The Consultative Committee for Space Data Systems

Research and Development for Space Data System Standards

SPACE LINK EXTENSION— ENHANCED FORWARD CLTU SERVICE SPECIFICATION

EXPERIMENTAL SPECIFICATION

CCSDS 912.11-O-1

ORANGE BOOK

July 2012

The Consultative Committee for Space Data Systems

Research and Development for Space Data System Standards

SPACE LINK EXTENSION— ENHANCED FORWARD CLTU SERVICE SPECIFICATION

EXPERIMENTAL SPECIFICATION

CCSDS 912.11-O-1

ORANGE BOOK
July 2012

AUTHORITY

Issue:	Orange Book, Issue 1
Date:	July 2012
Location:	Washington, DC, USA

This document has been approved for publication by the Consultative Committee for Space Data Systems (CCSDS). The procedure for review and authorization of CCSDS documents is detailed in *Organization and Processes for the Consultative Committee for Space Data Systems* (CCSDS A02.1-Y-3).

This document is published and maintained by:

> CCSDS Secretariat
> Space Communications and Navigation Office, 7L70
> Space Operations Mission Directorate
> NASA Headquarters
> Washington, DC 20546-0001, USA

FOREWORD

This document is an **Experimental Specification** for use in exploring AOS data link protocol enhancements to the CCSDS Space Link Extension (SLE) Forward Communications Link Transmission Unit (CLTU) Service (reference [8]) and has been contributed to CCSDS by NASA.

This **Experimental Specification** specifies a data service that extends certain of the space-to-ground and ground-to-space communications services previously defined by CCSDS (references [2], [3], and [9]) within the framework established by the CCSDS SLE Reference Model (reference [1]). It allows implementing organizations within each Agency to proceed with experimentation, analysis, and modification of the specification in preparation for subsequent adoption of the content of the specification in one or more CCSDS **Recommended Standard**s, as this Experimental Specification will not itself become a CCSDS Recommended Standard. Rather, it is intended as a means for analyzing, validating, and prototyping some of the requirements that will eventually be realized in one or more transfer services that meet the requirements of the IOAG Forward Synchronous Encoded Frame service (informative reference [I10]) and/or the Forward Frame service (informative reference [I11]). Both of those services will be defined in terms of the Cross Support Transfer Service Framework (informative reference [I9]).

NOTE – At the time of the writing of this Experimental Specification, the Cross Support Transfer Service Framework does not define operations and procedures suitable for the aforementioned IOAG forward services. This Experimental Specification is based on the existing SLE Forward CLTU transfer service specification so that FCLTU implementations can be used as test beds for experimentation with key capabilities of the target Cross Support Transfer services. Once the target Cross Support Transfer service(s) are available, this EFCLTU Experimental Specification will be deprecated.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CCSDS shall not be held responsible for identifying any or all such patent rights.

Through the process of normal evolution, it is expected that expansion, deletion, or modification of this document may occur. This Experimental Specification is therefore subject to CCSDS document management and change control procedures, which are defined in the *Procedures Manual for the Consultative Committee for Space Data Systems*. Current versions of CCSDS documents are maintained at the CCSDS Web site:

http://www.ccsds.org/

Questions relating to the contents or status of this document should be addressed to the CCSDS Secretariat at the address indicated on page i.

PREFACE

This document is a CCSDS Experimental Specification. Its Experimental status indicates that it is part of a research or development effort based on prospective requirements, and as such it is not considered a Standards Track document. Experimental Specifications are intended to demonstrate technical feasibility in anticipation of a 'hard' requirement that has not yet emerged. Experimental work may be rapidly transferred onto the Standards Track should a hard requirement emerge in the future.

DOCUMENT CONTROL

Document	Title	Date	Status
CCSDS 912.11-O-1	Space Link Extension—Enhanced Forward CLTU Service Specification, Experimental Specification, Issue 1	July 2012	Current issue

CONTENTS

CONTENTS (continued)

CONTENTS (continued)

1 INTRODUCTION

1.1 PURPOSE OF THIS EXPERIMENTAL SPECIFICATION

This Experimental Specification defines the Space Link Extension (SLE) Enhanced Forward Communications Link Transmission Unit (CLTU) service, which provides functions of several of the transfer services originally specified in reference [1], *Cross Support Reference Model—Part 1: SLE Services* (CSRM).

NOTE – The CSRM (reference [1]) specifies completely different services for handling the data structure defined in references [2], [3], and [9]). If and when Recommended Standards are created for Cross Support Transfer Services (CSTSes) to perform the functions specified herein, CCSDS will determine if and how related Recommended Standards (including the CSRM) will need to be updated.

This Experimental Specification is built on the service type for the SLE Forward CLTU transfer service (reference [8]), which enables a mission to send CLTUs to a spacecraft using the Telecommand Physical Layer Operations Procedures (PLOPs—see reference [2]). In addition to Telecommand CLTUs, the enhancements of the Forward CLTU service alternatively allows the service to transfer AOS transfer frames (see reference [9]) or Channel Access Data Units (CADUs—see reference [10]). Concurrent with the ability to transfer these additional space link data units, the enhancements include additional production processing to encode and synchronize the transfer frames and CADUs in accordance with the TM Synchronization and Channel Coding functions (see reference [10]) that are appropriate to each of those space link data unit types. The Enhanced Forward CLTU Experimental Specification defines capabilities that are intended to be standardized by CCSDS in the form of transfer services based on the Cross Support Transfer Service Specification Framework (informative reference [I8]). Specifically, these capabilities are expected to be employed in CSTS Framework-based services that meet the requirements of the IOAG Forward Synchronous Encoded Frame service (informative reference [I10]) and/or the Forward Frame service (informative reference [I11]). As with the proposed Forward Frame service, the Enhanced Forward CLTU experimental service transfers data units that exist at multiple levels of the space link protocol stacks: fully coded Telecommand CLTUs, partially coded CADUs, and to-be-coded transfer frames.

Adding the transfer (and processing) of AOS transfer frames and CADUs (to be processed and transmitted in accordance with the TM synchronization and channel coding functions) to the existing Forward CLTU SLE transfer service allows the re-use of the great majority of the FCLTU design and code base, enabling more rapid prototyping for experimental purposes.

NOTES

1   The addition of support for AOS transfer frames and CADUs to the Telecommand CLTU-oriented SLE Forward CLTU transfer service has required the introduction of new terms that are not completely consistent with other existing CCSDS standards. The establishment and use of these terms is strictly local to this Experimental Specification and not intended to replace or redefine standard CCSDS terminology. Any real CCSDS Recommended Standards that use the techniques defined in this Experimental Specification will use terminology that is consistent with the larger body of CCSDS standards.

2.   Transferring AOS CADUs implies an ability to multiplex CADUs that is not currently prescribed by CCSDS Space Link Recommended Standards (see references [9] and [10]) within the layers implemented by EFCLTU. However, the net behavior of the protocol stack implemented by user and provider and the effect on the signal being transmitted is compliant with the standards and therefore interoperability is unaffected. In the future, if CADU multiplexing is to be included in a real CCSDS Recommended Standard, CCSDS may wish to edit the Space Link Recommended Standards to explicitly support this capability.

1.2   SCOPE

This Experimental Specification defines, in an abstract manner, the Enhanced Forward CLTU service in terms of:

   a) the operations necessary to provide the transfer service;

   b) the parameter data associated with each operation;

   c) the behaviors that result from the invocation of each operation; and

   d) the relationship between, and the valid sequence of, the operations and resulting behaviors.

This Experimental Specification also specifies the coding and synchronization functions that are to be performed when the Enhanced Forward CLTU service is transferring data that is to be transmitted over an AOS space link. These AOS space link coding and synchronization functions are not part of the provision of the Enhanced Forward CLTU, but rather they are part of the production of the service (see 2.4.2).

This Experimental Specification does not specify:

   a) individual implementations or products;

   b) the implementation of entities or interfaces within real systems;

c) the methods or technologies required to radiate data to a spacecraft and to acquire telemetry frames from the signals received from that spacecraft for extraction of the Operational Control Field (OCF);

d) the methods or technologies required for communications; or

e) the management activities necessary to schedule, configure, and control the Enhanced Forward CLTU service.

1.3 APPLICABILITY

1.3.1 APPLICABILITY OF THIS EXPERIMENTAL SPECIFICATION

This Experimental Specification provides a basis for experimentation and analysis leading to the development of one or more Cross Support Transfer Services capable of supporting the AOS space data link protocol. Implementation of the Enhanced Forward CLTU service in a real system would additionally require the availability of a communications service to convey invocations and returns of Enhanced Forward CLTU service operations between service users and providers. This Experimental Specification requires that such a communications service ensure that invocations and returns of operations are transferred:

a) in sequence;

b) completely and with integrity;

c) without duplication;

d) with flow control that notifies backpressure to the Application layer in the event of congestion; and

e) with notification to the Application layer in the event that communications between the Enhanced Forward CLTU service user and the Enhanced Forward CLTU service provider are disrupted, possibly resulting in a loss of data.

It is the specific intent of this Experimental Specification to define the Enhanced Forward CLTU service in a manner that is independent of any particular communications services, protocols, or technologies.

1.3.2 LIMITS OF APPLICABILITY

This Experimental Specification specifies the Enhanced Forward CLTU service that may be provided by an SLE System for inter-Agency cross support. It is neither a specification of, nor a design for, real systems that may be implemented for the control and monitoring of existing or future missions.

1.4 RATIONALE

The goal of this Experimental Specification is to create a standard for interoperability between the tracking stations or ground data handling systems of various agencies and the users of forward services.

1.5 DOCUMENT STRUCTURE

1.5.1 DOCUMENT ORGANIZATION

This Experimental Specification is organized as follows:

a) section 1 provides purpose, scope, applicability, and rationale of this Experimental Specification and lists definitions, nomenclature, conventions, and references used throughout the specification;

b) section 2 presents an overview of the Enhanced Forward CLTU service including a functional description, the service management context, and protocol considerations;

c) section 3 specifies the operations of the Enhanced Forward CLTU service;

NOTE – For the purposes of this Experimental Specification, the Enhanced Forward CLTU operations are defined as later versions of the corresponding Forward CLTU operations. If and when the capabilities of the Enhanced Forward CLTU service are realized as part of an actual CSTS, the operations will be written specific to the actual CSTS.

d) section 4 specifies the dynamic behavior of the Enhanced Forward CLTU service in terms of the state transitions of the Enhanced Forward CLTU service provider;

e) annex A is a formal specification of Enhanced Forward CLTU service data types, using the Abstract Syntax Notation One (ASN.1);

NOTE – For the purposes of this Experimental Specification, the Enhanced Forward CLTU ASN.1 service data types are defined as later versions of the corresponding Forward CLTU ASN.1 service data types. If and when the capabilities of the Enhanced Forward CLTU service are realized as part of an actual CSTS, the formal specification of the ASN.1 service data types will be written in accordance with the CSTS rules for ASN.1 service data types and will be specific to the actual CSTS.

f) annex B explains the relationship of the Enhanced Forward CLTU service to the production status of the forward space link channel;

g) annex C provides a conformance matrix that defines what capabilities must be provided for an implementation to be considered compliant with this Experimental Specification;

   h) annex D specifies the AOS forward space link synchronization and channel coding production functions;

   i) annex E discusses security, SANA, and patent considerations;

   j) annex F lists all terms used in this document and identifies where they are defined;

   k) annex G lists all acronyms used within this document;

   l) annex H contains examples of usage of the CLTU-THROW-EVENT operation; and

   m) annex I contains a list of informative references.

1.5.2 SLE SERVICES DOCUMENTATION TREE

This Experimental Specification is based on the architectural model for cross support defined in reference [1]. It expands upon the concept of an SLE transfer service as interactions between SLE Mission User Entities (MUEs) and an SLE transfer service provider for the purpose of providing the Enhanced Forward CLTU transfer service.

This Experimental Specification is developed within the framework of the suite of documents specifying the SLE Services. The SLE Services constitute one of the three types of Cross Support Services:

   a) Part 1: SLE Services;

   b) Part 2: Ground Communications Services;

   c) Part 3: Ground Domain Services.

The basic organization of the SLE services documentation is shown in figure 1-1. The documents are described in the following paragraphs.
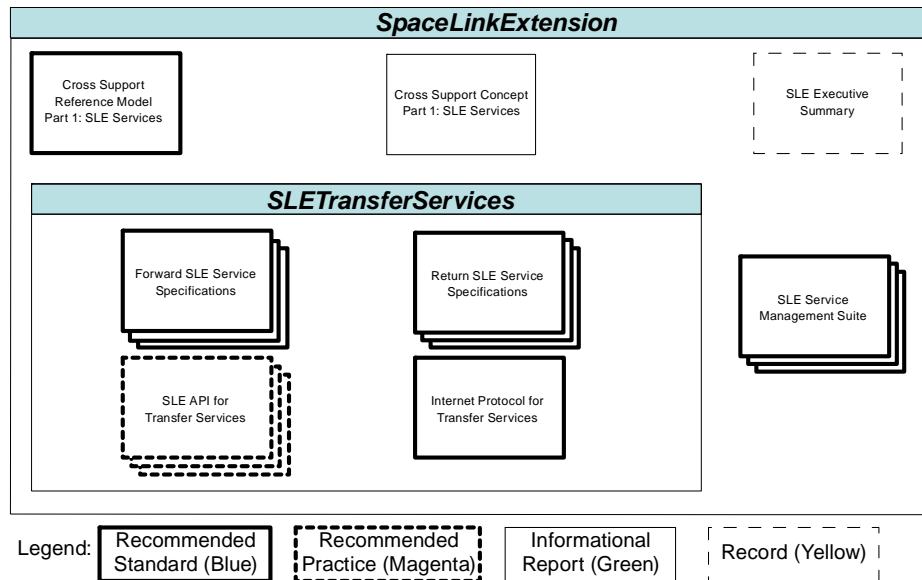
Figure 1-1: SLE Services Documentation

a) *Cross Support Concept—Part 1: Space Link Extension Services* (informative reference [I3]): a Report introducing the concepts of cross support and SLE services;

b) *Cross Support Reference Model—Part 1: Space Link Extension Services* (reference [1]): a Recommended Standard that defines the framework and terminology for the specification of SLE services;

c) *Return SLE Transfer Service Specifications*: a set of Recommended Standards that will provide specification of all return link SLE transfer services;

d) *Forward SLE Transfer Service Specifications*: a set of Recommended Standards that will provide specification of all forward link SLE transfer services (this Experimental Specification enhances one of the specifications in that set);

e) *SLE API for Transfer Services Specifications*: a set of Recommended Practices that provide specifications of an Application Program Interface; a set of Recommended Standards that provide specifications of an Application Program Interface and a mapping to TCP/IP as underlying communications service for SLE services;

f) *Internet Protocol for Transfer Services*: definition of a protocol for transfer of SLE Protocol Data Units using TCP/IP as underlying communications service for SLE services;

g) *SLE Service Management Specification Suite*: a set of Recommended Standards that establish the basis for SLE service management.

1.6   DEFINITIONS

1.6.1   DEFINITIONS FROM OPEN SYSTEMS INTERCONNECTION (OSI) BASIC REFERENCE MODEL

This Experimental Specification makes use of a number of terms defined in reference [7]. The use of those terms in this Experimental Specification are to be understood in a generic sense; i.e., in the sense that those terms are generally applicable to technologies that provide for the exchange of information between real systems. Those terms are:

    a) abstract syntax;

    b) application entity;

    c) Application layer;

    d) flow control;

    e) Open System Interconnection (OSI);

    f) real system;

    g) Service Access Point (SAP).

1.6.2   DEFINITIONS FROM ABSTRACT SYNTAX NOTATION ONE

This Experimental Specification makes use of the following terms defined in reference [6]:

    a) Abstract Syntax Notation One (ASN.1);

    b) object identifier;

    c) (data) type;

    d) (data) value.

NOTE – In annex A of this Experimental Specification, ASN.1 is used for specifying the abstract syntax of the invocations and returns of the operations of the Enhanced Forward CLTU service. The use of ASN.1 as a descriptive language is intended to support the specification of the abstract Enhanced Forward CLTU service; it is not intended to constrain implementations. In particular, there is no requirement for implementations to employ ASN.1 encoding rules. ASN.1 is simply a convenient tool for formally describing the abstract syntax of the invocations and returns of the Enhanced Forward CLTU service.

1.6.3   DEFINITIONS FROM TC SYNCHRONIZATION AND CHANNEL CODING

This Experimental Specification makes use of the following terms defined in reference [2]:

a) acquisition sequence;

b) Carrier Modulation Mode (CMM);

c) Communications Link Transmission Unit (CLTU);

d) idle sequence;

e) Physical Layer Operations Procedure (PLOP);

1.6.4 DEFINITIONS FROM TC SPACE DATA LINK PROTOCOL

This Experimental Specification makes use of the following term defined in reference [3]:

Communications Link Control Word (CLCW).

1.6.5 DEFINITIONS FROM SLE REFERENCE MODEL

This Experimental Specification makes use of the following terms defined in reference [1]:

a) abstract binding;

b) abstract object;

c) abstract port;

d) abstract service;

e) CLTU channel;

f) Forward CLTU service;

g) invoker;

h) Mission Data Operation System (MDOS);

i) Mission User Entity (MUE);

j) offline delivery mode;

k) online delivery mode;

l) operation;

m) performer;

n) physical channel;

o) service agreement;

p) service provider (provider);

q) service user (user);

r) SLE Complex;

    s) SLE Complex Management;

    t) SLE data channel;

    u) SLE functional group (SLE-FG);

    v) SLE protocol data unit (SLE-PDU);

    w) SLE service data unit (SLE-SDU);

    x) SLE service package;

    y) SLE transfer service instance;

    z) SLE transfer service production;

    aa) SLE transfer service provision;

    bb) SLE Utilization Management;

    cc) space link;

    dd) space link data channel;

    ee) Space Link Data Unit (SL-DU);

    ff) Space Link Protocol Data Unit (SL-PDU);

    gg) space link session.

1.6.6 DEFINITIONS FROM TM SYNCHRONIZATION AND CHANNEL CODING

This Experimental Specification makes use of the following terms defined in reference [10]:

    a) Attached Synchronization Marker (ASM);

    b) Channel Access Data Unit (CADU);

    c) codeblock;

NOTE –   The term 'codeblock' is also used for a component of a CLTU that is formed in accordance with Telecommand Synchronization and Channel Coding standards (see reference [2]). However, since the composition of the CLTU is opaque to the EFCLTU service, this Telecommand-style codeblock is not exposed by this service. Therefore, in this specification, the term 'codeblock' is used only in the telemetry (TM) sense.

    d) pseudo-randomization;

    e) transfer frame.

1.6.7 DEFINITIONS FROM AOS SPACE DATA LINK PROTOCOL

This Experimental Specification makes use of the following terms defined in reference [9]:

a) AOS idle transfer frame;

b) AOS transfer frame.

1.6.8 ADDITIONAL DEFINITIONS

1.6.8.1 General

For the purposes of this Experimental Specification, the following definitions also apply.

NOTE – Many of the terms defined for the purposes of this Experimental Specification have been chosen to resolve ambiguities resulting from the use of the Forward CLTU SLE transfer service as a basis for the specification, or to minimize changes needed in the Forward CLTU ASN.1 modules. These terms are not being proposed for permanent use within the CCSDS lexicon.

AOS Synchronization and Coding Procedure (ASCP): A procedure that incorporates a collection of AOS space data link protocol and TM synchronization and channel coding functions applied to the forward link: insertion of a fixed-length idle SL-PDU into the data stream whenever user-generated space link data is not available to the Enhanced Forward CLTU service provider; optional block forward error correction encoding; optional frame randomization; optional synchronization marker attachment; and optional convolutional encoding.

NOTE – In contrast with PLOP-1 and PLOP-2, ASCP is not a specific procedure of the Physical Layer and is therefore not technically a PLOP. However, ASCP is identified in the `plop-in-effect` parameter of various EFCLTU operations to indicate when the associated production is configured to process AOS data units. ASCP is also used to identify requirements in this Experimental Specification that apply only when the EFCLTU service and associated production are transferring and processing AOS data units.

association: A cooperative relationship between an SLE service-providing application entity and an SLE service-using application entity. An association is formed by the exchange of SLE protocol data units through use of an underlying communications service.

communications service: A capability that enables an SLE service-providing application entity and an SLE service-using application entity to exchange information.

NOTE – If an SLE service user and an SLE service provider are implemented using different communications services, then interoperability between them is possible only by means of a suitable gateway. Adherence to this Experimental Specification ensures, at least in principle, that it is possible to construct such a gateway.

confirmed operation: An operation that requires the performer to return a report of its outcome to the invoker.

idle SL-PDU: A fixed-length, service-management-defined block of octets that is inserted into the production of the forward AOS space link whenever user-generated space link data is not available to the Enhanced Forward CLTU service provider. The length of the idle SL-PDU is equal to the length of one of the user-generated AOS Transfer Frames or CADUs that are being transferred via the EFCLTU service. When the ECFLTU service is transferring AOS Transfer Frames, the Idle SL-PDUs are AOS Idle Transfer Frames that conform to the Idle Transfer Frame format specified in reference [9]. When the EFCLTU service is transferring CADUs, the Idle SL-PDU is a CADU that has been formed by processing an AOS Idle Transfer Frame with the same TM synchronization and channel coding functions as are performed on the AOS Transfer Frames that are transferred by the EFCLTU service instance.

initiator: The object that issues the request to bind to another object (the responder).

NOTE – In other words, the initiator is always the invoker of the request to bind to another object. Therefore, in the context of the request to bind, the terms 'initiator' and 'invoker' refer to the same object and are synonyms.

invocation: The making of a request by an object (the invoker) to another object (the performer) to carry out the operation.

parameter: Data that may accompany the operation's invocation or return.

NOTE – The term parameter is also used to refer to mission-dependent configuration information used in production or provision of the service.

performance: The carrying out of the operation by an object (the performer).

port identifier: Identifier for a source or a destination in a communications system.

NOTE – More information can be found in 2.6.4.6.

responder: The object that receives a request to bind and completes the binding (if possible) with the initiator in order for a service association to exist between the two objects.

NOTE – In other words, the responder is always the performer of the binding. Therefore, in the context of binding, the terms 'responder' and 'performer' refer to the same object and are synonyms.

return: A report, from the performer to the invoker, of the outcome of the performance of the operation.

service instance provision period: The time during which a service instance (i.e., the capability to transfer one or more SLE data channels of a given type) is scheduled to be provided.

SLE-Forward Transfer Data Unit (SLE-FTDU): The service data unit that is transferred by the EFCLTU transfer service. An SLE-FTDU may contain a single CLTU (see 1.6.3), one or more AOS transfer frames (see1.6.6e)), or one or more Channel Access Data Units (1.6.6b)).

AOS SL-PDU: A fixed-length space link protocol data unit that is carried by the SLE-FTDU for an EFCLTU service instance that supplies a forward AOS space link. The AOS SL-PDUs are: AOS transfer frame (1.6.6e) and Channel Access Data Unit (1.6.6b)) (see figure D-1).

unconfirmed operation: An operation that does not require a report of its outcome to be returned to the invoker by the performer.

1.7 NOMENCLATURE

1.7.1 NORMATIVE TEXT

The following conventions apply for the normative specifications in this Recommended Standard:

   a) the words 'shall' and 'must' imply a binding and verifiable specification;

   b) the word 'should' implies an optional, but desirable, specification;

   c) the word 'may' implies an optional specification;

   d) the words 'is', 'are', and 'will' imply statements of fact.

NOTE – These conventions do not imply constraints on diction in text that is clearly informative in nature.

1.7.2 INFORMATIVE TEXT

In the normative sections of this document (sections 3-4 and annexes A-D), informative text is set off from the normative specifications either in notes or under one of the following subsection headings:

   – Overview;

   – Background;

   – Rationale;

   – Discussion.

1.8 CONVENTIONS

1.8.1 SPECIFICATION OF OPERATIONS

1.8.1.1 General

Section 3 of this Experimental Specification specifies the operations that constitute the Enhanced Forward CLTU service. The specification of each operation is divided into subsections as follows:

1.8.1.2 Purpose Subsection

The Purpose subsection briefly describes the purpose and functioning of the operation. Additionally, it indicates whether the operation may be invoked by the user, provider, or both; whether the operation is confirmed or unconfirmed; and whether there are any constraints on when the operation may be invoked.

1.8.1.3 Invocation, Return, and Parameters Subsection

The Invocation, Return, and Parameters subsection describes the parameters associated with each operation, including their semantics. A table accompanying the description of each operation lists all parameters associated with the operation and, for both the invocation and return, whether the parameter is always present, always absent, or conditionally present.

For parameters that are conditionally present, the parameter description specifies the conditions for the presence or absence of the parameter. The condition is generally based on the value of another parameter in the same invocation or return; for example, in return of an operation, the `diagnostic` parameter is present if and only if the value of the `result` parameter is 'negative result'. For a conditional parameter in a return, the condition may be based on the value of a parameter in the corresponding invocation.

In the table, the following convention is used to indicate whether a parameter is always present, always absent, or conditionally present:

M   always present (mandatory)

C   conditionally present

Blank   always absent

NOTE – Even though a parameter may be characterized as always present, its description may specify that its value is permitted to be 'null' or 'unused' or the like.

1.8.1.4 Effects Subsection

The Effects subsection describes the effects an operation has on the invoker, the performer, the association between them, or any combination thereof. The details of how those effects occur or the mechanisms used are outside the scope of this Experimental Specification.

