt of notifications by proposing required transactions specifying the notification identifier provided by the ITL.

#### 6.0.3 Commitment Period

All tests cases are assumed to take place during Commitment Period 1 unless otherwise stated.

### 6.1 Issuance Transaction Tests

#### 6.1.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptNotification ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	None
<b>DES Behavior Diagrams</b>	Single Registry Transaction Behavior Diagram Time Synchronization
<b>Notification Types</b>	None

### 6.1.2 Test Steps

Steps:

1. The registry will propose an issuance transaction (Transaction Type 1).
2. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.
3. The ITL should call the registry's AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
4. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
5. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.

### 6.1.3 Test Cases

<b>Test ID</b>	2100
<b>Description</b>	
Issue 5,000 ICERs originating from Project BR-3	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	CDM
Acquiring Registry Code	CDM
Acquiring Account Type	110 (Pending Account)
Acquiring Account ID	5
Originating Registry Code	BR
Unit Type	7
LULUCF Activity	1
Project Identifier	3
Expiry Date	2028-01-16
Unit Quantity	5,000
<b>Expected Result</b>	
5,000 ICERs successfully issued to Pending Account #5.  The ITL's notification to the CDM registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).  The CDM registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).	

<b>Test ID</b>	2101
<b>Description</b>	
Issue 20,000 ICERs originating from Project BR-2345 to Pending Account	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	CDM
Acquiring Registry Code	CDM
Acquiring Account Type	110
Acquiring Account ID	5
Originating Registry Code	BR
Unit Type	7
LULUCF Activity	1
Project Identifier	2345
Expiry Date	2028-01-16
Unit Quantity	20,000
<b>Expected Result</b>	
20,000 ICERs successfully issued to Pending Account (Account ID #5).	
The ITL's notification to the CDM registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).	
The CDM registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).	

<b>Test ID</b>	2102	
<b>Description</b>		
Issue 2,200 tCERs originating from Project BR-6 to Pending Account		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	CDM	
Acquiring Registry Code	CDM	
Acquiring Account Type	110	
Acquiring Account ID	5	
Originating Registry Code	BR	
Unit Type	6	
LULUCF Activity	1	
Project Identifier	6	
Expiry Date	2017-12-31	
Unit Quantity	2,200	
<b>Expected Result</b>		
2,200 tCERs successfully issued to Pending Account (Account ID #5).		
The ITL's notification to the CDM registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).		
The CDM registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).		

<b>Test ID</b>	2103	
<b>Description</b>		
Issue 16,000 CERs originating from Project IN-4 to Pending Account		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	CDM	
Acquiring Registry Code	CDM	
Acquiring Account Type	110	
Acquiring Account ID	5	
Originating Registry Code	IN	
Unit Type	5	
Project Identifier	4	
Unit Quantity	16,000	
<b>Expected Result</b>		
16,000 CERs successfully issued to Pending Account (Account ID #5).		
The ITL's notification to the CDM registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).		
The CDM registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).		

<b>Test ID</b>	2104
<b>Description</b>	
Issue 10,000,000 tCERs originating from Project IN-2 to Pending Account	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	CDM
Acquiring Registry Code	CDM
Acquiring Account Type	110
Acquiring Account ID	5
Originating Registry Code	IN
Unit Type	6
LULUCF Activity	1
Project Identifier	2
Expiry Date	2017-12-31
Unit Quantity	10,000,000
<b>Expected Result</b>	
<p>10,000,000 tCERs successfully issued to Pending Account (Account ID #5).</p> <p>The ITL's notification to the CDM registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The CDM registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p>	

<b>Test ID</b>	2190
<b>Description</b>	
ITL will initiate Time Synchronization by calling the ProvideTime Web service.	
<b>Steps</b>	
The ITL will call the registry's ProvideTime Web service.	
<b>Expected Result</b>	
The time provided by the registry should be consistent with the time reflected by the ITL.	

<b>Test ID</b>	2191
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following totals:	
accountType accountCommitPeriod unitType unitCount	110 0 5 16000
accountType accountCommitPeriod unitType unitCount	110 0 6 10002200
accountType accountCommitPeriod unitType unitCount	110 0 7 25000
If requested by the ITL, the registry should return the following unit blocks:	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 5,000 units generated in Test ID 2100 BR 7 110 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 20,000 units generated in Test ID 2101 7 BR 110 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 2,200 units generated in Test ID 2102 BR 6 110 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 16,000 units generated in Test ID 2103 IN 5 110 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 10,000,000 units generated in Test ID 2104 IN 6 110 1

## 6.2 Internal Transfer Transaction Tests

### 6.2.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptNotification ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	Single Registry Transaction Behavior Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	None

### 6.2.2 Test Steps

1. The registry will propose an internal transfer transaction (Transaction Type 10).
2. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.
3. The ITL should call the registry's AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
4. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
5. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.



### 6.2.3 Test Cases

<b>Test ID</b>	2200	
<b>Description</b>		
Transfer 5,000 ICERs associated with Project BR-3 from Pending Account (Type 110, Account ID 5) into the NZ Holding Account (Type 100, Account ID 3).		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	CDM	
Transferring Account Type	110	
Transferring Account ID	5	
Acquiring Registry Code	CDM	
Acquiring Account Type	100	
Acquiring Account ID	3	
Originating Registry Code	BR	
Unit Type	7	
LULUCF Activity	1	
Project Identifier	3	
Unit Quantity	5000	
<b>Expected Result</b>		
5,000 ICERs should be successfully transferred from pending account (Account ID #5) to holding account (Account ID #3).		
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).		
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).		

<b>Test ID</b>	2201
<b>Description</b>	
Transfer 7,500 ICERs associated with Project BR-2345 from Pending Account (Type 110, Account ID 5) into the GB Holding Account (Type 100, Account ID 4).	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	CDM
Transferring Account Type	110
Transferring Account ID	5
Acquiring Registry Code	CDM
Acquiring Account Type	100
Acquiring Account ID	4
Originating Registry Code	BR
Unit Type	7
LULUCF Activity	1
Project Identifier	2345
Unit Quantity	7500
<b>Expected Result</b>	
7,500 ICERs should be successfully transferred from pending account (Account ID #5) to holding account (Account ID #4).	
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).	
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).	

<b>Test ID</b>	2202	
<b>Description</b>		
Transfer 2,500 ICERs associated with Project BR-2345 from Pending Account (Type 110, Account ID 5) into the NZ Holding Account (Type 100, Account ID 3).		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	CDM	
Transferring Account Type	110	
Transferring Account ID	5	
Acquiring Registry Code	CDM	
Acquiring Account Type	100	
Acquiring Account ID	3	
Originating Registry Code	BR	
Unit Type	7	
LULUCF Activity	1	
Project Identifier	2345	
Unit Quantity	2500	
<b>Expected Result</b>		
2,500 ICERs should be successfully transferred from pending account (Account ID #5) to holding account (Account ID #3).		
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).		
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).		

<b>Test ID</b>	2203
<b>Description</b>	
Transfer 10,000 ICERs associated with Project BR-2345 from Pending Account (Type 110, Account ID 5) into the BR Holding Account (Type 100, Account ID 1).	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	CDM
Transferring Account Type	110
Transferring Account ID	5
Acquiring Registry Code	CDM
Acquiring Account Type	100
Acquiring Account ID	1
Originating Registry Code	BR
Unit Type	1
LULUCF Activity	1
Project Identifier	2345
Unit Quantity	10000
<b>Expected Result</b>	
10,000 ICERs should be successfully transferred from pending account (Account ID #5) to holding account (Account ID #1).	
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).	
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).	

<b>Test ID</b>	2204
<b>Description</b>	
Transfer 2,000 tCERs associated with Project BR-6 from Pending Account (Type 110, Account ID 5) into the GB Holding Account (Type 100, Account ID 4).	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	CDM
Transferring Account Type	110
Transferring Account ID	5
Acquiring Registry Code	CDM
Acquiring Account Type	100
Acquiring Account ID	4
Originating Registry Code	BR
Unit Type	6
LULUCF Activity	1
Project Identifier	6
Unit Quantity	2000
<b>Expected Result</b>	
2,000 tCERs should be successfully transferred from pending account (Account ID #5) to holding account (Account ID #4).	
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).	
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).	

<b>Test ID</b>	2205	
<b>Description</b>		
Transfer 10,000 CERs associated with Project IN-4 from Pending Account (Type 110, Account ID 5) into the GB Holding Account (Type 100, Account ID 4).		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	CDM	
Transferring Account Type	110	
Transferring Account ID	5	
Acquiring Registry Code	CDM	
Acquiring Account Type	100	
Acquiring Account ID	4	
Originating Registry Code	IN	
Unit Type	5	
LULUCF Activity	1	
Project Identifier	4	
Unit Quantity	10000	
<b>Expected Result</b>		
10,000 CERs should be successfully transferred from pending account (Account ID #5) to holding account (Account ID #4).		
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).		
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).		

<b>Test ID</b>	2206	
<b>Description</b>		
Transfer 6,000 CERs associated with Project IN-4 from Pending Account (Type 110, Account ID 5) into the IN Holding Account (Type 100, Account ID 2).		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	CDM	
Transferring Account Type	110	
Transferring Account ID	5	
Acquiring Registry Code	CDM	
Acquiring Account Type	100	
Acquiring Account ID	2	
Originating Registry Code	IN	
Unit Type	5	
LULUCF Activity	1	
Project Identifier	4	
Unit Quantity	6000	
<b>Expected Result</b>		
6,000 CERs should be successfully transferred from pending account (Account ID #5) to holding account (Account ID #2).		
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).		
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).		

<b>Test ID</b>	2207
<b>Description</b>	
Transfer 9,500,000 tCERs associated with Project IN-2 from Pending Account (Type 110, Account ID 5) into the GB Holding Account (Type 100, Account ID 4).	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	CDM
Transferring Account Type	110
Transferring Account ID	5
Acquiring Registry Code	CDM
Acquiring Account Type	100
Acquiring Account ID	4
Originating Registry Code	IN
Unit Type	6
LULUCF Activity	1
Project Identifier	2
Unit Quantity	9500000
<b>Expected Result</b>	
9,500,000 tCERs should be successfully transferred from pending account (Account ID #5) to holding account (Account ID #4).	
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).	
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).	



<b>Test ID</b>	2208
<b>Description</b>	
Transfer 500,000 tCERs associated with Project IN-2 from Pending Account (Type 110, Account ID 5) into the IN Holding Account (Type 100, Account ID 2).	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	CDM
Transferring Account Type	110
Transferring Account ID	5
Acquiring Registry Code	CDM
Acquiring Account Type	100
Acquiring Account ID	2
Originating Registry Code	IN
Unit Type	6
LULUCF Activity	1
Project Identifier	2
Unit Quantity	500000
<b>Expected Result</b>	
500,000 tCERs should be successfully transferred from pending account (Account ID #5) to holding account (Account ID #2).	
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).	
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).	

<b>Test ID</b>	2209
<b>Description</b>	
Transfer 100 tCERs associated with Project BR-6 from Pending Account (Type 110, Account ID 5) into the FR Holding Account (Type 100, Account ID 13).	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	CDM
Transferring Account Type	110
Transferring Account ID	5
Acquiring Registry Code	CDM
Acquiring Account Type	100
Acquiring Account ID	13
Originating Registry Code	BR
Unit Type	6
LULUCF Activity	1
Project Identifier	6
Unit Quantity	100
<b>Expected Result</b>	
100 tCERs should be successfully transferred from pending account (Account ID #5) to holding account (Account ID #13).	
The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).	
The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).	

<b>Test ID</b>	2290
<b>Description</b>	
ITL will initiate Time Synchronization by calling the ProvideTime Web service.	
<b>Steps</b>	
The ITL will call the registry's ProvideTime Web service.	
<b>Expected Result</b>	
The time provided by the registry should be consistent with the time reflected by the ITL.	

<b>Test ID</b>	2291
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following totals:	
accountType accountCommitPeriod unitType unitCount	110 0 6 100
accountType accountCommitPeriod unitType unitCount	100 0 5 16000
accountType accountCommitPeriod unitType unitCount	100 0 6 10002100
accountType accountCommitPeriod unitType unitCount	100 0 7 25000
If requested by the ITL, the registry should return the following unit blocks:	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 5,000 units generated in Test ID 2100 BR 7 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 20,000 units generated in Test ID 2101 7 BR 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units generated in Test ID 2102 BR 6 110 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 16,000 units generated in Test ID 2103 IN 5 100 1

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 10,000,000 units generated in Test ID 2104 IN 6 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 2,100 units generated in Test ID 2102 BR 6 100 1

## 6.3 External Transfer Transaction Tests

### 6.3.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptProposal AcceptNotification ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	24 Hour Transaction Cleanup Time Synchronization
<b>DES Behavior Diagrams</b>	External Registry Transaction Behavior Diagram Discrepancy Notification Sequence Diagram Terminate External Transaction Diagram Transaction Clean-up Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	Excess Issuance for CDM Project (Type 6)

### 6.3.2 Test Steps

When the registry is the transferring registry:

1. The registry will propose an External transfer transaction (Transaction Type 3).
2. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.
3. The ITL then calls the simulated third-party registry (YY) AcceptProposal Web service and YY should call the ITL's AcceptNotification Web service after processing the proposal. For the purposes of testing, the simulated registry will return messages to the ITL, which will proceed to step 4.

4. The ITL should call the registry's (ZZ) AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
5. On receiving this notification, the registry (ZZ) will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
6. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.

When the registry is the acquiring registry:

1. For purposes of testing, the simulated third-party registry (YY) will generate proposals and pass them through the ITL to the registry being initialized.
2. The ITL will call the registry's AcceptProposal Web service and the registry should call the ITL's AcceptNotification Web service after processing the proposal.
3. The ITL would notify the third-party that the proposal was accepted, and finalize the transaction upon receiving confirmation that the third-party had finalized the transaction.

### 6.3.3 Test Cases

<b>Test ID</b>	2300	
<b>Description</b>		
Perform an external transfer of all units in the CDM registry temporary holding account for NZ to its national registry. NZ rejects the transaction because it does not contain the specified account.		
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	CDM	
Transferring Account Type	100	
Transferring Account ID	3	
Acquiring Registry Code	NZ	
Acquiring Account Type	100	
Acquiring Account ID	9999	
Originating Registry Code	BR	
Unit Type	7	
LULUCF Activity	1	
Project Identifier	3	
Unit Quantity	5000	
Originating Registry Code	BR	
Unit Type	7	
LULUCF Activity	1	
Project Identifier	2345	
Unit Quantity	2500	
<b>Expected Result</b>		
<p>Once the registry proposes the transaction, the ITL will send notification indicating that registry NZ has rejected the proposal. The notification will have transaction status 6 (Rejected) and response code 5902: Acquiring account does not exist.</p> <p>The CDM registry should terminate the transaction, sending notification to the ITL that it has been done. The notification will have transaction status 5 (Terminated).</p>		

<b>Test ID</b>	2301
<b>Description</b>	
Perform an external transfer of all units in the CDM registry temporary holding account for GB to its national registry.	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	CDM
Transferring Account Type	100
Transferring Account ID	4
Acquiring Registry Code	GB
Acquiring Account Type	100
Acquiring Account ID	8888
Originating Registry Code	BR
Unit Type	7
LULUCF Activity	1
Project Identifier	2345
Unit Quantity	7500
Originating Registry Code	BR
Unit Type	6
LULUCF Activity	1
Project Identifier	6
Unit Quantity	2000
Originating Registry Code	IN
Unit Type	5
LULUCF Activity	1
Project Identifier	4
Unit Quantity	10000
Originating Registry Code	IN
Unit Type	6
LULUCF Activity	1
Project Identifier	2
Unit Quantity	9500000
<b>Expected Result</b>	
<p>All units from GB temporary holding account #4 in the CDM registry should be successfully transferred to the holding account of external registry GB, holding account 8888.</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The ITL will then send a notification to registry with Transaction Status Code of 8 (Accepted).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p>	

<b>Test ID</b>	2302
<b>Description</b>	
Attempt to perform an external transfer of all units in the CDM registry temporary holding account for NZ to its national registry. The CDM registry then immediately calls the GetTransactionStatus Web service with the transaction number it specified.	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	CDM
Transferring Account Type	100
Transferring Account ID	3
Acquiring Registry Code	NZ
Acquiring Account Type	100
Acquiring Account ID	1234
Originating Registry Code	BR
Unit Type	7
LULUCF Activity	1
Project Identifier	3
Unit Quantity	5000
Originating Registry Code	BR
Unit Type	7
LULUCF Activity	1
Project Identifier	2345
Unit Quantity	2500
<b>Expected Result</b>	
<p>The registry will send a transaction proposal and then call the GetTransactionStatus Web service.</p> <p>The returned transaction status will be 2 - Checked, No Discrepancy.</p> <p>The CDM registry has not finalized the proposed transaction.</p>	



<b>Test ID</b>	2303
<b>Description</b>	
Registry NZ does not respond to the external transfer of the units specified in Test ID 2302 and the 24-Hour Transaction Clean-up is triggered (for the purposes of initialization, in less than 24 hours from initiating transaction).	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	CDM
Transferring Account Type	100
Transferring Account ID	3
Acquiring Registry Code	NZ
Acquiring Account Type	100
Acquiring Account ID	1234
Originating Registry Code	BR
Unit Type	7
LULUCF Activity	1
Project Identifier	3
Unit Quantity	5000
Originating Registry Code	BR
Unit Type	7
LULUCF Activity	1
Project Identifier	2345
Unit Quantity	2500
<b>Expected Result</b>	
<p>The ITL will send notification indicating that registry NZ has not responded with-in 24 hours. The notification will have transaction status 7 (Cancelled) and response code 8802: Transaction Clean-up.</p> <p>The CDM registry should cancel the transaction, sending notification to the ITL that it has been done. The notification will have transaction status 7 (Cancelled).</p>	

<b>Test ID</b>	2304
<b>Description</b>	
GB receives notification type 6: Excess Issuance for CDM Project and proposes a transaction transferring 5,000 AAUs originating from GB to the CDM registry.	
<b>Relevant Notification Parameters</b>	
Message Content	The CDM Executive Board has determined an excess number of units have been distributed for CDM Project IN-2. Move the number of units indicated to the Excess Issuance Cancellation Account.
Notification Type	6
Project Number	IN-2
Target Value	5000
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	GB
Transferring Account Type	100
Transferring Account ID	8888
Acquiring Registry Code	CDM
Acquiring Account Type	240
Acquiring Account ID	6
Originating Registry Code	GB
Unit Serial Block Start	1
Unit Serial Block End	5000
Unit Type	1
<b>Expected Result</b>	
<p>GB's AAUs should be successfully transferred to the CDM registry.</p> <p>After receiving the AcceptProposal call, the CDM registry should send an acknowledgement that it has received the proposal. CDM registry approves the transaction and finalizes it in the database.</p> <p>The CDM registry's notification to the ITL should include a Transaction Status Code of 8 (Accepted).</p>	

<b>Test ID</b>	2305
<b>Description</b>	
Attempt to perform an external transfer of all units in the CDM registry temporary holding account for FR to its national registry. The ITL rejects the transaction.	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	CDM
Transferring Account Type	100
Transferring Account ID	13
Acquiring Registry Code	FR
Acquiring Account Type	100
Acquiring Account ID	5678
Originating Registry Code	BR
Unit Type	6
LULUCF Activity	1
Project Identifier	6
Unit Quantity	100
<b>Expected Result</b>	
<p>The units are not transferred.</p> <p>After the registry proposes the transaction, the ITL will send notification indicating a forced failure of message validity checks with response code 1504: "Acquiring Registry Transaction Status." The notification will have transaction status 3 (Checked, Discrepancy).</p> <p>The registry should terminate the transaction, sending notification to the ITL that it has been done. The notification will have transaction status 5 (Terminated).</p>	

<b>Test ID</b>	2390
<b>Description</b>	
ITL will initiate Time Synchronization by calling the ProvideTime Web service.	
<b>Steps</b>	
The ITL will call the registry's ProvideTime Web service.	
<b>Expected Result</b>	
The time provided by the registry should be consistent with the time reflected by the ITL.	

<b>Test ID</b>	2391
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following totals:	
accountType accountCommitPeriod unitType unitCount	110 0 6 100
accountType accountCommitPeriod unitType unitCount	100 0 5 6000
accountType accountCommitPeriod unitType unitCount	100 0 6 500100
accountType accountCommitPeriod unitType unitCount	100 0 7 17500
accountType accountCommitPeriod unitType unitCount	240 1 1 5000
If requested by the ITL, the registry should return the following unit blocks:	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 5,000 units generated in Test ID 2100 BR 7 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 12,500 units generated in Test ID 2101 7 BR 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units generated in Test ID 2102 BR 6 110 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 6,000 units generated in Test ID 2103 IN 5 100 1

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 500,000 units generated in Test ID 2104 IN 6 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units generated in Test ID 2102 BR 6 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1 5000 GB 1 240 1

## 6.4 Cancellation Transaction Tests

### 6.4.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptNotification AcceptITLNotice ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	ITL Notification Single Registry Transaction Behavior Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	Reversal of Storage for CDM Project (Type 4) Non-submission of Certification Report for CDM Project (Type 5) Notification Update (Type 10)

### 6.4.2 Test Steps

1. The registry will receive a notification for Test ID 2400 before proceeding to step 2.
2. The registry will propose a Cancellation transaction (Transaction Type 4).
3. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.

4. The ITL should call the registry's AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
5. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
6. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.
7. The registry will receive a notification update indicating the requirement has been fulfilled.

### 6.4.3 Test Cases

<b>Test ID</b>	2400	
<b>Description</b>		
Registry receives reversal of storage for CDM Project notification (Type 4). The registry will cancel the affected 500 ICERs, which should be in the NZ temporary holding account to the Mandatory Cancellation Account.		
<b>Relevant Notification Parameters</b>		
messageContent	A reversal of storage of greenhouse gas has occurred at a CDM Project and all transactions involving units associated with that Project are suspended until the specified number of units is cancelled.	
notificationType	4	
notificationStatus	1	
projectNumber	BR-2345	
targetValue	500	
<b>Relevant Transaction Parameters</b>		
Transferring Registry Code	CDM	
Transferring Account Type	100	
Transferring Account ID	3	
Acquiring Registry Code	CDM	
Acquiring Account Type	250	
Acquiring Account ID	7	
Originating Registry Code	BR	
Project Identifier	2345	
Unit Type	7	
Unit Quantity	500	
Unit Serial Block Start	Selected from units transferred in test 2202 to include 500 units	
Unit Serial Block End		
<b>Expected Result</b>		
<p>The registry will propose a cancellation transaction (Type 4).</p> <p>The ITL's notification to the CDM registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The CDM registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p> <p>500 ICERs originating from Project BR-2345 from the NZ temporary holding account in the CDM registry should be transferred to the Mandatory Cancellation Account (Type 250).</p>		

<b>Test ID</b>	2401
<b>Description</b>	
The CDM registry receives notification type 6: Excess Issuance for CDM. CDM proposes a transaction transferring 500 ICERs originating from BR-3 and in the NZ holding account to the CDM registry Excess Issuance Account.	
<b>Relevant Notification Parameters</b>	
Message Content	The CDM Executive Board has determined an excess number of units have been distributed for CDM Project BR-3. Move the number of units indicated to the Excess Issuance Cancellation Account from the NZ holding account.
Notification Type	6
Project Number	BR-3
Target Value	500
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	CDM
Transferring Account Type	100
Transferring Account ID	3
Acquiring Registry Code	CDM
Acquiring Account Type	240
Acquiring Account ID	6
Originating Registry Code	BR
Project Identifier	3
Unit Type	7
Unit Quantity	500
<b>Expected Result</b>	
<p>NZ's 500 ICERs should be successfully transferred from pending account (Account ID #3) to excess issuance cancellation account (Account ID #6, type 240).</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p>	



<b>Test ID</b>	2402
<b>Description</b>	
Registry receives Non-submission of Certification Report for CDM Project notification (Type 5). The registry is directed to cancel the affected ICERs from Project BR-2345 in the holding account for BR. The CDM registry will partially fulfill the notification direction by transferring 10,000 ICERs into the cancellation account.	
<b>Relevant Notification Parameters</b>	
messageContent	The CDM Executive Board has not received a Certification Report, and it has requested that all ICERs generated by the specified CDM be cancelled. For the purposes of initialization, cancel ICERs from the same Project held by the host Party.
notificationType	5
notificationStatus	1
projectNumber	BR-2345
targetValue	12000
originatingRegistryCode	BR
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	CDM
Transferring Account Type	100
Transferring Account ID	1
Acquiring Registry Code	CDM
Acquiring Account Type	250
Acquiring Account ID	7
Originating Registry Code	BR
Project Identifier	2345
Unit Type	7
Unit Quantity	10000
<b>Expected Result</b>	
<p>The CDM registry will propose a cancellation transaction (Type 4).</p> <p>The ITL's notification to the registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p> <p>10,000 ICERs from the BR holding account should be transferred to the Mandatory Cancellation Account (Type 250).</p>	

<b>Test ID</b>	2403
<b>Description</b>	
CDM registry receives a notification update (Type 10) for the Non-submission of Certification Report for CDM Project notification in test 2402. The registry will cancel an additional 2,000 affected ICERs from the same affected Project in the NZ holding account.	
<b>Relevant Notification Parameters</b>	
messageContent	The CDM Executive Board has not received a Certification Report, and it has requested that all ICERs generated by the specified CDM be cancelled.
notificationType	10
notificationStatus	2
projectNumber	BR-2345
targetValue	2000
originatingRegistryCode	BR
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	CDM
Transferring Account Type	100
Transferring Account ID	3
Acquiring Registry Code	CDM
Acquiring Account Type	250
Acquiring Account ID	7
Originating Registry Code	BR
Unit Type	7
Unit Quantity	2000
<b>Expected Result</b>	
<p>The CDM registry will propose a cancellation transaction (Type 4).</p> <p>The ITL's notification to the CDM registry should include a Transaction Status Code of 2 (Checked, No Discrepancy).</p> <p>The CDM registry's notification to the ITL should include a Transaction Status Code of 4 (Completed).</p> <p>2000 ICERs from the NZ holding account should be transferred to the Mandatory Cancellation Account (Type 250).</p>	

<b>Test ID</b>	2490
<b>Description</b>	
ITL will initiate Time Synchronization by calling the ProvideTime Web service.	
<b>Steps</b>	
The ITL will call the registry's ProvideTime Web service.	
<b>Expected Result</b>	
The time provided by the registry should be consistent with the time reflected by the ITL.	

<b>Test ID</b>	2491
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following totals:	
accountType accountCommitPeriod unitType unitCount	110 0 6 100
accountType accountCommitPeriod unitType unitCount	100 0 5 6000
accountType accountCommitPeriod unitType unitCount	100 0 6 500100
accountType accountCommitPeriod unitType unitCount	100 0 7 4500
accountType accountCommitPeriod unitType unitCount	240 1 1 5000
accountType accountCommitPeriod unitType unitCount	240 1 7 500
accountType accountCommitPeriod unitType unitCount	250 1 7 12500
If requested by the ITL, the registry should return the following unit blocks:	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 4,500 units generated in Test ID 2100 BR 7 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units generated in Test ID 2102 BR 6 110 1

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 6,000 units generated in Test ID 2103 IN 5 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 500,000 units generated in Test ID 2104 IN 6 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units generated in Test ID 2102 BR 6 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1 5000 GB 1 240 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 500 units generated in Test ID 2100 BR 7 240 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 500 units generated in Test ID 2101 7 BR 250 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 12500 units generated in Test ID 2101 7 BR 250 1

## 6.5 Expiry Date Change Transaction Tests

### 6.5.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptNotification AcceptITLNotice ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization
<b>DES Behavior Diagrams</b>	ITL Notification Single Registry Transaction Behavior Diagram Time Synchronization Reconciliation Send Reconciliation Results
<b>Notification Types</b>	Expiry Date Change (Type 9)

### 6.5.2 Test Steps

1. The ITL will send the registry a notification (Notification Type 9) through its AcceptITLNotice Web service alerting it that certain units require an expiry date change. In one case, the unit type specified is Type 6 (tCERs), which indicates that all tCERs must have their expiry date changed. In another case, the unit type specified is Type 7 (ICERs), and the Project Number of the Project for which the ICERs were issued is specified in the notification.
2. The registry will propose an Expiry Date Change transaction (Transaction Type 8).
3. The registry will submit the proposal to the ITL's AcceptProposal Web service, and verify that it receives a response indicating that the proposal was received.
4. The ITL should call the registry's AcceptNotification Web service to notify the registry whether the proposal is approved or rejected. The ITL will verify that it receives a response indicating that the notification was received.
5. On receiving this notification, the registry will finalize the transaction in its database or terminate the transaction depending on the notification received from the ITL.
6. The registry will then call the ITL's AcceptNotification Web service to notify the ITL of its result.

### 6.5.3 Test Cases

<b>Test ID</b>	2500
<b>Description</b>	
The ITL sends an Expiry Date Change notification (Type 9) to change the expiry date on all tCERs.	
<b>Relevant Notification Parameters</b>	
messageContent	Due to a change in the crediting period for the relevant CDM Projects, the units specified within this message must have their expiry date changed.
notificationType	9
notificationStatus	1
unitType	6
targetValue	500200
targetDate	2018-11-30
<b>Expected Result</b>	
The registry will change the expiry date to the one specified for all tCERs in one or more transactions. The transaction proposal will include the notification identifier.	

<b>Test ID</b>	2501
<b>Description</b>	
The ITL sends an Expiry Date Change notification (Type 9) to change the expiry date on all ICERs associated with Project BR-3.	
<b>Relevant Notification Parameters</b>	
messageContent	Due to a change in the crediting period for the relevant CDM Project, the units specified within this message must have their expiry date changed.
notificationType	9
notificationStatus	1
projectNumber	BR-3
unitType	7
targetValue	5000
targetDate	2029-11-25
<b>Expected Result</b>	
The registry will change the expiry date of all ICERs originated with Project BR-3 in one or more transactions. The transaction proposal will include the notification identifier.	

<b>Test ID</b>	2590
<b>Description</b>	
ITL will initiate Time Synchronization by calling the ProvideTime Web service.	
<b>Steps</b>	
The ITL will call the registry's ProvideTime Web service.	
<b>Expected Result</b>	
The time provided by the registry should be consistent with the time reflected by the ITL.	

<b>Test ID</b>	2591
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following totals:	
accountType accountCommitPeriod unitType unitCount	110 0 6 100
accountType accountCommitPeriod unitType unitCount	100 0 5 6000
accountType accountCommitPeriod unitType unitCount	100 0 6 500100
accountType accountCommitPeriod unitType unitCount	100 0 7 4500
accountType accountCommitPeriod unitType unitCount	240 1 1 5000
accountType accountCommitPeriod unitType unitCount	240 1 7 500
accountType accountCommitPeriod unitType unitCount	250 1 7 12500

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 4,500 units generated in Test ID 2100 BR 7 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units generated in Test ID 2102 BR 6 110 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 6,000 units generated in Test ID 2103 IN 5 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 500,000 units generated in Test ID 2104 IN 6 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units generated in Test ID 2102 BR 6 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1 5000 GB 1 240 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 500 units generated in Test ID 2100 BR 7 240 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 500 units generated in Test ID 2101 7 BR 250 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 12500 units generated in Test ID 2101 7 BR 250 1



## 6.6 Additional Tests

This section contains tests to ensure the CDM registry has correctly implemented the functional components of the Data Exchange Standards for the commitment period life cycle. The section begins with a test to check implementation of the Web service to support the general message administrative function and continues with testing to evaluate registry performance in a subsequent commitment period.

### 6.6.1 Components Tested

<b>ITL Web Services</b>	AcceptProposal AcceptNotification ReceiveTotals ReceiveUnitBlocks
<b>Registry Web Services</b>	AcceptNotification AcceptProposal ProvideTime InitiateReconciliation ProvideTotals ProvideUnitBlocks ProvideAuditTrail ReceiveReconciliationResult
<b>Administrative Functions</b>	Time Synchronization General Messages
<b>DES Behavior Diagrams</b>	ITL Notification Single Registry Transaction Behavior Diagram External Transfer Behavior Diagram Time Synchronization Diagram Reconciliation Send Reconciliation Results
<b>Notification Types</b>	None

### 6.6.2 Test Steps

1. The ITL will send a general message to inform the registry that it is switching to commitment period 2.
2. The registry will switch to commitment period 2 and repeat the issuance tests as described.
3. The registry will then receive a notification to mark the start of the next test. The registry will proceed through the remaining tests, concluding with a full reconciliation through the audit trail phase for the unit blocks specified.

### 6.6.3 Test Cases

<b>Test ID</b>	2600
<b>Description</b>	
<p>The ITL will call the AcceptMessage Web service at the registry to deliver a general message to the registry administrator. The message will inform the registry administrator that the ITL is switching to the second commitment period. The registry will now function as if it were in the second commitment period. The registry administrator will call the ITL operator to indicate receipt of the message before proceeding to the next test.</p>	
<b>Expected Result</b>	
<p>The registry will call the ITL operator to indicate receipt of the message and report the notification identifier.</p>	

<b>Description</b>	
<p>Repeat the issuance tests 2100, 2101, 2102, 2103, and 2104 in the second commitment period. The tests are the same with the exceptions that:</p> <ol style="list-style-type: none"> <li>1. the original commitment period for the units is 2;</li> <li>2. the serial blocks issued are unique; and</li> <li>3. the expiry date for projects BR-6 (test 2102) and IN-2 (test 2104) are now 11-30-2011 as a result of test 2500 and the expiry date for project BR-3 (test 2100) is now 11-25-2027 as a result of test 2501.</li> </ol>	
<b>CP 1 Test ID</b>	<b>CP 2 Test ID</b>
2100	2601
2102	2602
2103	2603
2104	2604
<b>Expected Result</b>	
<p>The units issued should have an original commitment period = 2.</p>	

<b>Test ID</b>	2691
<b>Description</b>	
ITL Request Totals	
<b>Relevant Parameters</b>	
Reconciliation Identifier	ITL to determine
Reconciliation Snapshot DateTime	Now ()
<b>Expected Result</b>	
The registry should return the following totals:	
accountType accountCommitPeriod unitType unitCount	110 0 6 10002300
accountType accountCommitPeriod unitType unitCount	100 0 5 6000
accountType accountCommitPeriod unitType unitCount	100 0 6 500100
accountType accountCommitPeriod unitType unitCount	100 0 7 4500
accountType accountCommitPeriod unitType unitCount	240 1 1 5000
accountType accountCommitPeriod unitType unitCount	240 1 7 500
accountType accountCommitPeriod unitType unitCount	250 1 7 12500
accountType accountCommitPeriod unitType unitCount	110 0 5 16000
accountType accountCommitPeriod unitType unitCount	110 0 7 25000
If requested by the ITL, the registry should return the following unit blocks:	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 4,500 units generated in Test ID 2100 BR 7 100 1

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units generated in Test ID 2102 BR 6 110 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 6,000 units generated in Test ID 2103 IN 5 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 500,000 units generated in Test ID 2104 IN 6 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 100 units generated in Test ID 2102 BR 6 100 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	1 5000 GB 1 240 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 500 units generated in Test ID 2100 BR 7 240 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 500 units generated in Test ID 2101 7 BR 250 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 12000 units generated in Test ID 2101 7 BR 250 1
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 5,000 units generated in Test ID 2601 BR 7 110 2

(cont.)

<b>Expected Result (cont.)</b>	
If requested by the ITL, the registry should return the following unit blocks (cont.):	
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 2,200 units generated in Test ID 2602 BR 6 110 2
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 16,000 units generated in Test ID 2603 IN 5 110 2
unitSerialBlockStart unitSerialBlockEnd originatingRegistryCode unitType AccountType ApplicableCommitmentPeriod	Determined by registry to include 10,000,000 units generated in Test ID 2604 IN 6 110 2

<b>Test ID</b>	2692
<b>Description</b>	
<p>GB proposes a transaction transferring 100 AAUs originating from GB to the CDM registry to fulfill an Excess Issuance for CDM Project notification (possibly the same notification ID from test 2304). The CDM registry will accept the proposal and finalize the transaction in its database. However, GB does not complete the transaction before the ITL initializes a reconciliation. The registry will then be directed to roll-back the transaction as part of manual intervention.</p>	
<b>Relevant Transaction Parameters</b>	
Transferring Registry Code	GB
Transferring Account Type	100
Transferring Account ID	8888
Acquiring Registry Code	CDM
Acquiring Account Type	240
Acquiring Account ID	16
Originating Registry Code	GB
Unit Serial Block Start	8001
Unit Serial Block End	8100
Unit Type	1
<b>Expected Result</b>	
<p>After receiving the AcceptProposal call, the CDM registry should send an acknowledgement that it has received the proposal. The CDM registry approves the transaction and finalizes it in its database. The CDM registry's notification to the ITL should include a Transaction Status Code of 8 (Accepted).</p> <p>The ITL initiates a reconciliation test immediately after receiving the CDM registry's notification on the AcceptNotification Web service and, hypothetically, before the transferring registry (GB) finalizes the transaction and submits a notification with transaction status code of 4 (Completed). For the purposes of initialization testing, the transaction is cancelled without notifying the CDM registry.</p> <p>The reconciliation finds an inconsistency at the unit totals level and continues automatically through the audit trail phase, specifying the units involved in the transaction described in this test. The ITL operator will call the registry administrator with instructions to roll-back the transaction. The registry administrator will perform the manual intervention.</p> <p>The ITL will send notification that the reconciliation is "ITL Completed with Manual Intervention" (Status Code 6) and begin a new reconciliation that should be successful. The new reconciliation must reach reconciliation status code 2: "Validated."</p> <p>The manual intervention should be logged.</p>	

<b>Test ID</b>	2693
<b>Description</b>	
<p>The registry submits its transaction log, reconciliation history log, notification log, and message archive.</p>	
<b>Expected Result</b>	
<p>The ITL operator can verify randomly selected transactions resulting from Annex H testing. The logs will reflect only the registry activity described in Annex H and will conform with Data Logging Specifications described in Section 7 of the Data Exchange Standards.</p>	

**17 7. Annex H Testing for CP2**

## 18 7.1 Introduction

### 7.1.1 Purpose

This document provides the specification primarily for the functional testing component of registry initialization and/or other testing as determined to be required for commitment period 2 (CP2) by the ITL Change Advisory Board. The test plan also covers non-functional areas identified for inspection based on ITL production experience. In addition to successfully meeting the conditions for live operations with the ITL prescribed by the Data Exchange Standards (DES) and the Common Operational Procedures, a registry must successfully complete these test cases prior to being authorized to submit transactions to the ITL in the Production environment. The ITL Administrator and/or ITL Service Desk, drawing on operational experience, may require additional or modified tests in order to verify a registry is able to interoperate with the ITL as required by the DES. This document specifies eight test scenarios, each comprised of numerous test cases, each of which must be successfully executed, evaluated, and recorded by the registry being tested, and the ITL.

The successful completion of this functional test plan is a necessary condition that must be met by a registry prior to going live with the ITL.

### 7.1.2 Guiding Principles and Assumptions

#### 7.1.2.1 Principles and Scope

This document provides a functional test plan between a registry and the ITL for the purpose of evaluating the registry's ability to operate with the ITL for CP2. Successful completion of this test plan does NOT indicate that a registry meets all requirements of the DES, but successful completion is a necessary condition for a registry to connect to the ITL in CP2. It is likewise important to note that this test plan focuses on testing the registry. By necessity, the ITL must be involved, but this test plan is not designed as an ITL test plan. In fact, it assumes that the ITL has been fully tested and will behave as expected.

The following functional and non-functional components of the registry will be tested to evaluate the registry's implementation of relevant DES components.

- Every web service defined by the DES will be tested, with the exception of EU ETS-specific web services for Account Management;
- Every Kyoto transaction type will be tested;
- Every transaction flow (including negative flows) described in behaviour diagrams contained in the DES technical specification will be tested;
- A registry's ability to receive and act upon notifications received from the ITL will be tested for every notification type defined by the DES (Section 6);
- A registry's ability to respond and act upon ITL administrative requests will be tested; and
- Non-functional performance (e.g.; large number of unit blocks, unit blocks representing a single unit, long transaction and account identifiers, and multiple simultaneous transactions) will be tested.

#### 7.1.2.2 Assumptions

The following global assumptions regarding the execution of this test plan apply:

- Test execution will be coordinated by the ITL Service Desk. Third-party registries will be simulated and administrated by the ITL Service Desk, with support from scripts and utilities provided by the ITL Developer.
- This test plan can be executed for either a single registry, or by two registries in parallel (except for Scenario 8 which must be done by one registry at a time). Execution plans should be coordinated with the ITL Service Desk ahead of time.
- Transaction, unit holding, notification, and message log data will be archived and cleared down at the end of each test scenario.
- Registry ZZ refers to the registry being tested under this test plan.



### 7.1.2.3 Test Coverage

Table 7.4 – Test Coverage

Test Coverage Area	Relevant Test Cases
<b>Successful Transactions</b>	
Issuance	1.1, 1.3, 1.4, 1.5, 2.1, 2.3, 2.4, 6.2, 6.8, 7.2
Conversion	2.6, 2.7
External Transfer (Registry ZZ as transferring registry)	1.8, 1.10, 2.9, 4.5, 7.3
Incoming External Transfer (Registry ZZ as acquiring registry)	1.7, 1.11, 2.2, 3.1, 3.6, 4.1, 5.1, 6.1, 6.7, 7.1, 8.1, 8.2, 8.3, 8.4, 8.6, 8.8
Cancellation (Voluntary)	3.3, 6.3, 6.10
Cancellation (Notification-based)	3.4, 3.5, 4.2, 4.3, 5.2
Retirement	5.3, 6.4, 6.9, 7.4, 8.5
Replacement	5.2, 5.5
Carry-over	6.6
Expiry Date Change	5.4, 5.6
<b>Rejected and Cancelled Transactions</b>	
Issuance rejected by ITL	1.2, 1.6
External Transfer rejected by third party acquiring registry	1.9, 4.4
External Transfer rejected by ITL	1.13
Incoming External Transfer rejected by Registry ZZ	1.12
Transaction clean-up (of “stalled” transaction)	1.14
Conversion rejected by ITL	2.5, 2.8
Voluntary Cancellation rejected by ITL	3.2
Carry-over rejected by ITL	6.5
<b>Notifications and Notification Fulfilment</b>	
Net Source Cancellation (Notification Type 1)	3.4
Non-compliance Cancellation (Notification Type 2)	3.5
Impending Expiry of tCER/ICER (Notification Type 3)	5.5
Reversal of Storage (Notification Type 4)	4.2, 5.2
Non-submission of Certification Report (Notification Type 5)	4.3
Excess Issuance of CDM Project (Notification Type 6)	4.4, 4.5
Commitment Period Reserve (Notification Type 7)	3.5
Carry-over (Notification Type 8)	6.5, 6.6
Expiry Date Change (Notification Type 9)	5.4, 5.6
Notification Update (Notification Type 10)	4.6
EU15 CPR (Notification Type 11)	3.6
<b>Reconciliation</b>	
Reconciliation (Totals match)	1.15, 2.10, 3.7, 4.7, 5.7, 6.11, 7.11, 8.7, 8.9
Reconciliation (Inconsistency) and Manual Intervention	7.5, 8.7
Reconciliation with account filtering (EU ETS registries only)	7.5, 8.7
<b>Non-functional</b>	
Multiple simultaneous incoming External Transfers (Registry ZZ as acquiring registry)	8.1
Transactions with long transaction identifier	8.2
Transactions with long unit block serial numbers	8.3
Small unit block size (1 unit)	8.3
Long account number	8.4
Transactions with large number of units blocks	8.6, 8.8

Test Coverage Area	Relevant Test Cases
Graceful rejection of excessively large incoming transfer	8.8
Fragmented block impact on reconciliation	8.7, 8.9

### 7.1.3 Prerequisites

The registry must complete all internal functional testing before commencing the coordinated testing of this test plan. In addition, VPN connectivity and SSL connectivity must be successfully established, with connectivity demonstrated by:

- The registry sending a generic message to the ITL via the AcceptMessage web service; and
- The registry receiving a generic message from the ITL via the AcceptMessage web service; and
- The registry receiving and responding to a ProvideTime request from the ITL in which the registry's response is within allowable tolerances defined in the current DES.

## 19 7.2. CP2 Seed Data, Conventions and Execution

### 7.2.1 Seed Data Specifications

#### 7.2.1.1 Registry Accounts

The ITL must be aware of the Government Accounts<sup>3</sup> at the registry. In this context, Government Accounts refer to those account defined in Section 9.11 of the DES and include cancellation, retirement and replacement accounts. To provide flexibility for registries that may have a specific system for account identifiers, account identifiers are not prescribed. As such, the ITL Service Desk must solicit this information from the registry before testing commences. While not strictly a Government Account, the Type 100 holding account identifier must also be specified. This account will serve as the acquiring account for all external transfer transactions TO the registry. Table 7.2 below describes the account information required with the “Identifier” column blank as this information is solicited from the registry.

Table 7.5 – Registry Accounts

Type	Description	CP	Identifier
100	Holding Account <sup>4</sup>	0	
210	Net Source Cancellation Account	2	
220	Non-compliance Cancellation Account	2	
230	Voluntary Cancellation Account	2	
230	Voluntary Cancellation Account	3	
250	Mandatory Cancellation Account	2	
300	Retirement Account	2	
300	Retirement Account	3	
411	tCER Replacement Account for Expiry	2	
421	ICER Replacement Account for Expiry	2	
422	ICER Replacement Account for Reversal of Storage	2	
423	ICER Replacement Account for Non-submission of Certification Report	2	

#### 7.2.1.2 Registry and ITL Limits and CPR

The ITL and the registry maintain information about limits assigned to the registry. Certain test cases depend on an intentional disparity between the values stored at the registry and the values stored at the ITL representing limits for the registry. Please note that in all cases, “Registry ZZ” refers to registry being tested.

<sup>3</sup> Please note that Scenario 3 requires that Government Accounts for the registry are not loaded at the ITL until after test case 3.2 is completed successfully.

<sup>4</sup> Please note that this account should be used as the acquiring account for all external transfers TO the registry during testing.

**Table 7.6 – Registry Limits (Registry and ITL values)**

Limit Description	ITL	Registry ZZ
<b>Commitment Period 2</b>		
AAU Issuance Limit	3,000,000,000	3,500,000,000
CPR (90% of AAU Issuance Limit) <sup>5</sup>	2,700,000,000	3,150,000,000
EU15 CPR	2,700,000,000	3,150,000,000
RMU Issuance – LULUCF Activity 1	1,500,000	2,000,000
RMU Issuance – LULUCF Activity 2	1,500,000	2,000,000
RMU Issuance – LULUCF Activity 3	1,500,000	2,000,000
RMU Issuance – LULUCF Activity 4	1,500,000	2,000,000
RMU Issuance – LULUCF Activity 5	1,500,000	2,000,000
RMU Issuance – LULUCF Activity 6	1,500,000	2,000,000
CER Carry-over limit (2.5% of AAU Issuance Limit)	75,000,000	87,500,000
ERU Carry-over limit (2.5% of AAU Issuance Limit)	75,000,000	87,500,000
tCER/ICER Retirement Limit	1,000,000	1,000,000
<b>Commitment Period 3</b>		
AAU Issuance Limit	3,000,000,000	3,500,000,000
CPR (90% of AAU Issuance Limit)	2,700,000,000	3,150,000,000
EU15 CPR	3,000,000,000	3,500,000,000
RMU Issuance – LULUCF Activity 1	1,500,000	2,000,000
RMU Issuance – LULUCF Activity 2	1,500,000	2,000,000
CER Carry-over limit (2.5% of AAU Issuance Limit)	75,000,000	87,500,000
ERU Carry-over limit (2.5% of AAU Issuance Limit)	75,000,000	87,500,000

### 7.2.1.3 Third-Party Registries

The test plan relies on third party registries to both transfer units to, and receive units from the registry being tested. As with the registry being tested (Registry ZZ), these registries will also have limits recorded at the ITL. Further, the registry being tested must be aware of the account number for a holding account at each registry (for the purposes of specifying these accounts as the acquiring account for external transfers). All third party registries will be configured at the ITL to be fully eligible to engage in trading. Table 7.4 specifies the limits to be recorded at the ITL for each third party registry, and Table 7.5 provides the holding account numbers.

<sup>5</sup> Please note that this is currently based off of the existing CPR calculation which is expected to be reviewed. If the CPR calculation is changed, a corresponding change in this test plan will be required.

**Table 7.7 – Third Party Registry Limits**

Limit Description	Registry XX	Registry YY	Registry QQ	Registry RR
AAU Issuance Limit	3,000,000,000	3,000,000,000	3,000,000,000	3,000,000,000
CPR	0	0	0	0
EU15 CPR	0	0	0	0
RMU Issuance – LULUCF Activity 1	2,500,000	1,500,000	1,500,000	1,500,000
RMU Issuance – LULUCF Activity 2	2,500,000	1,500,000	1,500,000	1,500,000
RMU Issuance – LULUCF Activity 3	2,500,000	1,500,000	1,500,000	1,500,000
RMU Issuance – LULUCF Activity 4	2,500,000	1,500,000	1,500,000	1,500,000
RMU Issuance – LULUCF Activity 5	2,500,000	1,500,000	1,500,000	1,500,000
RMU Issuance – LULUCF Activity 6	2,500,000	1,500,000	1,500,000	1,500,000

**Table 7.8 – Third Party Registry Holding Accounts<sup>6</sup>**

Limit Description	Registry XX	Registry YY	Registry QQ	Registry RR
Holding Account Number <sup>7</sup>	XX-100-1000-0	YY-100-1000-0	QQ-100-1000-0	RR-100-1000-0

During execution of this test plan, the actual Party codes must be designated for the placeholders XX, YY, QQ, RR. The ITL Service Desk is responsible for determining which parties to use in lieu of these placeholders, and for communicating this information to the registry. To avoid complication with the EU 15 Commitment Period Reserve check, the third party registries should not be members of the EU 15. To test a registry's ability to handle message flows involving an STL, XX will be an EU registry subject to review by the EUTL but not part of the EU 15 (such as Bulgaria). YY, QQ, and RR may be any other available registry. All third-party registries (XX, YY, QQ, and RR) will remain the same for the duration of a given execution of this test plan. Table 7.6 below provides a template for recording these assignments.

**Table 7.9 – Third Party Registry – Placeholder Party Assignments**

Placeholder Party	XX (EU Registry)	YY	QQ	RR
Actual Party				

<sup>6</sup> Please see Table 7.10 for CDM Excess Issuance Accounts.

<sup>7</sup> Please note that these account numbers are specified in the following format: XX-100-1000-0. This indicates an account held by Party XX, with Account Type = 100, Account Identifier = 1000, and Account CP = 0

### 7.2.1.4 JI and CDM Projects

The test plan requires that both the registry and the ITL recognize JI projects for the purposes of testing the conversion transaction to generate ERUs. Table 7.7 specifies the JI projects that must be loaded in the registry, and Table 7.8 specifies the JI projects that must be loaded at the ITL.

**Table 7.10 – JI Projects Loaded at the Registry**

Project Name	Project Number	Unit Type	Track	Conversion Limit
Seed Project – ERU from AAU, Track 2	ZZ-1101	ERU from AAU (3)	2	1,500,000
Seed Project – ERU from RMU, Track 1	ZZ-1102	ERU from RMU (4)	1	1,000,000
Seed Project – Type Issue (Reg), Track 2	ZZ-1103	ERU from AAU (3)	2	1,500,000
Seed Project – Limit Issue (Reg), Track 2	ZZ-1104	ERU from RMU (4)	2	1,500,000

**Table 7.11 – JI Projects Loaded at the ITL**

Project Name	Project Number	Unit Type	Track	Conversion Limit
Seed Project – ERU from AAU, Track 2	ZZ-1101	ERU from AAU (3)	2	1,500,000
Seed Project – ERU from RMU, Track 1	ZZ-1102	ERU from RMU (4)	1	1,000,000
Seed Project – Type Issue (ITL), Track 2	ZZ-1103	ERU from RMU (4) <sup>8</sup>	2	1,500,000
Seed Project – Limit Issue (ITL), Track 2	ZZ-1104	ERU from RMU (4)	2	1,000,000 <sup>9</sup>
Registry XX Seed Project – ERU from RMU, Track 1	XX-11004	ERU from RMU (4)	1	1,500,000

The test plan also includes units generated from CDM projects (CERs, tCERs, and ICERs). The ITL must be aware of these projects in order to support the issuance and transfer of the units generated under these projects. The registry does not need to load data on these projects, however it must be able to receive units from these projects and be able to identify the project under which a given unit was issued. Table 7.9 below provides the basic information on CDM projects used.

**Table 7.12 – CDM Projects Loaded at the ITL**

Project Name	Project Number	Unit Type	Notes
CER Project 1	CN-1	CER	
CER Project 11	CN-11	CER	
CER Project 111	CN-111	CER	
tCER Project 2	AR-2	tCER	LULUCF 1; Expiry Date: 31/12/2022
tCER Project 22	AR-22	tCER	LULUCF 2; Expiry Date: 31/12/2022
ICER Project 3	BR-3	ICER	LULUCF 1; Expiry Date: 31/12/2023
ICER Project 33	BR-33	ICER	LULUCF 2; Expiry Date: 31/03/2024

<sup>8</sup> Please note that the unit type for JI Project ZZ-1103 has a different unit type recorded at the ITL than at the registry. This is intentional and serves the purposes of test case 2.5.

<sup>9</sup> Please note that the conversion limit for JI Project ZZ-1104 has a different conversion limit recorded at the ITL than at the registry. This is intentional and serves the purposes of test case 2.8.

**Table 7.13 – CDM Excess Issuance Account Numbers**

<b>Account</b>	<b>Account Number</b>
CDM Excess Issuance Account for CP2	CDM-240-2240-2
CDM Excess Issuance Account for CP3	CDM-240-3240-3

### 7.2.1.5 ITL Configuration Settings

In addition to the seed data specified in this document, execution of this test plan requires specializing ITL configuration settings. These are described below.

1. 24-hour transaction cleanup, which cancels incomplete transactions that have been inactive for 24 hours, should be configured to clean up incomplete transactions that have been inactive for a period less than 24 hours in order to shorten the time for test execution. The ITL Service Desk will configure the time to be shorter interval for the purposes of test case 1.14. For testing purposes, this clean-up time should be set to 5 minutes.
2. The InitiateReconciliation message sent to the registry which specifies the snapshot time should be configured to be shorter than in production in order to shorten the time for test execution. For testing purposes, this window should be set to 3 minutes.

## 7.2.2 Test Scenario and Test Case Conventions

### 7.2.2.1 Registry ZZ

Please note that throughout this document, the registry being tested is referred to as “Registry ZZ”. Other registries, such as “Registry XX”, “Registry YY”, “Registry QQ” and “Registry RR” refer to third party registries (simulated) that are involved in the test plan.

### 7.2.2.2 Test Scenarios and Test Cases

Section 7.3 below contains the specifications for the test scenarios and the test cases comprising each scenario. Each test scenario includes a brief overview of the scenario’s theme, as well as any assumptions that apply specifically to the scenario. Following this information are the test cases for that scenario, which are designed to be executed in order as the test cases are often interdependent. Each test case is specified with the following information.

- ID – Each test case is assigned a unique identifier comprised of two numbers. The first number represents the test scenario that the test case belongs to, and the second represents the order of the test case within the scenario. (i.e. ID 2.5 indicates that the test case is the 5<sup>th</sup> test case of scenario 2.)
- Description – A brief, one line description of the test case.
- Initiator – Indicates the initiator of the test case. If the registry initiates the test case (i.e. proposing a transaction), this field will indicate “Registry”. If the ITL Service Desk must initiate the test case (i.e. initiating reconciliation, or an incoming transfer from a third party registry), this field will indicate “ITL Service Desk”. If the test case requires actions initiated by the registry and the ITL service desk, the initiator field will indicate this, and the test steps will be clearly separated by initiator.
- Test Steps – Describes the test steps that must be executed. **Test steps highlighted in grey** indicate “setup” steps that do not directly involve the registry being tested. These steps are taken by the ITL Service Desk to prepare for test execution and may be achieved via scripts that represent the actions described. These “setup” steps are included for clarity and information purposes only.
- Success Criteria – Defines the expected results for the test case, including guidance on evaluating whether the test was successfully executed. Compliance with test case success criteria will be evaluated by the ITL Service Desk. Note that the specific transaction status code is not identified in the success criteria but does specify if the transaction is expected to be completed, terminated, or cancelled.



## **7.2.3 Test Execution**

### **7.2.3.1 General**

The ITL Service Desk is responsible for orchestrating the execution of the test plan in coordination with the registry. The steps for test case execution are specified in each test case below.

### **7.2.3.2 Reconciliation Tests**

Each test scenario ends with a reconciliation test case as a further criterion of success for the preceding test cases in the scenario. Failure in these confirmation reconciliations will extend the duration of test execution. In the event a reconciliation test case fails with inconsistent holdings, the ITL Service Desk will complete the inconsistent reconciliation with manual intervention and initiate a follow-on reconciliation to determine if the inconsistency persists. The ITL Services Desk and registry are responsible for an investigation to determine the source of the inconsistency. The ITL Service Desk and registry will document the cause of the inconsistency, if identified. To help mitigate the impact on test duration, the ITL Service Desk and registry should conclude the investigation of an inconsistent reconciliation in a timely manner. Generally, failure of an end-of-test-scenario reconciliation test case will require clearing-down the ITL and registry and repeating the test scenario.

## 20 7.3. Test Scenarios

### 7.3.1 Scenario 1 – Issuance and External Transfer of AAUs and RMUs

#### 7.3.1.1 Overview

This test scenario is designed to verify the registry's ability to issue AAUs and RMUs and to transfer these units via the External Transfer transaction. In addition, it ensures that the registry is capable of receiving AAUs and RMUs from foreign registries via the External Transfer transaction.

#### 7.3.1.2 Assumptions

- Scenario begins with all transaction, unit holding, notification, and message log data cleared.
- All unit blocks involved in this scenario have an original and applicable commitment period of CP2.

#### 7.3.1.3 Test Cases

<b>ID</b>	1.1	<b>Description</b>	Successful AAU issuance
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry proposes issuance of 2,500,000,000 AAUs		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Issuance transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	1.2	<b>Description</b>	AAU issuance in excess of the AAU issuance limit
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry proposes issuance of 600,000,000 AAUs		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• Issuance transaction is rejected by the ITL with response code 5008 – the final transaction status at both the registry and the ITL reflects termination.</li> </ul>		

<b>ID</b>	1.3	<b>Description</b>	Successful AAU issuance, within limit
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry proposes issuance of 500,000,000 AAUs		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Issuance transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	1.4	<b>Description</b>	Successful RMU issuance, LULUCF activity 1
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry proposes issuance of 1,500,000 RMUs with LULUCF activity 1		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Issuance transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	1.5	<b>Description</b>	Successful RMU issuance, LULUCF activity 2
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry proposes issuance of 1,500,000 RMUs with LULUCF activity 2		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Issuance transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	1.6	<b>Description</b>	RMU LULUCF activity 1 in excess of the limit
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry proposes issuance of 100,000 RMUs with LULUCF activity 1		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• Issuance transaction is rejected by the ITL with response code 5009 – the final transaction status at both the registry and the ITL reflects termination.</li> </ul>		

<b>ID</b>	1.7	<b>Description</b>	Successfully receive AAUs via external transfer from another registry
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>Registry XX (operated by ITL Service Desk) issues 3,000,000,000 AAUs</li> <li>Registry XX proposes an external transfer of 1,000,200,000 AAUs (10 transaction blocks of 20,000 AAUs each with originating party XX, and 1 transaction block of 1,000,000,000 AAUs) to registry ZZ.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>The External transfer transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	1.8	<b>Description</b>	Successful external transfer of AAUs
<b>Initiator</b>	Registry		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>Registry ZZ proposes an external transfer of 100,000 AAUs to Registry XX</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>The External transfer transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	1.9	<b>Description</b>	External transfer of AAUs rejected by acquiring registry
<b>Initiator</b>	Registry		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>Registry proposes an external transfer of 100,000 AAUs to registry YY (which is configured to reject incoming transfers)</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>External transfer transaction is rejected by the acquiring registry – final transaction status at both the registry and the ITL reflects termination.</li> </ul>		

<b>ID</b>	1.10	<b>Description</b>	Successful external transfer of AAUs and RMUs
<b>Initiator</b>	Registry		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>Registry proposes an external transfer of 1,000,000 AAUs and 50,000 RMUs with LULUCF activity 1 to Registry XX. (The registry may elect to accomplish this using a single transaction, or multiple transactions).</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>The External transfer transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	1.11	<b>Description</b>	Successfully receive AAUs and RMUs via external transfer from another registry
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>Registry XX (operated by ITL Service Desk) issues 1,500,000 RMUs with LULUCF activity 4.</li> <li>Registry XX (operated by ITL Service Desk) proposes an external transfer of 1,000,000 AAUs (with originating party XX) and 150,000 RMUs with LULUCF activity 4 to registry ZZ.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>The External transfer transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	1.12	<b>Description</b>	Reject incoming external transfer of AAUs from another registry
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>Confirm with Registry ZZ test contact that account ZZ-100-9999-0 does not exist in the registry. If this account does exist, identify a Type 100 holding account number that does not exist in registry ZZ.</li> <li>Registry XX (operated by ITL Service Desk) proposes external transfer of AAUs to Registry ZZ specifying acquiring account ZZ-100-9999-0 (or alternately identified non-existent account number).</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>Registry ZZ rejects the incoming external transfer proposal with response code 5902 (and possibly other response codes in the 5900-range) – final transaction status at both the registry and the ITL reflects termination.</li> </ul>		

<b>ID</b>	1.13	<b>Description</b>	External transfer of AAUs rejected because acquiring registry not eligible
<b>Initiator</b>	Registry		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>Registry proposes an external transfer of 1,000,000 AAUs to Registry RR.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>External transfer transaction is rejected by the ITL with response code 5103 – the final transaction status at both the registry and the ITL reflects termination.</li> </ul>		

<b>ID</b>	1.14	<b>Description</b>	External transfer of AAUs to non-responsive acquiring registry is cancelled.
<b>Initiator</b>	Registry, ITL Service Desk		
<b>Test Steps</b>	<u>Registry</u> 1. Registry proposes an external transfer of 1,000,000 AAUs to non-responsive Registry QQ.  <u>ITL Service Desk</u> 2. After 3 minutes have passed, run the transaction clean-up job for the ITL.		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>External transfer transaction is approved by the ITL with a status of 2-Checked, No Discrepancy (or 9-STL Checked, No Discrepancy if Registry QQ or Registry ZZ is an EU registry), but the acquiring registry QQ does not respond to the proposal.</li> <li>When the transaction clean-up job is run, the transaction is successfully cancelled. The final transaction status at both the registry and the ITL reflects cancellation.</li> </ul>		

*Note: Tests 1.15 and 1.16 below are only applicable for EU ETS registries. Non-EU ETS registries should skip directly to 1.17 below.*

<b>ID</b>	1.15	<b>Description</b>	External transfer rejected by EUTL with 7102 ( <b>EU ETS Registries Only</b> )
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry proposes an external transfer of 1,000,000 AAUs to Registry XX which is rejected by the EUTL with response code 7102.		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>External transfer is rejected by the EUTL with response code 7102 – the final transaction status at the EUTL, ITL, and Registry reflects termination.</li> </ul>		

<b>ID</b>	1.16	<b>Description</b>	External transfer, EUTL non-responsive ( <b>EU ETS Registries Only</b> )
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	1. Registry proposes an external transfer of 1,000,000 AAUs to Registry XX and the EUTL is non-responsive.		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>External transfer is stalled when the EUTL does not respond to the transaction proposal. After 5 minutes, the transaction is cancelled. The final transaction status at the EUTL, ITL, and Registry reflects transaction cancellation.</li> </ul>		

<b>ID</b>	1.17	<b>Description</b>	Reconciliation
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	1. ITL Service Desk initiates a ProvideTime request to the Registry. 2. ITL Service Desk initiates reconciliation with Registry.		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>ProvideTime response received from the registry within tolerances (five seconds) defined in current ITL operating procedures.</li> <li>Reconciliation completes successfully – final reconciliation status at both the registry and the ITL is 5-ITL Completed.</li> </ul>		

<b>ID</b>	1.18	<b>Description</b>	Account balance check	
<b>Initiator</b>	ITL Service Desk			
<b>Test Steps</b>	1. ITL Service Desk compares records of account holdings to the expected results defined in the Success Criteria.			
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>The account holdings match the following:</li> </ul>			
	<b>Type</b>	<b>Description</b>	<b>CP</b>	<b>Holdings</b>
	100	Holding Account	0	<ul style="list-style-type: none"> <li>3,000,100,000 AAUs</li> <li>3,100,000 RMUs</li> </ul>
	210	Net Source Cancellation Account	2	
	220	Non-compliance Cancellation Account	2	
	230	Voluntary Cancellation Account	2	
	230	Voluntary Cancellation Account	3	
	250	Mandatory Cancellation Account	2	
	300	Retirement Account	2	
	300	Retirement Account	3	
	411	tCER Replacement Account for Expiry	2	
	421	ICER Replacement Account for Expiry	2	
	422	ICER Replacement Account for Reversal of Storage	2	
423	ICER Replacement Account for Non-submission of Certification Report	2		

## 7.3.2 Scenario 2 – Conversion

### 7.3.2.1 Overview

This test scenario tests the registry's ability to convert AAUs and RMUs into ERUs under JI projects.

### 7.3.2.2 Assumptions

- Scenario begins with all transaction, unit holding, notification, and message log data cleared.
- All unit blocks involved in this scenario have an original and applicable commitment period of CP2.

### 7.3.2.3 Test Cases

<b>ID</b>	2.1	<b>Description</b>	Successful AAU issuance
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry proposes issuance of 2,700,000,000 AAUs.		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Issuance transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	2.2	<b>Description</b>	Receive AAUs and RMUs
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. Registry XX (operated by ITL Service Desk) issues 3,000,000,000 AAUs</li> <li>2. Registry XX (operated by ITL Service Desk) issues 1,500,000 RMUs with LULUCF activity 1.</li> <li>3. Registry XX (operated by ITL Service Desk) issues 1,500,000 RMUs with LULUCF activity 2.</li> <li>4. Registry XX (operated by ITL Service Desk) issues 1,500,000 RMUs with LULUCF activity 4.</li> <li>5. Registry XX (operated by ITL Service Desk) proposes an external transfer to Registry ZZ 3,000,000,000 AAUs, and 4,500,000 RMUs (activities 1, 2 and 4).</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The External transfer transaction includes the quantity and type of units specified in Test Step 5 and completes successfully.</li> </ul>		

<b>ID</b>	2.3	<b>Description</b>	Successful RMU issuance, LULUCF activity 1
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry proposes issuance of 1,500,000 RMUs with LULUCF activity 1.		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Issuance transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	2.4	<b>Description</b>	Successful RMU issuance, LULUCF activity 2
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry proposes issuance of 1,500,000 RMUs with LULUCF activity 2.		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Issuance transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	2.5	<b>Description</b>	AAU conversion under wrong JI project
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry proposes conversion of 123,000 AAUs under project ZZ-1103.		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• Conversion transaction is rejected by the ITL with response code 5060 – the final transaction status at both the registry and the ITL reflects termination.</li> </ul>		

<b>ID</b>	2.6	<b>Description</b>	Successful AAU conversion
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry proposes conversion of 1,500,000 AAUs under project ZZ-1101.		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Conversion transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	2.7	<b>Description</b>	Successful RMU conversion
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry proposes conversion of 1,500,000 RMUs with LULUCF activity 1 under project ZZ-1102.		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>The Conversion transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	2.8	<b>Description</b>	RMU conversion exceeding JI project conversion limit
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry proposes conversion of 1,250,000 RMUs with LULUCF activity 2 under project ZZ-1104.		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>Conversion transaction is rejected by the ITL with response code 5061 – the final transaction status at both the registry and the ITL reflects termination.</li> </ul>		

<b>ID</b>	2.9	<b>Description</b>	Successful ERU transfer
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry ZZ proposes an External Transfer transaction to Registry YY including the following unit types and quantities <ol style="list-style-type: none"> <li>150,000 ERUs from AAUs</li> <li>999,999 ERUs from RMUs</li> </ol> (The registry may elect to accomplish this using a single External Transfer transaction, or multiple External Transfer transactions)		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>The External transfer transaction(s) includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	2.10	<b>Description</b>	Reconciliation
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	1. ITL Service Desk initiates a ProvideTime request to the Registry. 2. ITL Service Desk initiates reconciliation with Registry.		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>ProvideTime response received from the registry within tolerances defined in the DES.</li> <li>Reconciliation completes successfully – final reconciliation status at both the registry and the ITL is 5-ITL Completed.</li> </ul>		

<b>ID</b>	2.11	<b>Description</b>	Account balance check																																																
<b>Initiator</b>	ITL Service Desk																																																		
<b>Test Steps</b>	1. ITL Service Desk compares records of account holdings to the expected results defined in the Success Criteria.																																																		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>The account holdings match the following:</li> </ul> <table border="1" data-bbox="411 1435 1404 2016"> <thead> <tr> <th>Type</th> <th>Description</th> <th>CP</th> <th>Holdings</th> </tr> </thead> <tbody> <tr> <td>100</td> <td>Holding Account</td> <td>0</td> <td> <ul style="list-style-type: none"> <li>5,698,500,000 AAUs</li> <li>6,000,000 RMUs</li> <li>1,350,000 ERU (from AAU)</li> <li>500,001 ERU (from RMU)</li> </ul> </td> </tr> <tr> <td>210</td> <td>Net Source Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>220</td> <td>Non-compliance Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>230</td> <td>Voluntary Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>230</td> <td>Voluntary Cancellation Account</td> <td>3</td> <td></td> </tr> <tr> <td>250</td> <td>Mandatory Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>300</td> <td>Retirement Account</td> <td>2</td> <td></td> </tr> <tr> <td>300</td> <td>Retirement Account</td> <td>3</td> <td></td> </tr> <tr> <td>411</td> <td>tCER Replacement Account for Expiry</td> <td>2</td> <td></td> </tr> <tr> <td>421</td> <td>ICER Replacement Account for Expiry</td> <td>2</td> <td></td> </tr> <tr> <td>422</td> <td>ICER Replacement Account for Reversal of Storage</td> <td>2</td> <td></td> </tr> </tbody> </table>			Type	Description	CP	Holdings	100	Holding Account	0	<ul style="list-style-type: none"> <li>5,698,500,000 AAUs</li> <li>6,000,000 RMUs</li> <li>1,350,000 ERU (from AAU)</li> <li>500,001 ERU (from RMU)</li> </ul>	210	Net Source Cancellation Account	2		220	Non-compliance Cancellation Account	2		230	Voluntary Cancellation Account	2		230	Voluntary Cancellation Account	3		250	Mandatory Cancellation Account	2		300	Retirement Account	2		300	Retirement Account	3		411	tCER Replacement Account for Expiry	2		421	ICER Replacement Account for Expiry	2		422	ICER Replacement Account for Reversal of Storage	2	
Type	Description	CP	Holdings																																																
100	Holding Account	0	<ul style="list-style-type: none"> <li>5,698,500,000 AAUs</li> <li>6,000,000 RMUs</li> <li>1,350,000 ERU (from AAU)</li> <li>500,001 ERU (from RMU)</li> </ul>																																																
210	Net Source Cancellation Account	2																																																	
220	Non-compliance Cancellation Account	2																																																	
230	Voluntary Cancellation Account	2																																																	
230	Voluntary Cancellation Account	3																																																	
250	Mandatory Cancellation Account	2																																																	
300	Retirement Account	2																																																	
300	Retirement Account	3																																																	
411	tCER Replacement Account for Expiry	2																																																	
421	ICER Replacement Account for Expiry	2																																																	
422	ICER Replacement Account for Reversal of Storage	2																																																	

	423	ICER Replacement Account for Non-submission of Certification Report	2	
--	-----	---	---	--



### 7.3.3 Scenario 3 – Cancellation unrelated to CDM projects

#### 7.3.3.1 Overview

This test scenario tests the registry's ability to cancel units unrelated to CDM projects, including voluntary cancellation, and cancellation in response to Net Source Cancellation (Notification Type 1) and Non-compliance Cancellation (Notification Type 2) notifications. This scenario also tests the registry's ability to receive the CPR notification (Notification Type 7).

#### 7.3.3.2 Assumptions

- Scenario begins with all transaction, unit holding, notification, and message log data cleared.
- Government Accounts are NOT loaded until test case 3.1 has been successfully completed.
- All unit blocks involved in this scenario have an original and applicable commitment period of CP2.

#### 7.3.3.3 Test Cases

<b>ID</b>	3.1	<b>Description</b>	Receive AAUs, RMUs, ERUs, and CERs
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. Registry XX (operated by ITL Service Desk) issues 3,000,000,000 AAUs</li> <li>2. Registry XX (operated by ITL Service Desk) issues 1,500,000 RMUs with LULUCF activity 1.</li> <li>3. Registry XX (operated by ITL Service Desk) issues 1,500,000 RMUs with LULUCF activity 2.</li> <li>4. Registry XX (operated by ITL Service Desk) converts 1,500,000 RMUs with LULUCF activity 2 under XX11004.</li> <li>5. CDM Registry (operated by ITL Service Desk) issues 3,000,000,000 CERs under project CN-1.</li> <li>6. CDM Registry (operated by ITL Service Desk) transfers 3,000,000,000 CERs to registry XX.</li> <li>7. Registry XX (operated by ITL Service Desk) proposes an external transfer to Registry ZZ 255,000,000 AAUs, and 1,500,000 RMUs, 1,500,000 ERUs and 3,000,000,000 CERs.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The External transfer transaction includes the quantity and type of units specified in Test Step 7 and completes successfully.</li> </ul>		

<b>ID</b>	3.2	<b>Description</b>	Voluntary cancellation of AAUs without government accounts loaded at ITL
<b>Initiator</b>	Registry		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. Registry proposes voluntary cancellation of 1,000,000 AAUs to Type 230 account.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• Cancellation transaction is rejected by the ITL with response codes 4018, and 5155 – the final transaction status at both the registry and the ITL reflects termination.</li> </ul>		

<b>ID</b>	3.3	<b>Description</b>	Successful voluntary cancellation of AAUs
<b>Initiator</b>	ITL Service Desk, Registry		
<b>Test Steps</b>	<p>ITL Service Desk</p> <ol style="list-style-type: none"> <li>1. ITL Service Desk loads government accounts for Registry.</li> <li>2. ITL Service Desk notifies Registry that government accounts have been loaded and to proceed with voluntary cancellation.</li> </ol> <p>Registry</p> <ol style="list-style-type: none"> <li>3. Registry proposes voluntary cancellation of 1,000,000 AAUs to Type 230 account.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Cancellation transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	3.4	<b>Description</b>	Cancellation to fulfil Net Source Cancellation notification
<b>Initiator</b>	ITL Service Desk, Registry		
<b>Test Steps</b>	<p><u>ITL Service Desk</u></p> <ol style="list-style-type: none"> <li>ITL Service Desk executes script emulating the CAD to change the LULUCF activity 1 accounting quantity, generating a Net Source Cancellation notification (Notification Type 1) with a target value of 2,000,000 applicable to CP2.</li> </ol> <p><u>Registry</u></p> <ol style="list-style-type: none"> <li>After receiving the Net-Source Cancellation notification, the Registry proposes a cancellation transaction to cancel the following unit types and quantities <ol style="list-style-type: none"> <li>500,000 ERUs</li> <li>1,500,000 RMUs</li> </ol> (The registry may elect to accomplish this using a single Cancellation transaction, or multiple Cancellation transactions)</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>Registry receives Net Source Cancellation notification (Notification Type 1).</li> <li>The Cancellation transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	3.5	<b>Description</b>	Cancellation to fulfil Non-compliance Cancellation notification
<b>Initiator</b>	ITL Service Desk, Registry		
<b>Test Steps</b>	<p><u>ITL Service Desk</u></p> <ol style="list-style-type: none"> <li>ITL Service Desk executes script emulating the CAD, generating a Non-compliance Cancellation notification (Notification Type 2) with a target value of 600,000,000 applicable to CP2.</li> </ol> <p><u>Registry</u></p> <ol style="list-style-type: none"> <li>After receiving the Non-compliance Cancellation notification, the Registry proposes a cancellation transaction to cancel the following unit types and quantities <ol style="list-style-type: none"> <li>200,000,000 AAUs</li> <li>400,000,000 CERs</li> </ol> (The registry may elect to accomplish this using a single Cancellation transaction, or multiple Cancellation transactions).</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>Registry receives Non-compliance Cancellation notification (Notification Type 2).</li> <li>The Cancellation transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	3.6	<b>Description</b>	Receive Commitment Period Reserve notification(s) and Receive CERs
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>The last test case generated a Commitment Period Reserve notification (Notification Type 7) with a target value of 45,000,000. If the Registry is an EU 15 registry, then the last test case should have also generated a EU15 Commitment Period Reserve notification (Notification Type 11) with a target value of 45,000,000 for CP2.</li> <li>CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 CERs under project CN-11.</li> <li>CDM Registry (operated by ITL Service Desk) transfers 1,000,000,000 CERs to Registry XX.</li> <li>Registry XX (operated by ITL Service Desk) proposes an external transfer to Registry ZZ 1,000,000,000 CERs.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>Registry receives Commitment Period Reserve notification (Notification Type 7). If an EU 15 registry, Registry also receives EU15 Commitment Period Reserve notification (Notification Type 11).</li> <li>The External transfer transaction to Registry ZZ includes the quantity and type of units specified and completes successfully.</li> </ul>		

<b>ID</b>	3.7	<b>Description</b>	Reconciliation
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>ITL Service Desk initiates a ProvideTime request to the Registry.</li> <li>ITL Service Desk initiates reconciliation with Registry.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>ProvideTime response received from the registry within tolerances defined in the DES.</li> <li>Reconciliation completes successfully – final reconciliation status at both the registry and the ITL is 5-ITL Completed.</li> </ul>		

<b>ID</b>	3.8	<b>Description</b>	Account balance check																																																					
<b>Initiator</b>	ITL Service Desk																																																							
<b>Test Steps</b>	1. ITL Service Desk compares records of account holdings to the expected results defined in the Success Criteria.																																																							
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>The account holdings match the following:</li> </ul> <table border="1"> <thead> <tr> <th>Type</th> <th>Description</th> <th>CP</th> <th>Holdings</th> </tr> </thead> <tbody> <tr> <td>100</td> <td>Holding Account</td> <td>0</td> <td> <ul style="list-style-type: none"> <li>54,000,000 AAUs</li> <li>1,000,000 ERU (from RMU)</li> <li>3,600,000,000 CERs</li> </ul> </td> </tr> <tr> <td>210</td> <td>Net Source Cancellation Account</td> <td>2</td> <td> <ul style="list-style-type: none"> <li>1,500,000 RMUs</li> <li>500,000 ERU (from RMU)</li> </ul> </td> </tr> <tr> <td>220</td> <td>Non-compliance Cancellation Account</td> <td>2</td> <td> <ul style="list-style-type: none"> <li>200,000,000 AAUs</li> <li>400,000,000 CERs</li> </ul> </td> </tr> <tr> <td>230</td> <td>Voluntary Cancellation Account</td> <td>2</td> <td> <ul style="list-style-type: none"> <li>1,000,000 AAUs</li> </ul> </td> </tr> <tr> <td>230</td> <td>Voluntary Cancellation Account</td> <td>3</td> <td></td> </tr> <tr> <td>250</td> <td>Mandatory Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>300</td> <td>Retirement Account</td> <td>2</td> <td></td> </tr> <tr> <td>300</td> <td>Retirement Account</td> <td>3</td> <td></td> </tr> <tr> <td>411</td> <td>tCER Replacement Account for Expiry</td> <td>2</td> <td></td> </tr> <tr> <td>421</td> <td>ICER Replacement Account for Expiry</td> <td>2</td> <td></td> </tr> <tr> <td>422</td> <td>ICER Replacement Account for Reversal of Storage</td> <td>2</td> <td></td> </tr> <tr> <td>423</td> <td>ICER Replacement Account for Non-submission of Certification Report</td> <td>2</td> <td></td> </tr> </tbody> </table>				Type	Description	CP	Holdings	100	Holding Account	0	<ul style="list-style-type: none"> <li>54,000,000 AAUs</li> <li>1,000,000 ERU (from RMU)</li> <li>3,600,000,000 CERs</li> </ul>	210	Net Source Cancellation Account	2	<ul style="list-style-type: none"> <li>1,500,000 RMUs</li> <li>500,000 ERU (from RMU)</li> </ul>	220	Non-compliance Cancellation Account	2	<ul style="list-style-type: none"> <li>200,000,000 AAUs</li> <li>400,000,000 CERs</li> </ul>	230	Voluntary Cancellation Account	2	<ul style="list-style-type: none"> <li>1,000,000 AAUs</li> </ul>	230	Voluntary Cancellation Account	3		250	Mandatory Cancellation Account	2		300	Retirement Account	2		300	Retirement Account	3		411	tCER Replacement Account for Expiry	2		421	ICER Replacement Account for Expiry	2		422	ICER Replacement Account for Reversal of Storage	2		423	ICER Replacement Account for Non-submission of Certification Report	2	
Type	Description	CP	Holdings																																																					
100	Holding Account	0	<ul style="list-style-type: none"> <li>54,000,000 AAUs</li> <li>1,000,000 ERU (from RMU)</li> <li>3,600,000,000 CERs</li> </ul>																																																					
210	Net Source Cancellation Account	2	<ul style="list-style-type: none"> <li>1,500,000 RMUs</li> <li>500,000 ERU (from RMU)</li> </ul>																																																					
220	Non-compliance Cancellation Account	2	<ul style="list-style-type: none"> <li>200,000,000 AAUs</li> <li>400,000,000 CERs</li> </ul>																																																					
230	Voluntary Cancellation Account	2	<ul style="list-style-type: none"> <li>1,000,000 AAUs</li> </ul>																																																					
230	Voluntary Cancellation Account	3																																																						
250	Mandatory Cancellation Account	2																																																						
300	Retirement Account	2																																																						
300	Retirement Account	3																																																						
411	tCER Replacement Account for Expiry	2																																																						
421	ICER Replacement Account for Expiry	2																																																						
422	ICER Replacement Account for Reversal of Storage	2																																																						
423	ICER Replacement Account for Non-submission of Certification Report	2																																																						

## 7.3.4 Scenario 4 – Cancellation Related to CDM Projects

### 3.4.1 Overview

This test scenario tests the registry's ability to cancel units in response to ITL notifications related to CDM projects. This includes cancellation in response to Reversal of Storage (Notification Type 4), Non-submission of Certification Report (Notification Type 5), and Excess Issuance (Notification Type 6) notifications. This scenario ensures that the registry is capable of handling transactions involving CERs, tCERs, and ICERs.

### 7.3.4.2 Assumptions

- Scenario begins with all transaction, unit holding, notification, and message log data cleared.
- All unit blocks involved in this scenario have an original and applicable commitment period of CP2.