1.8.2 TYPOGRAPHIC CONVENTIONS

1.8.2.1 Overview

Typographic conventions used in this Experimental Specification are described in the following subsections.

1.8.2.2 Operation Names

Names of Enhanced Forward CLTU service operations appear in uppercase and begin with the characters 'CLTU-' (e.g., CLTU-TRANSFER-DATA).

1.8.2.3 Parameter Names

In the main text, names of parameters of Enhanced Forward CLTU service operations appear in lowercase and are typeset in a fixed-width font (e.g., `responder-port-identifier`). In annex A, the corresponding name is formed by omitting any hyphens contained in the name and using mixed-case (e.g., `responderPortIdentifier`).

1.8.2.4 Value Names

The values of many parameters discussed in this Experimental Specification are represented by names. In the main text, these names are shown in single quotation marks (e.g., 'no such service instance'). The corresponding name in annex A is formed by omitting any hyphens or white space contained in the name and using mixed-case (e.g., `noSuchServiceInstance`). The actual value associated with the name is constrained by the type of the parameter taking on this value. Parameter types are specified in annex A of this Experimental Specification.

NOTE – The name of a value does not imply anything about type. For example, the value 'no such service instance' has the appearance of a character string but might be assigned to a parameter whose type is integer.

1.8.2.5   State Names

This Experimental Specification specifies the states of Enhanced Forward CLTU service providers. States may be referred to by number (e.g., state 3) or by name. State names are always shown in single quotation marks (e.g., 'active').

1.8.2.6   SLE-PDU Names

The names of SLE-PDUs appear in mixed-case (e.g., cltuBindInvocation).

1.8.2.7   Data Type Definitions

Data type definitions for the Enhanced Forward CLTU service are presented in annex A in the form of a set of ASN.1 modules. Regardless of the conventions used elsewhere in this Experimental Specification, the text of the ASN.1 modules is typeset entirely in a fixed-width font.

1.8.3   OTHER CONVENTIONS

This Experimental Specification uses the conventions specified in reference [1].

1.9   REFERENCES

The following documents contain provisions that, through reference in this text, constitute provisions of this Experimental Specification. At the time of publication, the editions indicated were valid. All documents are subject to revision, and users of this Experimental Specification are encouraged to investigate the possibility of applying the most recent editions of the documents indicated below. The CCSDS Secretariat maintains a register of currently valid CCSDS Recommended Standards.

NOTES

1   A list of informative references is provided in annex I.

2   This document takes advantage of the harmonized terminology introduced by restructured documentation of the space link protocols (references [2], [3], and [4]). From an interoperability point of view, they do not introduce any incompatibilities with respect to the original set of space link protocol documents (informative references [I4], [I5], [I6], and [I7]).

[1]   *Cross Support Reference Model—Part 1: Space Link Extension Services*. Recommendation for Space Data System Standards, CCSDS 910.4-B-2. Blue Book. Issue 2. Washington, D.C.: CCSDS, October 2005.

[2] *TC Synchronization and Channel Coding*. Recommendation for Space Data System Standards, CCSDS 231.0-B-2. Blue Book. Issue 2. Washington, D.C.: CCSDS, September 2010.

[3] *TC Space Data Link Protocol*. Recommendation for Space Data System Standards, CCSDS 232.0-B-2. Blue Book. Issue 2. Washington, D.C.: CCSDS, September 2010.

[4] *Communications Operation Procedure-1*. Recommendation for Space Data System Standards, CCSDS 232.1-B-2. Blue Book. Issue 2. Washington, D.C.: CCSDS, September 2010.

[5] *Time Code Formats*. Recommendation for Space Data System Standards, CCSDS 301.0-B-4. Blue Book. Issue 4. Washington, D.C.: CCSDS, November 2010.

[6] *Information Technology—Abstract Syntax Notation One (ASN.1): Specification of Basic Notation*. International Standard, ISO/IEC 8824-1:2002. 3rd ed. Geneva: ISO, 2002.

[7] *Information Technology—Open Systems Interconnection—Basic Reference Model: The Basic Model*. International Standard, ISO/IEC 7498-1:1994. 2nd ed. Geneva: ISO, 1994.

[8] *Space Link Extension—Forward CLTU Service Specification*. Recommendation for Space Data System Standards, CCSDS 912.1-B-3. Blue Book. Issue 3. Washington, D.C.: CCSDS, July 2010.

[9] *AOS Space Data Link Protocol*. Recommendation for Space Data System Standards, CCSDS 732.0-B-2. Blue Book. Issue 2. Washington, D.C.: CCSDS, July 2006.

[10] *TM Synchronization and Channel Coding*. Recommendation for Space Data System Standards, CCSDS 131.0-B-2. Blue Book. Issue 2. Washington, D.C.: CCSDS, August 2011.

2   DESCRIPTION OF THE ENHANCED FORWARD CLTU SERVICE

2.1   OVERVIEW

The Enhanced Forward CLTU service enables the user of the service to send either an stream of CLTUs (1.6.8) or a stream of AOS SL-PDUs (1.6.8) to a spacecraft via an established forward space link channel.

The service user submits SLE-FTDUs, encapsulated in SLE-SDUs, by means of the CLTU-TRANSFER-DATA operation. SLE-FTDUs contain CLTUs or AOS SL-PDUs, as appropriate to the use of the transfer service. Production of the Enhanced Forward CLTU service by the provider entails processing the SLE-FTDUs transferred by the user through the necessary transformations to (optionally) encode and modulate the Radio Frequency (RF) carrier channel providing uplink communications with the spacecraft.

The Enhanced Forward CLTU service transmits the SLE-FTDUs in the order in which they are submitted by the service user. The provider may perform checks to determine if the SLE-FTDU complies with applicable constraints, e.g., that the length of the SLE-FTDU is within the maximum size set by service management.

The data transferred by the Enhanced Forward CLTU service is processed in one of two ways, depending on the nature of the space link across which that data is transmitted.

If the space link is configured for Telecommand, each SLE-FTDU contains a single CLTU, and the provider does not interpret, interrogate, or modify the contents of a CLTU. CLTUs are transmitted bit for bit as received from the service user.

Although the CLTUs are formally defined in accordance with reference [2], the EFCLTU service does not enforce this format. The Enhanced Forward CLTU service may be used to uplink any octet-aligned bit pattern across the Telecommand space link.

If the space link is AOS, the content of the SLE-FTDUs and their resultant processing by the service provider have more options:

a) The SLE-FTDUs may each contain one or more CADUs, for which the provider of the Enhanced Forward CLTU service is configured to extract the CADUs and transmit them as received from the service user without interpretation, interrogation, or modification. However, CADUs containing AOS idle transfer frames may be inserted among the user-generated CADUs to maintain a continuous and contiguous (back-to-back) stream of CADUs, and the resulting CADU stream may be optionally convolutionally encoded prior to injection into the physical channel. Although the CADUs are formally defined in accordance with reference [9], the EFCLTU does not enforce this format. The EFCLTU service may be used to transmit any octet-aligned blocks as long as they conform to the AOS SL-PDU length configured for the transfer service instance;

b) The SLE-FTDUs may each contain one or more fixed-length AOS transfer frames, for which the provider of the Enhanced Forward CLTU service is configured to extract the transfer frames and forward-error-correction encode the transfer frames to form codeblocks, (optionally) pseudo-randomize the transfer frames or codeblocks, and prepend Attached Sync Markers (ASMs) to the (possibly randomized) transfer frames or codeblocks to form the CADUs that are to be transmitted. Transfer frames are processed without interpretation or interrogation. However, they may be modified as a result of forward error correction encoding and/or pseudo-randomization to form CADUs. CADUs containing AOS idle transfer frames may be inserted among the user-generated CADUs to maintain a continuous and contiguous (back-to-back) stream of CADUs, and the resulting CADU stream may be optionally convolutionally encoded prior to injection into the physical channel. Although the CADUs are formally defined in accordance with reference [9], the EFCLTU does not enforce this format. The EFCLTU service may be used to transmit any octet-aligned blocks as long as they conform to the AOS SL-PDU length configured for the transfer service instance.

The operations defined in section 3 of this Experimental Specification enable a single Enhanced Forward CLTU service user to interact with an Enhanced Forward CLTU service provider to:

a) establish an association between the user and the provider;

b) send annotated SLE-FTDUs to the provider;

c) obtain notifications and reports regarding status, configuration, and performance of the service;

d) temporarily stop and later restart the sending of SLE-FTDUs;

e) release an association.

The Sequence Controlled (AD) and Expedited (BD) Services, as defined in the Communications Operation Procedures (COP) Recommended Standard (reference [3]) are accomplished by higher-layer SLE services.

The Enhanced Forward CLTU service is provided only in the online delivery mode, as defined in reference [1]. The offline delivery mode is the subject of further study.

The provision of Enhanced Forward CLTU service for access to one physical channel by one service user constitutes one instance of service. Only a single service instance of the Enhanced Forward CLTU service may exist per physical channel at a time.

2.2   SPACE LINK EXTENSION REFERENCE MODEL

2.2.1   INTRODUCTION

The Enhanced Forward CLTU service is specified within the framework defined by the SLE Reference Model (reference [1]).   The following subsections summarize selected concepts from the SLE reference model.

2.2.2   ABSTRACT OBJECT

An abstract object is a functional entity that interacts with other abstract objects. Objects are of different types, which determine their function and behavior. An object is characterized by its interfaces (one or more), which are called abstract ports, and the operations that are made available through those interfaces.

2.2.3   ABSTRACT SERVICE

An abstract service is the capability provided by a set of operations that an abstract object exposes at one or more of its abstract ports.

NOTE   –   The concept of an abstract service is to be distinguished from the concept of an (N)-service as defined in the OSI Basic Reference Model (reference [7]). The definition of (N)-service is in terms of the capability provided by one layer in the OSI architecture to the layer above it. The definition of abstract service is in terms of the capability provided by one abstract object to another abstract object. In a cross support scenario, where one Agency is providing an SLE service to another Agency, the object that provides the service typically is associated with one Agency, and the object that uses the service typically is associated with the other Agency.

2.2.4   ABSTRACT BINDING

When two abstract ports have an association established between them, they are said to be bound. The act of establishing such an association is called abstract binding. One object (the initiator) invokes a bind operation which is accepted (or rejected) by another object (the responder).

2.2.5   SERVICE USER/PROVIDER

An object that offers a service to another by means of one or more of its ports is called a service provider (provider). The other object is called a service user (user). An object may be a provider of some services and a user of others.

The terms user and provider are used to distinguish the roles of two interacting objects. In this Experimental Specification, when two objects are involved in provision of a service, the object closer to the space link is considered to be the provider of the service, and the object further from the space link is considered to be the user.

2.2.6   OPERATION

An operation is a procedure or task that one object (the invoker) can request of another (the performer) through a bound port pair.

The terms invoker and performer are used to describe the interaction between two objects as the operations that constitute the service occur. One object invokes an operation that is performed by the other. For most services, each object invokes some operations and performs others.

2.3   SERVICE MANAGEMENT

SLE service management determines the number and schedule of Enhanced Forward CLTU service instances to be provided, the resources required to enable those service instances, and the initial configuration of all service instances and their supporting resources. SLE service management is the subject of separate CCSDS Recommended Standards.

The SLE Reference Model (reference [1]) distinguishes between service provision and service production:

a) service provision makes available to the user the operations necessary to obtain the service;

b) service production transforms the Enhanced Forward CLTU data channel to the RF carrier channel.

Certain configuration parameters are associated with provision of Enhanced Forward CLTU service while others are associated with production. Configuration parameters that are associated with the production, such as bit rate and modulation index, can potentially impact SLE Complex resources. Consequently, only service management may modify production configuration parameter values. The Enhanced Forward CLTU service user may modify some provision configuration parameters through operations specified in this Experimental Specification.

2.4 ARCHITECTURE MODEL—FUNCTIONAL VIEW

2.4.1 FORWARD TC SPACE LINK PROCESSING AND FORWARD AOS SPACE LINK PROCESSING SLE FUNCTIONAL GROUP

A combination of functions of the Forward Telecommand (TC) Space Link Processing and the Forward AOS Space Link Processing SLE Functional Group (SLE-FG) (shown in figure 2-1) produces the Enhanced Forward CLTU service.

As described in reference [1], the Forward TC Space Link Processing SLE-FG consumes one Telecommand CLTU data channel consisting of a stream of CLTU SLE-SDUs. The SLE-SDUs that encapsulate the CLTUs contain control and annotation data that specify radiation time and other parameters to aid in processing the data (see 3.7). The Forward TC Space Link Processing SLE-FG uses these data to extract the CLTUs and inject them into the Telecommand physical channel.

As described in reference [1], the functions of the Forward AOS Space Link Processing SLE-FG include consuming 'proto' transfer frames, generating idle transfer frames as needed, optionally Reed-Solomon encoding of the transfer frames, optionally generating bit transitions in the transfer frames, attaching synchronization markers to the frames, synchronously inserting the frames into a continuous bitstream, and optionally convolutionally encoding the bitstream.

NOTES

1   The Forward AOS Space Link Processing SLE-FG performs other functions besides those listed above. The above list includes only those AOS Space Link Processing SLE-FG functions that are relevant to the production of the Enhanced Forward CLTU service.

2   The representation in figure 2-1 of the EFCLTU transfer service port as spanning both the Forward TC Space Link Processing and Forward AOS Space Link Processing SLE-FGs is for description purposes only and is not intended to imply that the Reference Model has been or will be changed. However, the development of real CSTSes that include capabilities of this Experimental Specification may cause the Reference Model to be revisited and modified.

3   Per physical forward channel, only a single instance of the Enhanced Forward CLTU service is supported at any point in time.
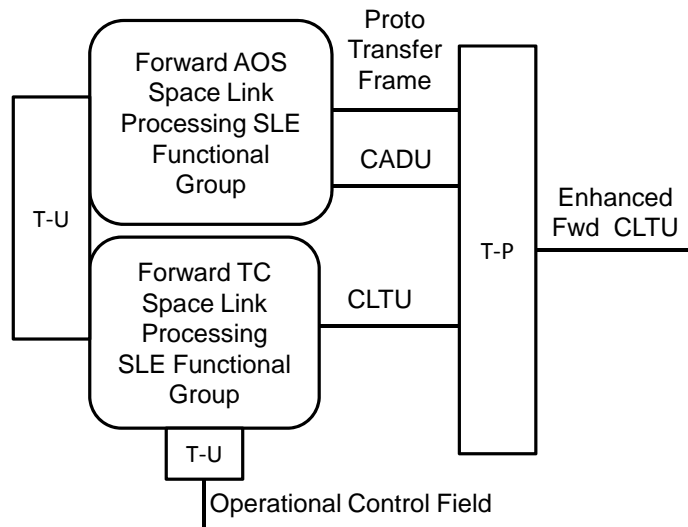
Figure 2-1: Forward TC and AOS Space Link Processing SLE-FGs

The combined Forward TC Space Link Processing and Forward AOS Space Link Processing SLE-FGs perform the following functions with respect to the Enhanced Forward CLTU service:

a) consumes one SLE-FTDU data channel and extracts SLE-FTDUs encapsulated in SLE-SDUs;

b) optionally consumes one OCF data channel and extracts the CLCWs; based on the values in the CLCWs, the Enhanced Forward CLTU service determines whether the physical channel is available;

NOTE – CLCWs may be ignored, as an option set by service management. (See 3.2.9 and table 3-11.)

c) performs the following:

1) for SLE-FTDUs carrying CLTUs that are to be transmitted over a Telecommand space link, generates the acquisition sequence and idle sequence on the physical channel in accordance with the PLOP in effect (PLOP-1 or PLOP-2);

2) for SLE-FTDUs carrying AOS SL-PDUs that are to be transmitted over an AOS data link:

i) as required and configured via service management, extracts one or more SL-PDUs from each SLE-FTDU;

ii) generates a fixed-length idle SL-PDU whenever an SLE-FTDU is not available (ASCP).

NOTE – When the SL-PDUs are CADUs, the idle SL-PDUs generated by the provider are idle CADUs. The ability to generate idle CADUs and multiplex CADUs is not currently supported by CCSDS Space Link Recommended Standards (see references [9] and [10]). The multiplexing of CADUs is strictly local to this Experimental Specification and not intended to replace or redefine the Space Link Recommended Standards. However, if CADU multiplexing is to be included in a future real CCSDS Recommended Standard, modifications to the Space Link Recommended Standards may be necessary.

iii) if the SL-PDUs are transfer frames, forward error correction-encodes each transfer frame to form a codeblock;

iv) if the SL-PDUs are transfer frames, pseudo-randomizes each codeblock if production is configured via service management to do so;

v) if the SL-PDUs are transfer frames, attaches the appropriate ASM; and

vi) if configured via service management to do so, convolutionally encodes the resulting stream of CADUs.

NOTE – The AOS data link functions listed above are paraphrased versions of requirements for the Forward AOS Space Link Processing SLE-FG (see reference [1]), modified to conform to the terminology necessary to integrate both AOS and TC functions into a single SLE transfer service. There is no intent in this Experimental Standard to change the terminology in reference [1]. However, the development of real CSTSes that include capabilities of this Experimental Specification may cause the Reference Model to be revisited and modified.

3) utilizes the underlying antenna steering capabilities provided by the ground element;

4) modulates the bit/symbol stream on the physical channel onto the ground-to-space RF carrier; and

5) radiates the signal to the spacecraft.

2.4.2   ENHANCED FORWARD CLTU SERVICE PRODUCTION AND PROVISION

Enhanced Forward CLTU **production** is concerned with radiating a symbol stream produced from a stream of SLE-SDUs according to the SLE-FTDU control and annotation information in the SLE-SDU and according to the configuration set up by service management.

Enhanced Forward CLTU service **provision** is concerned with receiving a stream of SLE-SDUs from an Enhanced Forward CLTU service user. Service provision addresses such

matters as when service is provided (e.g., service start and stop times), and how service is provided (e.g., which events are notified to the user).

The SLE-SDUs consumed by the Enhanced Forward CLTU service are sent by the service user by means of the Enhanced Forward CLTU service operations defined in section 3. These operations also provide additional functionality to facilitate the provision of service, i.e., enabling the exchange of SLE-SDUs across a remote interface. The service operations are realized as SLE-PDUs which are exchanged between the Enhanced Forward CLTU service provider and the Enhanced Forward CLTU service user by means of an underlying communications service. The general relationship between SL-DUs, SLE-SDUs, and SLE-PDUs is illustrated in figure 2-2.
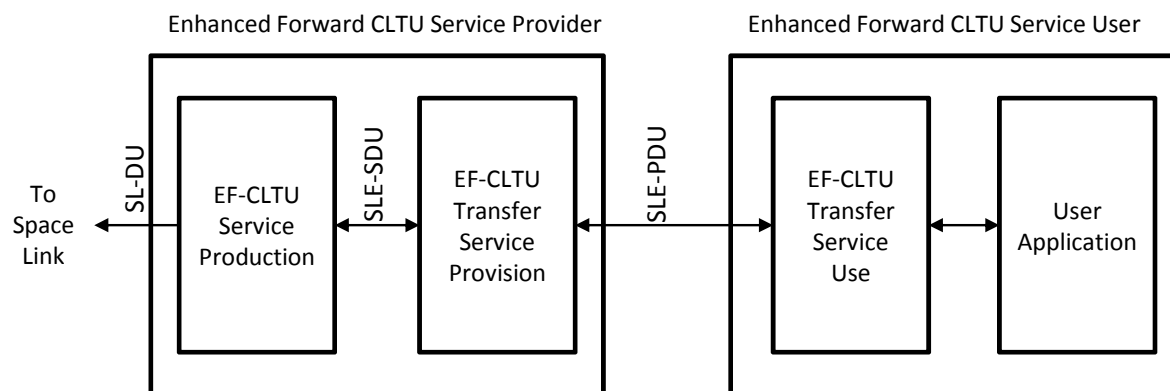
Figure 2-2: Enhanced Forward CLTU Service Production and Provision

Production of the Enhanced Forward CLTU service by the provider occurs during the space link session; in general, service production will largely overlap with service provision. Production status affects the provision of the service, as specified in sections 3 and 4, and reviewed in annex B.

2.5   ARCHITECTURE MODEL—CROSS-SUPPORT VIEW

The management and control of the production and provision of the SLE transfer services is described in general terms in reference [1]. Figure 2-3 shows an example operational scenario and the related binding of the Enhanced Forward CLTU transfer service ports and SLE management ports. This example shows an SLE Complex with one Forward Space Link Processing SLE-FG instance; it is providing one instance of Enhanced Forward CLTU service to a Mission Data Operations System (MDOS). As this figure shows, only a single Mission User Entity can use the Enhanced Forward CLTU service provided by a single Forward Space Link Processing SLE-FG.
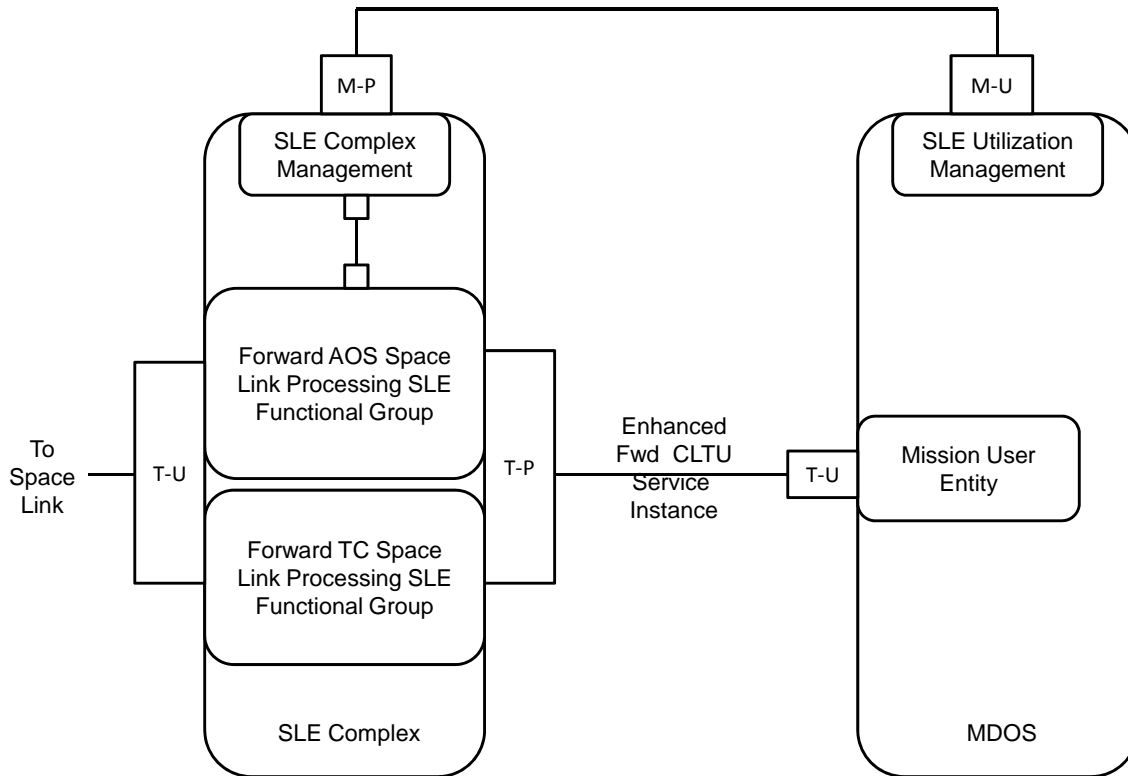
Figure 2-3: Example of Management and Provision of Enhanced Forward CLTU Service

2.6   FUNCTIONAL DESCRIPTION

2.6.1   GENERAL

This subsection describes the Enhanced Forward CLTU service with respect to scheduling, configuration, underlying services, provider states, and protocol considerations.

2.6.2   SCHEDULING AND CONFIGURATION

SLE Utilization Management negotiates with SLE Complex Management to establish mutually agreed-upon SLE service packages. Among other things, SLE service packages specify what service instances are to be provided, when those services are to be provided, and what resources are needed to enable those services.

Service packages also specify the initial values of the mission-dependent parameters required for service production and provision. Enhanced Forward CLTU service production parameters include such things as which forward error correction to apply (if any), PLOP, bit rate, modulation index, and subcarrier frequency. Provision parameters include such things as scheduled start and stop times of the Enhanced Forward CLTU service instance.

Service production is guaranteed to occur only as needed to support service packages that have been scheduled and mutually agreed upon by SLE Complex Management and SLE Utilization Management. Service provision occurs only within the bounds of the agreed upon schedule of service instances and only during those periods when there is an association between the service provider and the service user.

2.6.3   UNDERLYING SERVICES

The Enhanced Forward CLTU service does not depend on any other SLE transfer service. Provision of the Enhanced Forward CLTU Transfer service does depend on:

   a) service management for scheduling, resources, and configuration; the schedule for a Enhanced Forward CLTU service instance must be compliant with the schedule of the underlying equipment such as antennas, etc.;

   NOTE –   The Enhanced Forward CLTU service relies on service management actions for establishment of the space link, management of the PLOP and, when possible, recovery from production interruption.

   b) the availability of a suitable communications service to enable the exchange of information between the Enhanced Forward CLTU service user and provider; and

   c) the functioning of CLTU production resources (e.g., modulator(s), up-converter) to produce the forward physical channel.