### 7.3.4.3 Test Cases

ID	4.1	Description	Receive CERs, tCERs, ICERs, and other units
<b>Initiator</b>		ITL Service Desk	
<b>Test Steps</b>		<ol style="list-style-type: none"> <li>1. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 CERs under project CN-1</li> <li>2. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 CERs under project CN-11</li> <li>3. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 tCERs under project AR-2</li> <li>4. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 tCERs under project AR-22</li> <li>5. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 ICERs under project BR-3</li> <li>6. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 ICERs under project BR-33</li> <li>7. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 CERs from project CN-11 to Registry XX</li> <li>8. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 tCERs from project AR-22 to Registry XX</li> <li>9. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 ICERs from project BR-33 to Registry XX</li> <li>10. Registry XX (operated by ITL Service Desk) issues 3,000,000,000 AAUs</li> <li>11. Registry XX (operated by ITL Service Desk) issues 1,500,000 RMUs with LULUCF activity 1</li> <li>12. Registry XX (operated by ITL Service Desk) converts 1,500,000 AAUs into ERUs</li> <li>13. Registry XX (operated by ITL Service Desk) converts 1,000,000 RMUs into ERUs</li> <li>14. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 CERs from project CN-1 to Registry ZZ</li> <li>15. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 tCERs from project AR-2 to Registry ZZ</li> <li>16. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 ICERs from project BR-3 to Registry ZZ</li> <li>17. Registry XX (operated by ITL Service Desk) transfers the following units to Registry ZZ: <ol style="list-style-type: none"> <li>a. 200,000,000 AAUs</li> <li>b. 500,000 RMUs with LULUCF activity 1</li> <li>c. 1,500,000 ERUs from AAU</li> <li>d. 1,000,000 ERUs from RMU</li> <li>e. 1,000,000,000 CERs from project CN-11</li> <li>f. 1,000,000,000 tCERs from project AR-22</li> <li>g. 1,000,000,000 ICERs from Project BR-33</li> </ol> </li> </ol>	
<b>Success Criteria</b>		<ul style="list-style-type: none"> <li>• The External transfer transactions from the CDM Registry and from Registry XX to Registry ZZ include the quantity and type of units specified above and complete successfully.</li> </ul>	

<b>ID</b>	4.2	<b>Description</b>	Cancellation to fulfill Reversal of Storage notification
<b>Initiator</b>	ITL Service Desk, Registry		
<b>Test Steps</b>	<u>ITL Service Desk</u> 1. ITL Service Desk executes script emulating the CDMIS to generate a Reversal of Storage notification (Notification Type 4) for project BR-3 with a target value of 222,000,000  <u>Registry</u> 2. After receiving the Reversal of Storage notification, the Registry proposes cancellation of 222,000,000 ICERs to fulfill the Reversal of Storage notification.		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>Registry receives Reversal of Storage notification (Notification Type 4).</li> <li>The Cancellation transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	4.3	<b>Description</b>	Cancellation to fulfill Non-submission of Certification Report notification
<b>Initiator</b>	ITL Service Desk, Registry		
<b>Test Steps</b>	<u>ITL Service Desk</u> 1. ITL Service Desk executes script emulating the CDMIS to generate a Non-submission of Certification Report notification (Notification Type 5) for project BR-33 with a target value of 1,000,000,000.  <u>Registry</u> 2. After receiving the Non-submission of Certification Report notification (Notification Type 5), the Registry proposes cancellation of 1,000,000,000 ICERs.		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>Registry receives Non-submission of Certification Report notification (Notification Type 5).</li> <li>The Cancellation transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	4.4	<b>Description</b>	External transfer to fulfill Excess Issuance notification but acquiring account configured to reject the transaction
<b>Initiator</b>	ITL Service Desk, Registry		
<b>Test Steps</b>	<u>ITL Service Desk</u> 1. ITL Service Desk executes script emulating the CDMIS to generate a Excess Issuance notification (Notification Type 6) with a target value of 6,000,000.  <u>Registry</u> 2. After receiving the Excess Issuance notification (Notification Type 6), the Registry proposes transfer to CDM Excess Issuance Account of 3,000,000 CERs. (The CDM Excess Issuance Account is configured in the CDM Registry to reject incoming transfers.)		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>Registry receives Excess Issuance notification (Notification Type 6).</li> <li>External transfer transaction is rejected by the acquiring registry – final transaction status at both the registry and the ITL reflects termination.</li> </ul>		

<b>ID</b>	4.5	<b>Description</b>	Successful external transfer to partially fulfil Excess Issuance notification
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry again proposes transfer to CDM Excess Issuance Account of 3,000,000 CERs to fulfill the Excess Issuance notification (Notification Type 6) from Test Case 4.4. (The CDM Excess Issuance Account is configured in the CDM Registry to accept incoming transfers.)		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>The Cancellation transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	4.6	<b>Description</b>	Notification update and successful external transfer to fulfill Excess Issuance notification
<b>Initiator</b>	ITL Service Desk, Registry		
<b>Test Steps</b>	<p>ITL Service Desk</p> <ol style="list-style-type: none"> <li>ITL Service Desk executes script emulating the CDMIS to generate a Notification update (Notification Type 10) for the Excess Issuance notification in Test Case 4.4 with a target value of 3,000,000.</li> </ol> <p>Registry</p> <ol style="list-style-type: none"> <li>After receiving the Notification update (Notification Type 10), the Registry proposes a cancellation transaction in response to the Excess Issuance notification received to cancel the following unit types and quantities <ol style="list-style-type: none"> <li>1,500,000 AAUs</li> <li>500,000 RMUs</li> <li>1,000,000 ERUs</li> </ol> (The registry may elect to accomplish this using a single External Transfer transaction, or multiple External Transfer transactions).</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>Registry receives Notification update (Notification Type 10).</li> <li>The External transfer transaction(s) includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	4.7	<b>Description</b>	Reconciliation
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>ITL Service Desk initiates a ProvideTime request to the Registry.</li> <li>ITL Service Desk initiates reconciliation with Registry.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>ProvideTime response received from the registry within tolerances defined in the DES.</li> <li>Reconciliation completes successfully – final reconciliation status at both the registry and the ITL is 5-ITL Completed.</li> </ul>		

<b>ID</b>	4.8	<b>Description</b>	Account balance check																																																				
<b>Initiator</b>	ITL Service Desk																																																						
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>ITL Service Desk compares records of account holdings to the excepted results defined in the Success Criteria.</li> </ol>																																																						
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>The account holdings match the following:</li> </ul> <table border="1"> <thead> <tr> <th>Type</th> <th>Description</th> <th>CP</th> <th>Holdings</th> </tr> </thead> <tbody> <tr> <td>100</td> <td>Holding Account</td> <td>0</td> <td> <ul style="list-style-type: none"> <li>198,500,00 AAUs</li> <li>1,500,000 ERUs</li> <li>1,997,000,000 CERs</li> <li>2,000,000,000 tCERs</li> <li>778,000,000 ICERs</li> </ul> </td> </tr> <tr> <td>210</td> <td>Net Source Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>220</td> <td>Non-compliance Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>230</td> <td>Voluntary Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>230</td> <td>Voluntary Cancellation Account</td> <td>3</td> <td></td> </tr> <tr> <td>250</td> <td>Mandatory Cancellation Account</td> <td>2</td> <td> <ul style="list-style-type: none"> <li>1,222,000,000 ICERs</li> </ul> </td> </tr> <tr> <td>300</td> <td>Retirement Account</td> <td>2</td> <td></td> </tr> <tr> <td>300</td> <td>Retirement Account</td> <td>3</td> <td></td> </tr> <tr> <td>411</td> <td>tCER Replacement Account for Expiry</td> <td>2</td> <td></td> </tr> <tr> <td>421</td> <td>ICER Replacement Account for Expiry</td> <td>2</td> <td></td> </tr> <tr> <td>422</td> <td>ICER Replacement Account for Reversal of Storage</td> <td>2</td> <td></td> </tr> <tr> <td>423</td> <td>ICER Replacement Account for Non-submission of Certification Report</td> <td>2</td> <td></td> </tr> </tbody> </table>			Type	Description	CP	Holdings	100	Holding Account	0	<ul style="list-style-type: none"> <li>198,500,00 AAUs</li> <li>1,500,000 ERUs</li> <li>1,997,000,000 CERs</li> <li>2,000,000,000 tCERs</li> <li>778,000,000 ICERs</li> </ul>	210	Net Source Cancellation Account	2		220	Non-compliance Cancellation Account	2		230	Voluntary Cancellation Account	2		230	Voluntary Cancellation Account	3		250	Mandatory Cancellation Account	2	<ul style="list-style-type: none"> <li>1,222,000,000 ICERs</li> </ul>	300	Retirement Account	2		300	Retirement Account	3		411	tCER Replacement Account for Expiry	2		421	ICER Replacement Account for Expiry	2		422	ICER Replacement Account for Reversal of Storage	2		423	ICER Replacement Account for Non-submission of Certification Report	2	
Type	Description	CP	Holdings																																																				
100	Holding Account	0	<ul style="list-style-type: none"> <li>198,500,00 AAUs</li> <li>1,500,000 ERUs</li> <li>1,997,000,000 CERs</li> <li>2,000,000,000 tCERs</li> <li>778,000,000 ICERs</li> </ul>																																																				
210	Net Source Cancellation Account	2																																																					
220	Non-compliance Cancellation Account	2																																																					
230	Voluntary Cancellation Account	2																																																					
230	Voluntary Cancellation Account	3																																																					
250	Mandatory Cancellation Account	2	<ul style="list-style-type: none"> <li>1,222,000,000 ICERs</li> </ul>																																																				
300	Retirement Account	2																																																					
300	Retirement Account	3																																																					
411	tCER Replacement Account for Expiry	2																																																					
421	ICER Replacement Account for Expiry	2																																																					
422	ICER Replacement Account for Reversal of Storage	2																																																					
423	ICER Replacement Account for Non-submission of Certification Report	2																																																					

## 7.3.5 Scenario 5 – CDM Notifications

### 7.3.5.1 Overview

This test scenario tests the registry's ability to receive and act upon notifications related to CDM projects by executing both Cancellation and Replacement transactions. This includes Impending Expiry (Notification Type 3), Reversal of Storage (Notification Type 4), and Non-submission of Certification Report (Notification Type 5) notifications. In addition, this scenario includes tests for expiry date change of tCERs and ICERs.

### 7.3.5.2 Assumptions

- Scenario begins with all transaction, unit holding, notification, and message log data cleared.
- All unit blocks involved in this scenario have an original and applicable commitment period of CP2.

### 7.3.5.3 Test Cases

ID	5.1	Description	Receive AAUs, RMUs, ERUs, CERs, tCERs, and ICERs
<b>Initiator</b>		ITL Service Desk	
<b>Test Steps</b>		<ol style="list-style-type: none"> <li>1. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 CERs under project CN-1</li> <li>2. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 CERs under project CN-11</li> <li>3. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 tCERs under project AR-2</li> <li>4. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 tCERs under project AR-22</li> <li>5. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 ICERs under project BR-3</li> <li>6. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 ICERs under project BR-33</li> <li>7. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 CERs from project CN-11 to Registry XX</li> <li>8. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 tCERs from project AR-22 to Registry XX</li> <li>9. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 ICERs from project BR-33 to Registry XX</li> <li>10. Registry XX (operated by ITL Service Desk) issues 3,000,000,000 AAUs</li> <li>11. Registry XX (operated by ITL Service Desk) issues 2,500,000 RMUs with LULUCF activity 1</li> <li>12. Registry XX (operated by ITL Service Desk) converts 1,500,000 AAUs into ERUs</li> <li>13. Registry XX (operated by ITL Service Desk) converts 1,000,000 RMUs into ERUs</li> <li>14. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 CERs from project CN-1 to Registry ZZ</li> <li>15. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 tCERs from project AR-2 to Registry ZZ</li> <li>16. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 ICERs from project BR-3 to Registry ZZ</li> <li>17. Registry XX (operated by ITL Service Desk) transfers the following units to Registry ZZ: <ol style="list-style-type: none"> <li>a. 200,000,000 AAUs</li> <li>b. 1,500,000 RMUs with LULUCF activity 1</li> <li>c. 1,500,000 ERUs from AAU</li> <li>d. 1,000,000 ERUs from RMU</li> <li>e. 1,000,000,000 CERs from project CN-11</li> <li>f. 1,000,000,000 tCERs from project AR-22</li> <li>g. 1,000,000,000 ICERs from Project BR-33</li> </ol> </li> </ol>	
<b>Success Criteria</b>		<ul style="list-style-type: none"> <li>• The External transfer transactions from the CDM Registry and from Registry XX to Registry ZZ include the quantity and type of units specified above and complete successfully.</li> </ul>	

<b>ID</b>	5.2	<b>Description</b>	Reversal of Storage notification (Type 4) fulfilled with Cancellation and Replacement.
<b>Initiator</b>	ITL Service Desk, Registry		
<b>Test Steps</b>	<u>ITL Service Desk</u> 1. ITL Service Desk executes script emulating the CDMIS to indicate that there has been a reversal of storage for ICER project BR-3, generating a Reversal of Storage notification (Notification Type 4) with a target value of 10,000,000  <u>Registry</u> 2. After receiving the Reversal of Storage notification (Notification Type 4), the Registry proposes Cancellation of 5,000,000 ICERs from project BR-3 to partially fulfill the Reversal of Storage notification requirement. 3. Registry proposes Replacement of 5,000,000 ICERs from project BR-3 using 4,000,000 AAUs and 1,000,000 RMUs as the replacing units to fulfill the remainder of the Reversal of Storage notification requirement. (The registry may elect to accomplish this using a single Replacement transaction, or multiple Replacement transactions)		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>Registry receives Reversal of Storage notification (Notification Type 4)</li> <li>The Cancellation transaction includes the quantity and type of units specified in Test Step 2 and completes successfully. The Replacement transaction(s) includes the quantity and type of units specified in Test Step 3 and completes successfully.</li> </ul>		

<b>ID</b>	5.3	<b>Description</b>	Successful Retirement of tCERs
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry proposes a Retirement transaction of 1,000,000 tCERs from project AR-2.		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>The Retirement transaction includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	5.4	<b>Description</b>	Successful tCER Expiry Date Change
<b>Initiator</b>	ITL Service Desk, Registry		
<b>Test Steps</b>	<u>ITL Service Desk</u> 1. ITL Service Desk triggers an Expiry Date Change notification (Notification Type 9) for tCERs that is sent to the Registry indicating a new expiry date of 31/12/2023. As tCER expiry dates are tied to the end date of the commitment period subsequent to the Commitment Period in which the tCER was issued, this involves changing the end date of Commitment Period 3.  <u>Registry</u> 2. After receiving the Expiry Date Change notification (Notification Type 9), the Registry proposes an Expiry Date Change transaction to fulfill the Expiry Date Change notification requirement.		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>Registry receives Expiry Date Change notification (Notification Type 9)</li> <li>The Expiry Date Change transaction includes the quantity of units and target date specified in the ITL notification and completes successfully.</li> </ul>		



<b>ID</b>	5.5	<b>Description</b>	Replacement due to impending expiry of tCERs (Notification Type 3)
<b>Initiator</b>	ITL Service Desk, Registry		
<b>Test Steps</b>	<p><u>ITL Service Desk</u></p> <ol style="list-style-type: none"> <li>ITL Service Desk executes the “Impending tCER or ICER Expiry” job which triggers an Impending Expiry of tCER/ICER notification (Notification Type 3) affecting tCERs that were retired under test case 5.3.</li> </ol> <p><u>Registry</u></p> <ol style="list-style-type: none"> <li>After receiving the Impending Expiry of tCER/ICER notification, the Registry proposes a Replacement transaction to replace the 1,000,000 retired tCERs. The Registry should use the following replacing blocks for the replacement transaction: <ol style="list-style-type: none"> <li>500,000 AAUs</li> <li>200,000 RMUs</li> <li>200,000 ERUs</li> <li>100,000 CERs</li> </ol> (The registry may elect to accomplish this using a single Replacement transaction, or multiple Replacement transactions)</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>Registry receives Impending Expiry of tCER/ICER notification (Notification Type 3).</li> <li>The Replacement transaction(s) includes the quantity and type of units specified in Test Step 2 and completes successfully.</li> </ul>		

<b>ID</b>	5.6	<b>Description</b>	Successful ICER Expiry Date Change
<b>Initiator</b>	ITL Service Desk, Registry		
<b>Test Steps</b>	<p><u>ITL Service Desk</u></p> <ol style="list-style-type: none"> <li>ITL Service Desk executes a script which emulates the CDM IS updating the crediting period end date of project BR-33, which in turn triggers an Expiry Date Change notification (Notification Type 9) for ICERs from project BR-33 that is sent to the Registry.</li> </ol> <p><u>Registry</u></p> <ol style="list-style-type: none"> <li>After receiving the Expiry Date Change notification, the Registry proposes an Expiry Date Change transaction to fulfill the Expiry Date Change notification requirement.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>The Expiry Date Change transaction includes the quantity of units and target date specified in the ITL notification and completes successfully.</li> </ul>		

<b>ID</b>	5.7	<b>Description</b>	Reconciliation
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>ITL Service Desk initiates a ProvideTime request to the Registry.</li> <li>ITL Service Desk initiates reconciliation with Registry.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>ProvideTime response received from the registry within tolerances defined in the DES.</li> <li>Reconciliation completes successfully – final reconciliation status at both the registry and the ITL is 5-ITL Completed.</li> </ul>		

<b>ID</b>	5.8	<b>Description</b>	Account balance check																																																					
<b>Initiator</b>	ITL Service Desk																																																							
<b>Test Steps</b>	1. ITL Service Desk compares records of account holdings to the expected results defined in the Success Criteria.																																																							
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>The account holdings match the following:</li> </ul> <table border="1"> <thead> <tr> <th>Type</th> <th>Description</th> <th>CP</th> <th>Holdings</th> </tr> </thead> <tbody> <tr> <td>100</td> <td>Holding Account</td> <td>0</td> <td> <ul style="list-style-type: none"> <li>195,500,000 AAUs</li> <li>300,000 RMUs</li> <li>2,300,000 ERUs</li> <li>1,999,900,000 CERs</li> <li>1,999,000,000 tCERs</li> <li>1,995,000,000 ICERs</li> </ul> </td> </tr> <tr> <td>210</td> <td>Net Source Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>220</td> <td>Non-compliance Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>230</td> <td>Voluntary Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>230</td> <td>Voluntary Cancellation Account</td> <td>3</td> <td></td> </tr> <tr> <td>250</td> <td>Mandatory Cancellation Account</td> <td>2</td> <td> <ul style="list-style-type: none"> <li>5,000,000 ICERs</li> </ul> </td> </tr> <tr> <td>300</td> <td>Retirement Account</td> <td>2</td> <td> <ul style="list-style-type: none"> <li>1,000,000 tCERs</li> </ul> </td> </tr> <tr> <td>300</td> <td>Retirement Account</td> <td>3</td> <td></td> </tr> <tr> <td>411</td> <td>tCER Replacement Account for Expiry</td> <td>2</td> <td> <ul style="list-style-type: none"> <li>500,000 AAUs</li> <li>200,000 RMUs</li> <li>200,000 ERUs</li> <li>100,000 CERs</li> </ul> </td> </tr> <tr> <td>421</td> <td>ICER Replacement Account for Expiry</td> <td>2</td> <td></td> </tr> <tr> <td>422</td> <td>ICER Replacement Account for Reversal of Storage</td> <td>2</td> <td> <ul style="list-style-type: none"> <li>4,000,000 AAUs</li> <li>1,000,000 RMUs</li> </ul> </td> </tr> <tr> <td>423</td> <td>ICER Replacement Account for Non-submission of Certification Report</td> <td>2</td> <td></td> </tr> </tbody> </table>				Type	Description	CP	Holdings	100	Holding Account	0	<ul style="list-style-type: none"> <li>195,500,000 AAUs</li> <li>300,000 RMUs</li> <li>2,300,000 ERUs</li> <li>1,999,900,000 CERs</li> <li>1,999,000,000 tCERs</li> <li>1,995,000,000 ICERs</li> </ul>	210	Net Source Cancellation Account	2		220	Non-compliance Cancellation Account	2		230	Voluntary Cancellation Account	2		230	Voluntary Cancellation Account	3		250	Mandatory Cancellation Account	2	<ul style="list-style-type: none"> <li>5,000,000 ICERs</li> </ul>	300	Retirement Account	2	<ul style="list-style-type: none"> <li>1,000,000 tCERs</li> </ul>	300	Retirement Account	3		411	tCER Replacement Account for Expiry	2	<ul style="list-style-type: none"> <li>500,000 AAUs</li> <li>200,000 RMUs</li> <li>200,000 ERUs</li> <li>100,000 CERs</li> </ul>	421	ICER Replacement Account for Expiry	2		422	ICER Replacement Account for Reversal of Storage	2	<ul style="list-style-type: none"> <li>4,000,000 AAUs</li> <li>1,000,000 RMUs</li> </ul>	423	ICER Replacement Account for Non-submission of Certification Report	2	
Type	Description	CP	Holdings																																																					
100	Holding Account	0	<ul style="list-style-type: none"> <li>195,500,000 AAUs</li> <li>300,000 RMUs</li> <li>2,300,000 ERUs</li> <li>1,999,900,000 CERs</li> <li>1,999,000,000 tCERs</li> <li>1,995,000,000 ICERs</li> </ul>																																																					
210	Net Source Cancellation Account	2																																																						
220	Non-compliance Cancellation Account	2																																																						
230	Voluntary Cancellation Account	2																																																						
230	Voluntary Cancellation Account	3																																																						
250	Mandatory Cancellation Account	2	<ul style="list-style-type: none"> <li>5,000,000 ICERs</li> </ul>																																																					
300	Retirement Account	2	<ul style="list-style-type: none"> <li>1,000,000 tCERs</li> </ul>																																																					
300	Retirement Account	3																																																						
411	tCER Replacement Account for Expiry	2	<ul style="list-style-type: none"> <li>500,000 AAUs</li> <li>200,000 RMUs</li> <li>200,000 ERUs</li> <li>100,000 CERs</li> </ul>																																																					
421	ICER Replacement Account for Expiry	2																																																						
422	ICER Replacement Account for Reversal of Storage	2	<ul style="list-style-type: none"> <li>4,000,000 AAUs</li> <li>1,000,000 RMUs</li> </ul>																																																					
423	ICER Replacement Account for Non-submission of Certification Report	2																																																						

## 7.3.6 Scenario 6 – End of CP2

### 7.3.6.1 Overview

This test scenario focuses on events and transactions that are expected to take place at, or near, the end of Commitment Period 2 (CP2), and the start of Commitment Period 3 (CP3). This includes Carry-over and Retirement transactions, as well as Issuance, External Transfer (incoming), Retirement, and Voluntary Cancellation of units from CP3.

### 7.3.6.2 Assumptions

- Scenario begins with all transaction, unit holding, notification, and message log data cleared.
- All unit blocks involved in test cases 6.1 through 6.6 are from CP2. The remaining test cases involve units from CP3 (explicitly stated).

### 7.3.6.3 Test Cases

ID	6.1	Description	Receive AAUs, RMUs, ERUs, CERs, tCERs, and ICERs
<b>Initiator</b>		ITL Service Desk	
<b>Test Steps</b>		<ol style="list-style-type: none"> <li>1. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 CERs under project CN-1</li> <li>2. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 CERs under project CN-11</li> <li>3. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 tCERs under project AR-2</li> <li>4. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 tCERs under project AR-22</li> <li>5. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 ICERs under project BR-3</li> <li>6. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 ICERs under project BR-33</li> <li>7. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 CERs from project CN-11 to Registry XX</li> <li>8. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 tCERs from project AR-22 to Registry XX</li> <li>9. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 ICERs from project BR-33 to Registry XX</li> <li>10. Registry XX (operated by ITL Service Desk) issues 3,000,000,000 AAUs</li> <li>11. Registry XX (operated by ITL Service Desk) issues 2,500,000 RMUs with LULUCF activity 1</li> <li>12. Registry XX (operated by ITL Service Desk) converts 1,500,000 AAUs into ERUs</li> <li>13. Registry XX (operated by ITL Service Desk) converts 1,000,000 RMUs into ERUs</li> <li>14. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 CERs from project CN-1 to Registry ZZ</li> <li>15. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 tCERs from project AR-2 to Registry ZZ</li> <li>16. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 ICERs from project BR-3 to Registry ZZ</li> <li>17. Registry XX (operated by ITL Service Desk) transfers the following units to Registry ZZ:               <ol style="list-style-type: none"> <li>a. 200,000,000 AAUs</li> <li>b. 1,500,000 RMUs with LULUCF activity 1</li> <li>c. 1,500,000 ERUs from AAU</li> <li>d. 1,000,000 ERUs from RMU</li> <li>e. 1,000,000,000 CERs from project CN-11</li> <li>f. 1,000,000,000 tCERs from project AR-22</li> <li>g. 1,000,000,000 ICERs from Project BR-33</li> </ol> </li> </ol>	
<b>Success Criteria</b>		<ul style="list-style-type: none"> <li>• The External transfer transactions from the CDM Registry and from Registry XX to Registry ZZ include the quantity and type of units specified above and complete successfully.</li> </ul>	

<b>ID</b>	6.2	<b>Description</b>	Successful AAU and RMU Issuance
<b>Initiator</b>	Registry		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. Registry proposes issuance of 3,000,000,000 AAUs</li> <li>2. Registry proposes issuance of 1,500,000 RMUs with LULUCF activity 1</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Issuance transactions include the quantity and type of units specified above and complete successfully.</li> </ul>		

<b>ID</b>	6.3	<b>Description</b>	Successful voluntary Cancellation of AAUs and CERs
<b>Initiator</b>	Registry		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. Registry proposes cancellation of <ol style="list-style-type: none"> <li>a. 150,000,000 AAUs</li> <li>b. 50,000 CERs</li> </ol> (The registry may elect to accomplish this using a single Cancellation transaction, or multiple Cancellation transactions) </li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Cancellation transaction(s) includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	6.4	<b>Description</b>	Successful Retirement of AAUs, ERUs, CERs, and ICERs
<b>Initiator</b>	Registry		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. Registry proposes retirement of <ol style="list-style-type: none"> <li>a. 2,500,000,000 AAUs</li> <li>b. 1,000,000 ERUs</li> <li>c. 1,500,000 CERs</li> <li>d. 1,000,000 ICERs</li> </ol> (The registry may elect to accomplish this using a single Retirement transaction, or multiple Retirement transactions) </li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Retirement transaction(s) includes the quantity and type of units specified above and completes successfully.</li> </ul>		

**IMPORTANT NOTE**

*This point in the test scenario is intended to represent the commencement of Commitment Period 3. If the registry being tested requires explicit configuration changes to operate under these conditions, please make the required changes at this time.*

<b>ID</b>	6.5	<b>Description</b>	Rejected Carry-over transaction in response to Carry-over notification (Notification Type 8)
<b>Initiator</b>	ITL Service Desk, Registry		
<b>Test Steps</b>	<p><u>ITL Service Desk</u></p> <ol style="list-style-type: none"> <li>1. ITL Service Desk executes a script which emulates the CAD informing the ITL of the Carry-over limit for Registry ZZ, which in turn triggers a unit Carry-over notification (Notification Type 8) with a target value of 300,000,000 that may be carried over to Commitment Period 3 that is sent to the Registry.</li> </ol> <p><u>Registry</u></p> <ol style="list-style-type: none"> <li>2. Registry proposes a Carry-over transaction in response to the Carry-over notification received in Step 1 to carry over 80,000,000 CERs to Commitment Period 3.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Carry-over transaction includes the applicable commitment period, quantity and type of units specified above and– final transaction status at both the registry and the ITL reflects termination with response code 5313.</li> </ul>		

<b>ID</b>	6.6	<b>Description</b>	Successful Carry-over of AAUs, CERs, and ERUs in response to Carry-over (Notification Type 8)
<b>Initiator</b>	Registry		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. Registry proposes a Carry-over transaction in response to the Carry-over notification received in test 6.5 to Carry-over the following unit types and quantities to Commitment Period 3 <ol style="list-style-type: none"> <li>a. 249,500,000 AAUs that were issued by ZZ</li> <li>b. 50,000,000 CERs</li> <li>c. 500,000 ERUs</li> </ol> </li> </ol> <p>(The registry may elect to accomplish this using a single Retirement transaction, or multiple Carry-over transactions)</p>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Carry-over transaction(s) includes the applicable commitment period, quantity and type of units specified above and completes successfully.</li> <li>• ITL Service Desk verifies the AAUs carried-over have the same serial numbers as those in test case 6.2.</li> </ul>		

<b>ID</b>	6.7	<b>Description</b>	Receive Commitment Period 3 AAUs, RMUs, and CERs from Registry YY
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. Registry YY (operated by ITL Service Desk) issues 3,000,000,000 AAUs (applicable/original CP3)</li> <li>2. Registry YY (operated by ITL Service Desk) issues 1,500,000 RMUs with LULUCF activity 1 (applicable/original CP3)</li> <li>3. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 CERs under project CN-111 (applicable/original CP3)</li> <li>4. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 CERs from project CN-111 to Registry YY (applicable/original CP3)</li> <li>5. Registry YY (operated by ITL Service Desk) transfers the following units to Registry ZZ: <ol style="list-style-type: none"> <li>a. 200,000,000 AAUs (CP3)</li> <li>b. 1,500,000 RMUs with LULUCF activity 1 (CP3)</li> <li>c. 1,000,000,000 CERs from project CN-111 (CP3)</li> </ol> </li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The External transfer transactions from the CDM Registry and from Registry YY to Registry ZZ include the quantity and type of units specified above and complete successfully.</li> </ul>		

<b>ID</b>	6.8	<b>Description</b>	Successful issuance of AAUs for Commitment Period 3
<b>Initiator</b>	Registry		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. Registry proposes issuance of 3,000,000,000 AAUs with original and applicable Commitment Period of 3</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Issuance transaction includes the quantity and type of units specified above and completes successfully. The AAUs issued reflect both an applicable and original Commitment Period of 3.</li> </ul>		

<b>ID</b>	6.9	<b>Description</b>	Successful retirement of AAUs for Commitment Period 3
<b>Initiator</b>	Registry		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. Registry proposes retirement of 100,000,000 Commitment Period 3 AAUs carried-over during test case 6.6.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Retirement transaction includes the quantity and type of units specified above and completes successfully.</li> <li>• ITL Service Desk verifies the AAUs retired have the same serial numbers as those in test case 6.6.</li> </ul>		

<b>ID</b>	6.10	<b>Description</b>	Successful voluntary cancellation of RMUs from CP3.
<b>Initiator</b>	Registry		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. Registry proposes voluntary cancellation of 500,000 RMUs (applicable/original CP3)</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Cancellation transaction includes the applicable commitment period, quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	6.11	<b>Description</b>	Reconciliation
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>ITL Service Desk initiates a ProvideTime request to the Registry.</li> <li>ITL Service Desk initiates reconciliation with Registry.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>ProvideTime response received from the registry within tolerances defined in the DES.</li> <li>Reconciliation completes successfully – final reconciliation status at both the registry and the ITL is 5-ITL Completed.</li> </ul>		

<b>ID</b>	6.12	<b>Description</b>	Account balance check																																																				
<b>Initiator</b>	ITL Service Desk																																																						
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>ITL Service Desk compares records of account holdings to the excepted results defined in the Success Criteria.</li> </ol>																																																						
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>The account holdings match the following:</li> </ul> <table border="1" data-bbox="411 705 1404 1733"> <thead> <tr> <th>Type</th> <th>Description</th> <th>CP</th> <th>Holdings</th> </tr> </thead> <tbody> <tr> <td>100</td> <td>Holding Account</td> <td>0</td> <td> <ul style="list-style-type: none"> <li>300,500,000 CP2 AAUs</li> <li>3,000,000 CP2 RMUs</li> <li>1,000,000 CP2 ERUs</li> <li>1,948,450,000 CP2 CERs</li> <li>2,000,000,000 CP2tCERs</li> <li>1,999,000,000 CP2 ICERs</li> <li>3,349,500,000 CP3 AAUs</li> <li>1,000,000 CP3 RMUs</li> <li>500,000 CP3 ERUs</li> <li>1,050,000,000 CP3 CERs</li> </ul> </td> </tr> <tr> <td>210</td> <td>Net Source Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>220</td> <td>Non-compliance Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>230</td> <td>Voluntary Cancellation Account</td> <td>2</td> <td> <ul style="list-style-type: none"> <li>150,000,000 AAUs</li> <li>50,000 CERs</li> </ul> </td> </tr> <tr> <td>230</td> <td>Voluntary Cancellation Account</td> <td>3</td> <td> <ul style="list-style-type: none"> <li>500,000 RMUs</li> </ul> </td> </tr> <tr> <td>250</td> <td>Mandatory Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>300</td> <td>Retirement Account</td> <td>2</td> <td> <ul style="list-style-type: none"> <li>2,500,000,000 AAUs</li> <li>1,000,000 ERUs</li> <li>1,500,000 CERs</li> <li>1,000,000 ICERs</li> </ul> </td> </tr> <tr> <td>300</td> <td>Retirement Account</td> <td>3</td> <td> <ul style="list-style-type: none"> <li>100,000,000 AAUs</li> </ul> </td> </tr> <tr> <td>411</td> <td>tCER Replacement Account for Expiry</td> <td>2</td> <td></td> </tr> <tr> <td>421</td> <td>ICER Replacement Account for Expiry</td> <td>2</td> <td></td> </tr> <tr> <td>422</td> <td>ICER Replacement Account for Reversal of Storage</td> <td>2</td> <td></td> </tr> <tr> <td>423</td> <td>ICER Replacement Account for Non-submission of Certification Report</td> <td>2</td> <td></td> </tr> </tbody> </table>			Type	Description	CP	Holdings	100	Holding Account	0	<ul style="list-style-type: none"> <li>300,500,000 CP2 AAUs</li> <li>3,000,000 CP2 RMUs</li> <li>1,000,000 CP2 ERUs</li> <li>1,948,450,000 CP2 CERs</li> <li>2,000,000,000 CP2tCERs</li> <li>1,999,000,000 CP2 ICERs</li> <li>3,349,500,000 CP3 AAUs</li> <li>1,000,000 CP3 RMUs</li> <li>500,000 CP3 ERUs</li> <li>1,050,000,000 CP3 CERs</li> </ul>	210	Net Source Cancellation Account	2		220	Non-compliance Cancellation Account	2		230	Voluntary Cancellation Account	2	<ul style="list-style-type: none"> <li>150,000,000 AAUs</li> <li>50,000 CERs</li> </ul>	230	Voluntary Cancellation Account	3	<ul style="list-style-type: none"> <li>500,000 RMUs</li> </ul>	250	Mandatory Cancellation Account	2		300	Retirement Account	2	<ul style="list-style-type: none"> <li>2,500,000,000 AAUs</li> <li>1,000,000 ERUs</li> <li>1,500,000 CERs</li> <li>1,000,000 ICERs</li> </ul>	300	Retirement Account	3	<ul style="list-style-type: none"> <li>100,000,000 AAUs</li> </ul>	411	tCER Replacement Account for Expiry	2		421	ICER Replacement Account for Expiry	2		422	ICER Replacement Account for Reversal of Storage	2		423	ICER Replacement Account for Non-submission of Certification Report	2	
Type	Description	CP	Holdings																																																				
100	Holding Account	0	<ul style="list-style-type: none"> <li>300,500,000 CP2 AAUs</li> <li>3,000,000 CP2 RMUs</li> <li>1,000,000 CP2 ERUs</li> <li>1,948,450,000 CP2 CERs</li> <li>2,000,000,000 CP2tCERs</li> <li>1,999,000,000 CP2 ICERs</li> <li>3,349,500,000 CP3 AAUs</li> <li>1,000,000 CP3 RMUs</li> <li>500,000 CP3 ERUs</li> <li>1,050,000,000 CP3 CERs</li> </ul>																																																				
210	Net Source Cancellation Account	2																																																					
220	Non-compliance Cancellation Account	2																																																					
230	Voluntary Cancellation Account	2	<ul style="list-style-type: none"> <li>150,000,000 AAUs</li> <li>50,000 CERs</li> </ul>																																																				
230	Voluntary Cancellation Account	3	<ul style="list-style-type: none"> <li>500,000 RMUs</li> </ul>																																																				
250	Mandatory Cancellation Account	2																																																					
300	Retirement Account	2	<ul style="list-style-type: none"> <li>2,500,000,000 AAUs</li> <li>1,000,000 ERUs</li> <li>1,500,000 CERs</li> <li>1,000,000 ICERs</li> </ul>																																																				
300	Retirement Account	3	<ul style="list-style-type: none"> <li>100,000,000 AAUs</li> </ul>																																																				
411	tCER Replacement Account for Expiry	2																																																					
421	ICER Replacement Account for Expiry	2																																																					
422	ICER Replacement Account for Reversal of Storage	2																																																					
423	ICER Replacement Account for Non-submission of Certification Report	2																																																					

## 7.3.7 Scenario 7 –Complex Reconciliation

### 7.3.7.1 Overview

This scenario includes test cases involving complex reconciliation which includes totals, unit blocks, and audit trail phases.

### 7.3.7.2 Assumptions

- Scenario begins with all transaction, unit holding, notification, and message log data cleared.
- All unit blocks involved in this scenario have an original and applicable commitment period of CP2.

### 7.3.7.3 Test Cases

ID	7.1	Description	Receive AAUs, RMUs, ERUs, CERs, tCERs, and ICERs
<b>Initiator</b>		ITL Service Desk	
<b>Test Steps</b>		<ol style="list-style-type: none"> <li>1. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 CERs under project CN-1</li> <li>2. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 CERs under project CN-11</li> <li>3. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 tCERs under project AR-2</li> <li>4. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 tCERs under project AR-22</li> <li>5. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 ICERs under project BR-3</li> <li>6. CDM Registry (operated by ITL Service Desk) issues 1,000,000,000 ICERs under project BR-33</li> <li>7. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 CERs from project CN-11 to Registry XX</li> <li>8. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 tCERs from project AR-22 to Registry XX</li> <li>9. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 ICERs from project BR-33 to Registry XX</li> <li>10. Registry XX (operated by ITL Service Desk) issues 3,000,000,000 AAUs</li> <li>11. Registry XX (operated by ITL Service Desk) issues 2,500,000 RMUs with LULUCF activity 1</li> <li>12. Registry XX (operated by ITL Service Desk) converts 1,500,000 AAUs into ERUs</li> <li>13. Registry XX (operated by ITL Service Desk) converts 1,000,000 RMUs into ERUs</li> <li>14. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 CERs from project CN-1 to Registry ZZ</li> <li>15. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 tCERs from project AR-2 to Registry ZZ</li> <li>16. CDM Registry (operated by ITL Service Desk) forwards 1,000,000,000 ICERs from project BR-3 to Registry ZZ</li> <li>17. Registry XX (operated by ITL Service Desk) transfers the following units to Registry ZZ:               <ol style="list-style-type: none"> <li>a. 200,000,000 AAUs</li> <li>b. 1,500,000 RMUs with LULUCF activity 1</li> <li>c. 1,500,000 ERUs from AAU</li> <li>d. 1,000,000 ERUs from RMU</li> <li>e. 1,000,000,000 CERs from project CN-11</li> <li>f. 1,000,000,000 tCERs from project AR-22</li> <li>g. 1,000,000,000 ICERs from Project BR-33</li> </ol> </li> </ol>	
<b>Success Criteria</b>		<ul style="list-style-type: none"> <li>• The External transfer transactions from the CDM Registry and from Registry XX to Registry ZZ include the quantity and type of units specified above and complete successfully.</li> </ul>	

ID	7.2	Description	Successful AAU and RMU Issuance
<b>Initiator</b>		Registry	
<b>Test Steps</b>		<ol style="list-style-type: none"> <li>1. Registry proposes issuance of 3,000,000,000 AAUs</li> <li>2. Registry proposes issuance of 1,500,000 RMUs with LULUCF activity 1</li> </ol>	
<b>Success Criteria</b>		<ul style="list-style-type: none"> <li>• The Issuance transactions include the quantity and type of units specified above and complete successfully.</li> </ul>	

<b>ID</b>	7.3	<b>Description</b>	External transfer of AAUs and CERs – ZZ Status Remains “Proposed”
<b>Initiator</b>	Registry		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. Registry proposes External transfer to Registry XX of <ol style="list-style-type: none"> <li>a. 150,000,000 AAUs with originating party code XX received in Test Case 7.1</li> <li>b. 50,000 CERs</li> </ol> (The registry may elect to accomplish this using a single External transfer transaction, or multiple External transfer transactions) </li> <li>2. The acceptNotification message with status ‘Accepted’ is configured to not reach Registry ZZ, so transaction(s) remain in status ‘Proposed.’</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The External transfer transaction(s) includes the quantity and type of units specified above and remains in status ‘Proposed.’</li> </ul>		

<b>ID</b>	7.4	<b>Description</b>	Successful Retirement of AAUs, ERUs, CERs, and ICERs
<b>Initiator</b>	Registry		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. Registry proposes retirement of <ol style="list-style-type: none"> <li>a. 2,500,000,000 AAUs</li> <li>b. 1,000,000 ERUs</li> <li>c. 1,500,000 CERs</li> <li>d. 1,000,000 ICERs</li> </ol> (The registry may elect to accomplish this using a single Retirement transaction, or multiple Retirement transactions) </li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The Retirement transaction(s) includes the quantity and type of units specified above and completes successfully.</li> </ul>		

<b>ID</b>	7.5	<b>Description</b>	Reconciliation inconsistency and manual intervention
<b>Initiator</b>	ITL Service Desk, Registry		
<b>Test Steps</b>	<u>ITL Service Desk</u> <ol style="list-style-type: none"> <li>1. ITL Service Desk initiates reconciliation with Registry ZZ that reflects the External transfer transaction(s) in test case 7.3 is in status ‘Proposed’ at Registry ZZ and ‘ITL Completed’ at the ITL, thereby causing reconciliation inconsistency with the Registry: <ol style="list-style-type: none"> <li>a. Totals requested and do not match</li> <li>b. Unit blocks requested</li> <li>c. Audit trail is requested for all unit blocks identified</li> </ol> </li> <li>2. ITL Service Desk confirms the status of the reconciliation with Registry ZZ and provides notification it will proceed with the next step and specifies the transaction ID(s) (from test case 7.3) to be completed in the next step.</li> <li>3. ITL Service Desk triggers the final acceptNotification message to Registry ZZ for the External transfer transaction(s) (test case 7.3) via manual intervention.</li> <li>4. ITL Service Desk closes the reconciliation with a status of 6-ITL Completed with Manual Intervention.</li> <li>5. ITL Service Desk initiates reconciliation with the Registry.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• Registry ZZ response to the audit trail request provides the transactions relevant to the requested unit blocks.</li> <li>• After Test Step 3, Registry ZZ completes the External transfer transaction(s) from test case 7.3.</li> <li>• Upon completion of Test Step 4 – final reconciliation status at both the registry and the ITL is 6-ITL Completed with Manual Intervention.</li> <li>• The Reconciliation initiated in Test Step 5 completes successfully – final reconciliation status at both the registry and the ITL is 5-ITL Completed.</li> </ul>		



<b>ID</b>	7.6	<b>Description</b>	Account balance check																																																					
<b>Initiator</b>	ITL Service Desk																																																							
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>ITL Service Desk compares records of account holdings to the expected results defined in the Success Criteria.</li> <li>ITL Service Desk initiates reconciliation with Registry.</li> </ol>																																																							
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>The account holdings match the following:</li> </ul> <table border="1"> <thead> <tr> <th>Type</th> <th>Description</th> <th>CP</th> <th>Holdings</th> </tr> </thead> <tbody> <tr> <td>100</td> <td>Holding Account</td> <td>0</td> <td> <ul style="list-style-type: none"> <li>550,000,000 AAUs</li> <li>3,000,0000 RMUs</li> <li>1,500,000 ERUs</li> <li>1,998,450,000 CERs</li> <li>2,000,000,000 tCERs</li> <li>1,999,000,000 ICERs</li> </ul> </td> </tr> <tr> <td>210</td> <td>Net Source Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>220</td> <td>Non-compliance Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>230</td> <td>Voluntary Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>230</td> <td>Voluntary Cancellation Account</td> <td>3</td> <td></td> </tr> <tr> <td>250</td> <td>Mandatory Cancellation Account</td> <td>2</td> <td></td> </tr> <tr> <td>300</td> <td>Retirement Account</td> <td>2</td> <td> <ul style="list-style-type: none"> <li>2,500,000,000 AAUs</li> <li>1,000,000 ERUs</li> <li>1,500,000 CERs</li> <li>1,000,000 ICERs</li> </ul> </td> </tr> <tr> <td>300</td> <td>Retirement Account</td> <td>3</td> <td></td> </tr> <tr> <td>411</td> <td>tCER Replacement Account for Expiry</td> <td>2</td> <td></td> </tr> <tr> <td>421</td> <td>ICER Replacement Account for Expiry</td> <td>2</td> <td></td> </tr> <tr> <td>422</td> <td>ICER Replacement Account for Reversal of Storage</td> <td>2</td> <td></td> </tr> <tr> <td>423</td> <td>ICER Replacement Account for Non-submission of Certification Report</td> <td>2</td> <td></td> </tr> </tbody> </table>				Type	Description	CP	Holdings	100	Holding Account	0	<ul style="list-style-type: none"> <li>550,000,000 AAUs</li> <li>3,000,0000 RMUs</li> <li>1,500,000 ERUs</li> <li>1,998,450,000 CERs</li> <li>2,000,000,000 tCERs</li> <li>1,999,000,000 ICERs</li> </ul>	210	Net Source Cancellation Account	2		220	Non-compliance Cancellation Account	2		230	Voluntary Cancellation Account	2		230	Voluntary Cancellation Account	3		250	Mandatory Cancellation Account	2		300	Retirement Account	2	<ul style="list-style-type: none"> <li>2,500,000,000 AAUs</li> <li>1,000,000 ERUs</li> <li>1,500,000 CERs</li> <li>1,000,000 ICERs</li> </ul>	300	Retirement Account	3		411	tCER Replacement Account for Expiry	2		421	ICER Replacement Account for Expiry	2		422	ICER Replacement Account for Reversal of Storage	2		423	ICER Replacement Account for Non-submission of Certification Report	2	
Type	Description	CP	Holdings																																																					
100	Holding Account	0	<ul style="list-style-type: none"> <li>550,000,000 AAUs</li> <li>3,000,0000 RMUs</li> <li>1,500,000 ERUs</li> <li>1,998,450,000 CERs</li> <li>2,000,000,000 tCERs</li> <li>1,999,000,000 ICERs</li> </ul>																																																					
210	Net Source Cancellation Account	2																																																						
220	Non-compliance Cancellation Account	2																																																						
230	Voluntary Cancellation Account	2																																																						
230	Voluntary Cancellation Account	3																																																						
250	Mandatory Cancellation Account	2																																																						
300	Retirement Account	2	<ul style="list-style-type: none"> <li>2,500,000,000 AAUs</li> <li>1,000,000 ERUs</li> <li>1,500,000 CERs</li> <li>1,000,000 ICERs</li> </ul>																																																					
300	Retirement Account	3																																																						
411	tCER Replacement Account for Expiry	2																																																						
421	ICER Replacement Account for Expiry	2																																																						
422	ICER Replacement Account for Reversal of Storage	2																																																						
423	ICER Replacement Account for Non-submission of Certification Report	2																																																						

## 7.3.8 Scenario 8 – Non-functional and extraordinary cases

### 7.3.8.1 Overview

This test scenario tests focuses on non-functional requirements and special cases. This includes transactions and reconciliation involving large number of unit blocks, unit blocks representing a single unit, long transaction and account identifiers, and multiple simultaneous transactions.

### 7.3.8.2 Assumptions

- Scenario begins with all transaction, unit holding, notification, and message log data cleared.
- All unit blocks involved in this scenario have an original and applicable commitment period of CP2.

### 7.3.8.3 Test Cases

<b>ID</b>	8.1	<b>Description</b>	Multiple simultaneous incoming transfers
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. Registry XX (operated by ITL Service Desk) issues 3,000,000,000 AAUs</li> <li>2. Registry YY (operated by ITL Service Desk) issues 3,000,000,000 AAUs</li> <li>3. Registry QQ (operated by ITL Service Desk) issues 3,000,000,000 AAUs</li> </ol> <p><i>The following transfers are intended to be proposed simultaneously (in rapid succession)</i></p> <ol style="list-style-type: none"> <li>4. Registry XX (operated by ITL Service Desk) transfers 300,000,000 AAUs to Registry ZZ</li> <li>5. Registry YY (operated by ITL Service Desk) transfers 300,000,000 AAUs to Registry ZZ</li> <li>6. Registry QQ (operated by ITL Service Desk) transfers 300,000,000 AAUs to Registry ZZ</li> <li>7. Registry XX (operated by ITL Service Desk) transfers 300,000,000 AAUs to Registry ZZ</li> <li>8. Registry YY (operated by ITL Service Desk) transfers 300,000,000 AAUs to Registry ZZ</li> <li>9. Registry QQ (operated by ITL Service Desk) transfers 300,000,000 AAUs to Registry ZZ</li> <li>10. Registry XX (operated by ITL Service Desk) transfers 300,000,000 AAUs to Registry ZZ</li> <li>11. Registry YY (operated by ITL Service Desk) transfers 300,000,000 AAUs to Registry ZZ</li> <li>12. Registry QQ (operated by ITL Service Desk) transfers 300,000,000 AAUs to Registry ZZ</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The nine External transfer transactions to Registry ZZ include the quantity and type of units specified above and complete successfully.</li> <li>• All transfers complete successfully without timeouts occurring on either the ITL or Registry side.</li> </ul>		

<b>ID</b>	8.2	<b>Description</b>	Incoming transfer – long transaction identifier
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. Registry XX (operated by ITL Service Desk) transfers 100,000,000 AAUs to Registry ZZ using a transaction identifier with a numeric portion with 18 digits (e.g. 999,999,999,999,999,999 omitting commas).</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The External transfer transaction completes successfully.</li> </ul>		

<b>ID</b>	8.3	<b>Description</b>	Incoming transfer – long serial numbers and small blocks
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. Registry YY (operated by ITL Service Desk) transfers the following unit blocks to Registry ZZ <ol style="list-style-type: none"> <li>a. 5 blocks of a single AAU each, and each with an 15-digit number as the serial range start and end values.</li> <li>b. 5 blocks of two AAUs each, and each with an 15-digit number as the serial range start and end values.</li> </ol> </li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The External transfer transactions include the quantity and type of units specified above and complete successfully.</li> </ul>		

<b>ID</b>	8.4	<b>Description</b>	Incoming transfer – long transferring account number
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. Registry QQ (operated by ITL Service Desk) transfers 100,000,000 AAUs to Registry ZZ using a transferring account number with a numeric identifier 15 digits in length.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• The External transfer transaction completes successfully.</li> </ul>		

<b>ID</b>	8.5	<b>Description</b>	Successful Retirement
<b>Initiator</b>	Registry		
<b>Test Steps</b>	1. Registry proposes a Retirement transaction selecting 20 unit blocks (quantity and unit type unspecified).		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>The Retirement transaction completes successfully – final transaction status at both the registry and the ITL is 4-Completed.</li> </ul>		

<b>ID</b>	8.6	<b>Description</b>	Numerous successive incoming transfers – large block quantities								
<b>Initiator</b>	ITL Service Desk										
<b>Test Steps</b>	<p><i>Note: As the completion of multiple transactions with fragmented unit blocks may require an extended period of time to complete, this test case may be executed overnight.</i></p> <p>1. ITL Service Desk executes a script that emulates other registries proposing external transfer transactions to Registry ZZ. Based on the ITL Service Desk’s classification of Registry ZZ’s size as small, medium or large depending on the fragmentation in the production instance of the registry, the test will be executed with one of the three values in format small/medium/large. The classification of Registry ZZ as small, medium or large is based on the number of unit blocks in the most fragmented account type (for the most fragmented unit type) in the ITL Production environment as described below. If Registry ZZ does have ITL production data, then it is classified as small.</p> <table border="1" data-bbox="459 869 1404 1070"> <thead> <tr> <th></th> <th>Small</th> <th>Medium</th> <th>Large</th> </tr> </thead> <tbody> <tr> <td>Number of unit blocks of the most fragmented unit type in the most fragmented account type</td> <td>0-1,000</td> <td>1,001-10,000</td> <td>10,001+</td> </tr> </tbody> </table> <p>a. 10/50/100 transactions total  b. Each transaction will contain 100/300/600 unit blocks  c. Each unit block will range in size between 1 and 500 units  d. All unit types will be included</p>				Small	Medium	Large	Number of unit blocks of the most fragmented unit type in the most fragmented account type	0-1,000	1,001-10,000	10,001+
	Small	Medium	Large								
Number of unit blocks of the most fragmented unit type in the most fragmented account type	0-1,000	1,001-10,000	10,001+								
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>All transfers are complete successfully.</li> </ul>										

<b>ID</b>	8.7	<b>Description</b>	Reconciliation inconsistency – large data volume
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>ITL Service Desk executes a script emulating a significant data inconsistency in the Type 100 holding account receiving transaction in test case 8.6 and in the Type 300 retirement account in test case 8.5.</li> <li>ITL Service Desk initiates reconciliation with Registry: <ol style="list-style-type: none"> <li>Totals requested and do not match</li> <li>Unit blocks requested</li> <li>Audit trail is requested for all unit blocks identified</li> </ol> </li> <li>ITL Service Desk closes the reconciliation with a status of 6-ITL Completed with Manual Intervention.</li> <li>ITL Service Desk executes a script removing the data inconsistency created in Test Step 1.</li> <li>ITL Service Desk initiates reconciliation with Registry.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>Reconciliation initiated in Test Step 2 identifies totals inconsistency (if an ETS registry, inconsistent totals are identified at the account level for each the Type 100 account and the Type 300 account), unit block inconsistency, and audit trail is requested and provided. After totals, unit block, and audit trail data are provided to the ITL – reconciliation status at both the registry and the ITL is 4-ITL Unit Blocks Inconsistent.</li> <li>Upon completion of Test Step 3 – final reconciliation status at both the registry and the ITL is 6-ITL Completed with Manual Intervention.</li> <li>Reconciliation initiated in Test Step 5 completes successfully – final reconciliation status at both the registry and the ITL is 5-ITL Completed.</li> </ul>		

<b>ID</b>	8.8	<b>Description</b>	Handling numerous unit blocks
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<p><i>Note: The completion of transactions with fragmented unit blocks may require an extended period of time to complete.</i></p> <ol style="list-style-type: none"> <li>1. ITL Service Desk executes a script emulating third-party registries proposing external transfers to Registry ZZ that are comprised of progressively larger quantities of unit blocks. The following transfers should be performed in sequence: <ol style="list-style-type: none"> <li>a. Transfer of 100 blocks of AAUs to Registry ZZ</li> <li>b. If successful, transfer 200 blocks of AAUs to Registry ZZ</li> <li>c. If successful, transfer 500 blocks of AAUs to Registry ZZ</li> <li>d. If successful, transfer 1,000 blocks of AAUs to Registry ZZ</li> <li>e. If successful, transfer 2,000 blocks of AAUs to Registry ZZ</li> <li>f. If successful, transfer 3,000 blocks of AAUs to Registry ZZ</li> </ol> </li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• Each of the transfers described in Test Step 1 above either: <ul style="list-style-type: none"> <li>○ Completes successfully, OR</li> <li>○ Registry ZZ gracefully rejects the incoming transfer using a response code 5904 or 5905 from the DES indicating that the quantity is greater than it can handle.</li> </ul> </li> </ul>		

<b>ID</b>	8.9	<b>Description</b>	Reconciliation
<b>Initiator</b>	ITL Service Desk		
<b>Test Steps</b>	<ol style="list-style-type: none"> <li>1. ITL Service Desk initiates a ProvideTime request to the Registry.</li> <li>2. ITL Service Desk initiates reconciliation with Registry.</li> </ol>		
<b>Success Criteria</b>	<ul style="list-style-type: none"> <li>• ProvideTime response received from the registry within tolerances defined in the DES.</li> <li>• Reconciliation completes successfully – final reconciliation status at both the registry and the ITL is 5-ITL Completed.</li> </ul>		

**Note:** Because it cannot be reliably determined exactly how many units are transacted in Scenario 8 for a given registry, there is no account balance check included here. Reconciliation in test 8.9, along with verification of the preceding test cases, will be used to ensure successful completion of this scenario.

## **Annex I**

### **Messaging Service Specifications**

#### **1. Introduction**

This annex provides information on the required XML message structures. Messages are passed in the registry system using the SOAP 1.1 protocol defined by the W3C SOAP1.1, using the RPC/Encoded style.

#### **2. SOAP Message Exchange**

SOAP is a lightweight protocol for exchange of information in a decentralized, distributed environment. It is an XML-based protocol that consists of three parts:

- An envelope that defines a framework for describing what is in a message and how to process it;
- A set of encoding rules for validation of defined elements; and
- A convention for representing remote procedure calls and responses.

Requests (and responses) in the registry system are SOAP messages. Every SOAP message referred to in these specifications has a mandatory envelope and body element. The body contains the actual message to be delivered and processed.

#### **3. Standards and Requirements**

Due to the fact that the exchanges in the registry system will connect heterogeneous systems, the exchange of data must follow a common representation. Annex K (WSDL) describes the message structure and format of every message exchanged in the registry system. Any deviation from these named conventions shall result in the message being rejected.

##### **3.1 Versioning**

All messages shall comply with and be backward compatible with SOAP 1.1. The version of a SOAP message can be determined by the namespace defined for the SOAP message in the envelope. All SOAP messages will be transported by the HTTP1.1 protocol.

##### **3.2 Encoding Styles**

An encoding style is a set of rules that define exactly how data types are to be encoded into a common XML syntax. The SOAP1.1 messages in the registry system are defined using the RPC/Encoded encoding style, in order to guarantee the largest compatibility possible with existing tools and SOAP processors.

##### **3.3 SOAP Interoperability**

Although any implementation of SOAP must be designed to be platform- and language-independent, there are known interoperability problems between the different SOAP implementations. The WSDLs and SOAP messages submitted by the registries shall comply as much as possible with the Web Services Interoperability Organization (WS-I), Basic Profile 1.0a. This profile clarifies the grey areas of SOAP that are the root cause of interoperability problems. The WSDLs defined in Annex K are known to be compliant with the Web Service Interoperability Organization except for the use of RPC/Encoding style and the WSDL:arrayType as array. These two deviations are the result of limitations in one of the major development tools.

### 3.4 SOAP-Specific Port

It is recommended that SOAP messages in the registry system use TCP well-known port 80 for HTTP and 443 for HTTPS. See RFC 1700. Because of its applicative nature, SOAP usage must be restricted to known hosts in the registry system and network administrators should take all necessary measures to allow incoming HTTP and HTTPS requests only from recognized hosts. Additionally, only outgoing HTTP and HTTPS requests to recognized hosts should be authorized.

## 4. SOAP Message Specifications

### 4.1 SOAP Envelope Element

SOAP messages are well-formed XML documents, the envelope must start with an envelope declaration which contains namespace declarations. The envelope element must be prefixed with an indicator of the namespace that defines the SOAP version that is applicable. The version is indicated by the namespace attribute, `xmlns`, included in the envelope element start tag. The namespace prefix could be any valid XML namespace string, but the convention usually adopted is as follows:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/">
```

The namespace name SOAP-ENV is a symbol for `http://schema.xmlsoap.org/soap/envelope`. Although it may be referred to by any name, the URL part must be exactly as specified.

### 4.2 The Header Element

The Header element offers a framework for specifying additional application-level requirements. Due to various interpretations and incomplete support by SOAP processors and their WSDL compiling tools of the header information in SOAP messages, the registry system will not make use of the SOAP header functionality. Rather, information common to all messages will be embedded in the body of the messages.

### 4.3 The Body Element

The Body element is mandatory for all SOAP messages and contains information regarding the specific message being sent. Annex K defines the structure and layout of the messages.

#### 4.3.1 Namespaces

The primary use of namespaces in XML documents is to enable identification of logical structures in documents by software modules such as query processors, stylesheet-driven rendering engines, and schema-driven validators. Separate namespaces shall be used to distinguish elements that are associated with all types from those associated with operations. The following namespaces have been defined for the registry system:

- "urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0," the target namespace for all operations.
- "urn:KyotoProtocol:RegistrySystem:ITL:Types:1.0:0.0," the type namespace for all ITL-recognized elements and attributes.

Registries will use operations defined in the ITL namespaces (i.e. ITL for operations and ITL:Types for the data types).

## 5. WSDL Requirements

Web Services Description Language (WSDL) is a specification defining how a Web service is described in a common XML grammar. It represents a contract between the service requestor and service provider which is platform and language independent. The WSDLs defined in Annex K will be made available from the ITL public website.

## 21 Annex K

### Description Language (WSDL) Documentation

#### 1. Introduction

This annex contains the WSDL documents for data exchange for the Kyoto Protocol. The documents listed below specify how registries can communicate with the ITL and vice versa. These documents define the interfaces to the public Web service methods on both the ITL and registries. These Web service methods are the portals through which data can be passed into an application.

There are three WSDL documents listed below: Common, Registry and Transaction Log. The Common WSDL describes the data types and messages used by the Registry and Transaction Log WSDLs. The Registry WSDL describes the operations and the bindings that registries must support. The Registry WSDL imports the types and messages in the Common WSDL. The Transaction Log WSDL describes the operations and the bindings that the ITL will support. The Transaction Log WSDL also imports the types and messages in the Common WSDL.

In order to aid the understanding of non-technical persons, a summary of each operation is provided prior to the WSDL documents. WSDL documents are designed to be "read" by computers and not humans. The Web Service Method Summaries below reduce the numerous cross-references and repetitive nature of the WSDL documents to a more understandable form. The summaries are still detailed and reflect the complex nature of the data exchange system.

## 2. Summaries of Web Service Operations

### 2.1 AcceptMessage

**Figure K1: AcceptMessage**  
Role(s): Registry and ITL

#### MessageRequest

from	string
to	string
majorVersion	int
minorVersion	int
messageContent	string
messageDateTime	dateTime

#### MessageResponse

resultIdentifier	int
responseCodes	<b>ArrayOfInt</b> Opt

### 2.2 AcceptNotification

**Figure K2: AcceptNotification**  
Role(s): Registry and ITL

#### NotificationRequest

from	string												
to	string												
majorVersion	int												
minorVersion	int												
transactionIdentifier	string												
transactionStatus	int												
partyType	int												
evaluationResult	<b>ArrayOfEvaluationResult</b>												
	<table border="1"><tr><td>responseCode</td><td>int</td></tr><tr><td>unitBlockIdentifiers</td><td><b>ArrayOfUnitBlockIdentifier</b> Opt</td></tr><tr><td></td><td><table border="1"><tr><td>unitSerialBlockStart</td><td>long</td></tr><tr><td>unitSerialBlockEnd</td><td>long</td></tr><tr><td>originatingRegistryCode</td><td>string</td></tr></table></td></tr></table>	responseCode	int	unitBlockIdentifiers	<b>ArrayOfUnitBlockIdentifier</b> Opt		<table border="1"><tr><td>unitSerialBlockStart</td><td>long</td></tr><tr><td>unitSerialBlockEnd</td><td>long</td></tr><tr><td>originatingRegistryCode</td><td>string</td></tr></table>	unitSerialBlockStart	long	unitSerialBlockEnd	long	originatingRegistryCode	string
responseCode	int												
unitBlockIdentifiers	<b>ArrayOfUnitBlockIdentifier</b> Opt												
	<table border="1"><tr><td>unitSerialBlockStart</td><td>long</td></tr><tr><td>unitSerialBlockEnd</td><td>long</td></tr><tr><td>originatingRegistryCode</td><td>string</td></tr></table>	unitSerialBlockStart	long	unitSerialBlockEnd	long	originatingRegistryCode	string						
unitSerialBlockStart	long												
unitSerialBlockEnd	long												
originatingRegistryCode	string												

#### NotificationResponse

resultIdentifier	int
responseCodes	<b>ArrayOfInt</b> Opt



## 2.3 AcceptITLNotice

Figure K3: AcceptITLNotice  
Role(s): Registry

### ITLNoticeRequest

from	string	
to	string	
majorVersion	int	
minorVersion	int	
messageContent	string	
messageDate	dateTime	
notificationType	int	
notificationIdentifier	long	
notificationStatus	int	
projectNumber	string	Opt
unitType	int	Opt
targetValue	long	Opt
targetDate	date	Opt
LULUCFActivity	int	Opt
commitPeriod	int	Opt
actionDueDate	date	Opt
unitBlockIdentifiers	<b>ArrayOfUnitBlockIdentifier</b>	
	unitSerialBlockStart	long
	unitSerialBlockEnd	long
	originatingRegistryCode	string
		Opt

### ITLNoticeResponse

resultIdentifier	int	
responseCodes	<b>ArrayOfInt</b>	Opt

## 2.4 AcceptProposal

**Figure K4: AcceptProposal  
Role(s): Registry and ITL**

### ProposalRequest

from	string		
to	string		
majorVersion	int		
minorVersion	int		
proposedTransaction	<b>ProposalTransaction</b>		
	transactionIdentifier	string	
	transactionType	int	
	suppTransactionType	Int	Opt
	transferringRegistryCode	string	
	transferringRegistryAccountType	int	Opt
	transferringRegistryAccountIdentifier	long	Opt
	acquiringRegistryCode	string	
	acquiringRegistryAccountType	int	Opt
	acquiringRegistryAccountIdentifier	long	Opt
	notificationIdentifier	long	Opt
	proposalUnitBlocks	<b>ArrayOfTransactionUnitBlock</b>	
		unitSerialBlockStart	long
		unitSerialBlockEnd	long
		originatingRegistryCode	string
		unitType	int
		suppUnitType	int Opt
		originalCommitPeriod	int
		applicableCommitPeriod	int
		LULUCFActivity	int Opt
		projectIdentifier	int Opt
		track	int Opt
		blockRole	string Opt
		acquiringRegistryAccountType	int Opt
		acquiringRegistryAccountIdentifier	long Opt
		transferringRegistryAccountType	int Opt
		transferringRegistryAccountIdentifier	long Opt
		yearinCommitmentPeriod	int Opt
		installationIdentifier	long Opt
		expiryDate	date Opt

### ProposalResponse

resultIdentifier	int		
responseCodes	<b>ArrayOfInt</b>		Opt

## 2.5 ProvideTime

Figure K5: ProvideTime  
Role(s): Registry

### ProvideTimeRequest

from	string
to	string
majorVersion	int
minorVersion	int
ITLSystemTime	dateTime

### ProvideTimeResponse

systemTime	dateTime
resultIdentifier	int
responseCodes	<b>ArrayOfInt</b> Opt

## 2.6 InitiateReconciliation

Figure K6: InitiateReconciliation  
Role(s): Registry

### InitiateReconciliationRequest

from	string
to	string
majorVersion	int
minorVersion	int
reconciliationIdentifier	string
snapshotDatetime	datetime

### InitiateReconciliationResponse

resultIdentifier	int
responseCodes	<b>ArrayOfInt</b> Opt

## 2.7 ReceiveReconciliationResult

Figure K7: ReceiveReconciliationResult  
Role(s): Registry, ITL

### ReconciliationResultRequest

from	string
to	string
majorVersion	int
minorVersion	int
reconciliationIdentifier	string
reconciliationStatus	int
responseCodes	<b>ArrayOfInt</b> Opt

### ReconciliationResultResponse

resultIdentifier	int
responseCodes	<b>ArrayOfInt</b> Opt

## 2.8 ProvideAuditTrail

**Figure K8: ProvideAuditTrail**  
**Role(s): Registry, ITL**

### ProvideAuditTrailRequest

from	string	
to	string	
majorVersion	int	
minorVersion	int	
reconciliationIdentifier	string	
auditTrailBeginDatetime	dateTime	
auditTrailEndDatetime	dateTime	
accountType	int	Opt
accountIdentifier	long	Opt
accountCommitPeriod	int	Opt
unitType	int	Opt
suppUnitType	int	Opt
unitBlockIdentifiers	<b>ArrayOfReconciliationUnitBlockIdentifier</b>	Opt
	unitSerialBlockStart	long
	unitSerialBlockEnd	long
	originatingRegistryCode	string
responseCodes	<b>ArrayOfInt</b>	Opt

### ProvideAuditTrailResponse

resultIdentifier	int	
responseCodes	<b>ArrayOfInt</b>	Opt

## 2.9 ProvideTotals

**Figure K9: ProvideTotals**  
**Role(s): Registry, ITL**

### ProvideTotalsRequest

from	string	
to	string	
majorVersion	int	
minorVersion	int	
reconciliationIdentifier	string	
reconciliationSnapshotDatetime	dateTime	
reconciliationStatus	int	
unitType	int	Opt
suppUnitType	int	Opt
accountCommitPeriod	int	Opt
accountType	int	Opt
byAccountFlag	int	Opt
responseCodes	<b>ArrayOfInt</b>	Opt

### ProvideTotalsResponse

resultIdentifier	int	
responseCodes	<b>ArrayOfInt</b>	Opt

## 2.10 ProvideUnitBlocks

Figure K10: ProvideUnitBlocks  
Role(s): Registry, ITL

<u>ProvideUnitBlocksRequest</u>			
from	string		
to	string		
majorVersion	int		
minorVersion	int		
reconciliationIdentifier	string		
reconciliationSnapshotDatetime	dateTime		
unitBlockRequests	<b>ArrayOfUnitBlockRequest</b>		
	unitType	int	Opt
	suppUnitType	int	Opt
	accountCommitPeriod	int	Opt
	accountType	int	Opt
	accountIdentifier	long	Opt
responseCodes	<b>ArrayOfInt</b>		Opt
<u>ProvideUnitBlocksResponse</u>			
resultIdentifier	int		
responseCodes	<b>ArrayOfInt</b>		Opt

## 2.11 GetTransactionStatus

Figure K11: GetTransactionStatus  
Role(s): ITL

<u>TransactionStatusRequest</u>			
from	string		
to	string		
majorVersion	int		
minorVersion	int		
transactionIdentifier	string		
<u>TransactionStatusResponse</u>			
transactionIdentifier	string		
transactionStatus	int		
transactionStatusDate	dateTime		
resultIdentifier	int		
responseCodes	<b>ArrayOfInt</b>		Opt

## 2.12 ReceiveTotals

**Figure K12: ReceiveTotals**  
Role(s): ITL

### ReceiveTotalsRequest

from	String		
to	string		
majorVersion	int		
minorVersion	int		
reconciliationIdentifier	string		
totals	<b>ArrayOfTotal</b>		
	accountType	int	
	accountIdentifier	long	Opt
	accountCommitPeriod	int	
	unitType	int	
	suppUnitType	int	Opt
	unitCount	long	

### ReceiveTotalsResponse

resultIdentifier	int		
responseCodes	<b>ArrayOfInt</b>		Opt

## 2.13 ReceiveUnitBlocks

**Figure K13: ReceiveUnitBlocks**  
Role(s): ITL

### ReceiveUnitBlocksRequest

from	string		
to	string		
majorVersion	int		
minorVersion	int		
reconciliationIdentifier	string		
unitBlocks	<b>ArrayOfUnitBlock</b>		
	unitSerialBlockStart	long	
	unitSerialBlockEnd	long	
	originatingRegistryCode	string	
	unitType	int	
	suppUnitType	int	Opt
	accountType	int	
	accountIdentifier	long	Opt
	applicableCommitPeriod	int	

### ReceiveUnitBlocksResponse

resultIdentifier	int		
responseCode	<b>ArrayOfInt</b>		Opt

## 2.14 ReceiveAuditTrail

**Figure K14: ReceiveAuditTrail**  
Role(s): ITL

### ReceiveAuditTrailRequest

from	string		
to	string		
majorVersion	int		
minorVersion	int		
reconciliationIdentifier	string		
transactions	<b>ArrayOfTransaction</b>		
	transactionIdentifier	string	
	transactionType	int	
	suppTransactionType	int	Opt
	transactionStatusDateTime	dateTime	
	transferringRegistryCode	string	
	transferringRegistryAccountType	int	Opt
	transferringRegistryAccountIdentifier	long	Opt
	acquiringRegistryCode	string	
	acquiringRegistryAccountType	int	Opt
	acquiringRegistryAccountIdentifier	long	Opt
	notificationIdentifier	long	Opt
	transactionBlocks	<b>ArrayOfReconciliationTransactionUnitBlock</b>	
		unitSerialBlockStart	long
		unitSerialBlockEnd	long
		originatingRegistryCode	string
		unitType	int
		suppUnitType	int Opt
		originalCommitPeriod	int
		applicableCommitPeriod	int
		LULUCFActivity	int Opt
		projectIdentifier	int Opt
		track	int Opt
		blockRole	string Opt
		transferringRegistryAccountType	int Opt
		transferringRegistryAccountIdentifier	long Opt
		acquiringRegistryAccountType	int Opt
		acquiringRegistryAccountIdentifier	long Opt
		yearInCommitmentPeriod	int Opt
		installationIdentifier	long Opt
		expiryDate	date Opt

### ReceiveAuditTrailResponse

resultIdentifier	int		
responseCodes	<b>ArrayOfInt</b>		Opt

### 3. Registry WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="RegistryService"
targetNamespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"
xmlns:tns="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"
xmlns:typens="urn:KyotoProtocol:RegistrySystem:ITL:Types:1.0:0.0"
xmlns="http://schemas.xmlsoap.org/wsdl/" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/">
  <import namespace="urn:KyotoProtocol:RegistrySystem:ITL:Types:1.0:0.0"
location="common.wsdl"/>
  <portType name="RegistryPort">
    <operation name="provideTime">
      <input message="typens:provideTimeRequest"/>
      <output message="typens:provideTimeResponse"/>
    </operation>
    <operation name="acceptMessage">
      <input message="typens:acceptMessageRequest"/>
      <output message="typens:acceptMessageResponse"/>
    </operation>
    <operation name="acceptProposal">
      <input message="typens:acceptProposalRequest"/>
      <output message="typens:acceptProposalResponse"/>
    </operation>
    <operation name="acceptNotification">
      <input message="typens:acceptNotificationRequest"/>
      <output message="typens:acceptNotificationResponse"/>
    </operation>
    <operation name="acceptITLNotice">
      <input message="typens:acceptITLNoticeRequest"/>
      <output message="typens:acceptITLNoticeResponse"/>
    </operation>
    <operation name="initiateReconciliation">
      <input message="typens:initiateReconciliationRequest"/>
      <output message="typens:initiateReconciliationResponse"/>
    </operation>
    <operation name="receiveReconciliationResult">
      <input message="typens:receiveReconciliationResultRequest"/>
      <output message="typens:receiveReconciliationResultResponse"/>
    </operation>
    <operation name="provideAuditTrail">
      <input message="typens:provideAuditTrailRequest"/>
      <output message="typens:provideAuditTrailResponse"/>
    </operation>
    <operation name="provideTotals">
      <input message="typens:provideTotalsRequest"/>
      <output message="typens:provideTotalsResponse"/>
    </operation>
    <operation name="provideUnitBlocks">
      <input message="typens:provideUnitBlocksRequest"/>
      <output message="typens:provideUnitBlocksResponse"/>
    </operation>
  </portType>
  <binding name="RegistryBinding" type="tns:RegistryPort">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="rpc"/>
    <operation name="provideTime">
      <soap:operation soapAction="provideTime" style="rpc"/>
      <input>
        <soap:body use="encoded"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
      </input>
      <output>
        <soap:body use="encoded"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
      </output>
    </operation>
    <operation name="acceptMessage">
      <soap:operation soapAction="acceptMessage" style="rpc"/>
      <input>
        <soap:body use="encoded"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
      </input>
      <output>
        <soap:body use="encoded"
encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
      </output>
    </operation>
  </binding>
</definitions>
```



```

        </output>
    </operation>
    <operation name="acceptProposal">
        <soap:operation soapAction="acceptProposal" style="rpc"/>
        <input>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </input>
        <output>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </output>
    </operation>
    <operation name="acceptNotification">
        <soap:operation soapAction="acceptNotification" style="rpc"/>
        <input>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </input>
        <output>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </output>
    </operation>
    <operation name="acceptITLNotice">
        <soap:operation soapAction="acceptITLNotice" style="rpc"/>
        <input>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </input>
        <output>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </output>
    </operation>
    <operation name="initiateReconciliation">
        <soap:operation soapAction="initiateReconciliation" style="rpc"/>
        <input>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </input>
        <output>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </output>
    </operation>
    <operation name="receiveReconciliationResult">
        <soap:operation soapAction="receiveReconciliationResult" style="rpc"/>
        <input>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </input>
        <output>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </output>
    </operation>
    <operation name="provideAuditTrail">
        <soap:operation soapAction="provideAuditTrail" style="rpc"/>
        <input>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </input>
        <output>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"

```

```

        namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
    </output>
</operation>
<operation name="provideTotals">
    <soap:operation soapAction="provideTotals" style="rpc"/>
    <input>
        <soap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
    </input>
    <output>
        <soap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
    </output>
</operation>
<operation name="provideUnitBlocks">
    <soap:operation soapAction="provideUnitBlocks" style="rpc"/>
    <input>
        <soap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
    </input>
    <output>
        <soap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
    </output>
</operation>
</binding>
<service name="RegistryService">
    <port name="RegistryPort" binding="tns:RegistryBinding">
        <soap:address location="http://REPLACE.WITH.ACTUAL.URL"/>
    </port>
</service>
</definitions>

```

## 4. Transaction Log WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="TransactionLogService"
  targetNamespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"
  xmlns:tns="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"
  xmlns:typens="urn:KyotoProtocol:RegistrySystem:ITL:Types:1.0:0.0"
  xmlns="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  <import namespace="urn:KyotoProtocol:RegistrySystem:ITL:Types:1.0:0.0"
    location="common.wsdl"/>
  <portType name="TransactionLogPort">
    <operation name="acceptMessage">
      <input message="typens:acceptMessageRequest"/>
      <output message="typens:acceptMessageResponse"/>
    </operation>
    <operation name="acceptProposal">
      <input message="typens:acceptProposalRequest"/>
      <output message="typens:acceptProposalResponse"/>
    </operation>
    <operation name="acceptNotification">
      <input message="typens:acceptNotificationRequest"/>
      <output message="typens:acceptNotificationResponse"/>
    </operation>
    <operation name="getTransactionStatus">
      <input message="typens:getTransactionStatusRequest"/>
      <output message="typens:getTransactionStatusResponse"/>
    </operation>
    <operation name="receiveReconciliationResult">
      <input message="typens:receiveReconciliationResultRequest"/>
      <output message="typens:receiveReconciliationResultResponse"/>
    </operation>
    <operation name="provideAuditTrail">
      <input message="typens:provideAuditTrailRequest"/>
      <output message="typens:provideAuditTrailResponse"/>
    </operation>
    <operation name="provideTotals">
      <input message="typens:provideTotalsRequest"/>
      <output message="typens:provideTotalsResponse"/>
    </operation>
    <operation name="provideUnitBlocks">
      <input message="typens:provideUnitBlocksRequest"/>
      <output message="typens:provideUnitBlocksResponse"/>
    </operation>
    <operation name="receiveAuditTrail">
      <input message="typens:receiveAuditTrailRequest"/>
      <output message="typens:receiveAuditTrailResponse"/>
    </operation>
    <operation name="receiveTotals">
      <input message="typens:receiveTotalsRequest"/>
      <output message="typens:receiveTotalsResponse"/>
    </operation>
    <operation name="receiveUnitBlocks">
      <input message="typens:receiveUnitBlocksRequest"/>
      <output message="typens:receiveUnitBlocksResponse"/>
    </operation>
  </portType>
  <binding name="TransactionLogBinding" type="tns:TransactionLogPort">
    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="rpc"/>
    <operation name="acceptMessage">
      <soap:operation soapAction="acceptMessage" style="rpc"/>
      <input>
        <soap:body use="encoded"
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
      </input>
      <output>
        <soap:body use="encoded"
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
      </output>
    </operation>
    <operation name="acceptProposal">
      <soap:operation soapAction="acceptProposal" style="rpc"/>
      <input>
        <soap:body use="encoded"
          encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
          namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
      </input>
```

```

        </input>
      </output>
      <soap:body use="encoded"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
    </output>
  </operation>
  <operation name="acceptNotification">
    <soap:operation soapAction="acceptNotification" style="rpc"/>
    <input>
      <soap:body use="encoded"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
    </input>
    <output>
      <soap:body use="encoded"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
    </output>
  </operation>
  <operation name="getTransactionStatus">
    <soap:operation soapAction="getTransactionStatus" style="rpc"/>
    <input>
      <soap:body use="encoded"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
    </input>
    <output>
      <soap:body use="encoded"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
    </output>
  </operation>
  <operation name="receiveReconciliationResult">
    <soap:operation soapAction="receiveReconciliationResult" style="rpc"/>
    <input>
      <soap:body use="encoded"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
    </input>
    <output>
      <soap:body use="encoded"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
    </output>
  </operation>
  <operation name="provideAuditTrail">
    <soap:operation soapAction="provideAuditTrail" style="rpc"/>
    <input>
      <soap:body use="encoded"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
    </input>
    <output>
      <soap:body use="encoded"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
    </output>
  </operation>
  <operation name="provideTotals">
    <soap:operation soapAction="provideTotals" style="rpc"/>
    <input>
      <soap:body use="encoded"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
    </input>
    <output>
      <soap:body use="encoded"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
    </output>
  </operation>
  <operation name="provideUnitBlocks">
    <soap:operation soapAction="provideUnitBlocks" style="rpc"/>
    <input>
      <soap:body use="encoded"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
    </input>

```

```

        </input>
        <output>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </output>
    </operation>
    <operation name="receiveAuditTrail">
        <soap:operation soapAction="receiveAuditTrail" style="rpc"/>
        <input>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </input>
        <output>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </output>
    </operation>
    <operation name="receiveTotals">
        <soap:operation soapAction="receiveTotals" style="rpc"/>
        <input>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </input>
        <output>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </output>
    </operation>
    <operation name="receiveUnitBlocks">
        <soap:operation soapAction="receiveUnitBlocks" style="rpc"/>
        <input>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </input>
        <output>
            <soap:body use="encoded"
                encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
                namespace="urn:KyotoProtocol:RegistrySystem:ITL:1.0:0.0"/>
        </output>
    </operation>
</binding>
<service name="TransactionLogService">
    <port name="TransactionLogPort" binding="tns:TransactionLogBinding">
        <soap:address location="http://REPLACE.WITH.ACTUAL.URL"/>
    </port>
</service>
</definitions>

```

## 5. Common WSDL

```
<?xml version="1.0" encoding="UTF-8"?>
<definitions name="RegistryService" xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
  xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
  xmlns:typens="urn:KyotoProtocol:RegistrySystem:ITL:Types:1.0:0.0"
  targetNamespace="urn:KyotoProtocol:RegistrySystem:ITL:Types:1.0:0.0"
  xmlns="http://schemas.xmlsoap.org/wsdl/">
  <types>
    <schema targetNamespace="urn:KyotoProtocol:RegistrySystem:ITL:Types:1.0:0.0"
      xmlns="http://www.w3.org/2001/XMLSchema"
      <import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
      <complexType name="ArrayOfInt">
        <complexContent>
          <restriction base="soapenc:Array">
            <attribute ref="soapenc:arrayType"
              wsdl:arrayType="int[]" />
          </restriction>
        </complexContent>
      </complexType>
      <complexType name="Transaction">
        <sequence>
          <element name="transactionIdentifier" type="string" />
          <element name="transactionType" type="int" />
          <element name="suppTransactionType" type="int"
            minOccurs="0" />
          <element name="transactionStatusDateTime"
            type="dateTime" />
          <element name="transferringRegistryCode" type="string" />
          <element name="transferringRegistryAccountType"
            type="int"
            minOccurs="0" />
          <element name="transferringRegistryAccountIdentifier"
            type="long" minOccurs="0" />
          <element name="acquiringRegistryCode" type="string" />
          <element name="acquiringRegistryAccountType" type="int"
            minOccurs="0" />
          <element name="acquiringRegistryAccountIdentifier"
            type="long"
            minOccurs="0" />
          <element name="notificationIdentifier" type="long"
            minOccurs="0" />
          <element name="transactionBlocks"
            type="typens:ArrayOfReconciliationTransactionUnitBlock" />
        </sequence>
      </complexType>
      <complexType name="ArrayOfTransaction">
        <complexContent>
          <restriction base="soapenc:Array">
            <attribute ref="soapenc:arrayType"
              wsdl:arrayType="typens:Transaction[]" />
          </restriction>
        </complexContent>
      </complexType>
      <complexType name="ProposalTransaction">
        <sequence>
          <element name="transactionIdentifier" type="string" />
          <element name="transactionType" type="int" />
          <element name="suppTransactionType" type="int"
            minOccurs="0" />
          <element name="transferringRegistryCode" type="string" />
          <element name="transferringRegistryAccountType"
            type="int"
            minOccurs="0" />
          <element name="transferringRegistryAccountIdentifier"
            type="long" minOccurs="0" />
          <element name="acquiringRegistryCode" type="string" />
          <element name="acquiringRegistryAccountType" type="int"
            minOccurs="0" />
          <element name="acquiringRegistryAccountIdentifier"
            type="long"
            minOccurs="0" />
          <element name="notificationIdentifier" type="long"
            minOccurs="0" />
          <element name="proposalUnitBlocks"
            type="typens:ArrayOfTransactionUnitBlock" />
        </sequence>
    </types>
  </definitions>
```