2.6.4   PROTOCOL DESCRIPTION

2.6.4.1   CLTU Operations

The operations that constitute the Enhanced Forward CLTU service are listed in table 2-1. Section 3 of this Experimental Specification contains the detailed specification of these operations.

NOTE –   The Enhanced Forward CLTU service operations are identical to those of the Forward CLTU service (reference [8]), with the exception of the CLTU-TRANSFER-DATA and CLTU-GET-PARAMETER operations, which extend and/or modify the CLTU-TRANSFER-DATA and CLTU-GET-PARAMETER operations, respectively, of reference [8].

Table 2-1: Enhanced Forward CLTU Service Operations

Service Operation	Invoked By	Purpose	Con-firmed
CLTU-BIND	Initiator (User)	To establish an association with the provider	Yes
CLTU-UNBIND	Initiator (User)	To release an association previously established by a CLTU-BIND operation	Yes
CLTU-START	User	To request that the SLE service provider prepare to accept CLTU-TRANSFER-DATA operations	Yes
CLTU-STOP	User	To request that the SLE service provider stop service provision and production.	Yes
CLTU-TRANSFER-DATA	User	To transfer an SLE-FTDU to the service provider	Yes
CLTU-ASYNC-NOTIFY	Provider	To notify the user of an event affecting production or provision of the Enhanced Forward CLTU service	No
CLTU-SCHEDULE-STATUS-REPORT	User	To request that the provider send a status report immediately or periodically, or stop reporting	Yes
CLTU-STATUS-REPORT	Provider	To send a status report to the user	No
CLTU-GET-PARAMETER	User	To ascertain the value of an SLE service parameter (see table 3-11)	Yes
CLTU-THROW-EVENT	User	To forward an event that requires Complex Management to take the actions defined for this event	Yes
CLTU-PEER-ABORT	User or Provider	To notify the peer system that the local system detected an error that requires the association to be terminated	No

2.6.4.2   States of the Service Provider

Once an Enhanced Forward CLTU service instance is created, the Enhanced Forward CLTU service provider is in one of three states, as follows:

a)   State 1 ('unbound'): In state 1, all resources required to enable the provision of the Enhanced Forward CLTU service have been allocated, and all objects required to provide the service have been instantiated. However, no association yet exists between the user and the provider (i.e., the Enhanced Forward CLTU transfer service provider port is not bound).

b) State 2 ('ready'): In state 2, an association has been established between the user and the provider, and they may interact by means of the operations described in section 3 of this Experimental Specification. However, sending of SLE-FTDUs from the user to the provider (by means of the CLTU-TRANSFER-DATA operation) is not permitted. The user may enable the delivery of SLE-FTDUs by means of the appropriate service operation (CLTU-START), which, in turn, will cause the provider to transition to state 3 ('active').

c) State 3 ('active'): State 3 resembles state 2 ('ready'), except that now the user can send SLE-FTDUs and the provider is enabled to radiate CLTUs or CADUs to the spacecraft. The service continues in this state until the user invokes the CLTU-STOP operation to cause the provider to suspend transmission of CLTUs/CADUs and transition back to state 2.

A simplified state transition diagram for the Enhanced Forward CLTU service provider is shown in figure 2-4. A detailed state transition matrix is provided in 4.2.
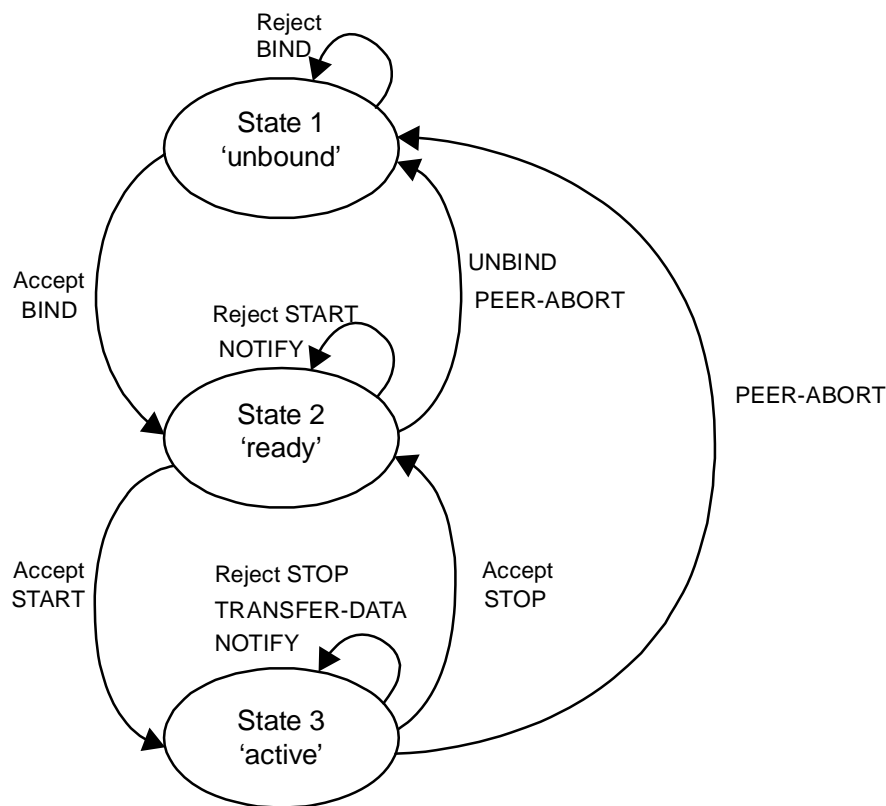
Figure 2-4: Simplified Enhanced Forward CLTU Service Provider State Transition Diagram

2.6.4.3 Terminating an Association

An association is released normally when a CLTU-UNBIND is issued by the user (the initiator of the association) and accepted by the provider (the responder). An association may be aborted

by either the user or the provider by means of the CLTU-PEER-ABORT operation. An association may also be aborted because of a failure in the underlying communications system. Such failures are signaled to the local application entity by the 'protocol abort' event described in 4.1.5.

2.6.4.4   Effects of Association Termination

The production processing of SLE-FTDUs stops immediately following the termination of an association, except for an SL-PDU in the process of being radiated. Any buffered SLE-FTDUs are discarded. The only exception to this occurs when the association is terminated due to a protocol abort, and the `protocol-abort-mode` option has been set to 'continue'; in this case, production processing of SLE-FTDUs continues, and buffered SLE-FTDUs are not discarded.

When an association is terminated, no further operations can be exchanged between the user and the provider. The systems may re-establish an association via a new CLTU-BIND operation, if that is consistent with the schedule for provision of service. Status information is not preserved after an association terminates and is not available to the new association with the following exceptions:

a)   statistics reported by means of the CLTU-STATUS-REPORT operation, such as the number of SLE-FTDUs processed (see 3.10), shall be accumulated for the entire service instance provision period;

b)   parameters that serve to relate notifications on an activity to operations that triggered this activity will not be altered when the association is released or aborted.

2.6.4.5   Technology-Specific Aspects—Interoperability and the Underlying Communications Service

This Experimental Specification defines the Enhanced Forward CLTU service. Provision of the Enhanced Forward CLTU service in a real system also requires a specification of how the service is mapped to a communications service, such that all invocations and returns of the Enhanced Forward CLTU service operations can be conveyed between the user and the provider. In order not to restrict the applicability of this Experimental Specification to a specific communications technology, as few assumptions as possible have been made about the characteristics of the underlying communications service (see 1.3.1).

The service interface between the user and the provider is specified in this Experimental Specification in terms of the operations that the service provides. Those operations are realized by mapping the service operation invocations and returns to protocol data units that can be conveyed by means of the underlying communications service. This Experimental Specification conceptualizes such mapping in two parts:

a)   Enhanced Forward CLTU service operation invocations and returns (defined in section 3) are mapped to SLE-PDUs (defined in annex A);

b) SLE-PDUs are mapped to protocol data units that can be conveyed by means of the underlying communications service.

The mapping of Enhanced Forward CLTU service operation invocations and returns to SLE-PDUs is specified by this Experimental Specification. The mapping of SLE-PDUs to an underlying communications service is intentionally outside the scope of this Experimental Specification (e.g., so that the Enhanced Forward CLTU service may be mapped to more than one communications technology). In order to achieve interoperability, the user and provider must conform not only to this Experimental Specification but also to an agreed upon specification of the mapping of the Enhanced Forward CLTU service to the underlying communications service. The specification of a mapping of the Enhanced Forward CLTU service onto a particular communications service must address such points as:

a) selection of communications network(s) to ensure connectivity;

b) compatible configuration of protocol stacks (e.g., timeout values);

c) specification of port-identifiers, and their translation onto the communications technology;

d) specification of security-related information.

Figure 2-5 illustrates a communications realization of the Enhanced Forward CLTU service that results from such a mapping. The specification of such mappings is the subject of separate CCSDS Recommended Standards.
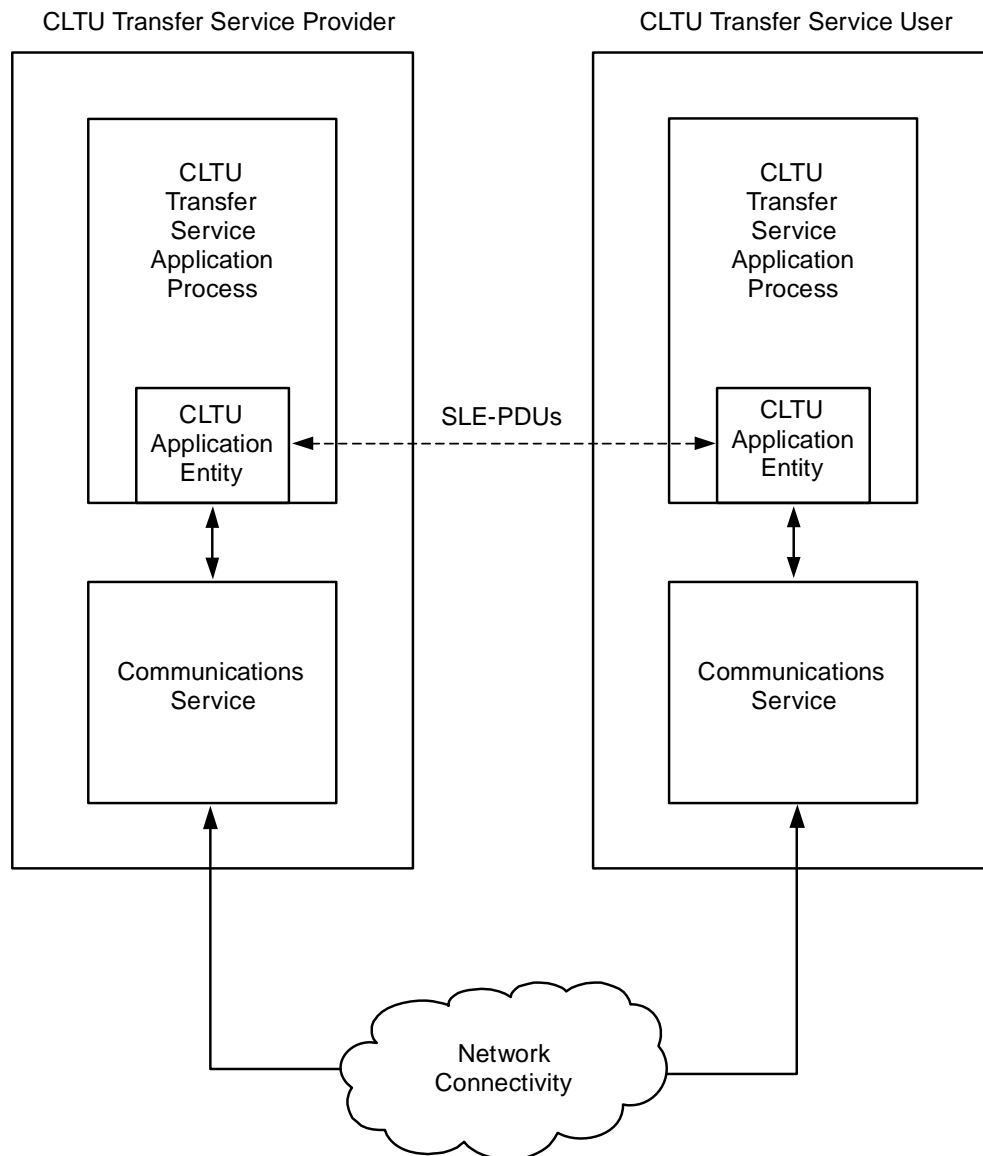
Figure 2-5: Communications Realization of Enhanced Forward CLTU Service

Because the operations of the Enhanced Forward CLTU service are relatively simple, once an association is in place between the service user and the service provider, the technology-specific elements involved in the exchange of SLE-PDUs are generally minor. However, the way an association is established (i.e., the binding) tends to vary significantly depending on the communications technology in use. Nonetheless, the CLTU-BIND and CLTU-UNBIND operations as specified in this document are intended to be 'technology neutral'. This neutrality is achieved as described in the following paragraphs.

For purposes of the communications mapping, the endpoints of an SLE association are identified by port identifiers, namely, an 'initiator port identifier' and a 'responder port identifier'. The port identifiers represent all the technology-specific addressing information needed to establish communications between the user and provider and to route SLE-PDUs

between them. The initiator port identifier identifies the endpoint that will invoke the CLTU-BIND operation (initiator). The responder port identifier identifies the endpoint that will perform the CLTU-BIND operation (responder). Generally speaking, the information represented by a port identifier consists of:

a) information needed to route data between two real systems over a communications channel or network; and

b) information needed to route data within a real system to a particular application entity.

For example, the information represented by a port identifier might be the combination of an Internet Protocol (IP) network address and a Transmission Control Protocol (TCP) port number or the combination of an OSI network address and an associated set of Service Access Points (SAPs).

The exact relationship between SLE port identifiers and communications ports provided by the underlying communications service must be specified by the mapping of the Enhanced Forward CLTU service to the underlying communications service.

One possible mapping of the SLE transfer service to the TCP/IP communications service is specified in informative reference [I8]. As part of this mapping, also issues such as sizing of TCP buffers in accordance with the bandwidth-delay product of the communication link and ways to manage relative priority of transfer services concurrently using the same connectivity are to be addressed.

In order for an SLE association to be established, SLE Complex Management and SLE Utilization Management must agree beforehand on the responder port identifier for the association. The responder needs the information represented by the responder port identifier to ensure that resources are allocated to recognize and respond to a CLTU-BIND invocation for that association. The initiator needs the information to ensure that the CLTU-BIND invocation will be communicated to the appropriate responder.

In general, it is not necessary for SLE Complex Management and SLE Utilization Management to agree beforehand on the initiator port identifier for the association. Rather, the initiator should communicate that information to the responder in conjunction with the CLTU-BIND invocation. The exact means by which the initiator port identifier is provided to the responder is technology-specific and must be specified by the mapping of the Enhanced Forward CLTU service to the underlying communications service.

The responder port identifier is included as a parameter of the CLTU-BIND operation. While it is only necessary that the SLE application entity communicate the information represented by the port identifiers to the underlying communications service, the responder port identifier is provided as a parameter of the CLTU-BIND operation to allow for the possible simplification of the implementation of a gateway.

The information represented by the responder port identifier is technology-specific. In order to define the CLTU-BIND operation in a way that is not technology-specific, the

`responder-port-identifier` parameter of the CLTU-BIND operation is defined to be a logical name. A logical name is an arbitrary identifier that has an appropriately chosen and agreed-upon translation to technology-specific information. Prior to the start time of a service instance, SLE Complex Management and SLE Utilization Management must mutually agree upon the value of the responder port identifier (and its translation) applicable to that service instance. The actual process of translating logical names to technology-specific information is considered a local matter. The translation methodology may rely on simple techniques such as look-up tables or may use more elaborate mechanisms such as naming or directory services.

The above discussion describes the case that both the user and provider applications are implemented using the same communications service. It is possible to achieve interoperability even if the user and provider use different communications services. However, in that case interoperability requires the use of an appropriate gateway.

2.6.4.6 Buffering

The Enhanced Forward CLTU service buffers the CLTUs for the primary purpose of maintaining radiation of a steady stream of CLTUs despite variable latency over the ground communications channel. All transfers of CLTUs from the service user to the service provider must occur within the scheduled service instance provision period.

2.7 OPERATIONAL SCENARIO

Prior to the actual provision of service, start and stop times for both the space link session and the associated Enhanced Forward CLTU service instance are negotiated between SLE Complex Management and SLE Utilization Management. Configuration and other information needed to enable the service are also agreed upon.

Some time before the scheduled start time of the Enhanced Forward CLTU service instance, the service instance is created by SLE Complex Management. Initially, the service provider is in state 1 ('unbound'). At the scheduled start time of the space link session, the SLE Complexes involved establish the forward link to the spacecraft and initiate the production of Enhanced Forward CLTU service. Typically (but not necessarily) the start time of the service instance will precede by a small margin the start time of the space link session to allow the user to bind to the service before the start of the space link session.

The following illustrates a typical sequence of operations between the user and the provider of an Enhanced Forward CLTU service. A complete definition of the operations is found in section 3; the formal specification of provider behavior is presented in section 4.

    a) The user invokes the CLTU-BIND operation to establish an association.

    b) The provider, when configured to monitor uplink status by examining the No RF Available and/or No Bit Lock flags returned from the spacecraft in the Communications Link Control Word (CLCW), performs the necessary operations to

receive the Operational Control Field (OCF) as provided by the Return Frame Processing SLE-FG.

c) The provider monitors equipment readiness, the status of the physical channel, and (when configured to do so) the uplink status. When production status changes to 'operational' the provider sends CLTU-ASYNC-NOTIFY to the user.

NOTE – Modulation of the uplink signal with acquisition sequence and idle sequence or idle CADUs, in accordance with the PLOP in effect, is under the control of service management.

d) The user sends CLTU-START, and the provider transitions to state 3, 'active'.

e) The user sends a CLTU-TRANSFER-DATA invocation to the provider. The provider verifies the invocation, and if acceptable, buffers the SLE-FTDU until the specified `earliest-radiation-time` is reached.

f) Additional CLTUs may be sent by the user and buffered.

NOTE – The user may perform steps d), e) and f), invoking CLTU-START and CLTU-TRANSFER-DATA, before the production status becomes operational as described in step c).

g) At the time specified for start of radiation of the first SLE-FTDU in the buffer, if production status is operational, the first SLE-FTDU is released to production processing. If no earliest radiation time was specified in the CLTU-TRANSFER-DATA operation, the first SLE-FTDU is released to production processing as soon as received, or whenever the production status becomes operational, if that is later.

h) If the SLE-FTDU contains a CLTU, the CLTU is radiated as soon as it is released to production processing.

i) If the SLE-FTDU contains one or more AOS transfer frames, the production process extracts the transfer frames and converts each into a CADU. The production process inserts CADUs containing idle transfer frames as necessary to maintain a contiguous and continuous data stream, The resulting data stream may optionally be convolutionally encoded and then injected into the physical channel and modulated onto the RF carrier. The signal is radiated to the spacecraft.

j) If the SLE-FTDU contains one or more CADUs, the production process extracts the CADUs from the SLE-FTDU. The production process inserts CADUs containing idle transfer frames as necessary to maintain a contiguous and continuous data stream. The resulting data stream may optionally be convolutionally encoded and then injected into the physical channel and modulated onto the RF carrier. The signal is radiated to the spacecraft.

NOTE – The start of radiation of a CADU may be delayed by the time needed to complete the radiation of an idle CADU.

k) Successive SLE-FTDUs are processed in similar fashion after the delay period (if any) specified in the preceding SLE-FTDU is satisfied.

l) The user transfers the last SLE-FTDU to the provider.

m) The provider completes processing of the buffered SLE-FTDUs. When the provider's SLE-FTDU buffer is empty, it sends CLTU-ASYNC-NOTIFY to inform the user.

n) The user sends CLTU-STOP and the provider transitions to state 2, 'ready'.

o) The user performs CLTU-UNBIND to release the association.

2.8 SECURITY ASPECTS OF THE SLE ENHANCED FORWARD CLTU TRANSFER SERVICE

2.8.1 SECURITY BACKGROUND/INTRODUCTION

The SLE transfer services explicitly provide authentication and access control. Additional security capabilities, if required, are levied on the underlying communication services that support the SLE transfer services. The SLE transfer services are defined as layered application services operating over underlying communication services that must meet certain requirements but which are otherwise unspecified. Selection of the underlying communication services over which real SLE implementations connect is based on the requirements of the communicating parties and/or the availability of CCSDS-standard communication technology profiles and proxy specifications. Different underlying communication technology profiles are intended to address not only different performance requirements but also different security requirements. Missions and service providers are expected to select from these technology profiles to acquire the performance and security capabilities appropriate to the mission. Specification of the various underlying communication technologies, and in particular their associated security provisions, are outside the scope of this Experimental Specification.

The SLE Enhanced Forward CLTU transfer service transfers data that is destined for a mission spacecraft. As such, the SLE Enhanced Forward CLTU transfer service has custody of the data for only a portion of the end-to-end data path between MDOS and mission spacecraft. Consequently the ability of an SLE transfer service to secure the transfer of mission spacecraft data is limited to that portion of the end-to-end path that is provided by the SLE transfer service (i.e., the terrestrial link between the MDOS and the ground termination of the ground-space link to the mission spacecraft). End-to-end security must also involve securing the data as it crosses the ground-space link, which can be provided by some combination of securing the mission data itself (e.g., encryption of the mission data within CCSDS space packets) and securing the ground-space link (e.g., encryption of the physical ground-space link). Thus while the SLE Enhanced Forward CLTU transfer service plays a role in the end-to-end security of the data path, it does not control and cannot ensure that end-to-end security. This component perspective is reflected in the security provisions of the SLE transfer services.

2.8.2   STATEMENTS OF SECURITY CONCERNS

2.8.2.1   General

This subsection identifies SLE Enhanced Forward CLTU transfer service support for capabilities that respond to security concerns in the areas of data privacy, data integrity, authentication, access control, availability of resources, and auditing.

2.8.2.2   Data Privacy (also known as Confidentiality)

This SLE Enhanced Forward CLTU transfer service specification does not define explicit data privacy requirements or capabilities to ensure data privacy. Data privacy is expected to be ensured outside of the SLE transfer service layer, by the mission application processes that communicate over the SLE transfer service, in the underlying communication service that lies under the SLE transfer service, or some combination of both. For example, mission application processes might apply end-to-end encryption to the contents of the CCSDS space link data units carried as data by the SLE transfer service. Alternatively or in addition, the network connection between the SLE entities might be encrypted to provide data privacy in the underlying communication network.

2.8.2.3   Data Integrity

The SLE Enhanced Forward CLTU service requires that each transferred CLTU be accompanied by a sequence number, which must increase monotonically. Failure of a CLTU to be accompanied by the expected sequence number causes the CLTU to be rejected (see 3.7.2.13.1d)). This constrains the ability of a third party to inject additional command data into an active Enhanced Forward CLTU transfer service instance.

The SLE Enhanced Forward CLTU transfer service defines and enforces a strict sequence of operations that constrain the ability of a third party to inject operation invocations or returns into the transfer service association between a service user and provider (see 4.2.2). This constrains the ability of a third party to seize control of an active Enhanced Forward CLTU transfer service instance without detection.

The SLE Enhanced Forward CLTU transfer service requires that the underlying communication service transfer data in sequence, completely and with integrity, without duplication, with flow control that notifies the Application layer in the event of congestion, and with notification to the Application layer in the event that communication between the service user and the service provider is disrupted (see 1.3.1). No specific mechanisms are identified, as they will be an integral part of the underlying communication service.

2.8.2.4 Authentication

This SLE Enhanced Forward CLTU transfer service specification defines authentication requirements (see 3.2.5), and defines `initiator-identifier`, `responder-identifier`, `invoker-credentials`, and `performer-credentials` parameters of the service operation invocations and returns that are used to perform SLE transfer service authentication. The procedure by which SLE transfer service operation invocations and returns are authenticated is described in annex F of the Cross Support Concept Green Book (informative reference [I3]). The SLE transfer service authentication capability can be selectively set to authenticate at one of three levels: authenticate every invocation and return, authenticate only the BIND operation invocation and return, or perform no authentication. Depending upon the inherent authentication available from the underlying communication network, the security environment in which the SLE service user and provider are operating, and the security requirements of the spaceflight mission, the SLE transfer service authentication level can be adapted by choosing the SLE operation invocation and returns that shall be authenticated. Furthermore, the mechanism used for generating and checking the credentials and thus the level of protection against masquerading (simple or strong authentication) can be selected in accordance with the results of a threat analysis.

2.8.2.5 Access Control

This SLE Enhanced Forward CLTU transfer service specification defines access control requirements (see 3.2.4), and defines `initiator-identifier` and `responder-identifier` parameters of the service operation invocations and returns that are used to perform SLE transfer service access control. The procedure by which access to SLE transfer services is controlled is described in annex F of the Cross Support Concept Green Book (informative reference [I3]).

2.8.2.6 Availability of Resources

The SLE transfer services are provided via communication networks that have some limit to the resources available to support those SLE transfer services. If these resources can be diverted from their support of the SLE transfer services (in what is commonly known as 'denial of service') then the performance of the SLE transfer services may be curtailed or inhibited. This SLE Enhanced Forward CLTU transfer service specification does not define explicit capabilities to prevent denial of service. Resource availability is expected to be ensured by appropriate capabilities in the underlying communication service. The specific capabilities will be dependent upon the technologies used in the underlying communication service and the security environment in which the transfer service user and provider operate.