```

</complexType>
<complexType name="ReconciliationUnitBlockIdentifier">
  <sequence>
    <element name="unitSerialBlockStart" type="long"/>
    <element name="unitSerialBlockEnd" type="long"/>
    <element name="originatingRegistryCode" type="string"/>
  </sequence>
</complexType>
<complexType name="ArrayOfReconciliationUnitBlockIdentifier">
  <complexContent>
    <restriction base="soapenc:Array">
      <attribute ref="soapenc:arrayType"
wsdl:arrayType="typens:ReconciliationUnitBlockIdentifier[]" />
    </restriction>
  </complexContent>
</complexType>
<complexType name="UnitBlockIdentifier">
  <sequence>
    <element name="unitSerialBlockStart" type="long"/>
    <element name="unitSerialBlockEnd" type="long"/>
    <element name="originatingRegistryCode" type="string"/>
  </sequence>
</complexType>
<complexType name="ArrayOfUnitBlockIdentifier">
  <complexContent>
    <restriction base="soapenc:Array">
      <attribute ref="soapenc:arrayType"
wsdl:arrayType="typens:UnitBlockIdentifier
[]" />
    </restriction>
  </complexContent>
</complexType>
<complexType name="UnitBlock">
  <sequence>
    <element name="unitSerialBlockStart" type="long"/>
    <element name="unitSerialBlockEnd" type="long"/>
    <element name="originatingRegistryCode" type="string"/>
    <element name="unitType" type="int"/>
    <element name="suppUnitType" type="int" minOccurs="0"/>
    <element name="accountType" type="int"/>
    <element name="accountIdentifier" type="long"
minOccurs="0"/>
    <element name="applicableCommitPeriod" type="int"/>
  </sequence>
</complexType>
<complexType name="ArrayOfUnitBlock">
  <complexContent>
    <restriction base="soapenc:Array">
      <attribute ref="soapenc:arrayType"
wsdl:arrayType="typens:UnitBlock[]" />
    </restriction>
  </complexContent>
</complexType>
<complexType name="ReconciliationTransactionUnitBlock">
  <sequence>
    <element name="unitSerialBlockStart" type="long"/>
    <element name="unitSerialBlockEnd" type="long"/>
    <element name="originatingRegistryCode" type="string"/>
    <element name="unitType" type="int"/>
    <element name="suppUnitType" type="int" minOccurs="0"/>
    <element name="originalCommitPeriod" type="int"/>
    <element name="applicableCommitPeriod" type="int"/>
    <element name="LULUCFActivity" type="int" minOccurs="0"/>
    <element name="projectIdentifier" type="int"
minOccurs="0"/>
    <element name="track" type="int" minOccurs="0"/>
    <element name="blockRole" type="string" minOccurs="0"/>
    <element name="transferringRegistryAccountType"
type="int"
minOccurs="0"/>
    <element name="transferringRegistryAccountIdentifier"
type="long" minOccurs="0"/>
    <element name="acquiringRegistryAccountType" type="int"
minOccurs="0"/>
    <element name="acquiringRegistryAccountIdentifier"
type="long"
minOccurs="0"/>
  </sequence>

```

```

        <element name="yearInCommitmentPeriod" type="int"
            minOccurs="0"/>
        <element name="installationIdentifier" type="long"
            minOccurs="0"/>
        <element name="expiryDate" type="date" minOccurs="0"/>
    </sequence>
</complexType>
<complexType name="ArrayOfReconciliationTransactionUnitBlock">
    <complexContent>
        <restriction base="soapenc:Array">
            <attribute ref="soapenc:arrayType"
                wsdl:arrayType=
                    "typens:ReconciliationTransactionUnitBlock[]"/>
        </restriction>
    </complexContent>
</complexType>
<complexType name="TransactionUnitBlock">
    <sequence>
        <element name="unitSerialBlockStart" type="long"/>
        <element name="unitSerialBlockEnd" type="long"/>
        <element name="originatingRegistryCode" type="string"/>
        <element name="unitType" type="int"/>
        <element name="suppUnitType" type="int" minOccurs="0"/>
        <element name="originalCommitPeriod" type="int"/>
        <element name="applicableCommitPeriod" type="int"/>
        <element name="LULUCFActivity" type="int" minOccurs="0"/>
        <element name="projectIdentifier" type="int"
            minOccurs="0"/>
        <element name="track" type="int" minOccurs="0"/>
        <element name="blockRole" type="string" minOccurs="0"/>
        <element name="transferringRegistryAccountType"
            type="long" minOccurs="0"/>
        <element name="transferringRegistryAccountIdentifier"
            type="long" minOccurs="0"/>
        <element name="acquiringRegistryAccountType" type="int"
            minOccurs="0"/>
        <element name="acquiringRegistryAccountIdentifier"
            type="long" minOccurs="0"/>
        <element name="yearInCommitmentPeriod" type="int"
            minOccurs="0"/>
        <element name="installationIdentifier" type="long"
            minOccurs="0"/>
        <element name="expiryDate" type="date" minOccurs="0"/>
    </sequence>
</complexType>
<complexType name="ArrayOfTransactionUnitBlock">
    <complexContent>
        <restriction base="soapenc:Array">
            <attribute ref="soapenc:arrayType"
                wsdl:arrayType="typens:TransactionUnitBlock[]"/>
        </restriction>
    </complexContent>
</complexType>
<complexType name="Total">
    <sequence>
        <element name="accountType" type="int"/>
        <element name="accountIdentifier" type="long"
            minOccurs="0"/>
        <element name="accountCommitPeriod" type="int"/>
        <element name="unitType" type="int"/>
        <element name="suppUnitType" type="int" minOccurs="0"/>
        <element name="unitCount" type="long"/>
    </sequence>
</complexType>
<complexType name="ArrayOfTotal">
    <complexContent>
        <restriction base="soapenc:Array">
            <attribute ref="soapenc:arrayType"
                wsdl:arrayType="typens:Total[]"/>
        </restriction>
    </complexContent>
</complexType>
<complexType name="UnitBlockRequest">
    <sequence>
        <element name="unitType" type="int" minOccurs="0"/>
        <element name="suppUnitType" type="int" minOccurs="0"/>
    </sequence>
</complexType>

```



```

minOccurs="0"/>
        <element name="accountCommitPeriod" type="int"
        <element name="accountType" type="int" minOccurs="0"/>
        <element name="accountIdentifier" type="long"
minOccurs="0"/>
    </sequence>
</complexType>
<complexType name="ArrayOfUnitBlockRequest">
    <complexContent>
        <restriction base="soapenc:Array">
            <attribute ref="soapenc:arrayType"
                wsdl:arrayType="typens:UnitBlockRequest[]" />
        </restriction>
    </complexContent>
</complexType>
<complexType name="EvaluationResult">
    <sequence>
        <element name="responseCode" type="int"/>
        <element name="unitBlockIdentifiers"
            type="typens:ArrayOfUnitBlockIdentifier"
minOccurs="0"/>
    </sequence>
</complexType>
<complexType name="ArrayOfEvaluationResult">
    <complexContent>
        <restriction base="soapenc:Array">
            <attribute ref="soapenc:arrayType"
                wsdl:arrayType="typens:EvaluationResult[]" />
        </restriction>
    </complexContent>
</complexType>
<complexType name="ProvideTimeRequest">
    <sequence>
        <element name="from" type="string"/>
        <element name="to" type="string"/>
        <element name="majorVersion" type="int"/>
        <element name="minorVersion" type="int"/>
        <element name="ITLSystemTime" type="dateTime"/>
    </sequence>
</complexType>
<complexType name="ProvideTimeResponse">
    <sequence>
        <element name="systemTime" type="dateTime"/>
        <element name="resultIdentifier" type="int"/>
        <element name="responseCodes" type="typens:ArrayOfInt"
minOccurs="0"/>
    </sequence>
</complexType>
<complexType name="MessageRequest">
    <sequence>
        <element name="from" type="string"/>
        <element name="to" type="string"/>
        <element name="majorVersion" type="int"/>
        <element name="minorVersion" type="int"/>
        <element name="messageContent" type="string"/>
        <element name="messageDateTime" type="dateTime"/>
    </sequence>
</complexType>
<complexType name="MessageResponse">
    <sequence>
        <element name="resultIdentifier" type="int"/>
        <element name="responseCodes" type="typens:ArrayOfInt"
minOccurs="0"/>
    </sequence>
</complexType>
<complexType name="ProposalRequest">
    <sequence>
        <element name="from" type="string"/>
        <element name="to" type="string"/>
        <element name="majorVersion" type="int"/>
        <element name="minorVersion" type="int"/>
        <element name="proposedTransaction"
            type="typens:ProposalTransaction"/>
    </sequence>
</complexType>
<complexType name="ProposalResponse">

```

```

        <sequence>
            <element name="resultIdentifier" type="int"/>
            <element name="responseCodes" type="typens:ArrayOfInt"
                minOccurs="0"/>
        </sequence>
    </complexType>
    <complexType name="NotificationRequest">
        <sequence>
            <element name="from" type="string"/>
            <element name="to" type="string"/>
            <element name="majorVersion" type="int"/>
            <element name="minorVersion" type="int"/>
            <element name="transactionIdentifier" type="string"/>
            <element name="transactionStatus" type="int"/>
            <element name="partyType" type="int"/>
            <element name="evaluationResult"
                type="typens:ArrayOfEvaluationResult"/>
        </sequence>
    </complexType>
    <complexType name="NotificationResponse">
        <sequence>
            <element name="resultIdentifier" type="int"/>
            <element name="responseCodes" type="typens:ArrayOfInt"
                minOccurs="0"/>
        </sequence>
    </complexType>
    <complexType name="ITLNoticeRequest">
        <sequence>
            <element name="from" type="string"/>
            <element name="to" type="string"/>
            <element name="majorVersion" type="int"/>
            <element name="minorVersion" type="int"/>
            <element name="messageContent" type="string"/>
            <element name="messageDate" type="dateTime"/>
            <element name="notificationType" type="int"/>
            <element name="notificationIdentifier" type="long"/>
            <element name="notificationStatus" type="int"/>
            <element name="projectNumber" type="string"
                minOccurs="0"/>
            <element name="unitType" type="int" minOccurs="0"/>
            <element name="targetValue" type="long" minOccurs="0"/>
            <element name="targetDate" type="date" minOccurs="0"/>
            <element name="LULUCFActivity" type="long"
                minOccurs="0"/>
            <element name="commitPeriod" type="int" minOccurs="0"/>
            <element name="actionDueDate" type="date" minOccurs="0"/>
            <element name="unitBlockIdentifiers"
                type="typens:ArrayOfUnitBlockIdentifier"
                minOccurs="0"/>
        </sequence>
    </complexType>
    <complexType name="ITLNoticeResponse">
        <sequence>
            <element name="resultIdentifier" type="int"/>
            <element name="responseCodes" type="typens:ArrayOfInt"
                minOccurs="0"/>
        </sequence>
    </complexType>
    <complexType name="ReconciliationResultRequest">
        <sequence>
            <element name="from" type="string"/>
            <element name="to" type="string"/>
            <element name="majorVersion" type="int"/>
            <element name="minorVersion" type="int"/>
            <element name="reconciliationIdentifier" type="string"/>
            <element name="reconciliationStatus" type="int"/>
            <element name="responseCodes" type="typens:ArrayOfInt"
                minOccurs="0"/>
        </sequence>
    </complexType>
    <complexType name="ReconciliationResultResponse">
        <sequence>
            <element name="resultIdentifier" type="int"/>
            <element name="responseCodes" type="typens:ArrayOfInt"
                minOccurs="0"/>
        </sequence>
    </complexType>
    <complexType name="ProvideTotalsRequest">

```

```

        <sequence>
            <element name="from" type="string"/>
            <element name="to" type="string"/>
            <element name="majorVersion" type="int"/>
            <element name="minorVersion" type="int"/>
            <element name="reconciliationIdentifier" type="string"/>
            <element name="reconciliationSnapshotDatetime"
                type="dateTime"/>
            <element name="reconciliationStatus" type="int"/>
            <element name="unitType" type="int" minOccurs="0"/>
            <element name="suppUnitType" type="int" minOccurs="0"/>
            <element name="accountCommitPeriod" type="int"
                minOccurs="0"/>
            <element name="accountType" type="int" minOccurs="0"/>
            <element name="byAccountFlag" type="int" minOccurs="0"/>
            <element name="responseCodes" type="typens:ArrayOfInt"
                minOccurs="0"/>
        </sequence>
    </complexType>
    <complexType name="ProvideTotalsResponse">
        <sequence>
            <element name="resultIdentifier" type="int"/>
            <element name="responseCodes" type="typens:ArrayOfInt"
                minOccurs="0"/>
        </sequence>
    </complexType>
    <complexType name="ProvideUnitBlocksRequest">
        <sequence>
            <element name="from" type="string"/>
            <element name="to" type="string"/>
            <element name="majorVersion" type="int"/>
            <element name="minorVersion" type="int"/>
            <element name="reconciliationIdentifier" type="string"/>
            <element name="reconciliationSnapshotDatetime"
                type="dateTime"/>
            <element name="unitBlockRequests"
                type="typens:ArrayOfUnitBlockRequest"/>
            <element name="responseCodes" type="typens:ArrayOfInt"
                minOccurs="0"/>
        </sequence>
    </complexType>
    <complexType name="ProvideUnitBlocksResponse">
        <sequence>
            <element name="resultIdentifier" type="int"/>
            <element name="responseCodes" type="typens:ArrayOfInt"
                minOccurs="0"/>
        </sequence>
    </complexType>
    <complexType name="ProvideAuditTrailRequest">
        <sequence>
            <element name="from" type="string"/>
            <element name="to" type="string"/>
            <element name="majorVersion" type="int"/>
            <element name="minorVersion" type="int"/>
            <element name="reconciliationIdentifier" type="string"/>
            <element name="auditTrailBeginDatetime" type="dateTime"/>
            <element name="auditTrailEndDatetime" type="dateTime"/>
            <element name="accountType" type="int" minOccurs="0"/>
            <element name="accountIdentifier" type="long"
                minOccurs="0"/>
            <element name="accountCommitPeriod" type="int"
                minOccurs="0"/>
            <element name="unitType" type="int" minOccurs="0"/>
            <element name="suppUnitType" type="int" minOccurs="0"/>
            <element name="unitBlockIdentifiers"
                type="typens:ArrayOfReconciliationUnitBlockIdentifier"
                minOccurs="0"/>
            <element name="responseCodes" type="typens:ArrayOfInt"
                minOccurs="0"/>
        </sequence>
    </complexType>
    <complexType name="ProvideAuditTrailResponse">
        <sequence>
            <element name="resultIdentifier" type="int"/>
            <element name="responseCodes" type="typens:ArrayOfInt"
                minOccurs="0"/>
        </sequence>
    </complexType>

```

```

</complexType>
<complexType name="ReceiveTotalsRequest">
  <sequence>
    <element name="from" type="string"/>
    <element name="to" type="string"/>
    <element name="majorVersion" type="int"/>
    <element name="minorVersion" type="int"/>
    <element name="reconciliationIdentifier" type="string"/>
    <element name="totals" type="typens:ArrayOfTotal"/>
  </sequence>
</complexType>
<complexType name="ReceiveTotalsResponse">
  <sequence>
    <element name="resultIdentifier" type="int"/>
    <element name="responseCodes" type="typens:ArrayOfInt"
      minOccurs="0"/>
  </sequence>
</complexType>
<complexType name="TransactionStatusRequest">
  <sequence>
    <element name="from" type="string"/>
    <element name="to" type="string"/>
    <element name="majorVersion" type="int"/>
    <element name="minorVersion" type="int"/>
    <element name="transactionIdentifier" type="string"/>
  </sequence>
</complexType>
<complexType name="TransactionStatusResponse">
  <sequence>
    <element name="transactionIdentifier" type="string"/>
    <element name="transactionStatus" type="int"/>
    <element name="transactionStatusDateTime"
      type="dateTime"/>
    <element name="resultIdentifier" type="int"/>
    <element name="responseCodes" type="typens:ArrayOfInt"
      minOccurs="0"/>
  </sequence>
</complexType>
<complexType name="ReceiveAuditTrailRequest">
  <sequence>
    <element name="from" type="string"/>
    <element name="to" type="string"/>
    <element name="majorVersion" type="int"/>
    <element name="minorVersion" type="int"/>
    <element name="reconciliationIdentifier" type="string"/>
    <element name="transactions"
      type="typens:ArrayOfTransaction"/>
  </sequence>
</complexType>
<complexType name="ReceiveAuditTrailResponse">
  <sequence>
    <element name="resultIdentifier" type="int"/>
    <element name="responseCodes" type="typens:ArrayOfInt"
      minOccurs="0"/>
  </sequence>
</complexType>
<complexType name="ReceiveUnitBlocksRequest">
  <sequence>
    <element name="from" type="string"/>
    <element name="to" type="string"/>
    <element name="majorVersion" type="int"/>
    <element name="minorVersion" type="int"/>
    <element name="reconciliationIdentifier" type="string"/>
    <element name="unitBlocks"
      type="typens:ArrayOfUnitBlock"/>
  </sequence>
</complexType>
<complexType name="ReceiveUnitBlocksResponse">
  <sequence>
    <element name="resultIdentifier" type="int"/>
    <element name="responseCodes" type="typens:ArrayOfInt"
      minOccurs="0"/>
  </sequence>
</complexType>
<complexType name="InitiateReconciliationRequest">
  <sequence>
    <element name="from" type="string"/>
    <element name="to" type="string"/>
  </sequence>
</complexType>

```

```

        <element name="majorVersion" type="int"/>
        <element name="minorVersion" type="int"/>
        <element name="reconciliationIdentifier" type="string"/>
        <element name="snapshotDatettime" type="dateTime"/>
    </sequence>
</complexType>
<complexType name="InitiateReconciliationResponse">
    <sequence>
        <element name="resultIdentifier" type="int"/>
        <element name="responseCodes" type="typens:ArrayOfInt"
            minOccurs="0"/>
    </sequence>
</complexType>
</schema>
</types>
<message name="provideTimeRequest">
    <part name="provideTimeRequest" type="typens:ProvideTimeRequest"/>
</message>
<message name="provideTimeResponse">
    <part name="provideTimeResponse" type="typens:ProvideTimeResponse"/>
</message>
<message name="acceptMessageRequest">
    <part name="acceptMessageRequest" type="typens:MessageRequest"/>
</message>
<message name="acceptMessageResponse">
    <part name="acceptMessageResponse" type="typens:MessageResponse"/>
</message>
<message name="acceptProposalRequest">
    <part name="acceptProposalRequest" type="typens:ProposalRequest"/>
</message>
<message name="acceptProposalResponse">
    <part name="acceptProposalResponse" type="typens:ProposalResponse"/>
</message>
<message name="acceptNotificationRequest">
    <part name="acceptNotificationRequest" type="typens:NotificationRequest"/>
</message>
<message name="acceptNotificationResponse">
    <part name="acceptNotificationResponse" type="typens:NotificationResponse"/>
</message>
<message name="acceptITLNoticeRequest">
    <part name="acceptITLNoticeRequest" type="typens:ITLNoticeRequest"/>
</message>
<message name="acceptITLNoticeResponse">
    <part name="acceptITLNoticeResponse" type="typens:ITLNoticeResponse"/>
</message>
<message name="receiveReconciliationResultRequest">
    <part name="receiveReconciliationResultRequest"
        type="typens:ReconciliationResultRequest"/>
</message>
<message name="receiveReconciliationResultResponse">
    <part name="receiveReconciliationResultResponse"
        type="typens:ReconciliationResultResponse"/>
</message>
<message name="provideAuditTrailRequest">
    <part name="provideAuditTrailRequest" type="typens:ProvideAuditTrailRequest"/>
</message>
<message name="provideAuditTrailResponse">
    <part name="provideAuditTrailResponse"
type="typens:ProvideAuditTrailResponse"/>
</message>
<message name="provideTotalsRequest">
    <part name="provideTotalsRequest" type="typens:ProvideTotalsRequest"/>
</message>
<message name="provideTotalsResponse">
    <part name="provideTotalsResponse" type="typens:ProvideTotalsResponse"/>
</message>
<message name="provideUnitBlocksRequest">
    <part name="provideUnitBlocksRequest" type="typens:ProvideUnitBlocksRequest"/>
</message>
<message name="provideUnitBlocksResponse">
    <part name="provideUnitBlocksResponse"
type="typens:ProvideUnitBlocksResponse"/>
</message>
<message name="getTransactionStatusRequest">
    <part name="getTransactionStatusRequest"
type="typens:TransactionStatusRequest"/>
</message>
<message name="getTransactionStatusResponse">

```

```

        <part name="getTransactionStatusResponse"
type="typens:TransactionStatusResponse"/>
    </message>
    <message name="receiveAuditTrailRequest">
        <part name="receiveAuditTrailRequest" type="typens:ReceiveAuditTrailRequest"/>
    </message>
    <message name="receiveAuditTrailResponse">
        <part name="receiveAuditTrailResponse"
type="typens:ReceiveAuditTrailResponse"/>
    </message>
    <message name="receiveTotalsRequest">
        <part name="receiveTotalsRequest" type="typens:ReceiveTotalsRequest"/>
    </message>
    <message name="receiveTotalsResponse">
        <part name="receiveTotalsResponse" type="typens:ReceiveTotalsResponse"/>
    </message>
    <message name="receiveUnitBlocksRequest">
        <part name="receiveUnitBlocksRequest" type="typens:ReceiveUnitBlocksRequest"/>
    </message>
    <message name="receiveUnitBlocksResponse">
        <part name="receiveUnitBlocksResponse"
type="typens:ReceiveUnitBlocksResponse"/>
    </message>
    <message name="initiateReconciliationRequest">
        <part name="initiateReconciliationRequest"
type="typens:InitiateReconciliationRequest"/>
    </message>
    <message name="initiateReconciliationResponse">
        <part name="initiateReconciliationResponse"
type="typens:InitiateReconciliationResponse"/>
    </message>
</definitions>

```

### WSDL Examples and Instructions

#### 1. Using the Web Services and WSDLs for Specific Transactions

This section provides additional information on how data must be provided for each transaction type. It defines the guiding principles, transaction-specific rules or guidelines and examples, both for the Kyoto Protocol and the European Union (EU) emissions trading scheme.

##### 1.1 Principles

The following principles have been agreed upon to guide the rules relating to submission of data via the Web services required in this document.

- Transaction messages will contain non duplicative data, wherever possible. Redundant data are discouraged because it creates the potential for conflicting data and the need for additional checks. Non-essential data are not expected, except where these data may improve the readability of the XML message or provide additional information to clarify a process.
- For transaction types which are designed to modify the characteristics of a unit block (such as conversion, expiry date change, or the removal or addition of a supplementary unit type code), the incoming data will contain the values proposed to be changed, not the existing values.
- Wherever possible, the content of both the Kyoto and Supplementary Program transactions will adhere to the same rules and adopt similar conventions for receipt, checking and handling potential conflicts.

##### 1.2 Transaction Rules

The following "rules" have been applied to the examples below and should be used to guide the development of requirements for other Kyoto transactions or transactions needed for a supplementary transaction log. Information about transactions for the EU Community Independent Transaction Log (CITL) is also provided since the CITL will be operating concurrently with the ITL.

- If the same data are provided at both the transaction level and the unit block level, the data at the transaction level will be presumed to be correct and the unit block level data will be ignored.
- Transactions may contain multiple "transfers" between accounts or "updates" as long as the unit blocks involved are for the same Commitment Period (with the exception for EU transaction 04-03 which retires allowances for Commitment Period 0 and cancels CERs for Commitment Period 1, and transaction 10-41 which cancels unit blocks for one Commitment Period and replaces them with unit blocks for the next Commitment Period). Extremely complex or large transactions result in a higher risk that an error will be identified and the entire transaction rejected.
- In general, the Kyoto Protocol does not require Account Identifiers to be specified. However, when the account type is cancellation, replacement, or retirement, the Account Identifier must be provided. For EU purposes, all transactions involving movement of an allowance from one account to another requires the Account Identifier.
- Figure L1 identifies by unit type the data for unit characteristics expected under the Kyoto Protocol. Figure L2 describes the circumstances for which data are provided for other parameters specific to transaction types. Figure L3 summarizes by

transaction type how Registry, Account Identifiers and account types should be provided.

**Figure L1: Unit Type Specific Parameters**

Parameter	Unit Type						
	AAU (1)	RMU (2)	ERU from AAU (3)	ERU from RMU (4)	CER (5)	tCER (6)	ICER (7)
LULUCF Activity	Null	Provide value	Null	Provide value	Null	Provide value	Provide value
Project Identifier	Null	Null	Provide value	Provide value	Provide value	Provide value	Provide value
Track	Null	Null	Provide value	Provide value	Null	Null	Null
Expiry Date	Null	Null	Null	Null	Null	Provide value	Provide value

**Figure L2: Transaction Type Specific Parameters**

Parameter	Applicable Transaction Types
Notification Identifier	Only used in replacement transactions if the transaction is prompted by non-submission of certification report, by reversal of storage, or cancellation transactions for excess issuance, net source cancellation transactions, or impending tCER or ICER expiry. Used in all carry-over transactions and all expiry date change transactions.
Block Role	Only used during replacement (transaction type 6). Value is "REP" for the blocks that are being replaced.
Year in commitment period	Only used in EU transactions during Issue of Force Majeure Allowances (transaction type 01-54), Surrender of Allowances (transaction type 10-02) and Allocation of Allowances (transaction type 10-53).
Installation Identifier	Only used in EU transactions during Issue of Force Majeure Allowances (transaction type 01-54), Allocation of Allowances (transaction type 10-53).



**Figure L3: Transaction and Block Information for Transferring and Acquiring Parties**

Transaction Type	Transferring Party	Acquiring Party
Issuance	Registry Code	Registry Code Account Type Account ID (EU only)
Conversion	Registry Code Account Type	Registry Code
External Transfers	Registry Code Account Type Account ID	Registry Code Account Type Account ID
Cancellation, Replacement, and Retirement	Registry Code Account Type Account ID (EU only)	Registry Code Account Type Account ID
Carry-over	Registry Code Account Type	Registry Code
Expiry Date Change	Registry Code Account Type	Registry Code
Internal (EU only)	Registry Code Account Type Account ID	Registry Code Account Type Account ID

For EU transactions for which the Account Type and Account ID are required, Account Type and Account ID must be included at either the transaction level or unit block level, but not both levels.

### 1.3 Transaction Element Descriptions

The following elements will be populated for all transactions and will conform to these standards:

- From - the code of the registry or entity sending the XML request. For messages received by registries, this value will always be the ITL. The codes used are the two-letter country codes in ISO3166, as of 01 January, 2005 using EU for the European Community Registry, CDM for the CDM Registry, ITL for the ITL, and CTL for the CITL.
- To - the code of the registry receiving or entity an XML request. This may not be the same registry as the acquiring registry for the transaction. For messages sent by registries, this value will always be the ITL. The codes used are the two-letter country codes in ISO3166, as of 01 January, 2005 using EU for the European Community Registry, CDM for the CDM Registry, ITL for the ITL, and CTL for the CITL.
- Major Version Number - Major Version number of the DES. This must be the current version or the message will be rejected
- Minor Version Number - Minor Version number of the DES. If this number does not equal the latest version, a warning will be generated, but the XML message will be accepted.
- Transaction Identifier - Unique transaction identifier generated by the registry initiating the transaction (transferring registry).
- Transaction Type - integer represented the type of transaction as specified in the DES.

- Transferring Registry Code - required for all transactions.
- Acquiring Registry Code - required for all transactions.

The use of the following elements for Kyoto Protocol transactions depends on the transaction type.

- Transferring Account Type - required for all transactions except Issuance.
- Transferring Account Identifier - required for External Transfers.
- Acquiring Account Type - required for Issuance, External Transfers, Cancellation, Retirement, and Replacement to designate the account type receiving the units.
- Acquiring Account Identifier - required for External Transfers. Also required for other Kyoto transactions whenever the acquiring account type is provided and is either cancellation, replacement, or retirement and not required for conversion, carry-over and expiry date change transactions.
- Notification Identifier - required for Replacement, Cancellation, Carry-over, and Expiry Date Change transactions that are in response to a notification sent by the ITL as defined in Section 6. This includes transactions for reversal of storage at a project site, the non submission of a certification report for a project, cancellation required in response to excess issuance of CERs, tCERs and ICERs, all carry-over transactions, and replacements initiated in response to impending tCER or ICER expiry. This value must be the Notification ID sent by the ITL in the original notification to which this transaction is responding.

Note: The Supplemental Transaction Type is never provided in Kyoto-only transactions.

## 1.4 Transaction Unit Block Element Descriptions

There is no restriction on the number of units blocks which may be included in a transaction. However, all transaction proposals must contain at least one unit block. Multiple blocks may be issued in a single transaction by submitting an array of unit blocks. Where multiple blocks are included they must be for the same Commitment Period.

For Kyoto transactions the following fields are always required:

- Serial block start;
- Serial block end;
- Originating registry code;
- Unit type;
- Original Commitment Period; and
- Applicable Commitment Period.

The use of the following fields depends on the unit characteristics and transaction type:

- LULUCF Activity Code - required for RMUs, tCERs, and ICERs in all transaction types.
- Project Identifier - required for ERUs, CERs, tCERs, and ICERs in all transaction types.
- Track - required for ERUs.

- Block Role - needed in Replacement transactions to distinguish between the replacing blocks and the block to be replaced. The block to be replaced will have a value of "REP" in this field. The replacing block value will be null.
- Expiry Date - required for tCERs and ICERs in all transaction types.

The following fields are never provided in the Transaction Unit Block for Kyoto-only transactions:

- Supplemental Unit Type;
- Transferring Registry Account Type;
- Transferring Registry Account Identifier;
- Acquiring Registry Account Type;
- Acquiring Registry Account Identifier;
- Year in Commitment Period; and
- Installation Identifier.

## 1.5 General Notes about each Kyoto Protocol Transaction Type

This section summarizes the rules articulated above for each transaction type:

- Issuance - Transaction Type 1. In a proposed Issuance transaction, the registry that is the proposing registry is identified in the transferring registry code data element. The registry and account type that receives the issuance of units must be listed in the acquiring registry and the acquiring account type fields. The acquiring Account Identifier must be left null for Kyoto-only transactions because for Kyoto transactions the account identifier is only used for cancellation, replacement, and retirement accounts and it is not possible to issue into those account types. The transferring account type, transferring Account Identifier and Notification ID must be null for all issuance transactions. The unit blocks will be completed as appropriate for their characteristics.
- Conversion - Transaction Type 2. The transferring registry code and transferring account type identify the holder of the units for conversion. The acquiring registry code will be the same value as the transferring registry code. The remaining acquiring registry fields and the Notification ID will be null. In the Transaction Unit Block fields, the unit type code will be the proposed ERU current unit type (code 3 or 4). The track and project identifier fields are required for ERUs and therefore for Conversion transactions. If the unit was previously an RMU, the LULUCF Activity Code is also required.
- External Transfer - Transaction Type 3. In External transactions the transferring registry, acquiring registry, transferring account type, acquiring account type, transferring account identifier and acquiring account identifier are required. The Notification ID will be null unless the transaction is to the CDM registry for excess issuance cancellation. It is not possible to "retire" units to another registry using an External transaction. The unit blocks will be completed as appropriate for their characteristics.
- Cancellation - Transaction Type 4. The transferring registry code and transferring account type describe the holder of the units prior to cancellation. The acquiring registry code, acquiring account type, and acquiring account identifier will describe the cancellation account. The Notification ID will be null for voluntary cancellations. The Notification ID is required for cancellations resulting from the impending expiry of

tCERs or ICERs, from non-compliance cancellations, net source cancellations or cancellations related to the non-submission of a certification report. The unit blocks will be completed as appropriate for their characteristics.

- Retirement - Transaction Type 5. The transferring registry code and transferring account type describe the holder of the units prior to retirement. The acquiring registry code, acquiring account type, and acquiring account identifier will describe the retirement account. The Notification ID will be null. The unit blocks will be completed as appropriate for their characteristics.
- Replacement - Transaction Type 6. The transferring registry code and transferring account type describe the holder of the replacing units prior to the transaction. The acquiring registry, acquiring account type, and acquiring account Identifier describe the replacement account receiving the units. The Notification ID will be populated if this replacement is in response to a notification due to reversal of storage, non-submission of a certification report, or the impending expiry of a tCER or ICER. If the replacement is not for these reasons, it will be null. The unit blocks listed first represent the units being transferred to the replacement account. After all these blocks are listed, the blocks that are being replaced are listed. The blocks being replaced are designated by a block role of "REP." (The block role of the replacing blocks will be null.) The remaining characteristics of the unit blocks (both replacing and replaced) will be completed as appropriate to their characteristics.
- Carry-over - Transaction Type 7. The transferring registry code and transferring account type describe the holder of the units for carry-over. The acquiring registry code will be the same value as the transferring registry code. The remaining acquiring registry fields will be null. A Notification ID must be provided. In the Unit Block fields, the Applicable Commitment Period will be the value of the new Applicable Commitment Period after the carry-over is completed. The remaining characteristics of the unit blocks will be completed as appropriate to their characteristics.
- Expiry Date Change - Transaction Type 8. The transferring registry code and transferring account type describe the holder of the units in the transaction. The acquiring registry code will be the same value as the transferring registry code. A valid Notification ID must be provided. The remaining acquiring registry fields will be null. In the Unit Block fields, the Expiry Date must be the value of the new expiry date after the transaction is completed. The remaining fields of the unit blocks will be completed as appropriate to their characteristics.

## 2. Transaction Examples

The following transaction examples provide guidance for which optional parameters are expected under which circumstances for each transaction type.

Note that if a parameter is specified as "not supplied," a value is not expected to be passed in the given scenario and the element may be omitted. If it is required in certain circumstances (for certain unit types, for example), it should be considered to be "not supplied" when the criteria is not met.

Although a transaction may involve as many unit blocks as desired, these examples only involve one or two for simplicity.

## 2.1 AcceptProposal

**Figure L4: Transaction Type 1-00 (Issuance of AAs)**

This example issues 1000 units to holding account(s) in Japan's registry. Note that there is only acquiring registry information.

<u>ProposalRequest</u>	Type	WSDL	KP Requirement	KP Sample Data	EU Requirement
from	string	required	required	JP	Not applicable
to	string	required	required	ITL	
major Version	int	required	required	1	
minor Version	int	required	required	1	
proposed Transaction	Proposal Transaction	required	required		
transaction Identifier	string	required	required	JP1	
transaction Type	int	required	required	1	
supp Transaction Type	int	optional	not supplied		
transferring Registry Code	string	required	required	JP	
transferring Registry Account Type	int	optional	not supplied		
transferring Registry Account Identifier	long	optional	not supplied		
acquiring Registry Code	string	required	required	JP	
acquiring Registry Account Type	int	optional	required	100	
acquiring Registry Account Identifier	long	optional	not supplied		
notification Identifier	long	optional	not supplied		
proposal Unit Blocks	Transaction Unit Block[ ]	required	required		
unit Serial Block Start	long	required	required	1001	
unit Serial Block End	long	required	required	2000	
originating Registry Code	string	required	required	JP	
unit Type	int	required	required	1	
supp Unit Type	int	optional	not supplied		
original Commit Period	int	required	required	1	
applicable Commit Period	int	required	required	1	
LULUCF Activity	int	optional	required for RMUs and for ICERs/tCERs issued by CDM		
project Identifier	int	optional	required for CERs, ICERs, and tCERs issued by CDM		
track	int	optional	not supplied		
block Role	string	optional	not supplied		
transferring Registry Account Type	int	optional	not supplied		
transferring Registry Account Identifier	long	optional	not supplied		
acquiring Registry Account Type	int	optional	not supplied		
acquiring Registry Account Identifier	long	optional	not supplied		
year in Commitment Period	int	optional	not supplied		
installation Identifier	long	optional	not supplied		
expiry Date	date	optional	required for ICERs/tCERs issued by CDM		

**Figure L5: Transaction Type 1  
Supp Transaction Type 51  
(Issue of Allowances 2005-2007)**

This example issues 1000 allowances to a national holding account in Belgium's registry.  
Note that there is only acquiring registry information.

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>EU Requirement</b>	<b>EU Sample Data</b>
from	string	required	Not applicable	required	BE
to	string	required		required	CTL
major Version	int	required		required	1
minor Version	int	required		required	1
proposed Transaction	Proposal Transaction	required		required	
transaction Identifier	string	required		required	BE1
transaction Type	int	required		required	1
supp Transaction Type	int	optional		required	51
transferring Registry Code	string	required		required	BE
transferring Registry Account Type	int	optional		optional	100
transferring Registry Account Identifier	long	optional		optional	1
acquiring Registry Code	string	required		required	BE
acquiring Registry Account Type	int	optional		optional	100
acquiring Registry Account Identifier	long	optional		optional	1
notification Identifier	long	optional		not supplied	
proposal Unit Blocks	Transaction Unit Block[ ]	required		required	
unit Serial Block Start	long	required		required	1001
unit Serial Block End	long	required		required	2000
originating Party Code	string	required		required	BE
unit Type	int	required		required	0
supp Unit Type	int	optional		required	2
original Commit Period	int	required		required	0
applicable Commit Period	int	required		required	0
LULUCF Activity	int	optional		not supplied	
project Identifier	int	optional		not supplied	
track	int	optional		not supplied	
block Role	string	optional		not supplied	
transferring Registry Account Type	int	optional		optional	
transferring Registry Account Identifier	long	optional		optional	
acquiring Registry Account Type	int	optional		optional	
acquiring Registry Account Identifier	long	optional		optional	
year in Commitment Period	int	optional	not supplied		
installation Identifier	long	optional	not supplied		
expiry Date	date	optional	not supplied		

**Figure L6: Transaction Type 1  
Supp Transaction Type 54  
(Issue of Force Majeure Allowances)**

This example issues 1000 allowances to a national holding account in Belgium's registry for installation 123 in 2005. Note that there is only acquiring registry information.

<u>ProposalRequest</u>	Type	WSDL	KP Requirement	EU Requirement	EU SampleData
from	string	required	Not applicable	required	BE
to	string	required		required	CTL
major Version	int	required		required	1
minor Version	int	required		required	1
proposed Transaction	Proposal Transaction	required		required	
transaction Identifier	string	required		required	BE1
transaction Type	int	required		required	1
supp Transaction Type	int	optional		required	54
transferring Registry Code	string	required		required	BE
transferring Registry Account Type	int	optional		optional	
transferring Registry Account Identifier	long	optional		optional	
acquiring Registry Code	string	required		required	BE
acquiring Registry Account Type	int	optional		optional	
acquiring Registry Account Identifier	long	optional		optional	
notification Identifier	long	optional		not supplied	
proposal Unit Blocks	Transaction Unit Block[ ]	required		required	
unit Serial Block Start	long	required		required	1001
unit Serial Block End	long	required		required	2000
originating Registry Code	string	required		required	BE
unit Type	int	required		required	0
supp Unit Type	int	optional		not supplied	3
original Commit Period	int	required		required	0
applicable Commit Period	int	required		required	0
LULUCF Activity	int	optional		not supplied	
project Identifier	int	optional		not supplied	
track	int	optional		not supplied	
block Role	string	optional		not supplied	
transferring Registry Account Type	int	optional		not supplied	
transferring Registry Account Identifier	long	optional	not supplied		
acquiring Registry Account Type	int	optional	optional	100	
acquiring Registry Account Identifier	long	optional	optional	1	
year in Commitment Period	int	optional	required	2005	
installation Identifier	long	optional	required	123	
expiry Date	date	optional	not supplied		

**Figure L7: Transaction Type 2  
(Conversion of AAUs and RMUs to ERUs)**

This example converts 100 RMUs into ERUs. Note that conversion requires a Project Identifier and track. Since RMUs are being converted, LULUCF activity code is also required.

ProposalRequest	Type	WSDL	KP Requirement	KP Sample Data	EU Requirement	EU Sample
from	string	required	required	CL		
to	string	required	required	ITL		
major Version	int	required	required	1		
minor Version	int	required	required	1		
proposed Transaction	Proposal Transaction	required	required			
transaction Identifier	string	required	required	CL101		
transaction Type	int	required	required	2		
supp Transaction Type	int	optional	not supplied			
transferring Registry Code	string	required	required	CL		
transferring Registry Account Type	int	optional	required	100		
transferring Registry Account Identifier	long	optional	not supplied			
acquiring Registry Code	string	required	required	CL		
acquiring Registry Account Type	int	optional	not supplied			
acquiring Registry Account Identifier	long	optional	not supplied			
notification Identifier	long	optional	not supplied			
proposal Unit Blocks	Transaction Unit Block[ ]	required	required			
unit Serial Block Start	long	required	required	301		
unit Serial Block End	long	required	required	400		
originating Registry Code	string	required	required	CL		
unit Type	int	required	required. Value will be unit type after conversion (3 or 4).	4		
supp Unit Type	int	optional	not supplied			
original Commit Period	int	required	required	1		
applicable Commit Period	int	required	required	1		
LULUCF Activity	int	optional	required if converting RMUs	1		
project Identifier	int	optional	required	5		
track	int	optional	required	2		
block Role	string	optional	not supplied			
transferring Registry Account Type	int	optional	not supplied			
transferring Registry Account Identifier	long	optional	not supplied			
acquiring Registry Account Type	int	optional	not supplied			
acquiring Registry Account Identifier	long	optional	not supplied			
year in Commitment Period	int	optional	not supplied			
installation Identifier	long	optional	not supplied			
expiry Date	date	optional	not supplied			



**Figure L8: Transaction Type 3  
(External Transfer of AAUs)**

This example transfers 35 AAUs from a holding account in one registry to a holding account in another registry.

<u>ProposalRequest</u>	Type	WSDL	KP Requirement	KP Sample Data	EU Requirement	EU Sample Data
from	string	required	required	AR	required	BE
to	string	required	required	ITL	required	CTL
major Version	int	required	required	1	required	1
minor Version	int	required	required	1	required	1
proposed Transaction	Proposal Transaction	required	required		required	
transaction Identifier	string	required	required	AR29	required	BE29
transaction Type	int	required	required	3	required	3
supp Transaction Type	int	optional	not supplied		not supplied	
transferring Registry Code	string	required	required	AR	required	BE
transferring Registry Account Type	int	optional	required	100	optional	120
transferring Registry Account Identifier	long	optional	required	50	optional	12
acquiring Registry Code	string	optional	required	PY	required	FR
acquiring Registry Account Type	int	optional	required	100	optional	120
acquiring Registry Account Identifier	long	optional	required	45	required	45
notification Identifier	long	optional	not supplied		not supplied	
proposal Unit Blocks	Transaction Unit Block[ ]	required	required		required	
unit Serial Block Start	long	required	required	601	required	601
unit Serial Block End	long	required	required	635	required	635
originating Registry Code	string	required	required	AR	required	BE
unit Type	int	required	required	1	required	1
supp Unit Type	int	optional	not supplied		required	1
original Commit Period	int	required	required	1	required	1
applicable Commit Period	int	required	required	1	required	1
LULUCF Activity	int	optional	required for RMUs, ERUs converted from RMUs, ICERs and tCERs		required for RMUs, ERUs converted from RMUs, ICERs and tCERs	
project Identifier	int	optional	required for ERUs, CERs, ICERs and tCERs		required for ERUs, CERs, ICERs and tCERs	
track	int	optional	required for ERUs		required for ERUs	
block Role	string	optional	not supplied		not supplied	
transferring Registry Account Type	int	optional	not supplied		optional	
transferring Registry Account Identifier	long	optional	not supplied		optional	
acquiring Registry Account Type	int	optional	not supplied		optional	
acquiring Registry Account Identifier	long	optional	not supplied		optional	

(cont.)

**Figure L8: Transaction Type 3  
(External Transfer of AAUs) (cont.)**

<u>ProposalRequest</u>	Type	WSDL	KP Requirement	KP Sample Data	EU Requirement	EU Sample Data
year in Commitment Period	int	optional	not supplied		not supplied	
installation Identifier	long	optional	not supplied		not supplied	
expiry Date	date	optional	required for ICERs, tCERs		required for ICERs, tCERs	

**Figure L9: Transaction Type 3  
Supp Transaction Type 21  
(External Transfer 2005-2007)**

This example transfers 35 allowances from a holding account in one registry to a holding account in another registry.

<u>ProposalRequest</u>	Type	WSDL	KP Requirement	EU Requirement	EU Sample Data
from	String	required	Not applicable	required	BE
to	String	required		required	CTL
major Version	Int	required		required	1
minor Version	Int	required		required	1
proposed Transaction	Proposal Transaction	required		required	
transaction Identifier	String	required		required	BE29
transaction Type	Int	required		required	3
supp Transaction Type	Int	optional		required	21
transferring Registry Code	String	required		required	BE
transferring Registry Account Type	Int	optional		optional	120
transferring Registry Account Identifier	long	optional		optional	56
acquiring Registry Code	string	required		required	FR
acquiring Registry Account Type	Int	optional		optional	120
acquiring Registry Account Identifier	Long	optional		optional	135
notification Identifier	long	optional		not supplied	
proposal Unit Blocks	Transaction Unit Block[ ]	required		required	
unit Serial Block Start	long	required		required	601
unit Serial Block End	long	required		required	635
originating Registry Code	string	required		required	BE
unit Type	int	required		required	0
supp Unit Type	int	optional	required	2	
original Commit Period	int	required	required	0	
applicable Commit Period	int	required	required	0	

(cont.)

**Figure L9: Transaction Type 3  
 Supp Transaction Type 21  
 (External Transfer 2005-2007) (cont.)**

<u>ProposalRequest</u>	Type	WSDL	KP Requirement	EU Requirement	EU Sample Data
LULUCF Activity	int	optional	Not applicable (cont.)	required for RMUs, ERUs converted from RMUs, ICERs and tCERs	
project Identifier	int	optional		required for ERUs, CERs, ICERs and tCERs	
track	int	optional		required for ERUs	
block Role	string	optional		not supplied	
transferring Registry Account Type	int	optional		optional	
transferring Registry Account Identifier	long	optional		optional	
acquiring Registry Account Type	int	optional		optional	
acquiring Registry Account Identifier	long	optional		optional	
year in Commitment Period	int	optional		not supplied	
installation Identifier	long	optional		not supplied	
expiry Date	date	optional		required for ICERs, tCERs	

**Figure L10: Transaction Type 4  
(Cancellation of AAUs)**

This example transfers 55 AAUs to a voluntary cancellation account. Note that since the acquiring account is a cancellation account, the Kyoto Protocol requires that the account identifier be specified.

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>	<b>EU Requirement</b>
from	string	required	required	CA	Not applicable
to	string	required	required	ITL	
major Version	int	required	required	1	
minor Version	int	required	required	1	
proposed Transaction	Proposal Transaction	required	required		
transaction Identifier	string	required	required	CA101	
transaction Type	int	required	required	4	
supp Transaction Type	int	optional	not supplied		
transferring Registry Code	string	required	required	CA	
transferring Registry Account Type	int	optional	required	100	
transferring Registry Account Identifier	long	optional	not supplied		
acquiring Registry Code	string	required	required	CA	
acquiring Registry Account Type	int	optional	required	230	
acquiring Registry Account Identifier	long	optional	required	43	
notification Identifier	long	optional	not supplied		
proposal Unit Blocks	Transaction Unit Block[ ]	required	required		
unit Serial Block Start	long	required	required	236	
unit Serial Block End	long	required	required	290	
originating Registry Code	string	required	required	CA	
unit Type	int	required	required	1	
supp Unit Type	int	optional	not supplied		
original Commit Period	int	required	required	1	
applicable Commit Period	int	required	required	1	
LULUCF Activity	int	optional	required for RMUs, ERUs converted from RMUs, ICERs and tCERs		
project Identifier	int	optional	required for ERUs, CERs, ICERs and tCERs		
track	int	optional	required for ERUs		
block Role	string	optional	not supplied		
transferring Registry Account Type	int	optional	not supplied		
transferring Registry Account Identifier	long	optional	not supplied		
acquiring Registry Account Type	int	optional	not supplied		
acquiring Registry Account Identifier	long	optional	not supplied		
year in Commitment Period	int	optional	not supplied		
installation Identifier	long	optional	not supplied		
expiry Date	date	optional	required for ICERs, tCERs		

**Figure L11: Transaction Type 4  
Supp Transaction Type 3  
(Retirement 2005-2007)**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>EU Requirement</b>	<b>EU Sample Data</b>
from	string	required	Not applicable	required	BE
to	string	required		required	CTL
major Version	int	required		required	1
minor Version	int	required		required	1
proposed Transaction	Proposal Transaction	required		required	
transaction Identifier	string	required		required	BE29
transaction Type	int	required		required	4
supp Transaction Type	int	optional		required	3
transferring Registry Code	string	required		required	BE
transferring Registry Account Type	int	optional		optional	
transferring Registry Account Identifier	long	optional		optional	
acquiring Registry Code	string	required		required	BE
acquiring Registry Account Type	int	optional		optional	
acquiring Registry Account Identifier	long	optional		optional	
notification Identifier	long	optional		not supplied	
proposal Unit Blocks	Transaction Unit Block[ ]	required		required	
unit Serial Block Start	long	required		required	236
unit Serial Block End	long	required		required	290
originating Registry Code	string	required		required	BE
unit Type	int	required		required	0
supp Unit Type	int	optional		required	2
original Commit Period	int	required		required	0
applicable Commit Period	int	required		required	0
LULUCF Activity	int	optional		required for RMUs, ERUs converted from RMUs, ICERs and tCERs	
project Identifier	int	optional		required for ERUs, CERs, ICERs and tCERs	
track	int	optional		required for ERUs	
block Role	string	optional		not supplied	
transferring Registry Account Type	int	optional		optional	100
transferring Registry Account Identifier	long	optional		optional	34
acquiring Registry Account Type	int	optional		optional	230
acquiring Registry Account Identifier	long	optional	optional	43	
year in Commitment Period	int	optional	not supplied		
installation Identifier	long	optional	not supplied		
expiry Date	date	optional	required for ICERs, tCERs		

**Figure L12: Transaction Type 5  
(Retirement 2008-2012)**

This Kyoto example retires 500 tCERs held by a holding account in the Bolivian registry. Note that since the acquiring account is a retirement account, the Kyoto Protocol requires the account identifier to be specified.

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>	<b>EU Requirement</b>	<b>EU Sample Data</b>
from	string	required	required	BO	required	BE
to	string	required	required	ITL	required	CTL
major Version	int	required	required	1	required	1
minor Version	int	required	required	1	required	1
proposed Transaction	Proposal Transaction	required	required		required	
transaction Identifier	string	required	required	BO243	required	BE101
transaction Type	int	required	required	5	required	5
supp Transaction Type	int	optional	not supplied		not supplied	
transferring Registry Code	string	required	required	BO	required	BE
transferring Registry Account Type	int	optional	required	100	optional	100
transferring Registry Account Identifier	long	optional	not supplied		optional	12
acquiring Registry Code	string	required	required	BO	required	BE
acquiring Registry Account Type	int	optional	required	300	optional	300
acquiring Registry Account Identifier	long	optional	required	41	optional	41
notification Identifier	long	optional	not supplied		not supplied	
proposal Unit Blocks	Transaction Unit Block[ ]	required	required		required	
unit Serial Block Start	long	required	required	1001	required	1001
unit Serial Block End	long	required	required	1500	required	1500
originating Registry Code	string	required	required	BO	required	BE
unit Type	int	required	required	6	required	1
supp Unit Type	int	optional	not supplied		not supplied	
original Commit Period	int	required	required	1	required	1
applicable Commit Period	int	required	required	1	required	1
LULUCF Activity	int	optional	required for RMUs, ERUs converted from RMUs, ICERs and tCERs	1		
project Identifier	int	optional	required for ERUs, CERs, ICERs and tCERs	5	optional	
track	int	optional	required for ERUs		optional	
block Role	string	optional	not supplied		not supplied	
transferring Registry Account Type	int	optional	not supplied		optional	
transferring Registry Account Identifier	long	optional	not supplied		optional	
acquiring Registry Account Type	int	optional	not supplied		optional	
acquiring Registry Account Identifier	long	optional	not supplied		optional	

(cont.)

**Figure L12: Transaction Type 5  
(Retirement 2008-2012) (cont.)**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>	<b>EU Requirement</b>	<b>EU Sample Data</b>
year in Commitment Period	int	optional	not supplied		not supplied	
installation Identifier	long	optional	not supplied		not supplied	
expiry Date	date	optional	required for ICERs, tCERs	2017-12-31	not supplied	

**Figure L13: Transaction Type 6  
(Replacement of ICERs)**

This example replaces 500 ICERs with 500 AAUs. Note that the replacement blocks (the AAUs) are listed first, and that the blocks being replaced (the ICERs) are listed last and indicated by a block role of 'REP.'

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>	<b>EU Requirement</b>
from	string	required	required	BO	Not applicable
to	string	required	required	ITL	
major Version	int	required	required	1	
minor Version	int	required	required	1	
proposed Transaction	Proposal Transaction	required	required		
transaction Identifier	string	required	required	BO244	
transaction Type	int	required	required	6	
supp Transaction Type	int	optional	not supplied		
transferring Registry Code	string	required	required	BO	
transferring Registry Account Type	int	optional	required	100	
transferring Registry Account Identifier	long	optional	required	2000	
acquiring Registry Code	string	required	required	BO	
acquiring Registry Account Type	int	optional	required	422	
acquiring Registry Account Identifier	long	optional	not supplied		
notification Identifier	long	optional	required if prompted by non-submission of certification, by reversal of storage, or by pending expiration	687	
proposal Unit Blocks	Transaction Unit Block[ ]	required	required		
<b>First Block(s) – Replacing Units</b>					
unit Serial Block Start	long	required	required	1	
unit Serial Block End	long	required	required	500	
originating Registry Code	string	required	required	BO	
unit Type	int	required	required	1	
supp Unit Type	int	optional	not supplied		
original Commit Period	int	required	required	1	
applicable Commit Period	int	required	required	1	

(cont.)

**Figure L13: Transaction Type 6  
(Replacement of ICERs) (cont.)**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>	<b>EU Requirement</b>
LULUCF Activity	int	optional	required for RMUs, ERUs converted from RMUs, ICERs and tCERs		Not applicable (cont.)
project Identifier	int	optional	required for ERUs, CERs, ICERs and tCERs		
track	int	optional	required for ERUs		
block Role	string	optional	not supplied		
transferring Registry Account Type	int	optional	not supplied		
transferring Registry Account Identifier	long	optional	not supplied		
acquiring Registry Account Type	int	optional	not supplied		
acquiring Registry Account Identifier	long	optional	not supplied		
year in Commitment Period	int	optional	not supplied		
installation Identifier	long	optional	not supplied		
expiry Date	date	optional	required for ICERs and tCERs		
<b>Last Block(s) – Units to be replaced</b>					
unit Serial Block Start	long	required	required	1001	
unit Serial Block End	long	required	required	1500	
originating Registry Code	string	required	required	BO	
unit Type	int	required	required	7	
supp Unit Type	int	optional	not supplied		
original Commit Period	int	required	required	1	
applicable Commit Period	int	required	required	1	
LULUCF Activity	int	optional	required for RMUs, ERUs converted from RMUs, ICERs and tCERs	1	
project Identifier	int	optional	required for ERUs, CERs, ICERs and tCERs	5	
track	int	optional	required for ERUs		
block Role	string	optional	required	REP	
transferring Registry Account Type	int	optional	not supplied		
transferring Registry Account Identifier	long	optional	not supplied		
acquiring Registry Account Type	int	optional	not supplied		
acquiring Registry Account Identifier	long	optional	not supplied		
year in Commitment Period	int	optional	not supplied		
installation Identifier	long	optional	not supplied		
expiry Date	date	optional	required for ICERs and tCERs	2028-12-31	



**Figure L14: Transaction Type 7  
(Carry-over)**

This example carries over 300 AAUs in a New Zealand holding account into the second commitment period. Note that since only one account is involved, no acquiring account information is specified. Also note that the value specified for the Applicable Commitment Period for the block(s) is the new commitment period; aside from this, the unit block information is unchanged.

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>	<b>EU Requirement</b>
from	string	required	required	NZ	Not applicable
to	string	required	required	ITL	
major Version	int	required	required	1	
minor Version	int	required	required	1	
proposed Transaction	Proposal Transaction	required	required		
transaction Identifier	string	required	required	NZ624	
transaction Type	int	required	required	7	
supp Transaction Type	int	optional	not supplied		
transferring Registry Code	string	required	required	NZ	
transferring Registry Account Type	int	optional	required	100	
transferring Registry Account Identifier	long	optional	not supplied		
acquiring Registry Code	string	required	required	NZ	
acquiring Registry Account Type	int	optional	not supplied		
acquiring Registry Account Identifier	long	optional	not supplied		
notification Identifier	int	optional	required	338	
proposal Unit Blocks	Transaction Unit Block[ ]	required	required		
unit Serial Block Start	long	required	required	1	
unit Serial Block End	long	required	required	300	
originating Registry Code	string	required	required	NZ	
unit Type	int	required	required	1	
supp Unit Type	int	optional	not supplied		
original Commit Period	int	required	required	1	
applicable Commit Period	int	required	required. Value is new commitment period	2	
LULUCF Activity	int	optional	not supplied (unit types associated with LULUCFs are not eligible for carry-over)		
project Identifier	int	optional	required for ERUs and CERs (only ERUs converted from AAUs are eligible for carry-over)		
track	int	optional	required for ERUs (only ERUs converted from AAUs are eligible for carry-over)		
block Role	string	optional	not supplied		

(cont.)

**Figure L14: Transaction Type 7  
(Carry-over) (cont.)**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>	<b>EU Requirement</b>
transferring Registry Account Type	int	optional	not supplied		Not applicable (cont.)
transferring Registry Account Identifier	long	optional	not supplied		
acquiring Registry Account Type	int	optional	not supplied		
acquiring Registry Account Identifier	long	optional	not supplied		
year in Commitment Period	int	optional	not supplied		
installation Identifier	long	optional	not supplied		
expiry Date	date	optional	not supplied (ICERs and tCERs are not eligible for carry-over)		

**Figure L15: Transaction Type 8  
(Expiry Date Change)**

This example extends the Expiry Date on a block of tCERs by one year. Note that the Expiry Date specified is the new date; aside from this, the unit block information is unchanged.

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>	<b>EU Requirement</b>
from	string	required	required	BO	Not applicable
to	string	required	required	ITL	
major Version	int	required	required	1	
minor Version	int	required	required	1	
proposed Transaction	Proposal Transaction	required	required		
transaction Identifier	string	required	required	BO243	
transaction Type	int	required	required	8	
supp Transaction Type	int	optional	not supplied		
transferring Registry Code	string	required	required	BO	
transferring Registry Account Type	int	optional	required	100	
transferring Registry Account Identifier	long	optional	not supplied		
acquiring Registry Code	string	required	required	BO	
acquiring Registry Account Type	int	optional	not supplied		
acquiring Registry Account Identifier	long	optional	not supplied		
notification Identifier	long	optional	required	339	
proposal Unit Blocks	Transaction Unit Block[ ]	required	required		
unit Serial Block Start	long	required	required	2001	
unit Serial Block End	long	required	required	2500	
originating Registry Code	string	required	required	BO	
unit Type	int	required	required	6	
supp Unit Type	int	optional	not supplied		
original Commit Period	int	required	required	1	
applicable Commit Period	int	required	required	1	
LULUCF Activity	int	optional	required (units are ICERs or tCERs)	1	
project Identifier	int	optional	required (units are ICERs or tCERs)	5	
track	int	optional	not supplied		
block Role	string	optional	not supplied		
transferring Registry Account Type	int	optional	not supplied		
transferring Registry Account Identifier	long	optional	not supplied		
acquiring Registry Account Type	int	optional	not supplied		
acquiring Registry Account Identifier	long	optional	not supplied		
year in Commitment Period	int	optional	not supplied		
installation Identifier	long	optional	not supplied		
expiry Date	date	optional	required. Value is new Expiry Date	2018-12-31	

**Figure L16: Transaction Type 10  
Supp Transaction Type 0  
(Internal Transfer)**

This example transfers 1000 allowances from an operator holding account to another operator holding account in the Belgian registry.

<u>ProposalRequest</u>	Type	WSDL	KP Requirement	EU Requirement	EU Sample Data
from	string	required	Not applicable	required	BE
to	string	required		required	CTL
major Version	int	required		required	1
minor Version	int	required		required	1
proposed Transaction	Proposal Transaction	required		required	
transaction Identifier	string	required		required	BE1
transaction Type	int	required		required	10
supp Transaction Type	int	optional		required	0
transferring Registry Code	string	required		required	BE
transferring Registry Account Type	int	optional		optional	
transferring Registry Account Identifier	long	optional		optional	
acquiring Registry Code	string	required		required	BE
acquiring Registry Account Type	int	optional		optional	
acquiring Registry Account Identifier	long	optional		optional	
notification Identifier	long	optional		not supplied	
proposal Unit Blocks	Transaction Unit Block[ ]	required		required	
unit Serial Block Start	long	required		required	1001
unit Serial Block End	long	required		required	2000
originating Registry Code	string	required		required	BE
unit Type	int	required		required	0
supp Unit Type	int	optional		required	2
original Commit Period	int	required		required	0
applicable Commit Period	int	required		required	0
LULUCF Activity	int	optional		not supplied	
project Identifier	int	optional		not supplied	
track	int	optional		not supplied	
block Role	string	optional		not supplied	
transferring Registry Account Type	int	optional		optional	120
transferring Registry Account Identifier	long	optional		optional	56
acquiring Registry Account Type	int	optional		optional	120
acquiring Registry Account Identifier	long	optional		optional	57
year in Commitment Period	int	optional	not supplied		
installation Identifier	long	optional	not supplied		
expiry Date	date	optional	not supplied		

**Figure L17: Transaction Type 10  
Supp Transaction Type 52  
(Issue of Allowances 2008-2012)**

This example issues (convert AAUs) 1000 allowances to a national holding account.

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>EU Requirement</b>	<b>EU Sample Data</b>
from	string	required	Not applicable	required	BE
to	string	required		required	CTL
major Version	int	required		required	1
minor Version	int	required		required	1
proposed Transaction	Proposal Transaction	required		required	
transaction Identifier	string	required		required	BE1
transaction Type	int	required		required	10
supp Transaction Type	int	optional		required	52
transferring Registry Code	string	required		required	BE
transferring Registry Account Type	int	optional		optional	
transferring Registry Account Identifier	long	optional		optional	
acquiring Registry Code	string	required		required	BE
acquiring Registry Account Type	int	optional		optional	
acquiring Registry Account Identifier	long	optional		optional	
notification Identifier	long	optional		not supplied	
proposal Unit Blocks	Transaction Unit Block[ ]	required		required	
unit Serial Block Start	long	required		required	1001
unit Serial Block End	long	required		required	2000
originating Registry Code	string	required		required	BE
unit Type	int	required		required	1
supp Unit Type	int	optional		required	1
original Commit Period	int	required		required	1
applicable Commit Period	int	required		required	1
LULUCF Activity	int	optional		not supplied	
project Identifier	int	optional		not supplied	
track	int	optional		not supplied	
block Role	string	optional		not supplied	
transferring Registry Account Type	int	optional		optional	100
transferring Registry Account Identifier	long	optional		optional	1
acquiring Registry Account Type	int	optional		optional	100
acquiring Registry Account Identifier	long	optional	optional	1	
year in Commitment Period	int	optional	not supplied		
installation Identifier	long	optional	not supplied		
expiry Date	date	optional	not supplied		

**Figure L18: Transaction Type 10  
Supp Transaction Type 53  
(Allocation of Allowances 2005-2007)**

This example allocates 1000 allowances to installation 123 for year 2005.

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>EU Requirement</b>	<b>EU Sample Data</b>
from	string	required	Not applicable	required	BE
to	string	required		required	CTL
major Version	int	required		required	1
minor Version	int	required		required	1
proposed Transaction	Proposal Transaction	required		required	
transaction Identifier	string	required		required	BE1
transaction Type	int	required		required	10
supp Transaction Type	int	optional		required	53
transferring Registry Code	string	required		required	BE
transferring Registry Account Type	int	optional		optional	
transferring Registry Account Identifier	long	optional		optional	
acquiring Registry Code	string	required		required	BE
acquiring Registry Account Type	int	optional		optional	
acquiring Registry Account Identifier	long	optional		optional	
notification Identifier	long	optional		not supplied	
proposal Unit Blocks	Transaction Unit Block[ ]	required		required	
unit Serial Block Start	long	required		required	1001
unit Serial Block End	long	required		required	2000
originating Registry Code	string	required		required	BE
unit Type	int	required		required	0
supp Unit Type	int	optional		not supplied	2
original Commit Period	int	required		required	0
applicable Commit Period	int	required		required	0
LULUCF Activity	int	optional		not supplied	
project Identifier	int	optional		not supplied	
track	int	optional		not supplied	
block Role	string	optional		not supplied	
transferring Registry Code	string	optional		not supplied	
transferring Registry Account Type	int	optional		optional	100
transferring Registry Account Identifier	long	optional		optional	1
acquiring Registry Account Type	int	optional		optional	120
acquiring Registry Account Identifier	long	optional		optional	22
year in Commitment Period	int	optional	required	2005	
installation Identifier	long	optional	required	123	
expiry Date	date	optional	not supplied		

**Figure L19: Transaction Type 10  
Supp Transaction Type 55  
(Correction to allowance)**

This example cancels 1000 allowances for installation 123 following the correction of the NAP for that installation in 2009.

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>EU Requirement</b>	<b>EU Sample Data</b>
from	string	required	Not applicable	required	BE
to	string	required		required	CTL
major Version	int	required		required	1
minor Version	int	required		required	1
proposed Transaction	Proposal Transaction	required		required	
transaction Identifier	string	required		required	BE1
transaction Type	int	required		required	10
supp Transaction Type	int	optional		required	55
transferring Registry Code	string	required		required	BE
transferring Registry Account Type	int	optional		optional	
transferring Registry Account Identifier	long	optional		optional	
acquiring Registry Code	string	required		required	BE
acquiring Registry Account Type	int	optional		optional	
acquiring Registry Account Identifier	long	optional		optional	
notification Identifier	long	optional		not supplied	
proposal Unit Blocks	Transaction Unit Block[ ]	required		required	
unit Serial Block Start	long	required		required	1001
unit Serial Block End	long	required		required	2000
originating Registry Code	string	required		required	BE
unit Type	int	required		required	1
supp Unit Type	int	optional		required	0
original Commit Period	int	required		required	1
applicable Commit Period	int	required		required	1
LULUCF Activity	int	optional		not supplied	
project Identifier	int	optional		not supplied	
track	int	optional		not supplied	
block Role	string	optional		not supplied	
transferring Registry Account Type	int	optional		optional	120
transferring Registry Account Identifier	long	optional		optional	36
acquiring Registry Account Type	int	optional		optional	100
acquiring Registry Account Identifier	long	optional		optional	1
year in Commitment Period	int	optional		required	
installation Identifier	long	optional	required		
expiry Date	date	optional	not supplied		

**Figure L20: Transaction Type 10  
Supp Transaction Type 1  
(Cancellation of Allowances 2005-2007)**

This example cancels 1000 allowances from account 35 for 2005-2007.

<u>ProposalRequest</u>	Type	WSDL	KP Requirement	EU Requirement	EU Sample Data
from	string	required	Not applicable	required	BE
to	string	required		required	CTL
major Version	int	required		required	1
minor Version	int	required		required	1
proposed Transaction	Proposal Transaction	required		required	
transaction Identifier	string	required		required	BE1
transaction Type	int	required		required	10
supp Transaction Type	int	optional		required	1
transferring Registry Code	string	required		required	BE
transferring Registry Account Type	int	optional		optional	120
transferring Registry Account Identifier	long	optional		optional	35
acquiring Registry Code	string	required		required	BE
acquiring Registry Account Type	int	optional		optional	230
acquiring Registry Account Identifier	long	optional		optional	22
notification Identifier	long	optional		not supplied	
proposal Unit Blocks	Transaction Unit Block[ ]	required		required	
unit Serial Block Start	long	required		required	1001
unit Serial Block End	long	required		required	2000
originating Registry Code	string	required		required	BE
unit Type	int	required		required	0
supp Unit Type	int	optional		not supplied	2
original Commit Period	int	required		required	0
applicable Commit Period	int	required		required	0
LULUCF Activity	int	optional		not supplied	
project Identifier	int	optional		not supplied	
track	int	optional		not supplied	
block Role	string	optional		not supplied	
transferring Registry Account Type	int	optional		optional	
transferring Registry Account Identifier	long	optional		optional	
acquiring Registry Account Type	int	optional		optional	
acquiring Registry Account Identifier	long	optional	optional		
year in Commitment Period	int	optional	not supplied		
installation Identifier	long	optional	not supplied		
expiry Date	date	optional	not supplied		



**Figure L21: Transaction Type 10  
Supp Transaction Type 2  
(Surrender of Allowances)**

This example surrenders 1000 allowances for installation 123 and year 2005.

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>EU Requirement</b>	<b>EU Sample Data</b>
from	string	required	Not applicable	required	BE
to	string	required		required	CTL
major Version	int	required		required	1
minor Version	int	required		required	1
proposed Transaction	Proposal Transaction	required		required	
transaction Identifier	string	required		required	BE1
transaction Type	int	required		required	10
supp Transaction Type	int	optional		required	2
transferring Registry Code	string	required		required	BE
transferring Registry Account Type	int	optional		optional	
transferring Registry Account Identifier	long	optional		optional	
acquiring Registry Code	string	required		required	BE
acquiring Registry Account Type	int	optional		optional	
acquiring Registry Account Identifier	long	optional		optional	
notification Identifier	long	optional		not supplied	
proposal Unit Blocks	Transaction Unit Block[ ]	required		required	
unit Serial Block Start	long	required		required	1001
unit Serial Block End	long	required		required	2000
originating Registry Code	string	required		required	BE
unit Type	int	required		required	0
supp Unit Type	int	optional		required	2
original Commit Period	int	required		required	0
applicable Commit Period	int	required		required	0
LULUCF Activity	int	optional		not supplied	
project Identifier	int	optional		not supplied	
track	int	optional		not supplied	
block Role	string	optional		not supplied	
transferring Registry Account Type	int	optional		required	120
transferring Registry Account Identifier	long	optional		required	36
acquiring Registry Account Type	int	optional		required	100
acquiring Registry Account Identifier	long	optional	required	1	
year in Commitment Period	int	optional	required	2005	
installation Identifier	long	optional	required		
expiry Date	date	optional	not supplied		

**Figure L22: Transaction Type 10  
Supp Transaction Type 41  
(Cancellation and Replacement)**

This example cancels 1000 allowances from an operator holding account and replaces them by 1000 AAUs.

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>EU Requirement</b>	<b>EU Sample Data</b>	
from	string	required	Not applicable	required	BE	
to	string	required		required	CTL	
major Version	int	required		required	1	
minor Version	int	required		required	1	
proposed Transaction	Proposal Transaction	required		required		
transaction Identifier	string	required		required	BE1	
transaction Type	int	required		required	10	
supp Transaction Type	int	optional		required	41	
transferring Registry Code	string	required		required	BE	
transferring Registry Account Type	int	optional		optional		
transferring Registry Account Identifier	long	optional		optional		
acquiring Registry Code	string	required		required	BE	
acquiring Registry Account Type	int	optional		optional		
acquiring Registry Account Identifier	long	optional		optional		
notification Identifier	long	optional		not supplied		
proposal Unit Blocks	Transaction Unit Block[ ]	required		required		
<b>First Unit Block</b>						
unit Serial Block Start	long	required		required	1001	
unit Serial Block End	long	required		required	2000	
originating Registry Code	string	required		required	BE	
unit Type	int	required		required	0	
supp Unit Type	int	optional		required	2	
original Commit Period	int	required		required	0	
applicable Commit Period	int	required		required	0	
LULUCF Activity	int	optional		not supplied		
project Identifier	int	optional		not supplied		
track	int	optional		not supplied		
block Role	string	optional		not supplied		
transferring Registry Account Type	int	optional		required	120	
transferring Registry Account Identifier	long	optional		required	244	
acquiring Registry Account Type	int	optional		required	230	
acquiring Registry Account Identifier	long	optional		required	22	
year in Commitment Period	int	optional	required			
installation Identifier	long	optional	required			
expiry Date	date	optional	not supplied			

(cont.)

**Figure L22: Transaction Type 10  
 Supp Transaction Type 41  
 (Cancellation and Replacement) (cont.)**

<u>ProposalRequest</u>	Type	WSDL	KP Requirement	EU Requirement	EU Sample Data
<b>Second Unit Block</b>			Not Applicable (cont.)		
unit Serial Block Start	long	required		required	2001
unit Serial Block End	long	required		required	3000
originating Registry Code	string	required		required	BE
unit Type	int	required		required	1
supp Unit Type	int	optional		required	1
original Commit Period	int	required		required	1
applicable Commit Period	int	required		required	1
LULUCF Activity	int	optional		not supplied	
project Identifier	int	optional		not supplied	
track	int	optional		not supplied	
block Role	string	optional		not supplied	
transferring Registry Account Type	int	optional		required	100
transferring Registry Account Identifier	long	optional		required	1
acquiring Registry Account Type	int	optional		required	120
acquiring Registry Account Identifier	long	optional		required	244
year in Commitment Period	int	optional		required	
installation Identifier	long	optional	required		
expiry Date	date	optional	not supplied		

## 2.2 AcceptNotification

**Figure L23: No Discrepancy Found**

In this example, the ITL informs a registry that the transaction it proposed has been reviewed without discrepancy.

<u>NotificationRequest</u>	Type	WSDL	KP Requirement	KP Sample Data	EU Requirement	EU Sample Data
From	string	required	required	ITL	required	CTL
to	string	required	required	JP	required	BE
major Version	int	required	required	1	required	1
minor Version	int	required	required	1	required	1
transaction Identifier	string	required	required	JP243	required	BE1
transaction Status	int	required	required	2	required	2
party Type	int	required	required	1	required	1
evaluation Result	<b>ArrayOf Evaluation Result</b>	required	required		required	
response Code	int	required	required	4000	required	4000
unit Block Identifiers	<b>ArrayOf UnitBlock Identifier</b>	optional	specified when response code is specific to certain unit blocks		specified when response code is specific to certain unit blocks	
unit Serial Block Start	long	required	required when unit blocks are specified		required when unit blocks are specified	
unit Serial Block End	long	required	required when unit blocks are specified		required when unit blocks are specified	
originating RegistryCode	string	required	required when unit blocks are specified		required when unit blocks are specified	

**Figure L24: Transaction Completed**

In this example, a registry informs the ITL that it has completed the transaction.

<b>NotificationRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>	<b>EU Requirement</b>	<b>EU Sample Data</b>
from	string	required	required	JP	required	CTL
to	string	required	required	ITL	required	BE
major Version	int	required	required	1	required	1
minor Version	int	required	required	1	required	1
transaction Identifier	string	required	required	JP243	required	BE1
transaction Status	int	required	required	4	required	4
party Type	int	required	required	1	required	1
evaluation Result	<b>ArrayOf Evaluation Result</b>	required	required		required	
response Code	int	required	required	4000	required	4000
unit Block Identifiers	<b>ArrayOf UnitBlock Identifier</b>	optional	specified when response code is specific to certain unit blocks		specified when response code is specific to certain unit blocks	
unit Serial Block Start	long	required	required when unit blocks are specified		required when unit blocks are specified	
unit Serial Block End	long	required	required when unit blocks are specified		required when unit blocks are specified	
originating Registry Code	string	required	required when unit blocks are specified		required when unit blocks are specified	

**Figure L25: Discrepancy Found**

In this example, the ITL informs a registry that the transaction it proposed has been reviewed, and a discrepancy was found (some of the proposed units currently have a reconciliation inconsistency).

<b>NotificationRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>	<b>EU Requirement</b>	<b>EU Sample Data</b>
from	string	required	required	ITL	required	CTL
to	string	required	required	UY	required	BE
major Version	int	required	required	1	required	1
minor Version	int	required	required	1	required	1
transaction Identifier	string	required	required	UY56	required	BE1
transaction Status	int	required	required	3	required	3
party Type	int	required	required	1	required	1
evaluation Result	<b>ArrayOf Evaluation Result</b>	required	required. Repeated for each response code returned.		Required	