2.8.2.7   Auditing

This SLE Enhanced Forward CLTU transfer service specification does not define explicit security auditing requirements or capabilities. Security auditing is expected to be negotiated and implemented bilaterally between the spaceflight mission and the service provider.

2.8.3   POTENTIAL THREATS AND ATTACK SCENARIOS

The SLE Enhanced Forward CLTU transfer service depends on unspecified mechanisms operating above the SLE transfer service (between a mission spacecraft application process and its peer application process on the ground), underneath the SLE transfer service in the underlying communication service, or some combination of both, to ensure data privacy (confidentiality). If no such mechanisms are actually implemented, or the mechanisms selected are inadequate or inappropriate to the network environment in which the mission is operating, an attacker could read the command data contained in the Enhanced Forward CLTU protocol data units as they traverse the WAN between service user and service provider.

The SLE Enhanced Forward CLTU transfer service constrains the ability of a third party to seize control of an active SLE transfer service instance, or to inject extra command data into a service instance, but it does not specify mechanisms that would prevent an attacker from intercepting the protocol data units and replacing the contents of the `data` parameter. The prevention of such a replacement attack depends on unspecified mechanisms operating above the SLE transfer service (between a mission spacecraft application process and its peer application process on the ground), underneath the SLE transfer service in the underlying communication service, in bilaterally agreed extra capabilities applied to the SLE transfer service (e.g., encryption of the `data` parameter) or some combination of the three. If no such mechanisms are actually implemented, or the mechanisms selected are inadequate or inappropriate to the network environment in which the mission is operating, an attacker could 'hijack' an established SLE Enhanced Forward CLTU transfer service instance and overwrite the commands in the protocol data units to subvert or destroy the operation of the spacecraft.

If the SLE transfer service authentication capability is not used and if authentication is not ensured by the underlying communication service, attackers may somehow obtain valid `initiator-identifier` values and use them to initiate SLE transfer service instances by which they could subvert or destroy the mission.

The SLE Enhanced Forward CLTU transfer service depends on unspecified mechanisms operating in the underlying communication service to ensure that the supporting network has sufficient resources to provide sufficient support to legitimate users. If no such mechanisms are actually implemented, or the mechanisms selected are inadequate or inappropriate to the network environment in which the mission is operating, an attacker could prevent legitimate users from communicating with their spacecraft, causing degradation or even loss of the mission.

If the provider of the SLE Enhanced Forward CLTU transfer service provides no security auditing capabilities, or if a user chooses not to employ auditing capabilities that do exist, then attackers may delay or escape detection long enough to do serious (or increasingly serious) harm to the mission.

2.8.4 CONSEQUENCES OF NOT APPLYING SECURITY

The consequences of not applying security to the SLE Enhanced Forward CLTU transfer service are possible degradation and loss of ability to command the spacecraft, and even loss of the spacecraft itself.

3 ENHANCED FORWARD CLTU SERVICE OPERATIONS

3.1 OVERVIEW

This section (3) specifies the processing of valid SLE-PDUs (i.e., those that are recognized as the invocation or return of an Enhanced Forward CLTU service operation). Subsection 3.2 specifies behaviors that are generally applicable to all operations. Subsections 3.3 through 3.13 specify individual operations. Handling of invalid SLE-PDUs is specified in subsection 4.1.

NOTE – For the purposes of this Experimental Specification, the Enhanced Forward CLTU operations are defined as later versions of the corresponding Forward CLTU operations. If and when the capabilities of the Enhanced Forward CLTU service are realized as part of an actual CSTS, the operations will be written specific to the actual CSTS.

3.2 GENERAL CONSIDERATIONS

3.2.1 RESULT OF OPERATIONS

3.2.1.1 All confirmed operations shall report on the outcome of the operation in a return, except as specified otherwise in section 4.

3.2.1.2 All returns shall include a `result` parameter that indicates whether the outcome of the operation was successful ('positive result') or unsuccessful ('negative result').

3.2.1.3 In the event of a 'negative result', the return shall also include a `diagnostic` parameter that is descriptive of the reason for the 'negative result'.

NOTE – Possible values of the `diagnostic` parameter are listed in the description of each operation.

3.2.1.4 A `diagnostic` parameter value of 'other reason' shall be returned only if no other value in the list adequately describes the reason for the 'negative result'.

3.2.2 PARAMETER TYPES

The types of all parameters shall conform to the abstract syntax specified in annex A.

NOTE – Some parameter types in annex A are chosen such that possible future extensions of the range of allowed values of a parameter will not cause a type mismatch. For example, parameters that logically are of the 'enumerated' type are specified as being of the 'named integer' type.

3.2.3 PARAMETER CHECKING

3.2.3.1   Validity checks shall be performed on the values of parameters associated with an operation.

NOTE –   Rules governing the validity of parameter values are included in the specification of individual operations. General reasons for regarding a parameter value as invalid are specified in the following subsections.

3.2.3.2   A parameter shall be treated as invalid if:

   a) its value is outside the range or not in the set of values currently permitted by service management for the given parameter;

   NOTE –   A conformant implementation shall be capable of supporting the full range or set as specified in annex A.

   b) its value is in conflict with the value of another parameter in the same invocation (e.g., if in CLTU-TRANSFER-DATA the time specified in the `earliest-radiation-time` parameter is later than the time specified in the `latest-radiation-time` parameter);

   c) its value is in conflict with the current provider configuration (e.g., the minimum delay time between CLTUs parameter as set by service management is longer than the delay-time value the CLTU-TRANSFER-DATA invocation contains).

3.2.3.3   If a parameter value is not valid, the operation shall not be performed, and, for confirmed operations, a report of 'negative result' shall be returned to the invoker.

3.2.3.4   Except as noted in 3.3.2.11, checks for invalid parameters or for other conditions that can cause a report of 'negative result' should be performed in the order in which diagnostics are listed in the descriptions of the operations, and the `diagnostic` parameter should be set to the value defined for the first problem found.

3.2.3.5   In the case that an implementation does not adhere to the sequence of checks as specified by the sequence of diagnostic values, such implementation shall specify the sequence in which the checks are actually performed.

3.2.4 ACCESS CONTROL

3.2.4.1   The Enhanced Forward CLTU service shall implement access control based on the identity of the initiator and responder. Access control is performed at two levels:

   a) the initiator must be registered at the responder and the responder must be registered at the initiator;

   b) the initiator and responder must be authorized for the given service instance.

3.2.4.2   The initiator shall have access to a registry of authorized responders and the responder shall have access to a registry of authorized initiators. These registries shall be maintained by SLE Complex Management and SLE Utilization Management, respectively.

3.2.4.3   Service management shall specify the authorized initiator and responder for each service instance.

3.2.4.4   The initiator and responder shall indicate their identity by setting the parameters `initiator-identifier` and `responder-identifier` in the CLTU-BIND operation to the values assigned by service management.

3.2.5   AUTHENTICATION

NOTE – Requirements for security depend on the application and the environment of the SLE Complexes and the MDOS (e.g., whether closed or public networks are used or if access is only from physically restricted areas). In many environments, security may be provided by the communications service transparently to the SLE application entity. This Experimental Specification does not preclude the use of security features that are provided by the communications service or the local environment, nor does it assume the availability of such features.

3.2.5.1   The Enhanced Forward CLTU service shall provide the following options with respect to level of authentication of invocations and returns of operations:

a)   'all': all Enhanced Forward CLTU invocations and returns, except the invocation of CLTU-PEER-ABORT, shall be authenticated;

b)   'bind': only the CLTU-BIND invocation and return shall be authenticated;

c)   'none': no Enhanced Forward CLTU invocations or returns shall be authenticated.

3.2.5.2   SLE Complex Management and SLE Utilization Management shall agree on the level of authentication to be required for an association between a service user and a Enhanced Forward CLTU service provider and shall configure both entities accordingly.

3.2.5.3   SLE Complex Management and SLE Utilization Management shall agree on the algorithm used to generate and check credentials parameters and make this algorithm known to the service user and service provider, together with associated parameters such as passwords or keys as necessary for the adopted algorithm.

NOTES

1   The specification of the algorithms themselves is outside the scope of this Experimental Specification.

2   The `initiator-identifier` and `responder-identifier` parameters of the CLTU-BIND operation identify the user and provider, respectively, and therefore

the applicable authentication level and algorithm necessary to generate and check credentials.

3.2.5.4   For operations for which authentication of credentials is required by terms of the agreement between SLE Complex Management and SLE Utilization Management:

    a) invocations shall include an `invoker-credentials` parameter to permit the performer to authenticate the invocation; and

    b) returns shall include a `performer-credentials` parameter to permit the invoker to authenticate the return.

3.2.5.5   For operations for which authentication is not required, the `invoker-credentials` or `performer-credentials` parameter should be set to the value 'unused' to signify that the invocation or return does not carry credentials.

3.2.6   BLOCKING AND NON-BLOCKING OPERATIONS

3.2.6.1   To support applications that may need to invoke several operations concurrently, the `invoke-ID` parameter is specified for all confirmed operations except CLTU-BIND and CLTU-UNBIND.

NOTES

1   The `invoke-ID` parameter allows the invoker to correlate a particular return to the invocation that prompted it.

2   Confirmed operations that include the `invoke-ID` parameter are non-blocking operations; those that do not are blocking operations. Unconfirmed operations are always non-blocking.

3.2.6.2   After invoking a blocking operation, the invoker shall not invoke another operation for the same service instance until the return from the blocking operation is received; except that, if the return from a blocking operation is not received in a timely manner, the invoker may invoke CLTU-PEER-ABORT to terminate the association.

3.2.6.3   After invoking a non-blocking operation, the invoker may invoke another operation without waiting for the return from the first invocation.

3.2.6.4   The value of the `invoke-ID` parameter shall be an invoker-supplied arbitrary integer value that shall be returned, unchanged, by the performer.

3.2.6.5   The invocation of a non-blocking operation shall be rejected with the `diagnostic` 'duplicate invoke id' if it includes an `invoke-ID` whose value is the same as that of another invocation that is awaiting confirmation within the context of the same service instance.

3.2.6.6 To ensure that the Enhanced Forward CLTU service behaves in a predictable manner, the effects of operations shall be as though the operations were performed in the order that their invocations were received by the performer.

3.2.6.7 The invoker may choose not to exploit the non-blocking capability and always wait for the return from a non-blocking operation before invoking another operation.

NOTE – An invoker wishing to operate in blocking mode, i.e., to invoke a new operation only after the return from the previous operation has been received, may use a constant value for the `invoke-ID` parameter. As long as a return is still outstanding, the performer will reject any further invocations.

3.2.6.8 Compliance with this Experimental Specification does not require the performer to process invocations concurrently; however, the performer must accept invocations from a non-blocking invoker and buffer and serialize them by local means not visible externally.

3.2.7 TIME

3.2.7.1 The time reference for all parameters containing a time value shall be based on Coordinated Universal Time (UTC).

3.2.7.2 The type of parameters containing a time value shall be the CCSDS Day Segmented (CDS) time code format (reference [5]) with a resolution of microseconds, an epoch of 1958-01-01, and a 16-bit day segment. Depending on the F-CLTU service provider capabilities and/or the supported mission requirements, the time tag may have either a resolution of microseconds or a resolution of picoseconds.

3.2.7.3 All time values shall be expressed to a precision of at least one-thousandenth (0.001) of a second.

3.2.7.4 All time value shall be accurate to within one-thousandenth (0.001) of a second or better.

3.2.8 DELIVERY MODES

3.2.8.1 Forward Online Delivery

3.2.8.1.1 Forward online delivery service provision shall occur at the same time as service production, i.e., during a space link session.

3.2.8.1.2 SLE-FTDUs supplied by the service user shall be buffered by the service provider until they are processed.

3.2.8.1.3 The buffer used by the service provider exists only during service provisioning.

3.2.8.1.4   The timing of SLE-FTDU processing shall be determined by the order of SLE-FTDUs in the buffer and any annotation data provided with the SLE-FTDUs.

NOTE –   The forward online delivery mode is defined in this Experimental Specification.

3.2.8.2   Forward Offline Delivery

3.2.8.2.1   Service provision and service production shall not overlap.

3.2.8.2.2   SLE-FTDUs supplied by the service user during service provision shall be buffered by the provider in persistent storage until service production.

NOTE –   The forward offline delivery mode is outside the scope of this version of this Experimental Specification.

3.2.9   SETTING OF PARAMETERS

3.2.9.1   An Enhanced Forward CLTU service provider shall permit setting of the service configuration parameters as specified in table 3-1.

3.2.9.2   The range or set of values a parameter may assume is constrained by specification of its data type (see annex A).

3.2.9.3   When the service instance is configured with a `plop-in-effect` of 'PLOP-1' or 'PLOP-2', the following managed parameters shall have the following values:

3.2.9.3.1   block-encode: 'none';

3.2.9.3.2   convolutionally-encode: 'no';

3.2.9.3.3   sl-pdu-length: 0 (zero);

3.2.9.3.4   randomize: 'no'.

NOTE –   When the `plop-in-effect` is 'PLOP-1' or 'PLOP-2', the above parameters have no effect on production processing. The above values are the values that are to be returned for their respective parameters if queried via the CLTU-GET-PARAMETER operation (3.11).

3.2.9.4   When the service instance is configured with a `plop-in-effect` of 'ASCP', the following managed parameters shall have the following values:

3.2.9.4.1   acquisition-sequence-length: 0 (zero);

3.2.9.4.2   bit-lock-required: 'no';

3.2.9.4.3   clcw-physical-channel: 'none';

3.2.9.4.4   clcw-global-VCID: the VC ID component of this parameter shall be set to 63 (the Only Idle Data virtual channel).

NOTE –   The Transfer Frame Version Number and Spacecraft ID components of the `clcw-global-VCID` parameter should be set to legitimate values for the mission spacecraft.

3.2.9.4.5   minimum-delay-time: 0 (zero);

3.2.9.4.6   plop-1-idle-sequence-length: 0 (zero);

3.2.9.4.7   rf-available-required: 'no'.

NOTE –   When the `plop-in-effect` is 'ASCP', the above parameters have no effect on production processing. The above values are the values that are to be returned for their respective parameters if queried via the CLTU-GET-PARAMETER operation (3.11).

3.2.9.5   Service management may further constrain the allowed values for a given service instance.

Table 3-1: Setting of Enhanced Forward CLTU Service Configuration Parameters

Parameter	Service Management	CLTU-START Operation	CLTU-SCHEDULE-STATUS-REPORT Operation	CLTU-THROW-EVENT Operation (NOTE 3)
acquisition-sequence-length	X			
bit-lock-required	X			
block-encode	X			
clcw-physical-channel	X			
clcw-global-VCID	X			
convolutionally-encode	X			
delivery-mode	X			
expected-cltu-identification		X		
expected-event-invocation-identification				X
sl-pdu-length	X			
idle-sl-pdu	X (see note 5)			
maximum-cltu-length	X			
maximum-reporting-cycle	X			

Parameter	Service Management	CLTU-START Operation	CLTU-SCHEDULE-STATUS-REPORT Operation	CLTU-THROW-EVENT Operation (NOTE 3)
`minimum-delay-time`	X			
`minimum-reporting-cycle`	X			
`modulation-frequency`	X			
`modulation-index`	X			
`notification-mode`	X			
`plop-1-idle-sequence-length`	X			
`plop-in-effect`	X			
`protocol-abort-mode`	X			
`randomize`	X			
`reporting-cycle`			X	
`return-timeout-period`	X			
`rf-available-required`	X			
`service-instance-provision-period`	X			
`service-version-number`	X			
`subcarrier-to-bit-rate-ratio`	X			

NOTES

1   Further details on `protocol-abort-mode` are discussed in 4.1.5. The `notification-mode` parameter is described in 3.8.2.3. Other parameters are presented and described in table 3-11. A complete list of parameters that may affect service production is to be found in the service management suite of documents (see 1.5.2).

2   The user can ascertain the current value of the parameters presented in table 3-1 *except* `idle-sl-pdu` by means of the CLTU-GET-PARAMETER operation (see table 3-11).

3   The ability to modify selected service configuration parameters using the CLTU-THROW-EVENT operation is allowed but not mandated in this Experimental Specification.

4   Although listed in table 3-1, the managed parameters `block-encode`, `convolutionally-encode`, `randomize`, `idle-sl-pdu`, `sl-pdu-length`, `modulation-frequency`, `modulation-index`, `plop-1-idle-sequence-length`, `plop-in-effect`, and `subcarrier-to-carrier-`

bit-rate-ratio are production parameters and not part of Enhanced Forward CLTU transfer service provision. They are included in the table because their status can be retrieved via the Enhanced Forward CLTU transfer service and/or they are used to configure the PLOP production processes.

5   When the Enhanced Forward CLTU service instance is used to transfer AOS transfer frames, the idle-sl-pdu managed parameter should be formatted as an AOS Idle Transfer Frame as specified in reference [9] in order to ensure proper execution of the AOS space data link protocol. However, the Enhanced Forward CLTU service does not enforce this formatting.

6   When the Enhanced Forward CLTU service instance is used to transfer CADUs, the idle-sl-pdu managed parameter should be formatted as an AOS Idle Transfer Frame as specified in reference [9] with the same synchronization and channel coding functions as applied to the AOS transfer frames in the user-generated CADUs, in order to ensure proper execution of the AOS space data link protocol. However, the Enhanced Forward CLTU service does not enforce this formatting.

3.2.10 PROVIDER BUFFERING REQUIREMENTS

3.2.10.1 The service package shall specify the amount of buffering the provider must maintain.

3.2.10.2 The amount of buffer space shall be specified in terms of the number of octets that can be stored.

3.2.10.3 The service provider shall buffer only complete SLE-FTDUs.

3.2.11 ACCOUNTING SUMMARY

Statistical information to be collected over a period of time shall always refer to the service instance provision period.

3.3   CLTU-BIND

3.3.1   PURPOSE

3.3.1.1   The initiator shall invoke the CLTU-BIND operation to establish an association between the initiator and responder as defined in 1.6.8.

3.3.1.2   The responder shall return a report of the outcome of the performance of the CLTU-BIND operation to the initiator.

3.3.1.3   Except as provided in 3.3.1.4, the initiator shall not invoke any further CLTU operations for this service instance until the return from the responder is received.

3.3.1.4   If the return from the invocation of CLTU-BIND is not received after a sufficiently long time (to be determined by service management), the initiator may attempt to recover by invoking the CLTU-PEER-ABORT operation (see 3.13) followed by another CLTU-BIND.

NOTE –   The length of the duration that constitutes 'a sufficiently long time' is determined by service management.

3.3.1.5   The CLTU-BIND operation is valid only in state 1 ('unbound') and shall be invoked only by the user.

3.3.2   INVOCATION, RETURN, AND PARAMETERS

3.3.2.1   General

The parameters of the CLTU-BIND operation shall be present in the invocation and return as specified in table 3-2.

Table 3-2: CLTU-BIND Parameters

Parameter	Invocation	Return
invoker-credentials	M	
performer-credentials		M
initiator-identifier	M	
responder-identifier		M
responder-port-identifier	M	
service-type	M	
version-number	M	C
service-instance-identifier	M	
result		M
diagnostic		C

3.3.2.2   **invoker-credentials**

The **invoker-credentials** parameter shall provide information that enables the performer to authenticate the CLTU-BIND invocation (see 3.2.5).

3.3.2.3   **performer-credentials**

The **performer-credentials** parameter shall provide information that enables the invoker to authenticate the return from the performance of CLTU-BIND (see 3.2.5).

3.3.2.4 **initiator-identifier**

The **initiator-identifier** parameter shall identify the authority on whose behalf the SLE application entity is initiating an association.

NOTES

1   The `initiator-identifier` parameter permits the responder to determine if the initiator is registered at this responder and if the CLTU-BIND operation is being invoked by the authorized initiator for this service instance.

2   Each value of the `initiator-identifier` parameter is associated with exactly one authentication level and exactly one authentication scheme.

3   If authentication based on credentials is used, this parameter may be redundant since the `initiator-identifier` value may be one constituent of the `invoker-credentials` parameter. However, the encoding may differ, and it may be convenient to have this parameter also available in 'clear text' form.

3.3.2.5 **responder-identifier**

The **responder-identifier** parameter shall identify the authority on whose behalf the responding SLE application is acting.

NOTES

1   The `responder-identifier` parameter permits the initiator to determine if the responder from which the CLTU-BIND return originates is registered at this initiator.

2   The initiator uses this parameter, if applicable, after having successfully authenticated the CLTU-BIND return to determine if this return originates from the intended responder.

3.3.2.6 **responder-port-identifier**

The **responder-port-identifier** parameter shall specify the port identifier of the responding SLE application entity with which the initiator seeks to establish an association.

NOTES

1   The value of the `responder-port-identifier` parameter is a logical name that can be translated into the technology-specific addressing information required to establish a connection with the responder using the agreed upon communications service. (See 2.6.4.5 for more information.)

2   SLE Complex Management and SLE Utilization Management must have previously agreed on the `responder-port-identifier` and its translation that is applicable to a particular instance of service.

3   The `responder-port-identifier` parameter is included in the CLTU-BIND invocation to support its possible use by particular kinds of gateways.

3.3.2.7 `service-type`

The **`service-type`** parameter shall specify the type of service that will be provided if the bind operation succeeds.[1]

3.3.2.8 `version-number`

3.3.2.8.1   The **`version-number`** parameter shall identify the version number of the Enhanced Forward CLTU service specification that is to govern this association if the CLTU-BIND succeeds.

3.3.2.8.2   `version-number` is conditionally present in the return based on the `result` parameter:

   a) if the value of `result` is 'positive result', `version-number` shall be present in the return;

   b) if the value of `result` is 'negative result', `version-number` shall not be present in the return.

3.3.2.8.3   If a provider does not support version negotiation, the `version-number` value it will accept during the BIND operation is configured by means of the managed parameter `service-version-number` (see table 3-1).

3.3.2.8.4   If the value of `result` is 'positive result', the responder shall either:

   a) accept the version proposed by the initiator by putting the same version number into the positive return; or

   b) if the responder supports version negotiation, propose a lower (earlier) version number by putting the lower version number in the return.

---

[1] For the CLTU-BIND operation, the `service-type` parameter is redundant, because the only valid value of `service-type` is 'fwdCltu'. However, it is anticipated that future work by CCSDS will result in CLTU-BIND being superseded by a generic SLE-BIND operation that is invoked with any one of several SLE service types as specified `ApplicationIdentifier` in annex A. The CLTU-BIND `service-type` parameter is provided in an attempt to facilitate such a change.

3.3.2.8.5   If the responder implementation does not support the requested version and does not support a lower version (or does not support version negotiation), the responder shall reject the bind with the `diagnostic` parameter set to 'version not supported'.

3.3.2.8.6   If the responder proposes a lower version in the return and the initiator does not support version negotiation or does not support the version proposed by the responder, the initiator shall unbind the association.

3.3.2.8.7   The `version-number` value of the Enhanced Forward CLTU service defined by this issue of this Experimental Specification shall be '101'.

NOTES

1   For purposes of version control and object identification (see annex A) the Enhanced Forward CLTU service is considered to be an advanced version of the Forward CLTU service. The high version number (above 100) essentially guarantees that the version space of the Experimental Standard will not clash with that of any actual Recommended Standards.

2   The version negotiation process as outlined above is only feasible as long as future versions of the Enhanced Forward CLTU service retain the specification of the CLTU-BIND operation.

3.3.2.9 `service-instance-identifier`

The **`service-instance-identifier`** parameter shall uniquely identify this service instance within the scope of the service-providing SLE Complex.

3.3.2.10 `result`

3.3.2.10.1 The **`result`** parameter shall specify the result of the CLTU-BIND invocation and shall contain one of the following values:

   a) 'positive result'—the CLTU-BIND operation has been performed by the responder and the association is established;

   b) 'negative result'—the CLTU-BIND operation has not been performed by the responder for the reason specified in the `diagnostic` parameter, and the association is not established.

3.3.2.11 `diagnostic`

3.3.2.11.1 If `result` is 'negative result', **`diagnostic`** shall be present and its value shall be one of the following values:

a) 'access denied'—an initiator with the `initiator-identifier` value presented in the CLTU-BIND invocation is not registered at the responder;

b) 'service type not supported'—the value of the `service-type` parameter of the CLTU-BIND invocation does not identify a service type supported by the responder;

c) 'version not supported'—the responder does not support the requested version and the responder implementation does not permit version negotiation; or the responder does not support any version of the service lower than the one requested by the initiator;

d) 'no such service instance'—the requested service instance is not defined by any agreed upon service package known to the responder;

e) 'already bound'—the service instance is already bound via a different association;

f) 'service instance not accessible to this initiator'—the initiator identified by the `initiator-identifier` parameter of the CLTU-BIND invocation does not match the authorized initiator for the service instance identified by the `service-instance-identifier` parameter;

g) 'inconsistent service type'—the value of the service-type parameter of the CLTU-BIND invocation is not 'fwdCltu', or the value of the `service-type` parameter does not match the service type of the service instance identified by the `service-instance-identifier` parameter;

h) 'invalid time'—the CLTU-BIND operation was invoked outside the service instance provision period of the service instance identified by the `service-instance-identifier` parameter;

i) 'out of service'—the responder has been taken out of service for an indefinite period by management action, i.e., `production-status` is 'halted';

j) 'other reason'—the reason for the negative result will have to be found by other means.

NOTES

1   In some implementations, under some circumstances, it may not be possible for the intended performer to provide a return in the event of the conditions indicated by `diagnostic` d), h) or i).

2   Initiators should consider that, under some conditions, CLTU-BIND may fail with no return, e.g., if the value of the `responder-port-identifier` parameter is incorrect.

3.3.2.11.2 If `result` is 'positive result', the `diagnostic` parameter shall not be present in the return.

3.3.3 EFFECTS

3.3.3.1   If `result` is 'positive result', the CLTU-Bind operation shall have the following effects:

    a) an association between the user and the provider shall be established;

    b) the provider shall transition from state 1 ('unbound') to state 2 ('ready');

    c) upon receipt of the positive return, the user may proceed to invoke other Enhanced Forward CLTU service operations, e.g., to initialize the service and enable CLTU transfer.

3.3.3.2   If `result` is 'negative result', the CLTU-Bind operation shall have the following effects:

    a) the association between the user and the provider shall not be established;

    b) the provider shall remain in state 1 ('unbound');

    c) upon receipt of the negative return:

        1) the initiator should examine the `diagnostic` parameter for the cause;

        2) the initiator may attempt to re-invoke the CLTU-BIND.

3.4 CLTU-UNBIND

3.4.1 PURPOSE

3.4.1.1   The initiator shall invoke the CLTU-UNBIND to release an association previously established by CLTU-BIND (see 3.3).

3.4.1.2   The responder shall return a report of the outcome of the performance of the CLTU-UNBIND operation to the initiator.

3.4.1.3   Except as provided in 3.4.1.4, the initiator shall not invoke any further Enhanced Forward CLTU operations for this service instance until the return from CLTU-UNBIND is received; nor shall it perform any further operations invoked by the responder; nor shall it return to the responder any further reports of the outcome of operations invoked by the responder.

NOTE –   The initiator may invoke the CLTU-UNBIND operation even if it did not yet receive all returns from previously invoked operations. The initiator should be aware that the responder may choose not to send any further returns as soon as it has received the CLTU-UNBIND invocation. It may then happen that the CLTU-UNBIND return is not received before one of the missing returns causes a 'missing return' timeout (see 4.1.3).

3.4.1.4 If the return from the CLTU-UNBIND invocation is not received after a sufficiently long time, the initiator should invoke the CLTU-PEER-ABORT operation (see 3.13) to abort the association.

NOTES

1   The length of the duration that constitutes 'a sufficiently long time' is determined by service management.

2   Following receipt of the return from CLTU-UNBIND or following the invocation of CLTU-PEER-ABORT, the initiator may issue another CLTU-BIND if otherwise permitted (e.g., if the end of the service instance provision period has not been reached).

3.4.1.5 The CLTU-UNBIND operation is valid only in state 2 ('ready')

3.4.1.6 The CLTU-UNBIND operation shall be invoked only by the initiator (i.e., the invoker of the CLTU-BIND operation that established this association).

3.4.2   INVOCATION, RETURN, AND PARAMETERS

3.4.2.1   General

The parameters of the CLTU-UNBIND operation shall be present in the invocation and return as specified in table 3-3.

Table 3-3: CLTU-UNBIND Parameters

Parameter	Invocation	Return
invoker-credentials	M	
performer-credentials		M
unbind-reason	M	
result		M

3.4.2.2   **invoker-credentials**

The **invoker-credentials** parameter shall provide information that enables the performer to authenticate the CLTU-UNBIND invocation (see 3.2.5).

3.4.2.3   **performer-credentials**

The **performer-credentials** parameter shall provide information that enables the invoker to authenticate the return from the performance of CLTU-UNBIND (see 3.2.5).

3.4.2.4 `unbind-reason`

The **unbind-reason** parameter shall indicate the reason for which the CLTU-UNBIND operation is being invoked and shall contain one of the following values:

a) 'end'—the initiator has completed the transfer of its data and is releasing the association normally: the provider may delete the service instance and release all its resources associated with it;

NOTE – If `unbind-reason` is 'end', any subsequent attempt to invoke CLTU-BIND may fail even if the service instance provision period has not expired, since the service provider may release the resources allocated to that service instance.

b) 'suspend'—the initiator is suspending usage of this service instance for an unspecified period of time; the user may or may not re-bind to the provider to continue data transfer at some time prior to the end of the service instance provision period;

c) 'version not supported'—the initiator does not support the version of the Enhanced Forward CLTU service proposed by the provider in the return from CLTU-BIND: this value of unbind-reason shall be used only if the CLTU-UNBIND is the first operation invoked following the CLTU-BIND;

d) 'other reason'—the reason for the release will have to be found by other means.

3.4.2.5 `result`

The **result** parameter shall specify the result of the CLTU-UNBIND invocation and shall always contain the following value:

'positive result'—the CLTU-UNBIND invocation is accepted by the responder and the association is released.

NOTES

1   If CLTU-UNBIND is invoked while the provider is not in the 'ready' state, the responder invokes CLTU-PEER-ABORT; if the authentication level is 'all' and the invocation of CLTU-UNBIND cannot be authenticated, the responder ignores it; there is no situation in which the response to the invocation of CLTU-UNBIND is to return 'negative result'.

2   The `result` parameter is returned for the CLTU-UNBIND operation, even though the only permitted value is 'positive result', for consistency with other confirmed operations.

3   CLTU-UNBIND is a confirmed operation in order to provide a definite indication to the initiator that the responder has performed the operation and the association is released.

3.4.3 EFFECTS

The CLTU-UNBIND operation shall have the following effects:

a) The association between the initiator and the responder shall be released, and the initiator and the responder shall cease to communicate with each other.

b) The provider shall transition to state 1 ('unbound').

c) If `unbind-reason` is 'end', the provider may terminate the service instance and release its resources.

d) If `unbind-reason` is not 'end', the initiator may attempt to re-bind at any time prior to the end of the provision period.

3.5 CLTU-START

3.5.1 PURPOSE

3.5.1.1 The user shall invoke the CLTU-START operation to request that the Enhanced Forward CLTU service provider prepare to receive CLTU-TRANSFER-DATA invocations (see 3.7).

3.5.1.2 The Enhanced Forward CLTU service provider shall return a report of the outcome of the performance of the CLTU-START operation to the user.

3.5.1.3 The CLTU-START operation shall allow the Enhanced Forward CLTU service provider to return to the user the times scheduled for start and stop of production.

3.5.1.4 CLTU-START is valid only in state 2 ('ready') and shall be invoked only by the user.

3.5.2 INVOCATION, RETURN AND PARAMETERS

3.5.2.1 General

The parameters of the CLTU-START operation shall be present in the invocation and return as specified in table 3-4.

Table 3-4: CLTU-START Parameters

Parameters	Invocation	Return
invoker-credentials	M	
performer-credentials		M
invoke-ID	M	M
first-cltu-identification	M	
start-production-time		C
stop-production-time		C
result		M
diagnostic		C

3.5.2.2 `invoker-credentials`

The **`invoker-credentials`** parameter shall provide information that enables the performer to authenticate the CLTU-START invocation (see 3.2.5).

3.5.2.3 `performer-credentials`

The **`performer-credentials`** parameter shall contain information that enables the invoker to authenticate the return from the performance of CLTU-START (see 3.2.5).

3.5.2.4 `invoke-ID`

The Enhanced Forward CLTU service provider shall return unchanged the user-supplied value of the **`invoke-ID`** parameter (see 3.2.6).

3.5.2.5 `first-cltu-identification`

3.5.2.5.1 The `first-cltu-identification` parameter shall contain the **`cltu-identification`** that the Enhanced Forward CLTU service provider shall expect to receive in the first CLTU-TRANSFER-DATA operation after the CLTU-START.

NOTE – Following certain exceptions (e.g., 'sldu expired'; see 3.8.2.3), it is necessary to invoke CLTU-STOP followed by CLTU-START. A CLTU that was being processed when CLTU-STOP was invoked may still cause a notification to be invoked after the new CLTU-START invocation. The value of the `first-cltu-identification` parameter in the new CLTU-START invocation should be chosen such that an SLE-FTDU for which processing did not yet complete can be uniquely identified.

3.5.2.6 `start-production-time`

3.5.2.6.1 If the `production-status` is not yet 'operational':

a) **`start-production-time`** shall be the estimated time at which the Enhanced Forward CLTU service provider will be able to start radiating CLTUs/CADUs;

b) in this case `start-production-time` shall be determined from the schedule, with allowance for completion of the uplink acquisition sequence.

3.5.2.6.2 If the `production-status` is 'operational' and the provider is ready to start radiating CLTUs/CADUs, `start-production-time` shall contain the time the `production-status` changed to 'operational'.

3.5.2.6.3 If there is a subsequent CLTU-START operation within a service instance, the `start-production-time` parameter shall contain the latest time the provider became ready to radiate; i.e., the most recent time the production status changed to 'operational'.

3.5.2.6.4 The `start-production-time` parameter shall be returned only if the value of the `result` parameter is 'positive result'.

3.5.2.7 `stop-production-time`

3.5.2.7.1 **`stop-production-time`** shall be the estimated time at which the provider will stop radiating CLTUs/CADUs.

NOTE – The value of the `stop-production-time` can be a time later than the end of the service instance provision period.

3.5.2.7.2 If the estimated stop time is not available to the provider, the `stop-production-time` parameter shall contain a value of 'null'.

3.5.2.7.3 The `stop-production-time` parameter shall be returned only if the value of `result` is 'positive result'.

3.5.2.8 `result`

The **`result`** parameter shall specify the result of the CLTU-START operation and shall contain one of the following values:

a) 'positive result'—the CLTU-START operation has been performed by the provider and the initiator may start invoking CLTU-TRANSFER-DATA operations;

b) 'negative result'—the CLTU-START operation has not been performed by the provider for the reason specified in the `diagnostic` parameter; the provider cannot accept CLTU-TRANSFER-DATA invocations.

3.5.2.9 `diagnostic`

3.5.2.9.1 If `result` is 'negative result', **`diagnostic`** shall be present in the return, and its value shall be one of the following:

a) 'duplicate invoke-ID'—the value of the `invoke-ID` parameter is the same as the `invoke-ID` value of a previous, outstanding operation;

b) 'out of service'—the provider has been taken out of service for an indefinite period by management action; i.e., `production-status` is 'halted';

c) 'unable to comply'—the provider is unable to accept data at this time because of a possibly temporary fault condition affecting the provider; i.e., `production-status` is 'interrupted';

d) 'production time expired'—the time period scheduled for production of SLE-FTDUs is already past;

e) 'invalid cltu-ID'—the `first-cltu-identification` parameter value is less than that of an SLE-FTDU already in the buffer;

f) 'other reason'—the reason for the negative result will have to be found by other means.

3.5.2.9.2 If `result` is 'positive result', the `diagnostic` parameter shall not be present in the return.

3.5.3 EFFECTS

3.5.3.1 If `result` is 'positive result', the CLTU-START operation shall have the following effects:

a) the provider shall transition to state 3 ('active');

b) in the 'active' state, the provider shall accept CLTU-TRANSFER-DATA invocations.

3.5.3.2 If `result` is 'negative result', the CLTU-START operation shall have the following effects:

a) the provider shall remain in state 2 ('ready');

b) if `diagnostic` is 'unable to comply':

1) the user may re-invoke the CLTU-START operation at a later time, within the constraints of the service instance provision period;

2) if the provider's complex management determines that being 'unable to comply' is more than a transient problem, the provider may invoke the CLTU-PEER-ABORT operation.

3.6   CLTU-STOP

3.6.1   PURPOSE

3.6.1.1   The user shall invoke the CLTU-STOP operation to request that the provider stop production, i.e., stop radiation, of SLE-FTDUs already received but not yet radiated.

NOTE –   Within the constraints of the service instance provision period, the user may re-enable the reception and production of SLE-FTDUs on the provider side by invoking the CLTU-START operation.

3.6.1.2   The provider shall provide a report of the outcome of the performance of the CLTU-STOP operation to the user.

3.6.1.3   The CLTU-STOP operation is valid only in state 3 ('active') and shall be invoked only by the user.

3.6.2   INVOCATION, RETURN AND PARAMETERS

3.6.2.1   General

The parameters of the CLTU-STOP operation shall be present in the invocation and return as specified in table 3-5.

Table 3-5: CLTU-STOP Parameters

Parameters	Invocation	Return
invoker-credentials	M	
performer-credentials		M
invoke-ID	M	M
result		M
diagnostic		C

3.6.2.2   **invoker-credentials**

The **invoker-credentials** parameter shall provide information that enables the performer to authenticate the CLTU-STOP invocation (see 3.2.5).

3.6.2.3   **performer-credentials**

The **performer-credentials** parameter shall provide information that enables the invoker to authenticate the return from the performance of CLTU-STOP (see 3.2.5).

3.6.2.4 `invoke-ID`

The CLTU service provider shall return unchanged the user-supplied value of the **invoke-ID** parameter (see 3.2.6).

3.6.2.5 `result`

The **result** parameter shall specify the result of the CLTU-STOP operation and shall contain one of the following values:

a) 'positive result'—the CLTU-STOP operation has been performed by the provider and the user may invoke a CLTU-UNBIND operation or a new CLTU-START operation;

b) 'negative result'—the CLTU-STOP operation has not been performed by the provider for the reason specified by the diagnostic parameter.

3.6.2.6 `diagnostic`

3.6.2.6.1 If result is 'negative result', the **diagnostic** parameter shall be present in the return and its value shall contain one of the following:

a) 'duplicate invoke-ID'—the value of the invoke-ID parameter is the same as the invoke-ID value of a previous, outstanding operation;

b) 'other reason'—the reason for the negative result will have to be found by other means.

3.6.2.6.2 If result is 'positive result', the diagnostic parameter shall not be present in the return.

3.6.3 EFFECTS

3.6.3.1 If result is 'positive result', the CLTU-STOP operation shall have the following effects:

a) the provider shall transition to state 2 ('ready') and shall no longer accept CLTU-TRANSFER-DATA invocations;

b) for SLE-FTDUs already buffered:

1) the provider shall discard any SLE-FTDUs for which radiation has not yet begun;

2) the provider shall not discard, but shall complete radiation of, a CLTU or CADU in the process of being radiated;

NOTE – An additional CLTU/CADU could be radiated after the provider receives the CLTU-STOP invocation, depending on implementation specifics.

c) if a report has been requested by the user for an SLE-FTDU that is already being radiated, the report shall be generated upon completion of processing and sent to the user by means of a CLTU-ASYNC-NOTIFY operation.

3.6.3.2 If `result` is 'negative result':

a) processing shall continue;

b) the provider state shall remain unchanged;

c) any buffered SLE-FTDUs shall not be discarded.

3.7 CLTU-TRANSFER-DATA

3.7.1 PURPOSE

3.7.1.1 The user shall invoke the CLTU-TRANSFER-DATA operation to transfer SLE-FTDUs to the provider.

3.7.1.2 The provider shall provide a report of the outcome of the performance of the CLTU-TRANSFER-DATA operation to the user.

3.7.1.3 The CLTU-TRANSFER-DATA operation is valid only in state 3 ('active') and shall be invoked only by the user.

3.7.2 INVOCATION, RETURN, AND PARAMETERS

3.7.2.1 General

The parameters of the CLTU-TRANSFER-DATA operation shall be present in the invocation and return as specified in table 3-6.

Table 3-6: CLTU-TRANSFER-DATA Parameters

Parameters	Invocation	Return
invoker-credentials	M	
performer-credentials		M
invoke-ID	M	M
cltu-identification	M	M
earliest-radiation-time	M	
latest-radiation-time	M	
delay-time	M	
report	M	
data	M	
cltu-buffer-available		M
result		M
diagnostic		C

3.7.2.2 `invoker-credentials`

The **`invoker-credentials`** parameter shall provide information that enables the performer to authenticate the CLTU-TRANSFER-DATA invocation (see 3.2.5).

3.7.2.3 `performer-credentials`

The **`performer-credentials`** parameter shall provide information that enables the invoker to authenticate the return from the performance of CLTU-TRANSFER-DATA (see 3.2.5).

3.7.2.4 `invoke-ID`

The CLTU service provider shall return unchanged the user-supplied value of the **`invoke-ID`** parameter (see 3.2.6).

3.7.2.5 `cltu-identification`

3.7.2.5.1 The **`cltu-identification`** parameter shall contain a monotonically increasing sequence number that shall be set for the first CLTU-TRANSFER-DATA invocation after a successful CLTU-START invocation to the value of the `first-cltu-identification` parameter of that CLTU-START invocation.

3.7.2.5.2 When the `plop-in-effect` (managed parameter) is either 'PLOP-1' or 'PLOP-2', the `cltu-identification` parameter shall be incremented for each CLTU-TRANSFER-DATA invocation accepted by the provider.

3.7.2.5.3 When the `plop-in-effect` (managed parameter) is 'ASCP', the `cltu-identification` parameter shall be incremented for each CLTU-TRANSFER-DATA invocation generated by the user.

3.7.2.5.4 When the `plop-in-effect` (managed parameter) is either 'PLOP-1' or 'PLOP-2', the provider shall set the value of the `cltu-identification` parameter of the CLTU-TRANSFER-DATA return to the sequence number expected in the next CLTU-TRANSFER-DATA invocation:

  a) if the invocation is accepted, the `cltu-identification` in the return shall be one greater than the value in the invocation;

  b) if the invocation is rejected, the `cltu-identification` in the return shall contain the value expected by the Enhanced Forward CLTU service provider:

    1) in the case of the first CLTU-TRANSFER-DATA operation following a CLTU-START, it is the value specified in the `first-cltu-identification` parameter of the CLTU-START operation;

    2) otherwise, the value is one greater than the value of the `cltu-identification` parameter of the last accepted CLTU-TRANSFER-DATA operation.

NOTE – It is assumed that the supported range of this parameter is sufficiently large that no two CLTUs within the CLTU buffer will have the same `cltu-identification`.

3.7.2.5.5 When the `plop-in-effect` (managed parameter) is 'ASCP', the provider shall set the value of the `cltu-identification` parameter of the CLTU-TRANSFER-DATA return to the same value of the `cltu-identification` parameter of the corresponding CLTU-TRANSFER-DATA invocation.

3.7.2.6 `earliest-radiation-time`

3.7.2.6.1 The **earliest-radiation-time** parameter shall be used to specify the earliest time that the provider shall start radiating the CLTU or first CADU resulting from this SLE-FTDU.

3.7.2.6.2 The `earliest-radiation-time` parameter shall either be unspecified, i.e., contain a value of 'null', or shall specify the earliest time at which the leading edge of the first bit of the CLTU or CADU may begin to be radiated.

3.7.2.6.3 If the `earliest-radiation-time` parameter is unspecified, the provider shall begin processing immediately after any delay associated with the previous SLE-FTDU has timed out, as long as `production-status` is 'operational'.

3.7.2.7 `latest-radiation-time`

3.7.2.7.1 The **`latest-radiation-time`** parameter shall be used to specify the latest time at which the provider shall start radiating the CLTU or first CADU resulting from this SLE-FTDU.

3.7.2.7.2 The `latest-radiation-time` parameter shall either be unspecified (i.e., contain a value of 'null') or shall specify the latest time at which the leading edge of the first bit of the CLTU or CADU may begin to be radiated.

3.7.2.7.3 If the `latest-radiation-time` parameter is unspecified, the provider shall process the SLE-FTDU as long as the service instance is in state 3 ('active') and the production process is not halted.

3.7.2.7.4 If `latest-radiation-time` is specified, i.e., it is not 'null', the provider shall defer processing of an SLE-FTDU if the current `production-status` value is 'interrupted'. Processing shall be deferred until either recovery from a temporary problem is accomplished, i.e., the `production-status` value changes to 'operational' before `latest-radiation-time` expires, or `latest-radiation-time` expires, in which case the provider shall discard the SLE-FTDU.

3.7.2.7.5 If `latest-radiation-time` equals `earliest-radiation-time`, radiation shall occur at this time.

NOTE –   When ASCP is in effect (i.e., the link is AOS), it may not be possible to radiate the contents of the SLE-FTDU at a specific time, because an idle CADU may be in the process of being radiated at that moment. If the resulting CADU cannot be radiated at the designated moment, it will be discarded per 3.7.2.7.6. Therefore users of EFCLTU service with ASCP in effect are encouraged to not set the value of `latest-radiation-time` equal to the value of `earliest-radiation-time`.

3.7.2.7.6 If radiation has not begun at or before `latest-radiation-time`, an 'sldu expired' exception shall be notified to the user, and the contents of the SLE-FTDU shall not be radiated.

3.7.2.8 `delay-time`

NOTE – The delay-time parameter is applicable only for Telecommand links (i.e., when PLOP-1 or PLOP-2 is used). In this case, only CLTUs are transferred via the service.

3.7.2.8.1 The **`delay-time`** parameter shall contain the minimum radiation delay, in microseconds, between the PL-CLTU transferred in this operation and the next CLTU.

3.7.2.8.2 The effect of `delay-time` depends on the setting of the `plop-in-effect` parameter. The behavior of each of the PLOPs is determined not only by the value of the `delay-time` parameter, but by the values of managed parameters of the Enhanced F-CLTU service. The following subsections specify the behavior of the PLOPs as they are affected by the `delay-time` parameter and the relevant managed parameters, in the context of the Enhanced F-CLTU service.

    a) When `plop-in-effect` is 'PLOP-1' (reference [2]):

        1) Initially, the carrier shall be unmodulated (Carrier Modulation Mode 1 [CMM-1]—see reference [2]) until a CLTU is received via a CLTU-TRANSFER-DATA invocation.

        2) Upon receipt of a CLTU-TRANSFER-DATA invocation, an acquisition sequence of length `acquisition-sequence-length` (see table 3-1) shall be transmitted (CMM-2).

        NOTE –   The beginning of transmission of the acquisition sequence may be delayed to ensure that the CLTU does not begin radiation until that CLTU's `earliest-radiation-time`.

        3) If the value of `plop-1-idle-sequence-length` (see table 3-1) is greater than zero, a leading idle sequence of length `plop-1-idle-sequence-length` shall be transmitted after the acquisition sequence (CMM-4).

        NOTE –   The `acquisition-sequence-length` and the `plop-1-idle-sequence-length` may both be set to zero and the actually required acquisition sequence (and optional leading and trailing idle sequences) may be inserted into the CLTU proper, enabling commanding of spacecraft that do not fully adhere to PLOP-1 (as per reference [2]).

        4) Following transmission of the acquisition sequence and optional idle sequence, the CLTU shall be transmitted (CMM-3).

        5) If the value of `plop-1-idle-sequence-length` (see table 3-1) is greater than zero, a trailing idle sequence of length `plop-1-idle-sequence-length` shall be transmitted following the transmission of the CLTU (CMM-4).

        6) Following transmission of the CLTU or optional trailing idle sequence, the carrier shall cease being modulated and shall remain unmodulated (CMM-1) until the latest of

            i) the expiration of `delay-time` specified for the most-recently radiated CLTU, as measured from the trailing edge of the last radiated bit of the CLTU or optional trailing idle sequence,

ii) availability for radiation of a subsequent CLTU, or

iii) the end of the communications session (i.e., the end of production of the forward space link).

7) When both (a) a subsequent CLTU is available for radiation and (b) the delay-time specified for the previous CLTU has expired, the subsequent CLTU shall be transmitted in accordance with steps 2) through 7). This process shall repeat until the end of the communications session.

b) When `plop-in-effect` is 'PLOP-2' (reference [2]):

1) Initially, the carrier is unmodulated (CMM-1) during the uplink sweep.

2) Following the uplink sweep, an acquisition sequence of length `acquisition-sequence-length` (see table 3-1) is transmitted (CMM-2).

3) Following the transmission of the acquisition sequence, an idle sequence shall be transmitted (CMM-4).

NOTE – For PLOP-2 the idle sequence has no fixed length.

4) When a CLTU becomes available for radiation, the CLTU shall be transmitted (CMM-3).

5) Following transmission of the CLTU, the idle sequence shall be transmitted (CMM-4) until the latest of

i) the expiration of delay-time specified for the most recently radiated CLTU, as measured from the trailing edge of the last radiated bit of the CLTU, or

ii) availability for radiation of a subsequent CLTU.

6) When both (a) a subsequent CLTU is available for radiation and (b) the `delay-time` specified for the previous CLTU has expired, the subsequent CLTU shall be transmitted in accordance with steps 4) through 6) until the end of the communications session (i.e., end of production of the forward space link).

NOTE – According to (reference [2]), PLOP-2 includes a final transition to unmodulated carrier (CMM-1) prior to the end of the communications session. However, there is no standard mechanism defined for determining when the transition from idle sequence modulation to no modulation is to occur.

c) When `plop-in-effect` is 'ASCP', the `delay-time` parameter has no meaning and shall be ignored by the provider,

3.7.2.9 **report**

The **report** parameter shall specify whether the provider shall invoke the CLTU-ASYNC-NOTIFY operation upon completion of the radiation of the contents of the SLE-FTDU, and shall contain one of the following values:

    a) 'produce report'—invoke a CLTU-ASYNC-NOTIFY operation upon completion of the radiation;

    b) 'do not produce report'—do not invoke the CLTU-ASYNC-NOTIFY operation.

3.7.2.10 data

3.7.2.10.1 When `plop-in-effect` is 'PLOP-1' or 'PLOP-2', the **data** parameter shall contain a single CLTU.

3.7.2.10.2 When `plop-in-effect` is 'ASCP', the `data` parameter shall contain one or more fixed-length AOS SL-PDUs, where each AOS SL-PDU is:

    a) an AOS transfer frame; or

    b) a CADU.