response Code	int	required	required	4004	required	7305
unit Block Identifiers	<b>ArrayOf UnitBlock Identifier</b>	optional	specified when response code is specific to certain unit blocks		specified when response code is specific to certain unit blocks	
unit Serial Block Start	long	required	required when unit blocks are specified	201	required when unit blocks are specified	201
unit Serial Block End	long	required	required when unit blocks are specified	230	required when unit blocks are specified	230
originating Registry Code	string	required	required when unit blocks are specified	UY	required when unit blocks are specified	BE

**Figure L26: Transaction Terminated**

In this example, a registry informs the ITL that it has terminated the transaction.

<b>NotificationRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>	<b>EU Requirement</b>	<b>EU Sample Data</b>
from	string	required	required	UY	required	CTL
to	string	required	required	ITL	required	BE
major Version	int	required	required	1	required	1
minor Version	int	required	required	1	required	1
transaction Identifier	string	required	required	UY56	required	BE1
transaction Status	int	required	required	5	required	5
party Type	int	required	required	1	required	1
evaluation Result	<b>ArrayOf Evaluation Result</b>	required	required		required	
response Code	int	required	required	4001	required	4001
unit Block Identifiers	<b>ArrayOf UnitBlock Identifier</b>	optional	specified when response code is specific to certain unit blocks		specified when response code is specific to certain unit blocks	
unit Serial Block Start	long	required	required when unit blocks are specified		required when unit blocks are specified	
unit Serial Block End	long	required	required when unit blocks are specified		required when unit blocks are specified	
originating Registry Code	string	required	required when unit blocks are specified		required when unit blocks are specified	

### 3. Notification Examples

The following notification examples demonstrate the different notification types, and the types of transactions that registries might respond with in order to address the issue. There are also examples of Notification Update messages which might be sent upon fulfillment of a notification requirement or to remind the registry to complete the requirement.

#### 3.1 Net Source Cancellation (Type 1)

This example shows a net source cancellation notification and the subsequent transaction to address the notification. A net source cancellation is the result of a particular LULUCF activity of a Party that has resulted in a net source of emissions. The registry is required to initiate sufficient transfers within 30 days into a Net Source Cancellation Account (Account Type Code 210) and to reference the Notification ID in the transaction. There may be more than one cancellation transaction, but all must reference the Notification ID. The notification's targetValue field will indicate the number of units held by the registry that are affected by this notification. Periodically, or once the cancellation required has been fulfilled, the ITL will send a notification update.

**Figure L27: Net Source Cancellation Notification**

<b>ITLNoticeRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	The Compliance Committee has found that LULUCF activities have resulted in a net source of emissions. Units must be cancelled within 30 days to offset these emissions.
messageDate	dateTime	required	required	2005-10-30 11:05:34
notificationType	int	required	required	1
notificationIdentifier	long	required	required	331
notificationStatus	int	required	required	1
projectNumber	string	optional	not supplied	
unitType	int	optional	not supplied	
targetValue	long	optional	required for Notification Type 1	500
targetDate	date	optional	not supplied	
LULUCFActivity	int	optional	required for Notification Type 1	2
commitPeriod	int	optional	required for Notification Type 1	1
actionDueDate	date	optional	required	2005-11-30
unitBlockIdentifiers	ArrayOfUnit Block Identifier	optional	not supplied	
unitSerialBlockStart	long	required	not supplied	
unitSerialBlockEnd	long	required	not supplied	
originatingRegistryCode	string	required	not supplied	

The registry initiates a cancellation transaction that references the notification identifier and transfers the specified number of units into the net source cancellation account (Account Type Code 210).

**Figure L28: Net Source Cancellation Transaction**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP SampleData</b>
from	string	required	required	ZZ
to	string	required	required	ITL
major Version	int	required	required	1
minor Version	int	required	required	1
proposed Transaction	Proposal Transaction	required	required	
transaction Identifier	string	required	required	ZZ101
transaction Type	int	required	required	4
supp Transaction Type	int	optional	not supplied	
transferring Registry Code	string	required	required	ZZ
transferring Registry Account Type	int	optional	required	120
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Code	string	required	required	ZZ

(cont.)



**Figure L28: Net Source Cancellation Transaction (cont.)**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
acquiring Registry Account Type	int	optional	required to be Account Type 210 for Cancellation in response to Notification Type 1	210
acquiring Registry Account Identifier	long	optional	required for Cancellation	41
notification Identifier	long	optional	required for Cancellation in response to Notification Type 1	331
proposal Unit Blocks	Transaction Unit Block[ ]	required	required	
unit Serial Block Start	long	required	required	1001
unit Serial Block End	long	required	required	1500
originating Registry Code	string	required	required	ZZ
unit Type	int	required	required	1
supp Unit Type	int	optional	not supplied	
original Commit Period	int	required	required	1
applicable Commit Period	int	required	required	1
LULUCF Activity	int	optional	required for RMUs and ERUs converted from RMUs.	
project Identifier	int	optional	required for ERUs and CERs.	
track	int	optional	required for ERUs	
block Role	string	optional	not supplied	
transferring Registry Account Type	int	optional	not supplied	
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
year in Commitment Period	int	optional	not supplied	
installation Identifier	long	optional	not supplied	
expiry Date	date	optional	required for ICERs, tCERs	

The ITL sends a notification update indicating the requirement has been fulfilled. The notification update's targetValue field will indicate the number of units remaining that must be cancelled before the notification's requirements are fulfilled, which in this case, is 0.

**Figure L29: Notification Update (Requirement Fulfilled)**

<b>ITLNoticeRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	The requirement to cancel units due to a Net Source of emissions has been fulfilled.
messageDate	dateTime	required	required	2005-11-14 12:05:34
notificationType	int	required	required	10
notificationIdentifier	long	required	required	331
notificationStatus	int	required	required	3
projectNumber	string	optional	not supplied	
unitType	int	optional	not supplied	

(cont.)

**Figure L29: Notification Update (Requirement Fulfilled) (cont.)**

<u>ITLNoticeRequest</u>	Type	WSDL	KP Requirement	KP Sample Data
targetValue	long	optional	required for Notification Type 9 with status of 3 "Complete"	0
targetDate	date	optional	not supplied	
LULUCFActivity	int	optional	not supplied	
commitPeriod	int	optional	not supplied	
actionDueDate	date	optional	not supplied	
unitBlockIdentifiers	arrayOfUnit Block Identifier	optional	not supplied	
unitSerialBlockStart	long	required	not supplied	
unitSerialBlockEnd	long	required	not supplied	
originatingRegistryCode	string	required	not supplied	

### 3.2 Non-compliance Cancellation (Type 2)

A Non-compliance Cancellation Notification is sent to a registry when its Party has been found by the Compliance Committee to be in non-compliance with its emissions targets in the previous Commitment Period. The registry is required to cancel within 30 days the number of units specified by the target value in the notification. The units must be valid for the subsequent Commitment Period and must be transferred to a Non-compliance Cancellation Account (Account Type Code 220). The original notification will specify the Commitment Period for which the units must be cancelled. The notification's targetValue field will indicate the number of units held by the registry that are affected by this notification. The cancellation transaction(s) must reference the original Notification ID. Periodically, or once the cancellation requirement has been fulfilled, the ITL will send a Notification Update.

**Figure L30: Non-compliance Cancellation Notification**

<u>ITLNoticeRequest</u>	Type	WSDL	KP Requirement	KP Sample Data
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	The Compliance Committee has determined that the Party is in non-compliance with its emissions target under Articles 3.7 and 3.8 of the Kyoto Protocol. Units must be cancelled in order to bring the Party into compliance.
messageDate	dateTime	required	required	2005-10-30 11:05:34
notificationType	int	required	required	2
notificationIdentifier	long	required	required	332
notificationStatus	int	required	required	1
projectNumber	string	optional	not supplied	
unitType	int	optional	not supplied	
targetValue	long	optional	required for Notification Type 2	100

(cont.)

**Figure L30: Non-compliance Cancellation Notification (cont.)**

<b>ITLNoticeRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
targetDate	date	optional	not supplied	
LULUCFActivity	int	optional	not supplied	
commitPeriod	int	optional	required for Notification Type 2	2
actionDueDate	date	optional	required for Notification Type 2	2005-11-30
unitBlockIdentifiers	ArrayOfUnit Block Identifier	optional	not supplied	
unitSerialBlockStart	long	required	not supplied	
unitSerialBlockEnd	long	required	not supplied	
originatingRegistryCode	String	required	not supplied	

The registry initiates a cancellation transaction to cancel some of the required number of units. The units are transferred to a Non-compliance Cancellation account (Account Type Code 220).

**Figure L31: Non-compliance Cancellation Transaction**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ZZ
to	string	required	required	ITL
major Version	int	required	required	1
minor Version	int	required	required	1
proposed Transaction	Proposal Transaction	required	required	
transaction Identifier	string	required	required	ZZ1021
transaction Type	int	required	required	4
supp Transaction Type	int	optional	not supplied	
transferring Registry Code	string	required	required	ZZ
transferring Registry Account Type	int	optional	required	120
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Code	string	required	required	ZZ
acquiring Registry Account Type	int	optional	required to be Account Type 220 for Cancellation in response to Notification Type 2	220
acquiring Registry Account Identifier	long	optional	required for Cancellation	42
notification Identifier	long	optional	required for Cancellation in response to Notification Type 2	332
proposal Unit Blocks	Transaction Unit Block[ ]	required	required	
unit Serial Block Start	long	required	required	1501
unit Serial Block End	long	required	required	1550
originating Registry Code	string	required	required	ZZ
unit Type	int	required	required	1
supp Unit Type	int	optional	not supplied	
original Commit Period	int	required	required	2
applicable Commit Period	int	required	required	2
LULUCF Activity	int	optional	required for RMUs and ERUs converted from RMUs.	
project Identifier	int	optional	required for ERUs and CERs.	
track	int	optional	required for ERUs	

(cont.)

**Figure L31: Non-compliance Cancellation Transaction (cont.)**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
block Role	string	optional	not supplied	
transferring Registry Account Type	int	optional	not supplied	
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
year in Commitment Period	int	optional	not supplied	
installation Identifier	long	optional	not supplied	
expiry Date	date	optional	required for ICERs, tCERs	

The ITL sends a notification indicating the progress toward the requirement. The notification update's targetValue field will indicate the number of units remaining that must be cancelled before the notification's requirements are fulfilled, which in this case, is 50.

**Figure L32: Notification Update (Progress Update)**

<b>ITLNoticeRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	The Compliance Committee has determined that the Party is in non-compliance with its emissions target under Articles 3.7 and 3.8 of the Kyoto Protocol. Units must be cancelled in order to bring the Party into compliance.
messageDate	dateTime	required	required	2005-11-08 12:05:34
notificationType	int	required	required	10
notificationIdentifier	long	required	required	332
notificationStatus	int	required	required	2
projectNumber	string	optional	not supplied	
unitType	int	optional	not supplied	
targetValue	long	optional	required for Notification Type 9 in response to Notification Type 2	50
targetDate	date	optional	not supplied	
LULUCFActivity	int	optional	not supplied	
commitPeriod	int	optional	not supplied	
actionDueDate	date	optional	required for Notification Type 9 with status of 2 "Incomplete"	2005-11-30
unitBlockIdentifiers	ArrayOfUnit Block Identifier	optional	not supplied	
unitSerialBlockStart	long	required	not supplied	
unitSerialBlockEnd	long	required	not supplied	
originatingRegistryCode	string	required	not supplied	

The registry initiates another cancellation transaction to complete the requirement.

**Figure L33: Non-compliance Cancellation Transaction**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ZZ
to	string	required	required	ITL
major Version	int	required	required	1
minor Version	int	required	required	1
proposed Transaction	Proposal Transaction	required	required	
transaction Identifier	string	required	required	ZZ1022
transaction Type	int	required	required	4
supp Transaction Type	int	optional	not supplied	
transferring Registry Code	string	required	required	ZZ
transferring Registry Account Type	int	optional	required	120
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Code	string	required	required	ZZ
acquiring Registry Account Type	int	optional	required to be Account Type 220 for Cancellation in response to Notification Type 2	220
acquiring Registry Account Identifier	long	optional	required for Cancellation	42
notification Identifier	long	required	required for Cancellation in response to Notification Type 2	332
proposal Unit Blocks	Transaction Unit Block[ ]	required	required	
unit Serial Block Start	long	required	required	1551
unit Serial Block End	long	required	required	1600
originating Registry Code	string	required	required	ZZ
unit Type	int	required	required	1
supp Unit Type	int	optional	not supplied	
original Commit Period	int	required	required	2
applicable Commit Period	int	required	required	2
LULUCF Activity	int	optional	required for RMUs, ERUs converted from RMUs, ICERs and tCERs	
project Identifier	int	optional	required for ERUs, CERs, ICERs and tCERs	
track	int	optional	required for ERUs	
block Role	string	optional	not supplied	
transferring Registry Account Type	int	optional	not supplied	
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
year in Commitment Period	int	optional	not supplied	
installation Identifier	long	optional	not supplied	
expiry Date	date	optional	required for ICERs, tCERs	

The ITL sends a notification update to indicate the requirement has been fulfilled. The notification update's targetValue field will indicate the number of units remaining that must be cancelled before the notification's requirements are fulfilled, which in this case, is 0.

**Figure L34: Notification Update (Requirement Fulfilled)**

<u>ITLNoticeRequest</u>	Type	WSDL	KP Requirement	KP Sample Data
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	The requirement to cancel units due to non-compliance has been fulfilled.
messageDate	dateTime	required	required	2005-11-14 12:05:34
notificationType	int	required	required	10
notificationIdentifier	long	required	required	332
notificationStatus	int	required	required	3
projectNumber	string	optional	not supplied	
unitType	int	optional	not supplied	
targetValue	long	optional	required for Notification Type 9 with status of 3 "Complete"	0
targetDate	date	optional	not supplied	
LULUCFActivity	int	optional	not supplied	
commitPeriod	int	optional	not supplied	
actionDueDate	date	optional	not supplied	
unitBlockIdentifiers	ArrayOfUnit Block Identifier	optional	not supplied	
unitSerialBlockStart	long	required	not supplied	
unitSerialBlockEnd	long	required	not supplied	
originatingRegistryCode	string	required	not supplied	

### 3.3 Impending tCER or ICER Expiry (Type 3)

The ITL will notify each national registry of the unit blocks of any tCERs held in retirement and tCER replacement accounts or ICERs held in a retirement account that are to expire within 30 days. The notification indicates that the specified tCERs or ICERs are to be replaced before their expiry dates. The registry will initiate replacement transactions that reference the Notification ID sent by the ITL. Periodically, or once the replacement requirement has been fulfilled, the ITL will send a Notification Update.

**Figure L35: Impending Expiry Notification**

<u>ITLNoticeRequest</u>	Type	WSDL	KP Requirement	KP Sample Data
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	tCERs and/or ICERs held by this registry will expire within 30 days. The tCERs and ICERs specified in this message must be replaced or cancelled before they expire.

(cont.)

**Figure L35: Impending Expiry Notification (cont.)**

<b>ITLNoticeRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
messageDate	dateTime	required	required	2015-10-30 11:05:34
notificationType	int	required	required	3
notificationIdentifier	long	required	required	333
notificationStatus	int	required	required	1
projectNumber	string	optional	not supplied	
unitType	int	optional	not supplied	
targetValue	long	optional	not supplied	
targetDate	date	optional	not supplied	
LULUCFActivity	int	optional	not supplied	
commitPeriod	int	optional	not supplied	
actionDueDate	date	optional	required	2015-11-30
unitBlockIdentifiers	ArrayOfUnit Block Identifier	optional	required for Notification Type 3	
unitSerialBlockStart	long	required	required	8101
unitSerialBlockEnd	long	required	required	8200
originatingRegistryCode	string	required	required	YY

The registry initiates a replacement transaction. Note that the replacement blocks (the AAUs) are listed first and that the blocks being replaced (the ICERs) are listed last and indicated by a block role of 'REP.'

**Figure L36: Replacement Transaction**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ZZ
to	string	required	required	ITL
major Version	int	required	required	1
minor Version	int	required	required	1
proposed Transaction	Proposal Transaction	required	required	
transaction Identifier	string	required	required	ZZ103
transaction Type	int	required	required	6
supp Transaction Type	int	optional	not supplied	
transferring Registry Code	string	required	required	ZZ
transferring Registry Account Type	int	optional	required	100
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Code	string	required	required	ZZ
acquiring Registry Account Type	int	optional	required	421
acquiring Registry Account Identifier	long	optional	required	3000
notification Identifier	long	optional	required for Replacement in response to Notification Type 3	333
proposal Unit Blocks	Transaction Unit Block[ ]	required	required	
<b>First Block(s) – Replacing Units</b>				
unit Serial Block Start	long	required	required	1
unit Serial Block End	long	required	required	99
originating Party Code	string	required	required	ZZ
unit Type	int	required	required	1
supp Unit Type	int	optional	not supplied	
original Commit Period	int	required	required	1

(cont.)

**Figure L36: Replacement Transaction (cont.)**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
applicable Commit Period	int	required	required	1
LULUCF Activity	int	optional	required for RMUs, ERUs converted from RMUs, ICERs and tCERs	
project Identifier	int	optional	required for ERUs, CERs, ICERs and tCERs	
track	int	optional	required for ERUs	
block Role	string	optional	not supplied	
transferring Registry Account Type	int	optional	not supplied	
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
year in Commitment Period	int	optional	not supplied	
installation Identifier	long	optional	not supplied	
expiry Date	date	optional	required for ICERs and tCERs	
<b>Last Block(s) – Units to be replaced</b>				
unit Serial Block Start	long	required	required	8101
unit Serial Block End	long	required	required	8200
originating Party Code	string	required	required	YY
unit Type	int	required	required	7
supp Unit Type	int	optional	not supplied	
original Commit Period	int	required	required	1
applicable Commit Period	int	required	required	1
LULUCF Activity	int	optional	required for RMUs, ERUs converted from RMUs, ICERs and tCERs	1
project Identifier	int	optional	required for ERUs, CERs, ICERs and tCERs	5
track	int	optional	required for ERUs	
block Role	string	optional	required	REP
transferring Registry Account Type	int	optional	not supplied	
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
year in Commitment Period	int	optional	not supplied	
installation Identifier	long	optional	not supplied	
expiry Date	date	Optional	required for ICERs and tCERs	2015-11-30

The ITL sends a notification indicating the requirement has been fulfilled.

**Figure L37: Notification Update (Requirement Fulfilled)**

<b>ITLNoticeRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	The requirement to replace or cancel tCERS or ICERS about to expire has been fulfilled.

(cont.)



**Figure L37: Notification Update (Requirement Fulfilled) (cont.)**

<u>ITLNoticeRequest</u>	Type	WSDL	KP Requirement	KP Sample Data
messageDate	dateTime	required	required	2015-11-14 12:05:34
notificationType	int	required	required	10
notificationIdentifier	long	required	required	333
notificationStatus	int	required	required	3
projectNumber	string	optional	not supplied	
unitType	int	optional	not supplied	
targetValue	long	optional	required for Notification Type 9 with status of 3 "Complete"	0
targetDate	date	optional	not supplied	
LULUCFActivity	int	optional	not supplied	
commitPeriod	int	optional	not supplied	
actionDueDate	date	optional	not supplied	
unitBlockIdentifiers	ArrayOfUnit Block Identifier	optional	not supplied	
unitSerialBlockStart	long	required	not supplied	
unitSerialBlockEnd	long	required	not supplied	
originatingRegistryCode	string	required	not supplied	

### 3.4 Reversal of Storage for CDM Project (Type 4)

In the event that a reversal of storage of greenhouse gases occurs within the boundaries of a CDM Project for which ICERs have been issued, the ITL will send this notification type to all registries with holdings of ICERs from this project. This notification shall indicate that all ICERs associated with the project will be prevented from being transferred to holding and retirement accounts until the appropriate number of ICERs has been replaced (or cancelled). Each registry is required to initiate transfers into a ICER Replacement Account for Reversal of Storage (Account Type Code 422), or a Mandatory Cancellation Account (Account Type Code 250), and must reference the Notification ID within the transaction. Periodically, or once the replacement or cancellation requirement has been fulfilled, the ITL will send a Notification Update.

**Figure L38: Reversal of Storage for CDM Project Notification**

<u>ITLNoticeRequest</u>	Type	WSDL	KP Requirement	KP Sample Data
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	A reversal of storage of greenhouse gases has occurred at a CDM Project and all transactions involving units associated with that project are suspended until the specified number of units are replaced or cancelled.
messageDate	dateTime	required	required	2005-10-30 11:05:34
notificationType	int	required	required	4
notificationIdentifier	long	required	required	334

(cont.)

**Figure L38: Reversal of Storage for CDM Project Notification (cont.)**

<b>ITLNoticeRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
notificationStatus	int	required	required	1
projectNumber	string	optional	required for Notification Type 4	YY225
unitType	int	optional	not supplied	
targetValue	long	optional	required for Notification Type 4	250
targetDate	date	optional	not supplied	
LULUCFActivity	int	optional	not supplied	
commitPeriod	int	optional	not supplied	
actionDueDate	date	optional	required	2005-11-30
unitBlockIdentifiers	ArrayOfUnit Block Identifier	optional	not supplied	
unitSerialBlockStart	long	required	not supplied	
unitSerialBlockEnd	long	required	not supplied	
originatingRegistryCode	string	required	not supplied	

Registry submits a cancellation transaction for 100 affected units.

**Figure L39: Reversal of Storage Cancellation Transaction**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ZZ
to	string	required	required	ITL
major Version	int	required	required	1
minor Version	int	required	required	1
proposed Transaction	Proposal Transaction	required	required	
transaction Identifier	string	required	required	ZZ1041
transaction Type	int	required	required	4
supp Transaction Type	int	optional	not supplied	
transferring Registry Code	string	required	required	ZZ
transferring Registry Account Type	int	optional	required	120
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Code	string	required	required	ZZ
acquiring Registry Account Type	int	optional	required for Cancellation	230
acquiring Registry Account Identifier	long	optional	required for Cancellation	40
notification Identifier	long	optional	required for Cancellation in response to Notification Type 4	334
proposal Unit Blocks	Transaction Unit Block[ ]	required	required	
unit Serial Block Start	long	required	required	1501
unit Serial Block End	long	required	required	1600
originating Registry Code	string	required	required	YY
unit Type	int	required	required	7
supp Unit Type	int	optional	not supplied	
original Commit Period	int	required	required	1
applicable Commit Period	int	required	required	1
LULUCF Activity	int	optional	required for RMUs, ERUs converted from RMUs, ICERs and tCERs	1

(cont.)

**Figure L39: Reversal of Storage Cancellation Transaction (cont.)**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
project Identifier	int	optional	required for ERUs, CERs, ICERs and tCERs	225
track	int	optional	required for ERUs	
block Role	string	optional	not supplied	
transferring Registry Account Type	int	optional	not supplied	
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
year in Commitment Period	int	optional	not supplied	
installation Identifier	long	optional	not supplied	
expiry Date	date	optional	required for ICERs, tCERs	2015-11-01

The ITL sends a Notification Update indicating that 150 units still must be cancelled or replaced.

**Figure L40: Notification Update (Progress Update)**

<b>ITLNoticeRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	A reversal of storage of greenhouse gases has occurred at a CDM Project and all transactions involving units associated with that project are suspended until the specified number of units are replaced or cancelled.
messageDate	dateTime	required	required	2005-11-14 12:05:34
notificationType	int	required	required	10
notificationIdentifier	long	required	required	334
notificationStatus	int	required	required	2
projectNumber	string	optional	required for Notification Type 9 in response to Notification Type 4	YY225
unitType	int	optional	not supplied	
targetValue	long	optional	required for Notification Type 9 in response to Notification Type 4	150
targetDate	date	optional	not supplied	
LULUCFActivity	int	optional	not supplied	
commitPeriod	int	optional	not supplied	
actionDueDate	date	optional	required for Notification Type 9 with status of 2 "Incomplete"	2005-11-30
unitBlockIdentifiers	ArrayOfUnit Block Identifier	optional	not supplied	
unitSerialBlockStart	Long	required	not supplied	
unitSerialBlockEnd	Long	required	not supplied	
originatingRegistryCode	String	required	not supplied	

Registry submits a replacement transaction for 150 units. Replacing Units transferred to Replacement Account for Reversal of Storage (Account Type Code 422).

**Figure L41: Replacement Transaction**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ZZ
to	string	required	required	ITL
major Version	int	required	required	1
minor Version	int	required	required	1
proposed Transaction	Proposal Transaction	required	required	
transaction Identifier	string	required	required	ZZ1042
transaction Type	int	required	required	6
supp Transaction Type	int	optional	not supplied	
transferring Registry Code	string	required	required	ZZ
transferring Registry Account Type	int	optional	required	100
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Code	string	required	required	ZZ
acquiring Registry Account Type	int	optional	required	422
acquiring Registry Account Identifier	long	optional	required	3000
notification Identifier	long	optional	required for Replacement in response to Notification Type 4	334
proposal Unit Blocks	Transaction Unit Block[ ]	required	required	
<b>First Block(s) – Replacing Units</b>				
unit Serial Block Start	long	required	required	501
unit Serial Block End	long	required	required	650
originating Party Code	string	required	required	ZZ
unit Type	int	required	required	1
supp Unit Type	int	optional	not supplied	
original Commit Period	int	required	required	1
applicable Commit Period	int	required	required	1
LULUCF Activity	int	optional	required for RMUs, ERUs converted from RMUs, ICERs and tCERs	
project Identifier	int	optional	required for ERUs, CERs, ICERs and tCERs	
track	int	optional	required for ERUs	
block Role	string	optional	not supplied	
transferring Registry Account Type	int	optional	not supplied	
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
year in Commitment Period	int	optional	not supplied	
installation Identifier	long	optional	not supplied	
expiry Date	date	optional	required for ICERs and tCERs	
<b>Last Block(s) – Units to be replaced</b>				
unit Serial Block Start	long	required	Required	8201
unit Serial Block End	long	required	Required	8350
originating Party Code	string	required	Required	YY
unit Type	int	required	Required	7
supp Unit Type	int	optional	not supplied	
original Commit Period	int	required	Required	1
applicable Commit Period	int	required	Required	1

(cont.)

**Figure L41: Replacement Transaction (cont.)**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
LULUCF Activity	int	optional	required for RMUs, ERUs converted from RMUs, ICERs and tCERs	1
project Identifier	int	optional	required for ERUs, CERs, ICERs and tCERs	225
track	int	optional	required for ERUs	
block Role	string	optional	required	REP
transferring Registry Account Type	int	optional	not supplied	
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
year in Commitment Period	int	optional	not supplied	
installation Identifier	long	optional	not supplied	
expiry Date	date	optional	required for ICERs and tCERs	2015-11-01

The ITL sends a Notification Update indicating that the requirement has been fulfilled.

**Figure L42: Notification Update (Requirement Fulfilled)**

<b>ITLNoticeRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	The requirement to cancel or replace units due to a reversal of storage has been fulfilled.
messageDate	dateTime	required	required	2005-11-14 12:05:34
notificationType	int	required	required	10
notificationIdentifier	long	required	required	334
notificationStatus	int	required	required	3
projectNumber	string	optional	required for Notification Type 9 in response to Notification Type 4	YY225
unitType	Int	optional	not supplied	
targetValue	long	optional	required for Notification Type 9 with status of 3 "Complete"	0
targetDate	date	optional	not supplied	
LULUCFActivity	int	optional	not supplied	
commitPeriod	int	optional	not supplied	
actionDueDate	date	optional	not supplied	
unitBlockIdentifiers	ArrayOfUnit Block Identifier	optional	not supplied	
unitSerialBlockStart	long	required	not supplied	
unitSerialBlockEnd	long	required	not supplied	
originatingRegistryCode	string	required	not supplied	

### 3.5 Non-submission of Certification Report for CDM Project (Type 5)

If the designated operational entities in a CDM Project, for which ICERs have been issued, have not submitted a certification report by the appointed time, the ITL will send this notification type to all registries with holdings of ICERs from this project. This notification shall indicate that all ICERs associated with the project will be prevented from being transferred, other than to a Mandatory Cancellation Account (Account Type Code 250). Each registry is required to initiate transfers into a ICER Replacement Account for Non-submission of Certification Report (Account Type Code 423), or a Mandatory Cancellation Account, and must reference the Notification ID. Periodically, or once the replacement requirement has been fulfilled, the ITL will send a Notification Update.

**Figure L43: Non-submission of Certification Report for CDM Project Notification**

<u>ITLNoticeRequest</u>	Type	WSDL	KP Requirement	KP Sample Data
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	The CDM Executive Board has not received a Certification Report, and it has requested that all ICERs generated by the specified CDM Project be prevented from being transferred (except to cancellation or replacement accounts).
messageDate	dateTime	required	required	2005-10-30 11:05:34
notificationType	int	required	required	5
notificationIdentifier	long	required	required	335
notificationStatus	int	required	required	1
projectNumber	string	optional	required for Notification Type 5	YY226
unitType	int	optional	not supplied	
targetValue	long	optional	required for Notification Type 5	200
targetDate	date	optional	not supplied	
LULUCFActivity	int	optional	not supplied	
commitPeriod	int	optional	not supplied	
actionDueDate	date	optional	not supplied	
unitBlockIdentifiers	ArrayOfUnit Block Identifier	optional	required for Notification Type 5	
unitSerialBlockStart	long	required	not supplied	
unitSerialBlockEnd	long	required	not supplied	
originatingRegistryCode	string	required	not supplied	

Registry submits a cancellation transaction for the affected units.

**Figure L44: Non-submission of Certification Report Cancellation Transaction**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ZZ
to	string	required	required	ITL
major Version	int	required	required	1
minor Version	int	required	required	1
proposed Transaction	Proposal Transaction	required	required	
transaction Identifier	string	required	required	ZZ105
transaction Type	int	required	required	4
supp Transaction Type	int	optional	not supplied	
transferring Registry Code	string	required	required	ZZ
transferring Registry Account Type	int	optional	required	100
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Code	string	required	required	ZZ
acquiring Registry Account Type	int	optional	required for Cancellation	230
acquiring Registry Account Identifier	long	optional	required for Cancellation	40
notification Identifier	long	optional	required for Cancellation in response to Notification Type 5	335
proposal Unit Blocks	Transaction Unit Block[ ]	required	required	
unit Serial Block Start	long	required	required	401
unit Serial Block End	long	required	required	450
originating Registry Code	string	required	required	YY
unit Type	int	required	required	7
supp Unit Type	int	optional	not supplied	
original Commit Period	int	required	required	1
applicable Commit Period	int	required	required	1
LULUCF Activity	int	optional	required for RMUs, ERUs converted from RMUs, ICERs and tCERs	1
project Identifier	int	optional	required for ERUs, CERs, ICERs and tCERs	226
track	int	optional	not supplied	
block Role	string	optional	not supplied	
transferring Registry Account Type	int	optional	not supplied	
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
year in Commitment Period	int	optional	not supplied	
installation Identifier	long	optional	not supplied	
expiry Date	date	optional	required for ICERs and tCERs	2015-11-01
unit Serial Block Start	long	required	required	501
unit Serial Block End	long	required	required	550
originating Registry Code	string	required	required	YY
unit Type	int	required	required	7
supp Unit Type	int	optional	not supplied	
original Commit Period	int	required	required	1
applicable Commit Period	int	required	required	1
LULUCF Activity	int	optional	required for RMUs, ERUs converted from RMUs, ICERs and tCERs	1
project Identifier	int	optional	required for ERUs, CERs, ICERs and tCERs	226
track	int	optional	not supplied	
block Role	string	optional	not supplied	
transferring Registry Account Type	int	optional	not supplied	

(cont.)

**Figure L44: Non-submission of Certification Report Cancellation Transaction (cont.)**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
year in Commitment Period	int	optional	not supplied	
installation Identifier	long	optional	not supplied	
expiry Date	date	optional	required for ICERs and tCERs	2015-11-01
unit Serial Block Start	long	required	required	551
unit Serial Block End	long	required	required	650
originating Registry Code	string	required	required	YY
unit Type	int	required	required	7
supp Unit Type	int	optional	not supplied	
original Commit Period	int	required	not supplied	1
applicable Commit Period	int	required	required	1
LULUCF Activity	int	optional	required for RMUs, ERUs converted from RMUs, ICERs and tCERs	1
project Identifier	int	optional	required for ERUs, CERs, ICERs and tCERs	226
track	int	optional	not supplied	
block Role	string	optional	not supplied	
transferring Registry Account Type	int	optional	not supplied	
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
year in Commitment Period	int	optional	not supplied	
installation Identifier	long	optional	not supplied	
expiry Date	date	optional	required for ICERs and tCERs	2015-11-01

The ITL sends a Notification Update indicating that the requirement has been fulfilled.

**Figure L45: Notification Update (Requirement Fulfilled)**

<b>ITLNoticeRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	The requirement to cancel or replace units due to the non-submission of a certification report has been fulfilled.
messageDate	dateTime	required	required	2005-11-14 12:05:34
notificationType	int	required	required	10
notificationIdentifier	long	required	required	335
notificationStatus	int	required	required	3
projectNumber	string	optional	required for Notification Type 9 in response to Notification Type 5	YY226
unitType	int	optional	not supplied	

(cont.)



**Figure L45: Notification Update (Requirement Fulfilled) (cont.)**

<b>ITLNoticeRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
targetValue	long	optional	required for Notification Type 9 with status of 3 "Complete"	0
targetDate	date	optional	not supplied	
LULUCFActivity	int	optional	not supplied	
commitPeriod	int	optional	not supplied	
actionDueDate	date	optional	not supplied	
unitBlockIdentifiers	ArrayOfUnit Block Identifier	optional	not supplied	
unitSerialBlockStart	long	required	not supplied	
unitSerialBlockEnd	long	required	not supplied	
originatingRegistryCode	string	required	not supplied	

### 3.6 Excess Issuance for CDM Project (Type 6)

If excess CERs (tCERs or ICERs) are issued for a CDM Project, the CDM Executive Board may require the designated operational entity responsible for verifying the project to transfer, within 30 days, a specified number of units to the Excess Issuance Cancellation Account (Account Type Code 240) in the CDM registry. This notification will be sent to all registries. The cancellation transactions initiated by the designated operational entity must reference the Notification ID so that the ITL may track compliance with the notification. No notification updates are sent via the Web service. For affected units held in national registries, the cancellation is achieved via an External Transfer (Transaction Type Code 3) to the Excess Issuance Cancellation Account in the CDM registry. For affected units held in the CDM Registry, the cancellation is achieved via a cancellation transaction (Transaction Type Code 4).

**Figure L46: Excess Issuance for CDM Project Cancellation Transaction**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ZZ
to	string	required	required	ITL
major Version	int	required	required	1
minor Version	int	required	required	1
proposed Transaction	Proposal Transaction	required	required	
transaction Identifier	string	required	required	ZZ106
transaction Type	int	required	required	3
supp Transaction Type	int	optional	not supplied	
transferring Registry Code	string	required	required	ZZ
transferring Registry Account Type	int	optional	required	120
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Code	string	required	required	CDM
acquiring Registry Account Type	int	optional	required for Cancellation	240
acquiring Registry Account Identifier	long	optional	required for Cancellation	44
notification Identifier	long	optional	required for Cancellation in response to Notification Type 6	336
proposal Unit Blocks	Transaction Unit Block[ ]	required	required	
unit Serial Block Start	long	required	required	7001
unit Serial Block End	long	required	required	7200
originating Registry Code	string	required	required	YY

(cont.)

**Figure L46: Excess Issuance for CDM Project Cancellation Transaction (cont.)**

<u>ProposalRequest</u>	Type	WSDL	KP Requirement	KP Sample Data
unit Type	int	required	required	1
supp Unit Type	int	optional	not supplied	
original Commit Period	int	required	required	1
applicable Commit Period	int	required	required	1
LULUCF Activity	int	optional	required for RMUs, ERUs converted from RMUs, ICERs and tCERs	
project Identifier	int	optional	required for ERUs, CERs, ICERs and tCERs	6
track	int	optional	required for ERUs	
block Role	string	optional	not supplied	
transferring Registry Account Type	int	optional	not supplied	
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
year in Commitment Period	int	optional	not supplied	
installation Identifier	long	optional	not supplied	
expiry Date	date	optional	required for ICERs and tCERs	

### 3.7 Commitment Period Reserve Change (Type 7)

Where an upward revision of a Party's CPR level raises it above the registry's current holdings of units, or where the cancellation or replacement of units within the registry reduces unit holdings below the CPR level, the ITL will notify the Party of the quantity of units by which it is required to increase its unit holdings within 30 days. The relevant Commitment Period will be specified in this notification. The registry will acquire sufficient units with the relevant Applicable Commitment Period identifier from other registries to meet this requirement. Since transactions are submitted by the transferring, not acquiring, registry, these transactions will not reference any Notification ID. Periodically, or once the acquisitions requirement has been fulfilled, the ITL will send a Notification Update. The ITL will check each registry's unit holdings against its CPR level on a daily basis.

**Figure L47: Commitment Period Reserve Change**

<u>ITLNoticeRequest</u>	Type	WSDL	KP Requirement	KP Sample Data
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	The CPR level for this registry has been raised and it is necessary for the registry to acquire the number of units specified in this message within 30 days.
messageDate	dateTime	required	required	2005-10-30 11:05:34
notificationType	int	required	required	7
notificationIdentifier	long	required	required	337
notificationStatus	int	required	required	1
projectNumber	string	optional	not supplied	
unitType	Int	optional	not supplied	

(cont.)

**Figure L47: Commitment Period Reserve Change (cont.)**

<b>ITLNoticeRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
targetValue	Long	optional	required for Notification Type 7	120
targetDate	date	optional	not supplied	
LULUCFActivity	int	optional	not supplied	
commitPeriod	int	optional	required for Notification Type 7	1
actionDueDate	Date	optional	required	2005-11-30
unitBlockIdentifiers	ArrayOfUnit Block Identifier	optional	not supplied	
unitSerialBlockStart	Long	required	not supplied	
unitSerialBlockEnd	Long	required	not supplied	
originatingRegistryCode	String	required	not supplied	

This is an External Transaction proposal from a third Party in which the registry in question acquires enough units to exceed the CPR.

**Figure L48: External Transfer Transaction**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	CL
to	string	required	required	ITL
major Version	int	required	required	1
minor Version	int	required	required	1
proposed Transaction	Proposal Transaction	required	required	
transaction Identifier	string	required	required	CL107
transaction Type	int	required	required	3
supp Transaction Type	int	optional	not supplied	
transferring Registry Code	string	required	required	CL
transferring Registry Account Type	int	optional	required	100
transferring Registry Account Identifier	long	optional	required	50
acquiring Registry Code	string	required	required	ZZ
acquiring Registry Account Type	int	optional	required	100
acquiring Registry Account Identifier	long	optional	required	658
notification Identifier	long	optional	not supplied	
proposal Unit Blocks	Transaction Unit Block[ ]	required	required	
unit Serial Block Start	long	required	required	9001
unit Serial Block End	long	required	required	9120
originating Registry Code	string	required	required	CL
unit Type	int	required	required	1
supp Unit Type	int	optional	not supplied	
original Commit Period	int	required	required	1
applicable Commit Period	int	required	required	1
LULUCF Activity	int	optional	required for RMUs, ERUs converted from RMUs, ICERs and tCERs	
project Identifier	int	optional	required for ERUs, CERs, ICERs and tCERs	
track	int	optional	required for ERUs	
block Role	string	optional	not supplied	
transferring Registry Account Type	int	optional	not supplied	
transferring Registry Account Identifier	long	optional	not supplied	

(cont.)

**Figure L48: External Transfer Transaction (cont.)**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
year in Commitment Period	int	optional	not supplied	
installation Identifier	long	optional	not supplied	
expiry Date	date	optional	not supplied	

The ITL sends a Notification Update indicating that the requirement has been fulfilled.

**Figure L49: Notification Update (Requirement Fulfilled)**

<b>ITLNoticeRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	The requirement to meet the CPR limit has been fulfilled.
messageDate	dateTime	required	required	2005-11-14 12:05:34
notificationType	int	required	required	10
notificationIdentifier	long	required	required	337
notificationStatus	int	required	required	3
projectNumber	string	optional	not supplied	
unitType	int	optional	not supplied	
targetValue	long	optional	required for Notification Type 9 with status of 3 "Complete"	0
targetDate	date	optional	not supplied	
LULUCFActivity	int	optional	not supplied	
commitPeriod	int	optional	not supplied	
actionDueDate	date	optional	not supplied	
unitBlockIdentifiers	ArrayOfUnit Block Identifier	optional	not supplied	
unitSerialBlockStart	long	required	not supplied	
unitSerialBlockEnd	long	required	not supplied	
originatingRegistryCode	string	required	not supplied	

### 3.8 Unit Carry-over (Type 8)

After the end of the true-up period and after the Compliance Committee has completed its consideration of all information reviewed under Article 8 of the Kyoto Protocol, the ITL notifies each registry of:

- The number of units which the registry may carry-over within 30 days, and the Commitment Period to which they may be carried-over.

The registry will then initiate carry-over transactions, up to the limit specified in the notification, within 30 days. For all carry-over transactions, the transaction must contain the Notification ID.

**Figure L50: Unit Carry-over Notification**

<b>ITLNoticeRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	Due to the end of the Commitment Period, the units specified within this message may be carried-over to the subsequent Commitment Period within 30 days.
messageDate	dateTime	required	required	2005-10-30 11:05:34
notificationType	int	required	required	8
notificationIdentifier	long	required	required	338
notificationStatus	int	required	required	1
projectNumber	string	optional	not supplied	
unitType	int	optional	no supplied	
targetValue	long	optional	required for Notification Type 8	1000
targetDate	date	optional	not supplied	
LULUCFActivity	int	optional	not supplied	
commitPeriod	int	optional	required for Notification Type 8	2
actionDueDate	date	optional	required for Notification Type 8	2005-11-30
unitBlockIdentifiers	ArrayOfUnit Block Identifier	optional	not supplied	

The registry initiates a transaction to carry-over some of the outstanding units.

**Figure L51: Unit Carry-over Transaction**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ZZ
to	string	required	required	ITL
major Version	int	required	required	1
minor Version	int	required	required	1
proposed Transaction	Proposal Transaction	required	required	
transaction Identifier	string	required	required	ZZ1081
transaction Type	int	required	required	7
supp Transaction Type	int	optional	not supplied	
transferring Registry Code	string	required	required	ZZ
transferring Registry Account Type	int	required	required	100
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Code	string	required	required	ZZ
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
notification Identifier	long	optional	required for Unit Carry-over transaction	338
proposal Unit Blocks	Transaction Unit Block[ ]	required	required	
unit Serial Block Start	long	required	required	10001
unit Serial Block End	long	required	required	11000

(cont.)

Figure L51: Unit Carry-over Transaction (cont.)

ProposalRequest	Type	WSDL	KP Requirement	KP Sample Data
originating Registry Code	string	required	required	CL
unit Type	int	required	required	1
supp Unit Type	int	optional	not supplied	
original Commit Period	int	required	required	1
applicable Commit Period	int	required	required. Value is new commitment period	2
LULUCF Activity	int	optional	not supplied (unit types associated with LULUCFs are not eligible for carry-over)	
project Identifier	int	optional	required for ERUs and CERs (only ERUs converted from AAUs are eligible for carry-over)	
track	int	optional	required for ERUs (only ERUs converted from AAUs are eligible for carry-over)	
block Role	string	optional	not supplied	
transferring Registry Account Type	int	optional	not supplied	
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
year in Commitment Period	int	optional	not supplied	
installation Identifier	long	optional	not supplied	
expiry Date	date	optional	not supplied (ICERs and tCERs are not eligible for carry-over)	
unit Serial Block Start	long	required	required	13001
unit Serial Block End	long	required	required	14000
originating Registry Code	string	required	required	CL
unit Type	int	required	required	1
supp Unit Type	int	optional	not supplied	
original Commit Period	int	required	required	1
applicable Commit Period	int	required	required. Value is new commitment period	2
LULUCF Activity	int	optional	not supplied (unit types associated with LULUCFs are not eligible for carry-over)	
project Identifier	int	optional	required for ERUs and CERs (only ERUs converted from AAUs are eligible for carry-over)	
track	int	optional	required for ERUs (only ERUs converted from AAUs are eligible for carry-over)	
block Role	string	optional	not supplied	
transferring Registry Account Type	int	optional	not supplied	
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
year in Commitment Period	int	optional	not supplied	
installation Identifier	long	optional	not supplied	
expiry Date	date	optional	not supplied (ICERs and tCERs are not eligible for carry-over)	

**Figure L53: Notification Update (Requirement Fulfilled)**

<u>ITLNoticeRequest</u>	Type	WSDL	KP Requirement	KP Sample Data
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	The requirement to carry-over has been fulfilled.
messageDate	dateTime	required	required	2005-11-14 12:05:34
notificationType	int	required	required	10
notificationIdentifier	long	required	required	338
notificationStatus	int	required	required	3
projectNumber	string	optional	not supplied	
unitType	int	optional	required for Notification Type 9 in response to Notification Type 8	1
targetValue	long	optional	required for Notification Type 9 with status of 3 "Complete"	0
targetDate	date	optional	not supplied	
LULUCFActivity	int	optional	not supplied	
commitPeriod	int	optional	not supplied	
actionDueDate	date	optional	not supplied	
unitBlockIdentifiers	ArrayOfUnit Block Identifier	optional	not supplied	
unitSerialBlockStart	long	required	not supplied	
unitSerialBlockEnd	long	required	not supplied	
originatingRegistryCode	string	required	not supplied	

### 3.9 Expiry Date Change (Type 9)

The ITL will notify registries when an expiry date change is required for tCERs or ICERs. The notification will identify the unit blocks whose expiry date is changing by specifying the projectNumber for ICERs, and the commitPeriod for tCERs. The registry will initiate expiry date change transactions, providing reference to the notification identifier sent by the ITL so the ITL can track when the registry has completed the required transaction.

**Figure L54: Expiry Date Change Notification**

<u>ITLNoticeRequest</u>	Type	WSDL	KP Requirement	KP Sample Data
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	Due to a change in the crediting period for the relevant CDM Project, the units specified within this message must have their expiry date changed.
messageDate	dateTime	required	required	2005-10-30 11:05:34
notificationType	int	required	required	9

(cont.)

**Figure L54: Expiry Date Change Notification (cont.)**

<u>ITLNoticeRequest</u>	Type	WSDL	KP Requirement	KP Sample Data
notificationIdentifier	long	required	required	339
notificationStatus	int	required	required	1
projectNumber	string	optional	required for Notification Type 10 with unitType of 7 (ICER) not required when unitType is 6, (tCER)	
unitType	int	optional	required for Notification Type 10	6
targetValue	long	optional	required for Notification Type 10	500
targetDate	date	optional	required for Notification Type 10	2017-12-31
LULUCFActivity	int	optional	not supplied	
commitPeriod	int	optional	Required for Notification Type 10	1
actionDueDate	date	optional	required for Notification Type 10	2005-11-30
unitBlockIdentifiers	ArrayOfUnit Block Identifier	optional	not supplied	

Registry submits an expiry date change transaction for the affected units.

**Figure L55: Expiry Date Change Transaction**

<u>ProposalRequest</u>	Type	WSDL	KP Requirement	KP Sample Data
from	string	required	required	ZZ
to	string	required	required	ITL
major Version	int	required	required	1
minor Version	int	required	required	1
proposed Transaction	Proposal Transaction	required	required	
transaction Identifier	string	required	required	ZZ244
transaction Type	int	required	required	8
supp Transaction Type	int	optional	not supplied	
transferring Registry Code	string	required	required	NZ
transferring Registry Account Type	int	optional	required	100
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Code	string	required	required	NZ
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
notification Identifier	long	optional	not supplied	339
proposal Unit Blocks	Transaction Unit Block[ ]	required	required	
unit Serial Block Start	long	required	required	2001
unit Serial Block End	long	required	required	2500
originating Registry Code	string	required	required	ZZ
unit Type	int	required	required	6
supp Unit Type	int	optional	not supplied	
original Commit Period	int	required	required	1
applicable Commit Period	int	required	required	1
LULUCF Activity	int	optional	required (units are ICERs or tCERs)	1

(cont.)



**Figure L55: Expiry Date Change Transaction (cont.)**

<b>ProposalRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
project Identifier	int	optional	required (units are ICERs or tCERs)	5
track	int	optional	not supplied	
block Role	string	optional	not supplied	
transferring Registry Account Type	int	optional	not supplied	
transferring Registry Account Identifier	long	optional	not supplied	
acquiring Registry Account Type	int	optional	not supplied	
acquiring Registry Account Identifier	long	optional	not supplied	
year in Commitment Period	int	optional	not supplied	
installation Identifier	long	optional	not supplied	
expiry Date	date	optional	required. Value is new Expiry Date	2018-12-31

The ITL sends a Notification Update indicating that the Expiry Date Change requirement has been fulfilled.

**Figure L56: Notification Update (Requirement Fulfilled)**

<b>ITLNoticeRequest</b>	<b>Type</b>	<b>WSDL</b>	<b>KP Requirement</b>	<b>KP Sample Data</b>
from	string	required	required	ITL
to	string	required	required	ZZ
major Version	int	required	required	1
minor Version	int	required	required	1
messageContent	string	required	required	The requirement to change expiry dates has been fulfilled
messageDate	dateTime	required	required	2005-11-14 12:05:34
notificationType	int	required	required	10
notificationIdentifier	long	required	required	339
notificationStatus	int	required	required	3
projectNumber	string	optional	not supplied	
unitType	int	optional	required for Notification Type 9 in response to Notification Type 8	6
targetValue	long	optional	required for Notification Type 9 with status of 3 "Complete"	0
targetDate	date	optional	not supplied	
LULUCFActivity	int	optional	not supplied	
commitPeriod	int	optional	not supplied	
actionDueDate	date	optional	not supplied	
unitBlockIdentifiers	ArrayOfUnit Block Identifier	optional	not supplied	
unitSerialBlockStart	long	required	not supplied	
unitSerialBlockEnd	long	required	not supplied	
originatingRegistryCode	string	required	not supplied	

*[This page intentionally left blank.]*

---

## **23 Annex M**

### **EU-ETS Supplementary Scheme Web Service Documentation**

#### **1. Introduction**

This annex describes the standards for data exchange between EU-ETS registries and the CITL, a supplementary transaction log under the Kyoto Protocol. All communication within the Registry System is via the ITL, which acts as the communications hub. So in the case of communication between members of supplementary scheme and the associated transaction log, all messages go via the ITL. In the case of operations documented in this Annex, the ITL only performs sufficient processing to successfully pass on the authenticated message to the recipient.

The documents listed below specify how registries can communicate with the CITL and vice versa. These documents define the interfaces to the public web service methods on both the CITL, registries and ITL. These Web service methods are the portals through which data can be passed into an application.

In order to aid the understanding of non-technical persons, a summary of each operation is provided prior to the more technical WSDL documents. WSDL documents are designed to be "read" by computers and not humans. The Web Service Method Summaries below reduce the numerous cross-references and repetitive nature of the WSDL documents to a more understandable form. The summaries are still detailed and reflect the complex nature of the data exchange system.

## **2. Account Management Web Service**

The account management web service is provided for EU-ETS members to be able to communicate account management instructions to the CITL. It supports the:

1. Creation of accounts;
2. Modification of accounts;
3. Closure of accounts;
4. Update of verified emissions for an installation.

There is also an operation that allows the CITL to communicate the outcome of these requests back to the registry.

## 2.1 Account Management web service operations

### 2.1.1 CreateAccountRequest

**Figure M1: CreateAccountRequest**  
Role(s): CITL

<b><u>CreateAccountRequest</u></b>			
from	string		
to	string		
correlationID	long		
majorVersion	int		
minorVersion	int		
accounts	<b>CreatedAccount</b>		
	accountType	int	
	accountIdentifier	long	
	identifierInReg	string	
	commitPeriod	int	
	installation	<b>CreatedInstallation</b> Opt	
		installationIdentifier	long
		permitIdentifier	string
		name	string
		mainActivityType	int
		country	string
		postalCode	string
		city	string
		address1	string
		address2	string Opt
		parentCompany	string Opt
		subsidiaryCompany	string Opt
		EPERIdentification	string Opt
		latitude	double Opt
		longitude	double Opt
	contactPeople	<b>CreatedPeople</b>	
		relationshipCode	int
		personIdentifier	string
		firstName	string
		lastName	string
		country	string
		postalCode	string
		city	string
		address1	string
		address2	string Opt
		phoneNumber1	string
		phoneNumber2	string
		faxNumber	string
		email	string
peoples	<b>CreatedPeople</b>		
		relationshipCode	int
		personIdentifier	string
		firstName	string
		lastName	string
		country	string
		postalCode	string
		city	string
		address1	string
		address2	string Opt

(cont.)

**Figure M1: CreateAccount Request (cont.)**  
**Role(s): CITL**

	phoneNumber1	string	
	phoneNumber2	string	
	faxNumber	string	
	email	string	
<b><u>CreateAccountResponse</u></b>			
	resultIdentifier	int	
	responseCode	int	Opt

## 2.1.2 UpdateAccountRequest

**Figure M2: UpdateAccountRequest**  
Role(s): CITL

<b>UpdateAccountRequest</b>			
from		string	
to		string	
correlationID		long	
majorVersion		int	
minorVersion		int	
accounts	<b>UpdatedAccount</b>		
	accountIdentifier	long	
	identifierInReg	string	Opt
	installation	<b>UpdatedInstallation</b>	Opt
		permitIdentifier	string Opt
		name	string Opt
		mainActivityType	int Opt
		country	string Opt
		postalCode	string Opt
		city	string Opt
		address1	string Opt
		address2	string Opt
		parentCompany	string Opt
		subsidiaryCompany	string Opt
		EPERIdentification	string Opt
		latitude	double Opt
		longitude	double Opt
	contactPeople	<b>UpdatedPeople</b>	Opt
		action	int
		relationshipCode	int
		personIdentifier	string
		firstName	string Opt
		lastName	string Opt
		country	string Opt
		postalCode	string Opt
		city	string Opt
		address1	string Opt
		address2	string Opt
		phoneNumber1	string Opt
		phoneNumber2	string Opt
		faxNumber	string Opt
		email	string Opt
	peoples	<b>UpdatedPeople</b>	Opt
		action	int
		relationshipCode	int
		personIdentifier	string
		firstName	string Opt
		lastName	string Opt
		country	string Opt
		postalCode	string Opt
		city	string Opt
		address1	string Opt
		address2	string Opt
		phoneNumber1	string Opt
		phoneNumber2	string Opt
		faxNumber	string Opt
		email	string Opt
<b>UpdateAccountResponse</b>			
	resultIdentifier	int	
	responseCode	int	Opt

### 2.1.3 CloseAccountRequest

**Figure M3: CloseAccountRequest**  
Role(s): CITL

<u>CloseAccountRequest</u>			
from	string		
to	string		
correlationID	long		
majorVersion	int		
minorVersion	int		
accounts	<b>CloseAccount</b>		
	accountIdentifier	long	
<u>CloseAccountResponse</u>			
resultIdentifier	int		
responseCode	int		Opt

### 2.1.4 UpdateVerifiedEmissionsRequest

**Figure M4: UpdateVerifiedEmissionsRequest**  
Role(s): CITL

<u>UpdateVerifiedEmissionsRequest</u>			
from	string		
to	string		
correlationID	long		
majorVersion	int		
minorVersion	int		
verifiedEmissions	<b>VerifiedEmission</b>		
	yearInCommitPeriod		
	installations	<b>InstallationVerifiedEmission</b>	
	installationIdentifier	long	
	verifiedEmission	long	
<u>UpdateVerifiedEmissionsResponse</u>			
resultIdentifier	int		
responseCode	int		Opt



## 2.1.5 ReceiveAccountOperationOutcomeRequest

**Figure M5: ReceiveAccountOperationOutcomeRequest  
Role(s): Registry**

<u>ReceiveAccountOperationOutcomeRequest</u>															
from	string														
to	string														
correlationID	long														
majorVersion	int														
minorVersion	int														
outcome	int														
outcomeDetails	<b>OutcomeDetail</b>		Opt												
<table border="1"> <tr> <td>responseCode</td> <td>int</td> <td></td> <td></td> </tr> <tr> <td>accountIdentifier</td> <td>long</td> <td></td> <td>Opt</td> </tr> <tr> <td>installationIdentifier</td> <td>long</td> <td></td> <td>Opt</td> </tr> </table>				responseCode	int			accountIdentifier	long		Opt	installationIdentifier	long		Opt
responseCode	int														
accountIdentifier	long		Opt												
installationIdentifier	long		Opt												
<u>ReceiveAccountOperationOutcomeResponse</u>															
<table border="1"> <tr> <td>resultIdentifier</td> <td>int</td> <td></td> <td></td> </tr> <tr> <td>responseCode</td> <td>int</td> <td></td> <td>Opt</td> </tr> </table>				resultIdentifier	int			responseCode	int		Opt				
resultIdentifier	int														
responseCode	int		Opt												

## 2.2. Account Management WSDLs

### 2.2.1 Account Management Registry WSDL Description