3.7.2.10.3 When `plop-in-effect` is 'ASCP', the size of the `data` parameter shall be an integral multiple of the `sl-pdu-length` managed parameter, and the provider shall extract `sl-pdu-length`-sized SL-PDUs from the `data` parameter.

3.7.2.10.4 If the `data` parameter contains multiple SL-PDUs, they shall be concatenated front-to-back within the **data** parameter.

3.7.2.11 **cltu-buffer-available**

The **cltu-buffer-available** parameter shall specify the remaining number of octets available for buffering SLE-FTDUs.

NOTES

1   Real-world implementations in general will have a small additional buffer at the modulator input in order to prevent data underflow in particular at high command rates and when the interface towards the modulator is asynchronous, e.g., based on TCP. Users should be aware that such an additional buffer is not reflected in the reported remaining number of octets available for buffering SLE-FTDUs, as such a modulator buffer is part of service production and outside the service provisioning.

2   The `cltu-buffer-available` parameter indicates the remaining space available in terms of the contents of the `data` parameter. Therefore, when the `data`

parameter contains transfer frames that will be subsequently lengthened by the addition of check symbols and/or sync markers, the `cltu-buffer-available` parameter indicates the space available for the octet blocks prior to their lengthening by service production.

3.7.2.12 `result`

The **`result`** parameter shall specify the result of the CLTU-TRANSFER-DATA operation and shall contain one of the following values:

a) 'positive result'—the CLTU-TRANSFER-DATA operation has been performed by the provider and the data will be buffered until processing is due;

b) 'negative result'—the CLTU-TRANSFER-DATA operation has not been performed by the provider, for the reason returned in the `diagnostic` parameter.

3.7.2.13 `diagnostic`

3.7.2.13.1 If `result` is 'negative result', **`diagnostic`** shall be present in the return and its value shall be one of the following:

a) 'duplicate invoke-ID' —the value of the `invoke-ID` parameter is the same as the `invoke-ID` value of a previous, outstanding operation;

b) 'unable to process'—the provider cannot process SLE-FTDUs for one of the following reasons:

1) the provider has been taken out of service for an indefinite period by management action, i.e., `production-status` is 'halted';

2) the `production-status` is 'interrupted', and the provider has reported the fault condition to the user via a CLTU-ASYNC-NOTIFY operation containing a `notification-type` of 'production interrupted';

NOTE – The provider may defer notifying the user when the `production-status` becomes 'interrupted', until the contents of an SLE-FTDU become ready to radiate. In this case, the provider continues to accept CLTU-TRANSFER-DATA operations so long as the notification has not been sent. (See annex B.)

3) the `production-status` is 'operational' but the provider is blocked because of an earlier fault: the provider has reported the fault condition to the user by a CLTU-ASYNC-NOTIFY operation containing the `notification-type` value 'production interrupted' and has reported the recovery from the fault condition to the user by a CLTU-ASYNC-NOTIFY operation containing the `notification-type` value 'production operational';

4) the `production-status` is 'operational', but the provider is blocked due to an expired SLE-FTDU while the `plop-in-effect` is either 'PLOP-1' or 'PLOP-2': the provider has reported the fault condition to the user via a CLTU-ASYNC-NOTIFY operation containing a `notification-type` of 'sldu expired';

NOTE – When an SLE-FTDU expires when the `plop-in-effect` is 'ASCP' the provider reports the fault condition to the user via a CLTU-ASYNC-NOTIFY operation containing a `notification-type` of 'sldu expired', but the service is *not* blocked.

5) the `production-status` is 'operational', but the provider has temporarily suspended production to process a CLTU-THROW-EVENT invocation: no fault condition has been reported to the user;

c) 'unable to store'—there is not enough buffer space available to store this SLE-FTDU;

d) 'out of sequence'—when the `plop-in-effect` is either 'PLOP-1' or 'PLOP-2', the value of the `cltu-identification` parameter is not equal to the value expected by the provider, which is one of the following:

1) in the case of the first CLTU-TRANSFER-DATA operation following a CLTU-START, the value specified by the user in the `first-cltu-identification` parameter of the CLTU-START operation;

2) otherwise, the value of the `cltu-identification` parameter specified by the provider in the last CLTU-TRANSFER-DATA return.

NOTE – Sequence checking of `cltu-identification` parameter values is not performed when the `plop-in-effect` is 'ASCP'.

e) 'inconsistent time range'— the time specified in the `earliest-radiation-time` parameter is later than the time specified in the `latest-radiation-time` parameter;

f) 'invalid time'—the radiation time window is invalid, for one of the following reasons:

1) the period from `earliest-radiation-time` to `latest-radiation-time` does not overlap with the range of times for which service production is scheduled;

2) the period from `earliest-radiation-time` to `latest-radiation-time` does not overlap with the service instance provision period;

NOTE – The production may be scheduled to terminate earlier than the service instance provision period ends. An SLE Complex may do so to have the production engine available for support of a different mission as soon as possible, but permitting the users of the previous production period some extra time to retrieve, for example, a status report reflecting the final accounting information.

g) 'late sldu'—latest-radiation-time is earlier than the time the CLTU-TRANSFER-DATA invocation is received by the provider;

h) 'invalid delay time'—the value of the delay-time is less than the `minimum-delay-time` value set by service management;

i) 'invalid data size'—the size of the `data` parameter is not an integral multiple of the `frame-length` value set by service management. The error condition that results in this diagnostic can occur only when `plop-in-effect` is 'ASCP'.

j) 'CLTU error'—the provider has performed error checks as provided in the service agreement and has determined that this CLTU is in error; for example, the SLE-FTDU exceeds the maximum size allowed for this service instance;

k) 'other reason'— the reason for the negative result will have to be found by other means.

3.7.2.13.2 If `result` is 'positive result', the `diagnostic` parameter shall not be present in the return.

3.7.3 EFFECTS

3.7.3.1 If `result` is 'positive result', the CLTU-TRANSFER-DATA operation shall have the following effects:

a) the provider shall buffer the SLE-FTDU until it is due to be processed;

b) the provider shall process SLE-FTDUs in the sequence received and according to the radiation time and delay parameters.

3.7.3.2 If `result` is 'negative result', the SLE-FTDU contained in the CLTU-TRANSFER-DATA invocation shall not be buffered.

3.7.3.3 The provider shall remain in state 3 ('active').

3.8 CLTU-ASYNC-NOTIFY

3.8.1 PURPOSE

3.8.1.1 The CLTU service provider shall invoke the CLTU-ASYNC-NOTIFY operation to notify the user of an event affecting the production of the Enhanced Forward CLTU service.

NOTE – Notification of events may be of value to the user in understanding specific provider behavior, such as an interruption of the command radiation.

3.8.1.2 CLTU-ASYNC-NOTIFY shall be an unconfirmed operation.

NOTE – Notifications from the provider are delivered to the user asynchronously to the flow of CLTU-TRANSFER-DATA operations from the user to the provider.

3.8.1.3 The CLTU-ASYNC-NOTIFY operation is valid in states 2 ('ready') and 3 ('active') and shall be invoked only by the provider.

3.8.2 INVOCATION, RETURN, AND PARAMETERS

3.8.2.1 General

The parameters of the CLTU-ASYNC-NOTIFY operation shall be present in the invocation as specified in table 3-7.

Table 3-7: CLTU-ASYNC-NOTIFY Parameters

Parameters	Invocation
invoker-credentials	M
notification-type	M
event-thrown-identification	C
cltu-last-processed	M
cltu-last-OK	M
cltu-status	C
radiation-start-time	C
radiation-stop-time	C
production-status	M
uplink-status	M

3.8.2.2 `invoker-credentials`

The **`invoker-credentials`** parameter shall provide information that enables the performer to authenticate the CLTU-ASYNC-NOTIFY invocation (see 3.2.5).

3.8.2.3 `notification-type`

The **`notification-type`** parameter shall describe the event being notified to the user and shall contain one of the following values:

a) 'cltu radiated'—the contents of the SLE-FTDU identified by the value of the `cltu-last-processed` parameter successfully completed radiation. This value of `notification-type` shall be used only if the value of `report` in the associated CLTU-TRANSFER-DATA invocation was 'produce report'. If the `data` parameter of the associated CLTU-TRANSFER-DATA invocation contained multiple SL-PDUs this notification type shall be sent only upon radiation of all CADUs resulting from the contents of the SL-PDUs contained in the `data` parameter.

b) 'sldu expired'—radiation of the contents of the SLE-FTDU identified by the value of the `cltu-last-processed` parameter did not begin by the time specified in the `latest-radiation-time` parameter of the associated CLTU-TRANSFER-DATA invocation, and therefore the expired SLE-FTDU has been discarded. If the `data` parameter of the associated CLTU-TRANSFER-DATA invocation contained multiple SL-PDUs, this notification type shall indicate that all of the SL_PDUs contained within the `data` parameter have been discarded.

    1) When `plop-in-effect` is either 'PLOP-1' or 'PLOP-2', no further CLTUs shall be radiated, buffered SLE-FTDUs shall be discarded, and further CLTU-TRANSFER-DATA invocations shall be blocked, i.e., rejected with an 'unable to process' diagnostic.

        NOTE – After the 'sldu expired' event occurs when the `plop-in-effect` is either 'PLOP-1' or 'PLOP-2', the service user has to clear the blocking by invoking a CLTU-STOP operation. In order to resume the transfer and radiation of CLTUs, the user must successfully invoke CLTU-START.

    2) When the `plop-in-effect` is 'ASCP', only the expired SLE-FTDU is discarded and processing of the SLE-FTDUs that follow the expired SLE-FTDU shall continue without blocking.

c) 'production interrupted'—the production process has stopped due to a condition that may be temporary. This event occurs and the notification shall be sent:

    1) for notification mode 'immediate' when the production-status changes to 'interrupted'; or

    2) for notification mode 'deferred' when production status is interrupted and either an SLE-FTDU is in status 'radiation started' or the radiation of the contents of the SLE-FTDU has to be started based on the specified earliest-radiation-time and latest-radiation-time parameters.

No further CLTUs/CADUs shall be radiated; buffered SLE-FTDUs shall be discarded; and, in state 3 ('active'), further CLTU-TRANSFER-DATA invocations shall be blocked, i.e., rejected with an 'unable to process' diagnostic.

NOTE – The user can unblock the service instance by invoking a CLTU-STOP operation. After the condition causing the 'production interrupted' event is corrected, the provider notifies the user by means of a 'production operational' notification. The user can resume the transfer and radiation of CLTUs after successfully invoking CLTU-START.

d) 'production halted'—the production process has been stopped by management action. No further CLTUs/CADUs shall be radiated; buffered SLE-FTDUs shall be discarded; and, in state 3 ('active'), further CLTU-TRANSFER-DATA invocations shall be blocked, i.e., rejected with an 'unable to process' diagnostic.

NOTES

1   The production process enters the halted state by service management decision, for example, in case of malfunctioning equipment.

2   After the 'production halted' event, the service user has to clear the blocking by invoking a CLTU-STOP operation. After management action to remove the 'halted' production status, the user can resume the transfer and radiation of CLTUs/CADUs by successful invocation of CLTU-START.

e) 'production operational'—the production process is ready to radiate CLTUs/CADUs and production-status has changed to 'operational'.

NOTE – The production process achieves operational status after initial establishment of the uplink channel or following actions to restore production to operational status after a halt or an interruption.

f) 'buffer empty'—the provider has no further SLE-FTDUs buffered for this service instance. This event shall not be notified if the empty buffer is the result of a CLTU-STOP operation or a CLTU-ASYNC-NOTIFY operation with a notification of 'production interrupted' or 'production halted'.

g) 'action list completed'—all actions initiated by the CLTU-THROW-EVENT operation identified by the `event-thrown-identification` parameter have successfully completed.

h) 'action list not completed'—at least one of the actions initiated by the CLTU-THROW-EVENT operation identified in the `event-thrown-identification` parameter did not complete.

i) 'event condition evaluated to false'—a required condition was false for the event that the CLTU-THROW-EVENT operation identified by the `event-thrown-identification` parameter attempted to trigger. The actions associated with this event have not been performed.

3.8.2.4 `event-thrown-identification`

3.8.2.4.1 If `notification-type` is 'action list completed', 'action list not completed', or 'event condition evaluated to false', the **`event-thrown-identification`** parameter shall be present and indicate the `event-invocation-identification` value of the CLTU-THROW-EVENT invocation that triggered the actions whose results are reported.

3.8.2.4.2 For all other notifications, this parameter shall not be present.

3.8.2.5 `cltu-last-processed`

For all notifications, the **`cltu-last-processed`** parameter shall be present. Its value shall be set as follows:

a) if the provider has processed (i.e., attempted to radiate the contents of) no SLE-FTDUs during this service instance, the value of the `cltu-last-processed` parameter shall be set to 'null';

b) otherwise, the `cltu-last-processed` parameter shall specify the sequence number of the SLE-FTDU that the provider most recently processed or attempted to process, whether the contents of the SLE-FTDU were successfully radiated or an exception occurred. If the `data` parameter of a CLTU-TRANSFER-DATA invocation contains multiple SL-PDUs, this parameter shall identify the sequence number of the SLE-FTDU of which any contained SL-PDU has been processed or attempted to be processed.

3.8.2.6 `cltu-last-OK`

For all notifications, the **`cltu-last-OK`** parameter shall be present. Its value shall be set as follows:

a) if no SLE-FTDUs have been successfully processed (i.e., radiated completely) during this service instance, the value of the `cltu-last-OK` parameter shall be set to 'null';

b) otherwise, the `cltu-last-OK` parameter shall specify the sequence number of the most recent SLE-FTDU for which all of the contents were successfully radiated. If the `data` parameter of a CLTU-TRANSFER-DATA invocation contains multiple SL-PDUs, this parameter shall identify the sequence number of the SLE-FTDU from which all of the CADUs resulting from that SLE-FTDU have been successfully radiated.

3.8.2.7 `cltu-status`

3.8.2.7.1   For all notifications, if the value of the `cltu-last-processed` parameter is 'null', the **cltu-status** parameter shall not be present.

3.8.2.7.2   Whenever the value of the `cltu-last-processed` parameter is not 'null', the `cltu-status` parameter shall be present and shall contain one of the following values representing the processing state of the SLE-FTDU identified by `cltu-last-processed`:

a) 'radiated'—radiation of the contents of the SLE-FTDU completed without aborting: the SLE-FTDU is guaranteed to have been radiated nominally. If the `data` parameter of a CLTU-TRANSFER-DATA invocation contains multiple SL-PDUs, this value indicates that all of the CADUs resulting from the SLE-FTDU have been radiated nominally;

b) 'expired'—the `latest-radiation-time` of the SLE-FTDU was reached before beginning radiation: the contents of the SLE-FTDU are guaranteed not to have been radiated. If the `data` parameter of a CLTU-TRANSFER-DATA invocation contains multiple SL-PDUs, this value indicates that none of the SL-PDUs have been radiated;

c) 'interrupted'— radiation of the contents of the SLE-FTDU started but did not complete because the `production-status` became 'interrupted';

d) 'radiation started'—radiation of the contents of the SLE-FTDU started but processing is not yet complete;

e) 'radiation not started'— the provider attempted to process the SLE-FTDU but could not start radiation because the `production-status` was 'interrupted'.

3.8.2.8 `radiation-start-time`

3.8.2.8.1   For all notifications, if the `cltu-last-processed` is 'null', the **radiation-start-time** parameter shall not be present.

3.8.2.8.2   Whenever `cltu-last-processed` is not 'null', the `radiation-start-time` parameter shall be present and shall contain the time at which radiation of the contents of the SLE-FTDU identified by `cltu-last-processed` started, subject to the following stipulations:

a) nominally, the `radiation-start-time` parameter shall contain the radiation time of the leading edge of the first radiated bit of the CLTU or first CADU resulting from the SLE-FTDU;

b) if the `cltu-status` value is 'expired' or 'production not started', the value of `radiation-start-time` shall be set to 'null'.

3.8.2.9 `radiation-stop-time`

3.8.2.9.1 For all notifications, if the `cltu-last-OK` is 'null', the **radiation-stop-time** parameter shall not be present.

3.8.2.9.2 Whenever `cltu-last-OK` is not 'null', the `radiation-stop-time` parameter shall be present and shall contain the time at which radiation of the CLTU or last CADU resulting from the SLE-TFDU identified by `cltu-last-OK` completed, i.e., the time of the trailing edge of the last radiated bit.

3.8.2.10 `production-status`

The **production-status** parameter shall describe the current status of the production process and shall contain one of the following values:

a) 'configured'—equipment has been assigned to support the service instance, but the production process is not yet capable of radiating CLTUs/CADUs;

b) 'operational'—the production process has been configured for support, has completed the acquisition sequence, and is capable of radiating CLTUs/CADUs;

c) 'interrupted'—the production process is stopped due to a fault;

d) 'halted'—the production process is stopped and production equipment is out of service due to management action.

3.8.2.11 `uplink-status`

3.8.2.11.1 The **uplink-status** parameter shall report to the user the state of the forward link obtained from the CLCW present in the return telemetry stream from the spacecraft.

3.8.2.11.2 The `uplink-status` parameter shall contain one of the following values:

a) 'uplink status not available'—no CLCWs from the spacecraft have been received by the provider;

b) 'no rf available'—the provider has received at least one CLCW; in the last CLCW received by the provider, the bit that flags 'No RF Available' was set to '1';

c) 'no bit lock'—the provider has received at least one CLCW; in the last CLCW received by the provider, the bit that flags 'No RF Available' was set to '0', and the bit that flags 'No Bit Lock' was set to '1';

d) 'nominal'—the provider has received at least one CLCW; in the last CLCW received by the provider, the bit that flags 'No RF Available' was set to '0', and the bit that flags 'No Bit Lock' was set to '0'.

3.8.3 EFFECTS

3.8.3.1 If `notification-type` indicates 'production interrupted', or 'production halted':

    a) no further CLTUs/CADUs shall be radiated;

    b) buffered SLE-FTDUs shall be discarded; and

    c) further CLTU-TRANSFER-DATA invocations shall be rejected.

3.8.3.2 If `notification-type` indicates 'sldu expired' and `plop-in-effect` is 'PLOP-1' or 'PLOP 2':

    d) no further CLTUs shall be radiated;

    e) buffered SLE-FTDUs shall be discarded; and

    f) further CLTU-TRANSFER-DATA invocations shall be rejected.

3.8.3.3 If `notification-type` indicates 'sldu expired' and `plop-in-effect` is 'ASCP', the expired SLE-FTDU shall be discarded and its contents shall not be radiated.

3.8.3.4 There shall be no effect for any other values of `notification-type`.

3.8.3.5 The provider shall remain in its original state, i.e., state 2 ('ready') or state 3 ('active').

3.9 CLTU-SCHEDULE-STATUS-REPORT

3.9.1 PURPOSE

3.9.1.1 The user shall invoke the CLTU-SCHEDULE-STATUS-REPORT operation to request that the provider do one of the following:

    a) send a status report immediately (see 3.10);

    b) send periodic status reports (see 3.10); or

    c) stop sending periodic status reports.

3.9.1.2 The provider shall return a report of the outcome of the performance of the CLTU-SCHEDULE-STATUS-REPORT operation to the user

3.9.1.3 Initially (i.e., whenever the CLTU-BIND operation is performed and the provider transitions from state 1 to state 2), periodic reporting shall be inactive.

3.9.1.4 For periodic reporting, the user may change the reporting period by invoking another CLTU-SCHEDULE-STATUS-REPORT operation.

3.9.1.5 The CLTU-SCHEDULE-STATUS-REPORT operation is valid only in states 2 ('ready') and 3 ('active') and shall be invoked only by the user.

3.9.2   INVOCATION, RETURN, AND PARAMETERS

3.9.2.1   General

The parameters of the CLTU-SCHEDULE-STATUS-REPORT operation shall be present in the invocation and return as specified in table 3-8.

Table 3-8: CLTU-SCHEDULE-STATUS-REPORT Parameters

Parameters	Invocation	Return
invoker-credentials	M	
performer-credentials		M
invoke-ID	M	M
report-request-type	M	
reporting-cycle	C	
result		M
diagnostic		C

3.9.2.2   **invoker-credentials**

The **invoker-credentials** parameter shall provide information that enables the performer to authenticate the CLTU-SCHEDULE-STATUS-REPORT invocation (see 3.2.5).

3.9.2.3   **performer-credentials**

The **performer-credentials** parameter shall contain information that enables the invoker to authenticate the return from the performance of CLTU-SCHEDULE-STATUS-REPORT (see 3.2.5).

3.9.2.4   **invoke-ID**

The performer shall return unchanged the invoker-supplied value of the **invoke-ID** parameter (see 3.2.6).

3.9.2.5 `report-request-type`

3.9.2.5.1 The **`report-request-type`** parameter shall specify how reporting shall be done, and its value shall be one of the following values:

a) 'immediately'—send a single status report immediately;

b) 'periodically'—send a status report every `reporting-cycle` seconds;

c) 'stop'—do not send further status reports.

3.9.2.5.2 If `report-request-type` is 'immediately',

a) the provider shall send a status report immediately;

b) the provider shall stop sending status reports after the immediate status report has been sent.

NOTE   –   Periodic reporting may be (re-)started by means of another CLTU-SCHEDULE-STATUS-REPORT invocation.

3.9.2.6 `reporting-cycle`

3.9.2.6.1 If the value of the `report-request-type` parameter is 'periodically', then the **`reporting-cycle`** parameter shall be present and shall specify the requested interval between status reports in seconds.

3.9.2.6.2 If the value of the `report-request-type` parameter is not 'periodically', the `reporting-cycle` parameter shall not be present.

3.9.2.7 `result`

The **`result`** parameter shall specify the result of the CLTU-SCHEDULE-STATUS-REPORT operation, and its value shall be one of the following:

a) 'positive result'—the CLTU-SCHEDULE-STATUS-REPORT operation has been performed, and the provider will send the requested status report(s) or stop sending periodic status reports;

b) 'negative result'—the CLTU-SCHEDULE-STATUS-REPORT operation has not been performed for the reason specified in the `diagnostic` parameter;   the previous setting for status reporting remains in effect.

3.9.2.8 **diagnostic**

3.9.2.8.1 If `result` is 'negative result', the **diagnostic** parameter shall be present in the return and it value shall be one of the following:

   a) 'duplicate invoke-ID'—the value of the `invoke-ID` parameter is the same as the `invoke-ID` value of a previous, outstanding operation;

   b) 'already stopped'—the provider is not currently providing periodic reports (applicable only when `report-request-type` is 'stop');

   c) 'invalid reporting cycle'—the `reporting-cycle` value is outside the range agreed upon by service management;

   d) 'other reason'—the reason for the negative result will have to be found by other means.

3.9.2.8.2 If `result` is 'positive result', the `diagnostic` parameter shall not be present.

3.9.3 EFFECTS

3.9.3.1 If `result` is 'positive result', the CLTU-SCHEDULE-STATUS-REPORT operation shall have the following effects, depending on the value of the `report-request-type` parameter:

   a) if `report-request-type` is 'immediately':

      1) a status report shall be sent immediately; i.e., the provider shall invoke a CLTU-STATUS-REPORT operation;

      2) the sending of any previously requested periodic status reports shall cease;

   b) if `report-request-type` is 'periodically':

      1) a status report shall be sent immediately; i.e., the provider shall invoke a CLTU-STATUS-REPORT operation;

      2) subsequent status reports shall be sent at the interval specified in the `reporting-cycle parameter`;

   c) if `report-request-type` is 'stop', periodic reporting shall cease.

3.9.3.2 If `result` is 'negative result', the CLTU-SCHEDULE-STATUS-REPORT operation shall have no effect, and the previous setting for the status reporting shall not change.

3.9.3.3 The provider shall remain in its original state, i.e., state 2 ('ready') or state 3 ('active').

3.10 CLTU-STATUS-REPORT

3.10.1 PURPOSE

3.10.1.1 The provider shall invoke the CLTU-STATUS-REPORT operation to send a status report to the user.

3.10.1.2 CLTU-STATUS-REPORT shall be an unconfirmed operation.

3.10.1.3 Status reports shall be sent (or not sent) in accordance with user requests conveyed by means of the CLTU-SCHEDULE-STATUS-REPORT operation (see 3.9).

3.10.1.4 The CLTU-STATUS-REPORT operation is valid only in states 2 ('ready') and 3 ('active') and shall be invoked only by the provider.

3.10.2 INVOCATION AND PARAMETERS

3.10.2.1 General

The parameters of the CLTU-STATUS-REPORT operation shall be present in the invocation as specified in table 3-9.

Table 3-9: CLTU-STATUS-REPORT Parameters

Parameters	Invocation
invoker-credentials	M
cltu-last-processed	M
cltu-last-OK	M
cltu-status	C
radiation-start-time	C
radiation-stop-time	C
production-status	M
uplink-status	M
number-of-cltus-received	M
number-of-cltus-processed	M
number-of-cltus-radiated	M
cltu-buffer-available	M

3.10.2.2 **invoker-credentials**

The **invoker-credentials** parameter shall provide information that enables the performer to authenticate the CLTU-STATUS-REPORT invocation (see 3.2.5).

3.10.2.3 `cltu-last-processed`

3.10.2.3.1 If the provider has processed (i.e., attempted to radiate) no SLE-FTDUs during this service instance, the value of the **`cltu-last-processed`** parameter shall be set to 'null'.

3.10.2.3.2 Otherwise, the `cltu-last-processed` parameter shall specify the sequence number of the SLE-FTDU that the provider most recently processed or attempted to process, whether the contents of the SLE-FTDU were successfully radiated or an exception occurred.

3.10.2.4 `cltu-last-OK`

3.10.2.4.1 If no SLE-FTDUs have been successfully processed (i.e., their contents radiated completely) during this service instance, the value of the **`cltu-last-OK`** parameter shall be set to 'null'.

3.10.2.4.2 Otherwise, the `cltu-last-OK` parameter shall specify the sequence number of the most recent SLE-FTDU for which the contents were successfully radiated. If the `data` parameter of a CLTU-TRANSFER-DATA invocation contains multiple SL-PDUs, this parameter shall identify the sequence number of the SLE-FTDU from which all of the resulting CADUs have been successfully radiated.

3.10.2.5 `cltu-status`

3.10.2.5.1 If the value of the `cltu-last-processed` parameter is 'null', the **`cltu-status`** parameter shall not be present.

3.10.2.5.2 Whenever the value of the `cltu-last-processed` parameter is not 'null', the `cltu-status` parameter shall be present and shall contain one of the following values representing the processing state of the SLE-FTDU identified by `cltu-last-processed`:

    a) 'radiated'—Radiation of the contents of the SLE-FTDU completed without aborting: the SLE-FTDU is guaranteed to have been radiated nominally. If the `data` parameter of a CLTU-TRANSFER-DATA invocation contains multiple SL-PDUs, this value indicates that all of the resultant CADUs have been radiated nominally.

    b) 'expired'—The `latest-radiation-time` of the SLE-FTDU was reached before beginning radiation: the contents of the SLE-FTDU are guaranteed not to have been radiated. If the `data` parameter of a CLTU-TRANSFER-DATA invocation contains multiple SL-PDUs, this value indicates that none of the SL-PDUs have been radiated.

    c) 'interrupted'—Radiation of the SLE-FTDU was aborted when the production status changed from 'operational' to 'interrupted': status information available to the

provider indicates that the contents of the SLE-FTDU started radiating and aborted prior to nominal completion.

d) 'radiation started'—Radiation of the contents of the SLE FTDU started but processing is not yet complete.

e) 'radiation not started'—The reported SLE-FTDU did not start radiation: the SLE-FTDU was blocked because of the 'interrupted' status of the production process.

3.10.2.6 `radiation-start-time`

3.10.2.6.1 If the `cltu-last-processed` is 'null', the **`radiation-start-time`** parameter shall not be present.

3.10.2.6.2 Whenever `cltu-last-processed` is not 'null', the `radiation-start-time` parameter shall be present and shall contain the time at which radiation of the contents of the SLE-FTDU identified by `cltu-last-processed` started, subject to the following stipulations:

a) nominally, the `radiation-start-time` parameter shall contain the radiation time of the leading edge of the first radiated bit of the CLTU or first CADU resulting from the SLE-FTDU;

b) if the `cltu-status` value is 'expired' or 'radiation not started', the value of `radiation-start-time` shall be set to 'null'.

3.10.2.7 `radiation-stop-time`

3.10.2.7.1 If the `cltu-last-OK` is 'null', the **`radiation-stop-time`** parameter shall not be present.

3.10.2.7.2 Whenever `cltu-last-OK` is not 'null', the `radiation-stop-time` parameter shall be present and shall contain the time at which radiation of the PL-CLTU or last CADU resulting from the SLE-FTDU identified by `cltu-last-OK` completed, i.e., the time of the trailing edge of the last radiated bit.

3.10.2.8 `production-status`

The **`production-status`** parameter shall describe the current status of the production process and shall contain one of the following values:

a) 'configured'—equipment has been assigned to support the service instance, but the production process is not yet capable of radiating CLTUs/CADUs;

b) 'operational'—the production process has been configured for support, has completed the acquisition sequence, and is capable of radiating CLTUs/CADUs;

c) 'interrupted'—the production process is stopped due to a fault;

d) 'halted'—the production process is stopped and production equipment is out of service due to management action.

3.10.2.9 **uplink-status**

3.10.2.9.1 The **uplink-status** parameter shall report to the user the state of the forward link obtained from the CLCW present in the return telemetry stream from the spacecraft.

3.10.2.9.2 The uplink-status parameter shall contain one of the following values:

a) 'uplink status not available'—no CLCWs from the spacecraft have been received by the provider;

b) 'no rf available'—the provider has received at least one CLCW; in the last CLCW received by the provider, the bit that flags 'No RF Available' was set to '1';

c) 'no bit lock'—the provider has received at least one CLCW; in the last CLCW received by the provider, the bit that flags 'No RF Available' was set to '0', and the bit that flags 'No Bit Lock' was set to '1';

d) 'nominal'—the provider has received at least one CLCW; in the last CLCW received by the provider, the bit that flags 'No RF Available' was set to '0', and the bit that flags 'No Bit Lock' was set to '0'.

3.10.2.10 **number-of-cltus-received**

3.10.2.10.1 The **number-of-cltus-received** parameter shall specify the total number of CLTUs received during this service instance.

3.10.2.10.2 Only CLTUs that are checked successfully and buffered shall be counted in the total.

3.10.2.10.3 If no CLTUs have been received, the value of this parameter shall be zero.

3.10.2.11 **number-of-cltus-processed**

3.10.2.11.1 The **number-of-cltus-processed** parameter shall specify the number of SLE-FTDUs that the provider attempted to radiate during this service instance, including

– SLE-FTDUs for which the contents radiated successfully;

– SLE-FTDUs that expired;

– SLE-FTDUs that aborted;

   – an SLE-FTDU the contents of which are in the process of being radiated.

3.10.2.11.2 If no SLE-FTDUs have been processed, the value of this parameter shall be zero.

NOTE – If the data parameter of a CLTU-TRANSFER-DATA invocation contains multiple SL-PDUs, the `number-of-cltus-processed` parameter value must be multiplied by the number of SL-PDUs per SLE-FTDU to determine the number of SL-PDUs processed.

3.10.2.12 `number-of-cltus-radiated`

3.10.2.12.1 The **`number-of-cltus-radiated`** shall contain the count of SLE-FTDUs the complete contents of which have been successfully radiated during this service instance.

3.10.2.12.2 An SLE-FTDU the contents of which are in the process of being radiated shall not be included in this count.

3.10.2.12.3 If no SLE-FTDUs have been radiated, the value of this parameter shall be zero.

NOTE – If the data parameter of a CLTU-TRANSFER-DATA invocation contains multiple SL-PDUs, the `number-of-cltus-radiated` parameter value must be multiplied by the number of SL-PDUs per SLE-FTDU to determine the number of SL-PDUs radiated.

3.10.2.13 `cltu-buffer-available`

3.10.2.13.1 The **`cltu-buffer-available`** parameter shall specify the remaining number of octets available for buffering SLE-FTDUs.

3.10.2.13.2 Once the provider starts to process an SLE-FTDU, it shall be considered removed from the buffer.

3.10.3 EFFECTS

The CLTU-STATUS-REPORT operation shall have the following effects:

   a) Status information shall be delivered to the user.

   b) The provider shall remain in its original state, i.e., state 2 ('ready') or state 3 ('active').

3.11 CLTU-GET-PARAMETER

3.11.1 PURPOSE

3.11.1.1 The user shall invoke the CLTU-GET-PARAMETER operation to ascertain the value of a Enhanced Forward CLTU service parameter.

3.11.1.2 The provider shall return a report of the outcome of the performance of the CLTU-GET-PARAMETER operation to the user.

3.11.1.3 If the operation is successful, the current value of the specified CLTU service parameter shall be provided to the user in the return from the operation.

3.11.1.4 The CLTU-GET-PARAMETER operation is valid only in states 2 ('ready') and 3 ('active') and shall be invoked only by the user.

3.11.2 INVOCATION, RETURN, AND PARAMETERS

3.11.2.1 General

The parameters of the CLTU-GET-PARAMETER operation shall be present in the invocation and return as specified in table 3-10.

Table 3-10: CLTU-GET-PARAMETER Parameters

Parameters	Invocation	Return
invoker-credentials	M	
performer-credentials		M
invoke-ID	M	M
cltu-parameter	M	C
parameter-value		C
result		M
diagnostic		C

3.11.2.2 `invoker-credentials`

The **invoker-credentials** parameter shall provide information that enables the performer to authenticate the CLTU-GET-PARAMETER invocation (see 3.2.5).

3.11.2.3 `performer-credentials`

The **performer-credentials** parameter shall provide information that enables the invoker to authenticate the return from the performance of CLTU-GET-PARAMETER (see 3.2.5).

3.11.2.4 `invoke-ID`

The performer shall return unchanged the invoker-supplied value of the **invoke-ID** parameter (see 3.2.6).

3.11.2.5 `cltu-parameter`

3.11.2.5.1 The **cltu-parameter** parameter shall specify the Enhanced Forward CLTU service parameter whose value is to be returned to the user, and its value shall be one of the values listed in table 3-11.

3.11.2.5.2 cltu-parameter is conditionally present in the return based on the result parameter:

  a) if the value of result is 'positive result', cltu-parameter shall be present in the return;

  b) if the value of result is 'negative result', cltu-parameter shall not be present in the return.

3.11.2.6 `parameter-value`

3.11.2.6.1 The **`parameter-value`** parameter shall contain the value of the CLTU service parameter indicated by `cltu-parameter` (see 3.11.2.5).

3.11.2.6.2 The `parameter-value` parameter is conditionally present in the return based on the `result` parameter:

   a) if the value of `result` is 'positive result', `parameter-value` shall be present;

   b) if the value of `result` is 'negative result', `parameter-value` shall not be present.

3.11.2.7 `result`

The **`result`** parameter shall specify the result of the CLTU-GET-PARAMETER invocation and shall contain one of the following values:

   a) 'positive result'—the CLTU-GET-PARAMETER operation has been performed, and the value of the specified CLTU service parameter is provided in the return to the user;

   b) 'negative result'—the CLTU-GET-PARAMETER operation has not been performed for the reason specified in the `diagnostic` parameter.

Table 3-11: Enhanced Forward CLTU Parameters

Parameter	Description
`acquisition-sequence-length`	The size, in octets, of the bit pattern to be radiated to enable the spacecraft telecommand system to achieve bit lock.
	NOTE – This parameter is meaningful only when `plop-in-effect` is 'PLOP-1' or 'PLOP-2'. It has a value of zero ('0') when `plop-in-effect` is 'ASCP'.
`bit-lock-required`	If the value is 'yes', the 'No bit lock' flag in the CLCW must be false in order for the provider to set the production status to 'operational'.
	NOTE – This parameter is meaningful only when `plop-in-effect` is 'PLOP-1' or 'PLOP-2'. It has a value of 'no' when `plop-in-effect` is 'ASCP'.

Parameter	Description
`block-encode`	The forward error correction block encoding that is to be applied to the frames. The values are: - 'rs': Reed-Solomon; - 'ldpc': low density parity check; - 'none': no FEC block encoding is to be applied. NOTES 1 This parameter is used to configure the production processing specified in D3 of annex D. 2 This parameter is meaningful only when `plop-in-effect` is 'ASCP'. It has a value of 'none' when plop-in-effect is 'PLOP-1' or 'PLOP-2'. 3 Although the TM Synchronization and Channel Coding Recommended Standard also defines functions for Turbo coding of transfer frames, Turbo coding on the forward link is not among the requirements for the spaceflight missions that motivated the development of this Experimental Specification.
`clcw-global-VCID`	The Master or Virtual Channel that carries the CLCW to be used by the F-CLTU provider to determine the forward link RF and bit lock status. NOTES 1 This parameter is meaningful only when `plop-in-effect` is 'PLOP-1' or 'PLOP-2'. When `plop-in-effect` is 'ASCP', the VC ID component of this parameter shall be set to 63 (the Only Idle Data VC). 2 When `plop-in-effect` is 'PLOP-1' or 'PLOP-2' but there is no return virtual channel carrying CLCWs for this forward link, the VC ID component of this parameter shall be set to 63 (the Only Idle Data VC).
`clcw-physical-channel`	The RF return channel that carries the CLCW to be used by the F-CLTU provider to determine the forward link RF and bit lock status. NOTES 1. This parameter is meaningful only when `plop-in-effect` is 'PLOP-1' or 'PLOP-2'. When `plop-in-effect` is 'ASCP', this parameter has a value of 'none'. 2. When `plop-in-effect` is 'PLOP-1' or 'PLOP-2' but there is no return virtual channel carrying CLCWs for this forward link, this parameter has a value of 'none'.

Parameter	Description
`convolutionally-encode`	Specifies whether service production is to be configured to `convolutionally-encode` the data stream prior to radiation. The values are: 'yes', 'no'. NOTES 3 If the value of `block-encode` is 'ldpc', then `convolutionally-encode` must be 'no'. 4 This parameter is used to configure the production processing specified in D6 of annex D. 2 This parameter is meaningful only when `plop-in-effect` is 'ASCP'. It has a value of 'no' when `plop-in-effect` is 'PLOP-1' or 'PLOP-2'.
`delivery-mode`	'fwd online'
`expected-cltu-identification`	The expected value of the `cltu-identification` parameter to be received in the next CLTU-TRANSFER-DATA invocation. If no CLTU-START has been received, zero shall be returned as the default value of this parameter.
`expected-event-invocation-identification`	The expected value of the `event-invocation-identification` parameter to be received in the next CLTU-THROW-EVENT invocation. The initial value of this parameter is zero.
`sl-pdu-length`	The fixed length (in octets) of the SL-PDUs carried in the `data` parameters of the CLTU-TRANSFER-DATA invocations. NOTES 1 This parameter is used to configure the production processing specified in D2 of annex D. 2 This parameter is meaningful only when `plop-in effect` is 'ASCP'. It has a value of 0 (zero) when `plop-in-effect` is 'PLOP-1' or 'PLOP-2'.
`maximum-cltu-length`	**3.11.3** The size, in octets, of the maximum-length SLE-FTDU that will be accepted by the provider for this service instance.

Parameter	Description
minimum-delay-time	The minimum guard time the F-CLTU provider will accept between two consecutive CLTUs. NOTE – This parameter is meaningful only when plop-in-effect is 'PLOP-1' or 'PLOP-2'. It has a value of 0 (zero) when plop-in-effect is 'ASCP'.
modulation-frequency	The frequency of the primary modulation of the RF carrier, expressed in tenths of Hertz. Primary modulation means the subcarrier frequency, when applicable, otherwise the frequency of direct data modulation.
modulation-index	The angle by which the RF carrier is phase shifted with respect to the un-modulated RF carrier, expressed in milli-radians (10^{-3} rad).
notification-mode	The notification mode may be 'immediate' or 'deferred'. If 'immediate', the user is notified of a production-status change to 'interrupted' by means of a CLTU-ASYNC-NOTIFY invocation as soon as this transition is detected. If 'deferred', the user is notified about the production-status change by means of a CLTU-ASYNC-NOTIFY invocation only if and when the radiation of a CLTU or CADU is affected.
plop-1-idle-sequence-length	The size, in octets, of the optional idle sequence that shall be used in conjunction with PLOP-1. If 0, no idle sequence is applied. NOTE – This parameter is meaningful only when plop-in-effect is 'PLOP-1' or 'PLOP-2'. It has a value of 0 (zero) when plop-in-effect is 'ASCP'.
plop-in-effect	The PLOP being used: 'PLOP-1', 'PLOP-2',or 'ASCP'.
protocol-abort-mode	The protocol-abort-mode may be 'abort' or 'continue'. If it is 'abort', service production shall cease in the event of a protocol abort. If it is 'continue', service production shall disregard this event and continue radiating the contents of the SLE-FTDUs buffered at that time.

Parameter	Description
`randomize`	Specifies whether service production is configured by service management to randomize each transfer frame or codeblock prior to the attachment of the sync marker. The values are: 'yes', 'no'. NOTE 1 This parameter is used to configure the production processing specified in D4 of annex D. 2 This parameter is meaningful only when `plop-in-effect` is 'ASCP'. It has a value of 'no' when `plop-in-effect` is 'PLOP-1' or 'PLOP-2'
`reporting-cycle`	The current setting of the reporting cycle for status reports (see 3.9 and 3.10): the value is 'null' if cyclic reporting is off; otherwise it is the time (in seconds) between successive CLTU-STATUS-REPORT invocations (see 3.9).
`return-timeout-period`	The maximum time period (in seconds) permitted from when a confirmed Forward CLTU operation is invoked until the return is received by the invoker. (See 4.1.3.)
`rf-available-required`	If the value is 'yes', the 'No RF available' flag in the CLCW must be false in order for the provider to set production status to 'operational'. NOTE – This parameter is meaningful only when plop-in-effect is 'PLOP-1' or 'PLOP-2'. It has a value of 'no' when plop-in-effect is 'ASCP'.
`subcarrier-to-bit-rate-ratio`	When subcarrier modulation is used, the value represents the ratio of the subcarrier frequency to the uplink data rate (i.e., the bit rate). A value of one indicates that data will be directly modulated onto the carrier.

3.11.3.1 `diagnostic`

3.11.3.1.1 If `result` is 'negative result', **`diagnostic`** shall be present and shall contain one of the following values:

a) 'duplicate invoke-ID'—the value of the `invoke-ID` parameter is the same as the `invoke-ID` value of a previous, outstanding operation;

b) 'unknown parameter'—the value of `cltu-parameter` does not identify any CLTU service parameter that is recognized by the service provider;

c) 'other reason'—the reason for the negative result will have to be found by other means.

3.11.3.1.2 If `result` is 'positive result', the `diagnostic` parameter shall not be present in the return.

3.11.4 EFFECTS

3.11.4.1 If `result` is 'positive result', the value of the CLTU service parameter specified by the `cltu-parameter` parameter shall be returned to the user in the return.

3.11.4.2 If `result` is 'negative result', no CLTU service parameter specified by the `cltu-parameter` parameter shall be returned to the user.

3.11.4.3 The provider shall remain in its original state, i.e., state 2 ('ready') or state 3 ('active').

3.12 CLTU-THROW-EVENT

3.12.1 PURPOSE

3.12.1.1 The user shall invoke the CLTU-THROW-EVENT operation in order to cause the provider to forward to SLE Complex Management an event that requires management action.

NOTES

1   An example of such an event is change of the uplink bit rate. This will typically require sending a telecommand that reconfigures the onboard commanding system to the new bit rate. When this command is executed, the uplink bit rate can be changed by throwing the associated event.

2   The event is considered to occur when the user invokes the CLTU-THROW-EVENT operation; i.e., the provider is expected to react to the receipt of a CLTU-THROW-EVENT invocation with minimal delay. It is therefore asynchronous with the radiation of SL-DUs.

3   If synchronism is required between the event and the radiation of a specific SL-DU, it is the user's responsibility to ensure it, e.g., by means of controlling the radiation time of the critical CLTUs or by means of the notification capabilities provided by the Enhanced Forward CLTU service.

3.12.1.2 The performer shall return a report of the outcome of the performance of the CLTU-THROW-EVENT operation to the user.

NOTE – The return will only show whether the invocation itself has been accepted or rejected, but not if the actions associated with the event have been performed successfully. The provider will invoke a CLTU-ASYNC-NOTIFY operation to inform the user on the outcome of the actions triggered by the event.

3.12.1.3 The CLTU-THROW-EVENT operation is valid in states 2 ('ready') and 3 ('active') and shall be invoked only by the user.

3.12.2 INVOCATION, RETURN, AND PARAMETERS

3.12.2.1 General

The parameters of the CLTU-THROW-EVENT operation shall be present in the invocation and return as specified in table 3-12.

3.12.2.2 `invoker-credentials`

The **invoker-credentials** parameter shall provide information that enables the performer to authenticate the CLTU-THROW-EVENT invocation (see 3.2.5).

3.12.2.3 `performer-credentials`

The **performer-credentials** parameter shall provide information that enables the invoker to authenticate the return from the performance of CLTU-THROW-EVENT (see 3.2.5).

3.12.2.4 `invoke-ID`

The performer shall return unchanged the invoker-supplied value of the **invoke-ID** parameter (see 3.2.6).

Table 3-12: CLTU-THROW-EVENT Parameters

Parameters	Invocation	Return
invoker-credentials	M	
performer-credentials		M
invoke-ID	M	M
event-invocation-identification	M	M
event-identifier	M	
event-qualifier	M	
result		M
diagnostic		C

3.12.2.5 `event-invocation-identification`

3.12.2.5.1 The **event-invocation-identification** parameter shall contain a monotonically increasing sequence number and shall be

a) set to zero ('0') for the first CLTU-THROW-EVENT invocation following the first successful CLTU-BIND to the service instance;

b) incremented for each CLTU-THROW-EVENT invocation accepted by the provider.

3.12.2.5.2 The provider shall set the value of the event-invocation-identification parameter to the sequence number expected in the next CLTU-THROW-EVENT invocation:

a) if the invocation is accepted, the value of event-invocation-identification in the return shall be one greater than the value in the invocation;

b) if the invocation is rejected, the event-invocation-identification in the return shall contain the value expected by the Enhanced Forward CLTU service provider:

1) in the case of the first CLTU-THROW-EVENT operation following a successful CLTU-BIND operation, it is zero ('0');

2) otherwise, the value is one greater than the value of the event-invocation-identification parameter of the last accepted CLTU-THROW-EVENT operation.

3.12.2.5.3 The user shall set the value of event-invocation-identification in the next CLTU-THROW-EVENT invocation to the value returned by the provider.

3.12.2.5.4 The value expected by the provider for the given service instance shall not be altered when the association between user and provider is reestablished after being released or aborted.

3.12.2.5.5 The provider shall use the user-supplied value of `event-invocation-identification` to reference the respective event in all subsequent reports.

3.12.2.6 `event-identifier`

The **event-identifier** parameter shall identify the event to be forwarded to SLE Complex Management by the service provider.

3.12.2.7 `event-qualifier`

The **event-qualifier** parameter may be used to provide additional data constraining the actions to be performed by Complex Management in response to the event specified in `event-identifier` and shall be forwarded together with the event.

NOTES

1   For example, `event-identifier` may be 'bit-rate-change'. Then `event-qualifier` may specify the new bit rate.

2   The **actual** type and any constraints applicable to the data contained in the `event-qualifier` parameter depend on the event. These have to be checked by Complex Management. The Enhanced Forward CLTU provider only checks that the **formal** type (octet string) and the length of this parameter are in line with the specification in annex A.

3.12.2.8 `result`

The **result** parameter shall specify the result of the CLTU-THROW-EVENT invocation and shall contain one of the following values:

   a) 'positive result'—the CLTU-THROW-EVENT operation has been performed and the event together with the associated qualifier will be forwarded to SLE Complex Management;

   b) 'negative result'—the CLTU-THROW-EVENT operation has not been performed for the reason specified in the `diagnostic` parameter. The provider will not forward the event to Complex Management.

3.12.2.9 `diagnostic`

3.12.2.9.1 If `result` is 'negative result', the **diagnostic** parameter shall be present in the return, and its value shall be one of the following:

a) 'duplicate invoke-ID'—the value of the `invoke-ID` parameter is the same as the `invoke-ID` value of a previous, outstanding operation;

b) 'operation not supported'—the CLTU-THROW-EVENT operation is not supported by the provider;

c) 'out of sequence'—the value of the `event-invocation-identification` parameter is not equal to the value expected by the provider;

d) 'no such event'—the value of the `event-identifier` parameter in the CLTU-THROW-EVENT invocation is not in the set of `event-identifiers` values recognized by SLE Complex Management;

e) 'other reason'—the reason for the negative result will have to be found by other means.

3.12.2.9.2 If `result` is 'positive result', the `diagnostic` parameter shall not be present in the return.

3.12.3 EFFECTS

3.12.3.1 If `result` is 'positive result' the CLTU-THROW-EVENT operation shall have the following effects:

a) the `event-identifier` and the `event-qualifier` parameters shall be forwarded, with minimum delay, to the event handler within SLE Complex Management;

b) the provider shall pass notifications received from the event handler to the user by invoking the CLTU-ASYNC-NOTIFY operation.

NOTES

1   In order for the CLTU-THROW-EVENT to have an effect, there must be prior agreement between SLE Complex Management and SLE Utilization Management regarding the definition of event identifiers and the interpretation of any accompanying qualifying parameter.

2   The effects of the event thrown may consist of preparations made for a change to be triggered by a later CLTU-THROW-EVENT invocation.

3   Checking of the `event-qualifier` parameter may be performed within SLE Complex Management. If an error is found, CLTU-ASYNC-NOTIFY is sent to the user with the notification 'action list not completed'.

3.12.3.2 If `result` is 'negative result', the event shall not be forwarded to SLE Complex Management.

3.12.3.3 The provider shall remain in its original state, i.e., state 2 ('ready') or state 3 ('active').

3.13 CLTU-PEER-ABORT

3.13.1 PURPOSE

3.13.1.1 The user or provider shall invoke the CLTU-PEER-ABORT operation to notify the peer system that the local application detected an error that requires that the association between them be terminated abnormally.

3.13.1.2 CLTU-PEER-ABORT shall be an unconfirmed operation.

3.13.1.3 The CLTU-PEER-ABORT operation should be truly abortive, i.e., the transfer of the SLE-PDU carrying this invocation should not be delayed by other invocations submitted earlier and possibly being buffered by the underlying communications service.