```
<?xml version="1.0" encoding="UTF-8"?>

<definitions name="AccountManagement"
  targetNamespace="urn:KyotoProtocol:RegistrySystem:CITL:1.0:0.0"
  xmlns:tns="urn:KyotoProtocol:RegistrySystem:CITL:1.0:0.0"
  xmlns:typens="urn:KyotoProtocol:RegistrySystem:CITL:Types:1.0:0.0"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">

  <types>

    <schema targetNamespace="urn:KyotoProtocol:RegistrySystem:CITL:Types:1.0:0.0"
      xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
      xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
      xmlns="http://www.w3.org/2001/XMLSchema">

      <import namespace="http://schemas.xmlsoap.org/soap/encoding/" />

      <complexType name="OutcomeDetail">
        <sequence>
          <!-- An outcome detail contains a response code
            per account/installation where a check failed -->
          <element name="responseCode" type="int"/>
          <element name="accountIdentifier" minOccurs="0" type="long"/>
          <element name="installationIdentifier" minOccurs="0" type="long"/>
        </sequence>
      </complexType>

      <complexType name="ArrayOfOutcomeDetail">
        <complexContent>
          <restriction base="soapenc:Array">
            <attribute ref="soapenc:arrayType"
              wsdl:arrayType="typens:OutcomeDetail[]" />
          </restriction>
        </complexContent>
      </complexType>

      <complexType name="ReceiveAccountOperationOutcomeRequest">
        <sequence>
          <element name="from" type="string"/>
          <element name="to" type="string"/>
          <element name="correlationID" type="long"/>
          <element name="majorVersion" type="int"/>
          <element name="minorVersion" type="int"/>
          <element name="outcome" type="int"/>
          <element name="outcomeDetails" minOccurs="0"
            type="typens:ArrayOfOutcomeDetail"/>
        </sequence>
      </complexType>

      <complexType name="ReceiveAccountOperationOutcomeResponse">
        <sequence>
          <element name="resultIdentifier" type="int"/>
          <element name="responseCode" minOccurs="0" type="int"/>
        </sequence>
      </complexType>

    </schema>

  </types>

  <message name="receiveAccountOperationOutcomeRequest">
    <part name="ReceiveAccountOperationOutcomeRequest"
      type="typens:ReceiveAccountOperationOutcomeRequest" />
  </message>

  <message name="receiveAccountOperationOutcomeResponse">
    <part name="return"
      type="typens:ReceiveAccountOperationOutcomeResponse" />
  </message>

</definitions>
```

```

<portType name="AccountManagementPort">
  <operation name="receiveAccountOperationOutcome">
    <input message="tns:receiveAccountOperationOutcomeRequest"/>
    <output message="tns:receiveAccountOperationOutcomeResponse"/>
  </operation>
</portType>
<binding name="AccountManagementBinding" type="tns:AccountManagementPort">
  <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="rpc"/>
  <operation name="receiveAccountOperationOutcome">
    <soap:operation soapAction="" />
    <input>
      <soap:body use="encoded"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:KyotoProtocol:RegistrySystem:CITL:1.0:0.0"/>
    </input>
    <output>
      <soap:body use="encoded"
        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:KyotoProtocol:RegistrySystem:CITL:1.0:0.0"/>
    </output>
  </operation>
</binding>
<service name="AccountManagementService">
  <port name="AccountManagementPort" binding="tns:AccountManagementBinding">
    <soap:address location="http://REPLACE.WITH.ACTUAL.URL"/>
  </port>
</service>
</definitions>

```

## 2.2.2 Account Management Transaction Log WSDL Description

```

<?xml version="1.0" encoding="UTF-8"?>
<definitions name="AccountManagement"
  targetNamespace="urn:KyotoProtocol:RegistrySystem:CITL:1.0:0.0"
  xmlns:tns="urn:KyotoProtocol:RegistrySystem:CITL:1.0:0.0"
  xmlns:typens="urn:KyotoProtocol:RegistrySystem:CITL:Types:1.0:0.0"
  xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
  xmlns="http://schemas.xmlsoap.org/wsdl/">
  <types>
    <schema targetNamespace="urn:KyotoProtocol:RegistrySystem:CITL:Types:1.0:0.0"
      xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
      xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/"
      xmlns="http://www.w3.org/2001/XMLSchema">
      <import namespace="http://schemas.xmlsoap.org/soap/encoding/" />
      <complexType name="CreatedAccount">
        <sequence>
          <element name="accountType" type="int"/>
          <element name="accountIdentifier" type="long"/>
          <element name="identifierInReg" type="string"/>
          <element name="commitPeriod" type="int"/>
          <element name="installation" minOccurs="0"
            type="typens:CreatedInstallation"/>
          <element name="peoples" minOccurs="0"
            type="typens:ArrayOfCreatedPeople"/>
        </sequence>
      </complexType>
      <complexType name="ArrayOfCreatedAccount">
        <complexContent>
          <restriction base="soapenc:Array">
            <attribute ref="soapenc:arrayType"
              wsdl:arrayType="typens:CreatedAccount[]" />
          </restriction>
        </complexContent>
      </complexType>
    </schema>
  </types>

```

```

</complexType>

<complexType name="CreatedInstallation">
  <sequence>
    <element name="installationIdentifier" type="long"/>
    <element name="permitIdentifier" type="string"/>
    <element name="name" type="string"/>
    <element name="mainActivityType" type="int"/>
    <element name="country" type="string"/>
    <element name="postalCode" type="string"/>
    <element name="city" type="string"/>
    <element name="address1" type="string"/>
    <element name="address2" minOccurs="0" type="string"/>
    <element name="parentCompany" minOccurs="0" type="string"/>
    <element name="subsidiaryCompany" minOccurs="0" type="string"/>
    <element name="EPERIdentification" minOccurs="0" type="string"/>
    <element name="latitude" minOccurs="0" type="double"/>
    <element name="longitude" minOccurs="0" type="double"/>
    <element name="contactPeople" type="typens:CreatedPeople"/>
  </sequence>
</complexType>

<complexType name="CreatedPeople">
  <sequence>
    <element name="relationshipCode" type="int"/>
    <element name="personIdentifier" type="string"/>
    <element name="firstName" minOccurs="0" type="string"/>
    <!-- If the people is an organization (i.e. not a physical person)
         lastName should contain the organization's name -->
    <element name="lastName" type="string"/>
    <element name="country" type="string"/>
    <element name="postalCode" type="string"/>
    <element name="city" type="string"/>
    <element name="address1" type="string"/>
    <element name="address2" minOccurs="0" type="string"/>
    <element name="phoneNumber1" type="string"/>
    <element name="phoneNumber2" type="string"/>
    <element name="faxNumber" type="string"/>
    <element name="email" type="string"/>
  </sequence>
</complexType>

<complexType name="ArrayOfCreatedPeople">
  <complexContent>
    <restriction base="soapenc:Array">
      <attribute ref="soapenc:arrayType"
        wsdl:arrayType="typens:CreatedPeople[]"/>
    </restriction>
  </complexContent>
</complexType>

<complexType name="UpdatedAccount">
  <sequence>
    <element name="accountIdentifier" type="long"/>
    <element name="identifierInReg" minOccurs="0" type="string"/>
    <element name="installation" minOccurs="0"
      type="typens:UpdatedInstallation"/>
    <element name="peoples" minOccurs="0"
      type="typens:ArrayOfUpdatedPeople"/>
  </sequence>
</complexType>

<complexType name="ArrayOfUpdatedAccount">
  <complexContent>
    <restriction base="soapenc:Array">
      <attribute ref="soapenc:arrayType"
        wsdl:arrayType="typens:UpdatedAccount[]"/>
    </restriction>
  </complexContent>
</complexType>

<complexType name="UpdatedInstallation">
  <sequence>
    <element name="permitIdentifier" minOccurs="0" type="string"/>
    <element name="name" minOccurs="0" type="string"/>
    <element name="mainActivityType" minOccurs="0" type="int"/>
    <element name="country" minOccurs="0" type="string"/>
    <element name="postalCode" minOccurs="0" type="string"/>
  </sequence>

```

```

        <element name="city" minOccurs="0" type="string"/>
        <element name="address1" minOccurs="0" type="string"/>
        <element name="address2" minOccurs="0" type="string"/>
        <element name="parentCompany" minOccurs="0" type="string"/>
        <element name="subsidiaryCompany" minOccurs="0" type="string"/>
        <element name="EPERIdentification" minOccurs="0" type="string"/>
        <element name="latitude" minOccurs="0" type="double"/>
        <element name="longitude" minOccurs="0" type="double"/>
        <element name="contactPeople" minOccurs="0"
            type="typens:UpdatedPeople"/>
    </sequence>
</complexType>

<complexType name="UpdatedPeople">
    <sequence>
        <!-- action : add==1, update==2, delete==3 -->
        <element name="action" type="int"/>
        <element name="relationshipCode" type="int"/>
        <element name="personIdentifier" type="string"/>
        <element name="firstName" minOccurs="0" type="string"/>
        <element name="lastName" minOccurs="0" type="string"/>
        <element name="country" minOccurs="0" type="string"/>
        <element name="postalCode" minOccurs="0" type="string"/>
        <element name="city" minOccurs="0" type="string"/>
        <element name="address1" minOccurs="0" type="string"/>
        <element name="address2" minOccurs="0" type="string"/>
        <element name="phoneNumber1" minOccurs="0" type="string"/>
        <element name="phoneNumber2" minOccurs="0" type="string"/>
        <element name="faxNumber" minOccurs="0" type="string"/>
        <element name="email" minOccurs="0" type="string"/>
    </sequence>
</complexType>

<complexType name="ArrayOfUpdatedPeople">
    <complexContent>
        <restriction base="soapenc:Array">
            <attribute ref="soapenc:arrayType"
                wsdl:arrayType="typens:UpdatedPeople[]"/>
        </restriction>
    </complexContent>
</complexType>

<complexType name="ClosedAccount">
    <sequence>
        <element name="accountIdentifier" type="long"/>
    </sequence>
</complexType>

<complexType name="ArrayOfClosedAccount">
    <complexContent>
        <restriction base="soapenc:Array">
            <attribute ref="soapenc:arrayType"
                wsdl:arrayType="typens:ClosedAccount[]"/>
        </restriction>
    </complexContent>
</complexType>

<complexType name="VerifiedEmission">
    <sequence>
        <element name="yearInCommitPeriod" type="int"/>
        <element name="installations"
            type="typens:ArrayOfInstallationVerifiedEmission"/>
    </sequence>
</complexType>

<complexType name="ArrayOfVerifiedEmission">
    <complexContent>
        <restriction base="soapenc:Array">
            <attribute ref="soapenc:arrayType"
                wsdl:arrayType="typens:VerifiedEmission[]"/>
        </restriction>
    </complexContent>
</complexType>

<complexType name="InstallationVerifiedEmission">
    <sequence>
        <element name="installationIdentifier" type="long"/>
        <element name="verifiedEmission" type="long"/>
    </sequence>
</complexType>

```

```

    </sequence>
</complexType>

<complexType name="ArrayOfInstallationVerifiedEmission">
  <complexContent>
    <restriction base="soapenc:Array">
      <attribute ref="soapenc:arrayType"
        wsdl:arrayType="typens:InstallationVerifiedEmission[]" />
    </restriction>
  </complexContent>
</complexType>

<complexType name="CreateAccountRequest">
  <sequence>
    <element name="from" type="string"/>
    <element name="to" type="string"/>
    <element name="correlationID" type="long"/>
    <element name="majorVersion" type="int"/>
    <element name="minorVersion" type="int"/>

    <element name="accounts" type="typens:ArrayOfCreatedAccount"/>
  </sequence>
</complexType>

<complexType name="CreateAccountResponse">
  <sequence>
    <element name="resultIdentifier" type="int"/>
    <element name="responseCode" minOccurs="0" type="int"/>
  </sequence>
</complexType>

<complexType name="UpdateAccountRequest">
  <sequence>
    <element name="from" type="string"/>
    <element name="to" type="string"/>
    <element name="correlationID" type="long"/>
    <element name="majorVersion" type="int"/>
    <element name="minorVersion" type="int"/>
    <element name="accounts" type="typens:ArrayOfUpdatedAccount"/>
  </sequence>
</complexType>

<complexType name="UpdateAccountResponse">
  <sequence>
    <element name="resultIdentifier" type="int"/>
    <element name="responseCode" minOccurs="0" type="int"/>
  </sequence>
</complexType>

<complexType name="CloseAccountRequest">
  <sequence>
    <element name="from" type="string"/>
    <element name="to" type="string"/>
    <element name="correlationID" type="long"/>
    <element name="majorVersion" type="int"/>
    <element name="minorVersion" type="int"/>
    <element name="accounts" type="typens:ArrayOfClosedAccount"/>
  </sequence>
</complexType>

<complexType name="CloseAccountResponse">
  <sequence>
    <element name="resultIdentifier" type="int"/>
    <element name="responseCode" minOccurs="0" type="int"/>
  </sequence>
</complexType>

<complexType name="UpdateVerifiedEmissionsRequest">
  <sequence>
    <element name="from" type="string"/>
    <element name="to" type="string"/>
    <element name="correlationID" type="long"/>
    <element name="majorVersion" type="int"/>
    <element name="minorVersion" type="int"/>
    <element name="verifiedEmissions"
      type="typens:ArrayOfVerifiedEmission"/>
  </sequence>
</complexType>

```

```

        <complexType name="UpdateVerifiedEmissionsResponse">
            <sequence>
                <element name="resultIdentifier" type="int"/>
                <element name="responseCode" minOccurs="0" type="int"/>
            </sequence>
        </complexType>

    </schema>

</types>

<message name="createAccountRequest">
    <part name="CreateAccountRequest" type="typens:CreateAccountRequest"/>
</message>

<message name="createAccountResponse">
    <part name="return" type="typens:CreateAccountResponse"/>
</message>

<message name="updateAccountRequest">
    <part name="UpdateAccountRequest" type="typens:UpdateAccountRequest"/>
</message>

<message name="updateAccountResponse">
    <part name="return" type="typens:UpdateAccountResponse"/>
</message>

<message name="closeAccountRequest">
    <part name="CloseAccountRequest" type="typens:CloseAccountRequest"/>
</message>

<message name="closeAccountResponse">
    <part name="return" type="typens:CloseAccountResponse"/>
</message>

<message name="updateVerifiedEmissionsRequest">
    <part name="UpdateVerifiedEmissionsRequest"
        type="typens:UpdateVerifiedEmissionsRequest"/>
</message>

<message name="updateVerifiedEmissionsResponse">
    <part name="return"
        type="typens:UpdateVerifiedEmissionsResponse"/>
</message>

<portType name="AccountManagementPort">

    <operation name="createAccount">
        <input message="tns:createAccountRequest"/>
        <output message="tns:createAccountResponse"/>
    </operation>

    <operation name="updateAccount">
        <input message="tns:updateAccountRequest"/>
        <output message="tns:updateAccountResponse"/>
    </operation>

    <operation name="closeAccount">
        <input message="tns:closeAccountRequest"/>
        <output message="tns:closeAccountResponse"/>
    </operation>

    <operation name="updateVerifiedEmissions">
        <input message="tns:updateVerifiedEmissionsRequest"/>
        <output message="tns:updateVerifiedEmissionsResponse"/>
    </operation>

</portType>

<binding name="AccountManagementBinding" type="tns:AccountManagementPort">

    <soap:binding transport="http://schemas.xmlsoap.org/soap/http" style="rpc"/>

    <operation name="createAccount">
        <soap:operation soapAction="" />
        <input>
            <soap:body use="encoded"

```

```

        encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
        namespace="urn:KyotoProtocol:RegistrySystem:CITL:1.0:0.0"/>
    </input>
    <output>
        <soap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:KyotoProtocol:RegistrySystem:CITL:1.0:0.0"/>
    </output>
</operation>

<operation name="updateAccount">
    <soap:operation soapAction="" />
    <input>
        <soap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:KyotoProtocol:RegistrySystem:CITL:1.0:0.0"/>
    </input>
    <output>
        <soap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:KyotoProtocol:RegistrySystem:CITL:1.0:0.0"/>
    </output>
</operation>

<operation name="closeAccount">
    <soap:operation soapAction="" />
    <input>
        <soap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:KyotoProtocol:RegistrySystem:CITL:1.0:0.0"/>
    </input>
    <output>
        <soap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:KyotoProtocol:RegistrySystem:CITL:1.0:0.0"/>
    </output>
</operation>

<operation name="updateVerifiedEmissions">
    <soap:operation soapAction="" />
    <input>
        <soap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:KyotoProtocol:RegistrySystem:CITL:1.0:0.0"/>
    </input>
    <output>
        <soap:body use="encoded"
            encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
            namespace="urn:KyotoProtocol:RegistrySystem:CITL:1.0:0.0"/>
    </output>
</operation>

</binding>

<service name="AccountManagementService">
    <port name="AccountManagementPort" binding="tns:AccountManagementBinding">
        <soap:address location="http://REPLACE.WITH.ACTUAL.URL"/>
    </port>
</service>

</definitions>

```



## 2.3 Account Management Processes - transaction examples

### 2.4 Account Management Processes

Figure M6: Create Account Request

<u>CreateAccountRequest</u>	Type	optional / required	EU Sample Data	Comment
from	string	required	BE	
to	string	required	CTL	
correlationID	long	required	7	
majorVersion	int	required	1	
minorVersion	int	required	1	
accounts	CreatedAccount[ ]	required		
accountType	int	required	120	
accountIdentifier	long	required	456	
identifierInReg	string	required	ACCOUNT1	
commitPeriod	int	required		
installation	CreatedInstallation	optional		
installationIdentifier	long	required	999999999999999	
permitIdentifier	string	required	XCVXXCV-999	
name	string	required	Installation1	
mainActivityType	int	required	2	
country	string	required	BE	
postalCode	string	required	1200	
city	string	required	Brussels	
Address1	string	required	Avenue Ariane 7	
Address2	string	optional		
ParentCompany	string	optional		
SubsidiaryCompany	string	optional		
EPERIdentification	string	optional		
Latitude	double	optional		
longitude	double	optional		
ContactPeople	CreatedPeople	required	(see below)	
RelationshipCode	int	required	6	
PersonIdentifier	string	required	ZERZER-123	
FirstName	string	required	Jack	
LastName	string	required	Black	
Country	string	required	BE	
PostalCode	string	required	1000	
City	string	required	Bruxelles	
Address1	string	required	Rue Royale, 99	
Address2	string	optional		
PhoneNumber1	string	required	+32-2-9999999	
PhoneNumber2	string	required	+32-2-8888888	
FaxNumber	string	required	+32-2-7777777	
Email	string	required	<a href="mailto:jb@mlk.be">jb@mlk.be</a>	
People	CreatedPeople	required	(see below)	
RelationShipCode	int	required	1	
PersonallIdentifier	string	required	Gfds-g	
FirstName	string	required	John	
LastName	string	required	Watson	
Country	string	required	BE	
PostalCode	string	required	1000	
City	string	required	Bruxelles	
Address1	string	required	Rue Neuve, 88	
Address2	string	optional		
PhoneNumber1	string	required	+32-2-1234567	
PhoneNumber2	string	required	+32-2-9876543	
FaxNumber	string	required	+32-2-7539512	
E-Mail	string	required	<a href="mailto:jw@aaa.be">jw@aaa.be</a>	

**Figure M7: Create Account Response**

<b>CreateAccountResponse</b>	<b>Type</b>	<b>optional / required</b>	<b>EU Sample Data</b>	<b>Comment</b>
ResultIdentifier	int	required	1	1 if request accepted, 0 if request refused
ResponseCode	int	optional		

**Figure M8: Update Account Request**

<b>UpdateAccountRequest</b>	<b>Type</b>	<b>optional / required</b>	<b>EU Sample Data</b>	<b>Comment</b>
From	string	required	BE	
To	string	required	CTL	
CorrelationID	long	required	6	
MajorVersion	int	required	1	
MinorVersion	int	required	1	
Accounts	UpdatedAccount[ ]	required		
AccountIdentifier	long	required	345	
IdentifierInReg	string	optional	MYACCOUNT	
Installation	UpdatedInstallation	optional		
PermitIdentifier	string	optional	XCVXXCV-999	
Name	string	optional	Installation1	
MainActivityType	int	optional	2	
Country	string	optional	BE	
PostalCode	string	optional	1200	
City	string	optional	Brussels	
Address1	string	optional	Avenue Ariane 7	
Address2	string	optional		
ParentCompany	string	optional		
SubsidiaryCompany	string	optional		
EPERIdentification	string	optional		
Latitude	double	optional		
longitude	double	optional		
ContactPeople	UpdatedPeople	optional		
action	int	required	1	This is about adding a new or updating an existing person
RelationshipCode	int	required	6	
PersonIdentifier	string	required	ZERZER-123	
FirstName	string	optional	Jack	
LastName	string	optional	Black	
Country	string	optional	BE	
PostalCode	string	optional	1000	
City	string	optional	Bruxelles	
Address1	string	optional	Rue Royale, 99	
Address2	string	optional		
PhoneNumber1	string	optional	+32-2-9999999	
PhoneNumber2	string	optional	+32-2-8888888	
FaxNumber	string	optional	+32-2-7777777	
Email	string	optional	<a href="mailto:jb@mlk.be">jb@mlk.be</a>	
People	UpdatedPeople	required		
Action	int	required	1	This is about adding a new or updating an existing person
RelationShipCode	int	required	1	
PersonallIdentifier	string	required	Gfds-g	
FirstName	string	optional	John	
LastName	string	optional	Watson	

Country	string	optional	BE	
PostalCode	string	optional	1000	
City	string	optional	Bruxelles	
Address1	string	optional	Rue Neuve, 88	
Address2	string	optional		
PhoneNumber1	string	optional	+32-2-1234567	
PhoneNumber2	string	optional	+32-2-9876543	
FaxNumber	string	optional	+32-2-7539512	
E-Mail	string	optional	<a href="mailto:jw@aaa.be">jw@aaa.be</a>	

**Figure M9: Update Account Response**

<u>ProposalRequest</u>	Type	optional / required	EU Sample Data	Comment
ResultIdentifier	int	required	1	1 if request accepted, 0 if request refused
ResponseCode	int	optional		

**Figure M10: Close Account Request**

<u>CloseAccountRequest</u>	Type	optional / required	EU Sample Data	Comment
From	string	required	BE	
To	string	required	CTL	
CorrelationID	long	required	5	
MajorVersion	int	required	1	
MinorVersion	int	required	1	
Accounts	ClosedAccount[ ]	required		
AccountIdentifier	long	required	234	

**Figure M11: Close Account Response**

<u>CloseAccountResponse</u>	Type	optional / required	EU Sample Data	Comment
ResultIdentifier	int	required	1	1 if request accepted, 0 if request refused
ResponseCode	int	optional		

**Figure M12: Update Verified Emissions Request**

<u>UpdateVerifiedEmissions Request</u>	Type	optional / required	EU Sample Data	Comment
From	string	required	BE	
To	string	required	CTL	
CorrelationID	long	required	55	
MajorVersion	int	required	1	
MinorVersion	int	required	1	
VerifiedEmissions	VerifiedEmissions[ ]	required		
YearInCommitPeriod	int	required	2005	
Installations	InstallationVerified Emission[ ]	required		
InstallationIdentifier	long	required	123	
VerifiedEmissions	long	required	55555	

**Figure M13: Update Verified Emissions Response**

<u>UpdateVerifiedEmissions Response</u>	Type	optional / required	EU Sample Data	Comment
ResultIdentifier	int	required	1	1 if request accepted, 0 if request refused
ResponseCode	int	optional		

**Figure M14: Receive Account Operation Outcome Request**

<u>ReceiveAccountOperation OutcomeRequest</u>	Type	optional / required	EU Sample Data	Comment
From	string	required	CTL	
To	string	required	BE	
CorrelationID	long	required	OutcomeRequest	
MajorVersion	int	required	1	
MinorVersion	int	required	1	
Outcome	int	required	0	1 if request completed, 0 if request terminated
OutcomeDetails	OutcomeDetail[ ]	optional		
ResponseCode	int	required	7149	
AccountIdentifier	long	optional	1	
InstallationIdentifier	long	optional	123	

**Figure M15: Receive Account Operation Outcome Response**

<u>ReceiveAccountOperation OutcomeResponse</u>	Type	optional / required	EU Sample Data	Comment
ResultIdentifier	int	required	1	1 if request completed, 0 if request terminated
ResponseCode	int	optional		

## 24 3 Generic Web Service

The Generic Web Service is designed to provide a means of communicating messages for the function of a supplementary scheme between supplementary scheme members and the supplementary transaction log. The communication is via the ITL communications hub with the ITL simply forwarding messages to their destination. The ITL performs sufficient validation to authenticate the sender of a message and to send the message to its correct destination, but no more. The service is designed so that the ITL has no need to be aware of the STL specific content of the message.

### 3.1 Generic Web Service Process

The same process is used to send messages from an EU-ETS member to the CITL and from the CITL to the EU-ETS. The steps to send a message are:

Step 1 - The sender sends a SOAP request to the ITL. If the message is not a well formed based on the WSDL, the ITL returns the standard SOAP error and stops processing the message.

Step 2 - If the message is well formed, the authentication and DES version checks are performed. If the checks show discrepancies in the message, a synchronous SOAP response with a result identifier of zero and the appropriate response code is returned to the sender, and then the ITL stops processing the message.

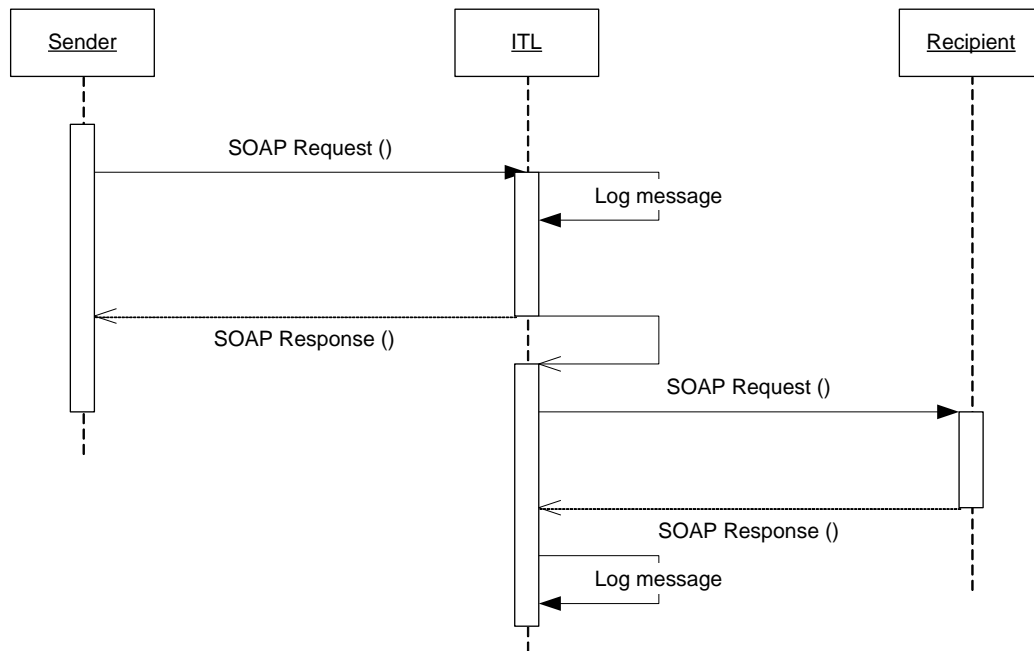
Step 3 - If the message is a well formed SOAP message and passes the checks, the ITL logs the inbound message, and the response by the ITL, and stores them.

Step 4 - The ITL sends the well formed message to the recipient. The recipient responds to confirm receipt and the outbound message, and the response from the recipient, is logged

#### 3.1.1 Behaviour diagram

The figure below shows the flow of a well-formed SOAP message through the Generic Web Service.

**Figure M16 - Handling of a SOAP request received by the generic WS**



### 3.1.2 Logging

Registries, the ITL and CITL should follow the message logging requirements articulated in the section 7.5 of the DES but the message logging should clearly distinguish between Generic Web Service messages and other logged messages.

All incoming and outgoing messages on the Generic Web Service that pass the authentication checks are logged in the ITL message logs.

### 3.1.3 Constraints on the operations support by the Generic Web Service

The design of the business operations to be supported by the Generic Web Service must take I to account the following constraints:

- (1) All operations provided through the Generic Web Service must tolerate delivery of duplicate messages.
- (2) The recipient must not reject messages if they are well formed.
- (3) All responses by the recipient to the sender are asynchronous and made using an operation supported by the Generic Web Service.
- (4) A registry may support either the Account Management Web Service or the Generic Web Service but not both.
- (5) The CITL must support both the Account Management Web Service and the Generic Web Service.

## 3.2 Generic Web Service message handling

### 3.2.1 Error handling and response codes

The ITL performs sufficient checks to ensure a message can be routed to its destination. These checks are limited to the fields defined in Figure M18. Any message to the Generic Web Service is first checked against the current WSDL to see that it is well formed. The ITL then applies only limited checks to the messages it receives and all the response codes are given in the synchronous SOAP response. A standard SOAP error is return to the send of the SOAP request if it is not.

The Generic Web Service makes use of the Message Validity Response Codes described in Annex E. The specific message validity checks the ITL performs on messages sent to the Generic Web Service are list below in figure M17 below.

**Figure M17: Checks performed by ITL on message to the Generic Web Service**

Response Code	Check Name	Check Description
1031	Major Version	Major Version number in the message must match current Major Version number for DES.
1032	Minor Version	Minor Version number in the message should match current Minor Version number for DES.
1511	Sender Authentication	The identity of the sender in the “from” element of the message must match the identity of the authenticated sender of the message.
Discarded (1)	Sender status	The sender’s status must allow Generic Web Service messages to be sent. (The ITL will maintain the current status of each registry. In this case, the ITL must recognize that the registry is fully operational.)
Discarded (1)	Recipient status	The recipient’s status must allow Generic Web Service messages to be received. (The ITL will maintain the current status of each registry. In this case, the ITL must recognize that the registry is fully operational.)
Discarded (1)	EU-ETS membership	The sender and recipient of a Generic Web Service message must be a member of the supplementary trading scheme, and either the sender or the recipient must be the supplementary transaction log.

Notes on Figure M17

- (1) Checks marked with a response code of “Discarded” do not result in a response code being returned to the sender of the message. The message is simply logged and discarded.
- (2) Response 1032 is accompanied by a resultIdentifier of one, indicating the message was successfully processed. This combination of response code and resultIdentifier of 1 is just a warning, and the message will be processed by the recipient.

### 3.2.2 Message life time

A message has a maximum life of 24 hours from the first attempt to send it. After 24 hours the sender must make not further attempts to process a message. This limit is taken in to account in the ITLs handling of messages that can not be delivered on the first attempt using the approach in section 3.2.3.

### 3.2.3 Recovery from problems

A message is successfully sent when the sender of a message receives a well formed response with a resultIdentifier of one, even if the response also contains a response code. If the sender of message using the Generic Web Service receives a different response they may attempt to re-send the message under the following circumstances:

- (1) They did not receive a well-formed SOAP response from the ITL.
- (2) They received a well formed SOAP response from the ITL, but have not received the expected message from the recipient.

Messages may be resent automatically or manually. If a well formed response with a resultIdentifier of zero is received the sender should not attempt automatically resend the message. The sender may attempt to manually resend a message after the problem that gave rise to the resultIdentifier of zero has been resolved. If messages are resent automatically, the mechanism used must **not**:

- (1) Attempt to resend messages so frequently that the performance of the services provided by the ITL or the intended recipient degrades;
- (2) Attempt to resend messages more than 24 hours old.

The international transaction log will use an automatic process for resending messages it could not deliver. If, on attempting to send a message to the recipient, the ITL does not receive a well-formed XML response, then the ITL will attempt to re-send the message to its recipient at intervals for up to 24 hours. If a message cannot be successfully sent in this time frame the ITL logs the message as a failed delivery and takes no further action.

### 3.3 Generic Web Service operations

The operations available via the Generic Web Service all take the following format with <generic operation> replaced by the name of the operation.

<generic operation>Request  
<generic operation>Response

All requests have some mandatory common elements that the ITL uses to correctly route the message. These are listed below. The specific data elements for each operation are agreed and maintained within the EU-ETS and are not documented in the data exchange standards.

The following figure shows the common elements for an operation know as <generic operation>

**Figure M18: <generic operation>Request  
Role(s): CITL**

<u>&lt;generic operation&gt;Request</u>			
from	String		
to	String		
majorVersion	Int		
minorVersion	Int		
correlationID	Long		<i>Use is defined by EU-ETS</i>
<i>generic content</i>	<i>Various</i>		<i>Defined by EU-ETS</i>
<u>&lt;generic operation&gt;Response</u>			
resultIdentifier	int		
responseCode	int		Opt

#### 3.3.1 Generic Web Service message elements

The mandatory elements of a generic web service request are defined as follows:

Element	Definition
from	The registry code of the registry or entity sending the XML request. For messages received by EU-ETS registries, this value will always be the CTL. The codes used are the two letter country codes in ISO3166, EU for the European Community Registry and CTL for the CITL.
to	The code of the registry receiving or entity an XML request. For messages sent by EU-ETS registries, this value will always be the CTL. The codes used are the two-letter country codes in ISO3166, EU for the European Community Registry and CTL for the CITL.
majorVersion	The Major Version number of the version of the DES to which this message complies.



Element	Definition
minorVersion	The Minor Version number of the version of the DES to which this message complies.
correlationID	An identifier that can be used to correlate two or more messages exchanged under the Generic Web Service. The usage of this identifier must be defined as part of the definition of the operations provided using the Generic Web Service.

The mandatory elements of a generic web service response are defined as follows:

Element	Definition
resultIdentifier	This value will indicate whether the receiving function succeeded or failed. A value of: <b>Zero</b> indicates permanent failure and the message must not be resent by the sender. <b>One</b> indicates the message has been successfully received.
responseCode	A message response code returned by the Generic Web Service as defined in Annex M of the DES. The response code is <u>not</u> populated if a result Identifier of one is supplied in the response.

### 3.4 Generic Web Service WSDLs

The Generic Web Service is available on a single endpoint and uses two WSDL files, one for the registries and one for the STL. Endpoint is named as follows:

- On the Registry it is named **RegistrySTLWebService**.
- On the CITL it is named **STLWebService**.

Since the Generic Web Service is designed to allow the WSDL to be changed without a change to the ITL or the DES, the WSDL is not published within the DES. Instead and electronic version of the WSDL support by the ITL will be made available on the RSA Extranet. The business logic, specifications and associated WSDL for the Generic Web Service is maintained and distributed by the European Commission. The WSDL will adhere to the constraints imposed by the design and implementation of the Generic Web Service to ensure its ongoing operation.

The target namespace for the Generic Web Service is defined in the WSDL. It is different from the namespace used by the account management web service, which is shown in the WSDLs in section 2.2 of Annex M.

### 3.5 Generic Web Service examples

Figure M19: <generic operation> Request

<generic operation>Request	Type	optional / required	EU Sample Data	Comment
From	string	required	XX	Registry code of and EU-ETS member or CTL
To	string	required	YY	
MajorVersion	int	required	1	
MinorVersion	int	required	1	
correlationID	long	required	12345	
<i>generic content</i>				As defined in WSDL

**Figure M20: <generic operation> Response**

<b>&lt;generic operation&gt;Response</b>	<b>Type</b>	<b>optional / required</b>	<b>EU Sample Data</b>	<b>Comment</b>
ResultIdentifier	int	required	1	1 if request completed, 0 if request terminated
ResponseCode	int	optional		No response code is supplied if a result identifier of 1 is given.