3.13.1.4 The CLTU-PEER-ABORT operation is valid in states 2 ('ready') and 3 ('active') and may be invoked in these states by either the user or the provider.

3.13.1.5 The CLTU-PEER-ABORT operation may be invoked by the user in state 1 ('unbound') subsequent to invocation of the CLTU-BIND operation, when the return is not received within the specified time limit (see 4.1.3).

3.13.2 INVOCATION AND PARAMETERS

3.13.2.1 General

The parameters of the CLTU-PEER-ABORT operation shall be present in the invocation as specified in table 3-13.

Table 3-13: CLTU-PEER-ABORT Parameters

Parameters	Invocation
diagnostic	M

3.13.2.2 `diagnostic`

The `diagnostic` parameter shall specify why the CLTU-PEER-ABORT is being issued and shall contain one of the following values:

a) 'access denied'—a responder with an identity as presented in the `responder-identifier` parameter of the CLTU-BIND return is not registered at the initiator;

b) 'unexpected responder ID'—the value of the `responder-identifier` parameter in the CLTU-BIND return does not match the identity of the authorized responder for this service instance as specified by service management;

c) 'operational requirement'—the local system had to terminate the association to accommodate some other operational need;

d) 'protocol error'—the local application entity detected an error in the sequencing of CLTU service operations;

e) 'communications failure'—the communications service on the other side of a gateway was disrupted;

NOTE – This `diagnostic` value is only applicable when the SLE applications entities are communicating via a gateway.

f) 'encoding error'—the local application entity detected an error in the encoding of one or more operation parameters or did not recognize the operation;

g) 'return timeout'—the local application entity detected that the return from a confirmed operation was not received within the time specified by the `return-timeout-period` parameter (see 4.1.3);

h) 'end of service instance provision period'—the local application entity detected that the service instance provision period has ended, and the initiator has not invoked the CLTU-UNBIND operation;

i) 'unsolicited invoke-ID'—the local application entity received a return with an invoke-ID that does not match the invoke-ID of any of the operations for which the return is pending;

j) 'other reason'—the local application detected an unspecified error during the processing of one or more operations.

NOTE – CLTU-PEER-ABORT does not carry an `invoker-credentials` parameter. It is conceivable that an intruder might use the CLTU-PEER-ABORT operation for a denial-of-service attack. If an intruder has this capability of breaking into an existing connection, then a denial-of-service attack can be much more easily accomplished by disrupting communications at a layer lower than the Application layer. Therefore, authentication of the CLTU-PEER-ABORT operation would not provide improved protection against such attacks.

3.13.3 EFFECTS

The CLTU-PEER-ABORT operation shall have the following effects:

a) the association between the initiator and the responder shall be aborted, and the initiator and the responder shall cease to communicate with each other;

b) the provider shall transition to state 1 ('unbound');

c) the provider shall discard buffered SLE-FTDUs;

d) the provider shall stop production, except for an SLE-FTDU the contents of which are in the process of being radiated;

e) the initiator may attempt to re-bind at any time prior to the end of the provision period scheduled for this service instance;

f) statistical information required for the generation of status reports shall be retained throughout the service instance provision period;

g) parameters that serve to relate notifications on an activity to operations that triggered that activity shall not be altered when the association is aborted.

4   CLTU PROTOCOL

4.1   GENERIC PROTOCOL CHARACTERISTICS

NOTE –   This section specifies the handling of invalid SLE-PDUs and other failures affecting the protocol.

4.1.1   UNEXPECTED SLE PROTOCOL DATA UNIT

If the peer application entity sends an invocation or return not allowed in the current state of the performer, the performer shall abort the association by invoking the CLTU-PEER-ABORT operation with the `diagnostic` parameter set to 'protocol error'.

4.1.2   INVALID OR UNDECODABLE PROTOCOL DATA UNIT

If the application entity receives an invocation or return that contains an unrecognized operation type, contains a parameter of a wrong type, or is otherwise not decodable, the application entity shall abort the association by invoking the CLTU-PEER-ABORT operation with the `diagnostic` parameter set to 'encoding error'.

4.1.3   MISSING RETURN

For confirmed operations, if the invoker does not receive the return from the performer within the return-timeout-period specified by service management, the invoker shall abort the association by invoking the CLTU-PEER-ABORT operation with the `diagnostic` parameter set to 'return timeout'.

NOTES

1   The return-timeout-period shall be chosen taking into account performance of user and provider applications as well as the delays introduced by the underlying communications service.

2   In order to provide responsive service and short return-timeout-periods, the generation of the return from an operation must not depend on any human interaction.

3   After invoking the CLTU-UNBIND operation, the initiator must not invoke any further operations (with the exception of the case addressed in 3.4.1.4) nor send any returns. The responder is not required to send any pending returns after having received the CLTU-UNBIND invocation. Therefore, following a CLTU-UNBIND invocation, the 'missing return' event may occur.

4.1.4   UNSOLICITED RETURN

If the application entity receives a return with an `invoke-ID` parameter value that does not correspond to any invocation for which the return is still pending, the application entity shall abort the association by invoking the CLTU-PEER-ABORT operation with the `diagnostic` parameter set to 'unsolicited invoke-ID'.

4.1.5   COMMUNICATIONS FAILURE

4.1.5.1   Every SLE entity (i.e., every SLE user or provider) that is in an association (bound) with a peer SLE entity shall maintain knowledge of the health of the communications interface with the peer.

4.1.5.2   If an SLE entity is informed by its underlying communications service that communications with the peer entity have been disrupted (e.g., due to a communications failure), then this entity shall consider that the association with the peer entity has been aborted. Occurrence of such communications failure event shall be referred to as a 'protocol abort'.

NOTE –   The exact criteria for determining when communications have been disrupted may depend on the characteristics of the communications service and may be specific to a given implementation. However, it is a firm requirement on any communications service underlying an SLE transfer service that it include suitable means to ascertain the status of the connection (see 1.3.1). In the case of communications protocols that do not intrinsically provide such a capability, the transmission of a periodic 'heartbeat' indicator or equivalent may need to be implemented.

4.1.5.3   Occurrence of the communications failure event described above shall be referred to as a 'protocol abort'.

4.1.5.4   Subsequent to a 'protocol abort' event:

a)   the provider shall transition to state 1 ('unbound');

b)   the provider shall complete processing an SLE-FTDU the contents of which are in the process of being radiated;

c)   if the value of the `protocol-abort-mode` parameter (see 3.2.10) is 'flush', all buffered SLE-FTDUs shall be discarded;

d)   if the value of the `protocol-abort-mode` parameter is 'continue', buffered SLE-FTDUs shall not be discarded, and SLE-FTDUs shall continue to be processed as though the association had not been aborted;

e)   neither the user nor the provider shall attempt further communications with the peer except that the user may attempt to re-establish the association by invoking the CLTU-BIND operation.

NOTES

1   When the CLTU buffer is not flushed following occurrence of a protocol abort, any notifications (e.g., 'cltu radiated') generated while the association is not available shall not cause the invocation of a CLTU-ASYNC-NOTIFY operation. Neither shall such CLTU-ASYNC-NOTIFY operations be buffered such that they can be invoked if the user should later re-BIND to the service instance. It is recommended, but not required, that such notifications be logged and thus available to the user via service management. The user, on the basis of the information conveyed by a CLTU-STATUS-REPORT when the association is re-established, can determine which SLE-FTDUs were successfully radiated.

2   In addition, when the CLTU buffer is not flushed following a protocol abort, if the user subsequently re-BINDs to the service instance, it is the user's responsibility to choose a `first-cltu-identification` value in the CLTU-START that is greater than the `cltu-identification` of any SLE-FTDU that remains in the buffer.

4.1.5.5   Statistical information required for the generation of the status report shall be retained during the service instance provision period, regardless of a 'protocol abort' event.

4.1.5.6   Parameters that serve to relate notifications on an activity to operations that triggered that activity (such as `cltu-identification`) shall not be altered in the event of a 'protocol abort'.

4.1.6   ACCESS CONTROL

4.1.6.1   The initiator of an association shall present its own identity in the `initiator-identifier` parameter of the CLTU-BIND invocation.

4.1.6.2   If the CLTU-BIND operation is invoked with a value of `initiator-identifier` that is not known to the responder, the responder shall not make any attempt to authenticate that invocation. Instead, the responder shall generate a CLTU-BIND return with `result` set to 'negative result', `diagnostic` set to 'access denied', and `performer-credentials` set to 'unused'.

4.1.6.3   If the initiator is known to the responder, the responder shall attempt to authenticate the CLTU-BIND invocation (see 3.2.5) as required for the given initiator. If authentication succeeds, but the initiator is not authorized for the service instance indicated in the `service-instance-identifier` parameter of the CLTU-BIND invocation, the responder shall generate a CLTU-BIND return with `result` set to 'negative result' and `diagnostic` set to 'service instance not accessible to this initiator'.

NOTE – If authentication fails, the responder shall behave as specified in 4.1.7. If authentication is not required for the given initiator, it shall be as if authentication was successful.

4.1.6.4   If the initiator receives a CLTU-BIND return with a `responder-identifier` value that is not known to the initiator, the initiator shall not make any attempt to authenticate this return but shall abort the association by invoking CLTU-PEER-ABORT with `diagnostic` set to 'access denied'.

4.1.6.5   If the responder is known to the initiator, the initiator shall attempt to authenticate the CLTU-BIND return (see 3.2.5) as required for the given responder. If authentication succeeds, but the `responder-identifier` value is different from the expected responder identity, the initiator shall abort the association by invoking CLTU-PEER-ABORT with `diagnostic` set to 'unexpected responder ID'.

4.1.6.6   If authentication fails, the initiator shall behave as specified in 4.1.7.

4.1.6.7   If authentication is not required for the given responder, it shall be as if authentication was successful.

4.1.7   FAILING AUTHENTICATION

4.1.7.1   An incoming invocation or return shall be ignored if the credentials parameter cannot be authenticated when, by management arrangement, credentials are required.

4.1.7.2   If an invocation is ignored, the operation shall not be confirmed, and a report of the outcome shall not be returned to the provider.

4.1.7.3   If a return is ignored, it shall be as if no report of the outcome of the operation has been received.

4.2   CLTU SERVICE PROVIDER BEHAVIOR

4.2.1   GENERAL REQUIREMENTS

4.2.1.1   The behavior of the Enhanced Forward CLTU service provider shall conform to the state transition matrix specified in table 4-1.

NOTE –   The state table specified in table 4-1 represents one instance of service, and thus one association.

4.2.1.2   All actions including state transitions specified for a given state and a given event shall be performed before a subsequent event is considered.

4.2.1.3   SLE-PDUs shall be sent in the sequence specified in table 4-1.

4.2.1.4   Implementations shall ensure that events are not lost while an earlier event is being processed but are buffered in first-in first-out order for processing as soon as processing of the earlier event has completed.

4.2.1.5   The state transition matrix specified in table 4-1 represents one instance of service and thus one association. Once the association is established, if a CLTU-BIND invocation for a different association but for the same service instance is received, it shall be rejected with a CLTU-BIND return with the `result` parameter set to 'negative result' and the `diagnostic` parameter set to 'already bound'. This event shall not affect the association already in place.

4.2.2   STATE TABLE

NOTES

1   The state table specifies operation interactions and state transitions for the service provider.

2   The first column simply numbers the rows of the table.

3   The second column lists all incoming events. Where these events correspond to the arrival of an incoming SLE-PDU, the ASN.1 type defined for this SLE-PDU in annex A is indicated in parentheses, ( ).

4   Where an event is internal to the provider, its description is put in quotation marks, ' '. These events are defined in table 4-2.

5   The three columns (one column per state) on the right side of the table specify the behavior the provider will exhibit, which depends on the current state and the incoming event.   In some cases, the behavior in addition depends on Boolean conditions, also referred to as predicates. Such predicates are put in double quotation marks, " ". The predicates are defined in table 4-3. Predicates that are simple Boolean variables set only by the state machine itself are referred to as Boolean flags and specified in table 4-4. The dependency on a predicate is presented in the form of an 'IF <condition> THEN <action> [ELSEIF <condition> THEN <action>] ELSE <action>' clause.

6   If the action given in the table is simply to send a specific SLE-PDU, that action is indicated by the appearance of the name of the ASN.1 type of the SLE-PDU to be sent in parentheses, ( ). If that SLE-PDU is a return, the name may be preceded by the plus symbol ('+') to indicate that `result` is 'positive result', or by the negative symbol ('-') to indicate 'negative result'. Where several actions are to be taken (referred to as a 'compound action'), the name of the compound action is put in curly braces { }. The individual actions making up each compound action are identified in table 4-5.

7   'Not applicable' is stated where the given event can only occur in the given state because of an implementation error on the provider side.

8   Where the consequences of an incoming event are not visible to the user because the provider does not send any SLE-PDU in reaction to the given event, the action is put in square brackets, [ ].

9   State transitions are indicated by an arrow and the number of the state that will be entered; for example, → 1 indicates the transition to state 1.

10   Whenever the provider invokes a confirmed operation with `invoke-ID` set to <n>, it starts an associated return <n> timer. Should this timer expire before the return <n> is received, the provider shall invoke CLTU-PEER-ABORT.

Table 4-1: Provider Behavior

	Incoming Event	Unbound (State 1)	Ready (State 2)	Active (State 3)
1.	(cltuBindInvocation)	IF "positive result" THEN (+cltuBindReturn) → 2 ELSE (-cltuBindReturn)	{peer abort 'protocol error'}　　→ 1	{peer abort 'protocol error'}　　→ 1
2.	'end of service instance provision period'	{clean up}	{peer abort ('end of service instance provision period')}　→ 1	{peer abort ('end of service instance provision period')}　→ 1
3.	(cltuUnbindInvocation)	[ignore]	(cltuUnbindReturn) → 1 stop reporting-cycle timer IF "end" THEN release resources ELSE [ignore]	{peer abort ('protocol error')}　　→ 1
4.	(cltuStartInvocation)	[ignore]	IF "positive result" THEN (+cltuStartReturn)　　→ 3 ELSE (-cltuStartReturn)	{peer abort ('protocol error')}　　→ 1
5.	(cltuStopInvocation)	[ignore]	{peer abort ('protocol error')}　→ 1	IF "positive result" THEN {initiate stop}　　→ 2 ELSE (-cltuStopReturn)
6.	(cltuTransferDataInvocation)	[ignore]	{peer abort ('protocol error')}　→ 1	IF "positive result" .AND. (.NOT. "service instance blocked") THEN buffer SLE-FTDU 　(+cltuTransferDataReturn) ELSE discard SLE-FTDU 　(-cltuTransferDataReturn)

	Incoming Event	Unbound (State 1)	Ready (State 2)	Active (State 3)
7.	(cltuScheduleStatusReportInvocation)	[ignore]	IF "positive result" THEN (+cltuScheduleStatusReportReturn) 　IF "immediately" 　THEN {immediate report} 　ELSE IF "periodically" 　THEN {periodic report} 　ELSE stop reporting-cycle timer ELSE (-cltuScheduleStatusReportReturn)	IF "positive result" THEN (+cltuScheduleStatusReportReturn) 　IF "immediately" 　THEN {immediate report} 　ELSE IF "periodically" 　THEN {periodic report} 　ELSE stop reporting-cycle timer ELSE (-cltuScheduleStatusReportReturn)
8.	'reporting-cycle timer expired'	Not applicable	{periodic report}	{periodic report}
9.	'return-timeout-period timer <n> expired'	Not applicable	{peer abort ('return timeout')} → 1	{peer abort ('return timeout')　　　　→ 1
10.	(cltuGetParameterInvocation)	[ignore]	IF "positive result" THEN (+cltuGetParameterReturn) ELSE (-cltuGetParameterReturn)	IF "positive result" THEN (+cltuGetParameterReturn) ELSE (-cltuGetParameterReturn)
11.	(cltuThrowEventInvocation)	[ignore]	IF "positive result" THEN (+cltuThrowEventReturn) 　forward event to Complex Management ELSE (-cltuThrowEventReturn)	IF "positive result" THEN (+cltuThrowEventReturn) 　forward event to Complex Management ELSE (-cltuThrowEventReturn)
12.	'cltu radiated'	[ignore]	IF "report" THEN {notify('cltu radiated')} ELSE [ignore]	IF "report" THEN {notify('cltu radiated')} ELSE [ignore]
13.	'sldu expired'	IF "continue" THEN clear CLTU buffer ELSE [ignore]	Not applicable	IF "telecommand plop" THEN {notify('sldu expired') and block} ELSE notify('sldu expired')
14.	'production interrupted'	IF "continue" THEN clear CLTU buffer ELSE [ignore]	{notify('production interrupted') and clear} set "notify production operational" to TRUE	{notify('production interrupted') and block} set "notify production operational" to TRUE
15.	'production halted'	IF "continue" THEN clear CLTU buffer ELSE [ignore]	{notify('production halted') and clear} set "notify production operational" to TRUE	{notify('production halted') and block} set "notify production operational" to TRUE

	Incoming Event	Unbound (State 1)	Ready (State 2)	Active (State 3)
16.	'production operational'	[ignore]	IF "notify production operational" THEN {notify('production operational')} set "notify production operational" to FALSE	IF "notify production operational" THEN {notify('production operational')} set "notify production operational" to FALSE
17.	'buffer empty'	[ignore]	Not applicable	{notify('buffer empty')}
18.	'action list completed'	Not applicable	{notify('action list completed')}	{notify('action list completed')}
19.	'action list not completed'	Not applicable	{notify('action list not completed')}	{notify('action list not completed')}
20.	'event condition evaluated to false'	Not applicable	{notify('event condition evaluated to false')}	{notify('event condition evaluated to false')}
21.	(cltuPeerAbortInvocation)	[ignore]	{clean up} → 1	{clean up} → 1
22.	'protocol abort'	[ignore]	{clean up} → 1	IF "continue" THEN stop reporting-cycle timer → 1 ELSE {clean up} → 1
23.	'unsolicited invoke-ID'	[ignore]	{peer abort ('unsolicited invoke-ID')} → 1	{peer abort ('unsolicited invoke-ID')} → 1
24.	'invalid SLE-PDU'	[ignore]	{peer abort ('encoding error')} → 1	{peer abort ('encoding error')} → 1
25.	'unexpected SLE-PDU'	[ignore]	{peer abort ('protocol error')} → 1	{peer abort ('protocol error')} → 1
26.	'not authenticated SLE-PDU'	[ignore]	[ignore]	[ignore]

Table 4-2: Event Description References

Event	Reference
'action list completed'	3.8.2.3
'action list not completed'	3.8.2.3
'buffer empty'	3.8.2.3
'cltu radiated'	3.8.2.3
'end of service instance provision period'	3.13.2.2
'event condition evaluated to false'	3.8.2.3
'invalid SLE-PDU'	4.1.2
'not authenticated SLE-PDU'	4.1.7
'production halted'	3.8.2.3
'production interrupted'	3.8.2.3
'production operational'	3.8.2.3
'protocol abort'	4.1.5.4
'reporting-cycle timer expired'	3.9.2.6
'return-timeout-period timer <n> expired'	4.1.3
'sldu expired'	3.8.2.3
'unexpected SLE-PDU'	4.1.1
'unsolicited invoke-ID'	3.13.2.2

Table 4-3: Predicate Definitions

Predicate	Evaluates to TRUE if
"telecommand plop"	the `plop-in-effect` is either 'PLOP-1' or 'PLOP-2'
"continue"	the `protocol-abort-mode` parameter value is 'continue'
"end"	all checks on the (cltuUnbindInvocation) PDU are passed and the `unbind-reason` parameter value is 'end'
"immediately"	all parameter checks on the (cltuScheduleStatusReportInvocation) PDU are passed and the `report-request-type` parameter value is 'immediately'
"periodically"	all parameter checks on the (cltuScheduleStatusReportInvocation) PDU are passed and the `report-request-type` parameter value is 'periodically'
"positive result"	all checks on the given invocation PDU are passed
"report"	the `report` parameter value in the associated (cltuTransferDataInvocation) PDU is 'produce report'

Table 4-4: Boolean Flags

Flag Name	Initial Value
"service instance blocked"	FALSE
"notify production operational"	FALSE

Table 4-5: Compound Action Definitions

Name	Actions Performed
{clean up}	a) stop reporting-cycle timer b) reset reporting-cycle c) clear CLTU buffer d) set "service instance blocked" to FALSE
{immediate report}	a) (cltuStatusReportInvocation) b) stop reporting-cycle timer
{initiate stop}	a) clear CLTU buffer b) set "service instance blocked" to FALSE c) (+cltuStopReturn)
{notify('xxxx')}	(cltuAsyncNotifyInvocation) with `notification-type` set to 'xxxx'; i.e., to the value corresponding to the incoming event
{notify('xxxx') and clear}	a) (cltuAsyncNotifyInvocation) with `notification-type` set to 'xxxx'; i.e., to the value corresponding to the incoming event b) clear CLTU buffer
{notify('xxxx') and block}	a) (cltuAsyncNotifyInvocation) with `notification-type` set to 'xxxx'; i.e., to the value corresponding to the incoming event b) set "service instance blocked" to TRUE c) clear CLTU buffer
{peer abort ('xxxx')}	a) (cltuPeerAbortInvocation) with `diagnostic` set to 'xxxx' b) stop reporting-cycle timer c) reset reporting-cycle d) clear CLTU buffer e) set "service instance blocked" to FALSE
{periodic report}	a) (cltuStatusReportInvocation) b) set reporting-cycle timer to the `reporting-cycle` value in the most recent (cltuScheduleStatusReportInvocation) c) start reporting-cycle timer

ANNEX A

DATA TYPE DEFINITIONS

(NORMATIVE)

A1 INTRODUCTION

A1.1 This annex defines the data types that are used by the Enhanced Forward CLTU service. It is intended to provide a clear specification of these data types and to avoid ambiguity. It is not intended to constrain how these data types are implemented or encoded. These definitions are suitable for inclusion in any type of ASN.1-based protocol that implements the Enhanced Forward CLTU service.

NOTE – For the purposes of this Experimental Specification, the Enhanced Forward CLTU ASN.1 service data types are defined as later versions (101) of the corresponding Forward CLTU ASN.1 service data types. In particular, the Forward Transfer Data Units (FTDUs) are called 'CLTUs' in the ASN.1 specifications. If and when the capabilities of the Enhanced Forward CLTU service are realized as part of an actual CSTS, the formal specification of the ASN.1 service data types will be written in accordance with the CSTS rules for ASN.1 service data types and will be specific to the actual CSTS.

A1.2 The data type definitions are presented in seven ASN.1 modules.

A1.3 Subsection A2.1 contains basic data types that are common with other SLE transfer services. As additional services are specified by CCSDS, further data types may be added to this module or existing types may be extended. However, that eventuality is not expected to invalidate the module in its present form because it is expected that an implementation compliant with a future extended version of this module will be interoperable with an implementation based on its present version.

A1.4 Subsection A2.2 specifies the SLE-PDUs exchanged between an SLE user and an SLE provider application in order to establish, release, or abort an association. They are common among SLE transfer service types.

A1.5 Subsection A2.3 specifies SLE-PDUs related to invocations and returns that are common to SLE transfer service types.

A1.6 Subsection A2.4 specifies the format of the Service Instance Identifiers.

A1.7 Subsection A2.5 specifies data types specific to the Enhanced Forward CLTU service. In part, these specific types are derived from types specified in A2.1 by means of subtyping.

A1.8   Subsection A2.6 specifies all incoming SLE-PDUs of the Enhanced Forward CLTU service, as seen from a provider point of view. Where applicable, these SLE-PDUs are mapped to the generic SLE-PDUs defined in A2.2 and A2.3.

A1.9   Subsection A2.7 specifies in the same way all outgoing SLE-PDUs.

A1.10 Although A2.2, A2.3, A2.6 and A2.7 define the SLE-PDUs that will be exchanged between the SLE provider and user applications, they shall not be interpreted as requiring that these SLE-PDUs be completely mapped to the user data field of the underlying communications protocol. For example, depending on the communications protocol(s) used, part of the SLE-PDUs may be used to determine the appropriate setting of protocol control information.

A2   CLTU DATA TYPE SPECIFICATION

A2.1   SLE TRANSFER SERVICE COMMON TYPES

```
CCSDS-SLE-TRANSFER-SERVICE-COMMON-TYPES

{  iso identified-organization(3)
   standards-producing-organization(112) ccsds(4)
   space-link-extension(3) sle-transfer-services(1)
   modules(1) common-modules(99) version-one-hundred-one(101)
   asn1-common-types(1)
}

DEFINITIONS
IMPLICIT TAGS
::=   BEGIN

EXPORTS  ConditionalTime
,        Credentials
,        DeliveryMode
,        Diagnostics
,        Duration
,        ForwardDuStatus
,        IntPosLong
,        IntPosShort
,        IntUnsignedLong
,        IntUnsignedShort
,        InvokeId
,        ParameterName
,        SlduStatusNotification
,        SpaceLinkDataUnit
,        Time
;

ConditionalTime        ::=   CHOICE
{  undefined        [0]   NULL
,  known            [1]   Time
}
```

```
-- If credentials are used, it will be necessary that
-- the internal structure of the octet string is known
-- to both parties. Since the structure will depend on
-- algorithm used, it is not specified here. However,
-- the peer entities may use ASN.1 encoding to make the
-- internal structure visible.
Credentials              ::=   CHOICE
{ unused           [0]   NULL
, used             [1]   OCTET STRING (SIZE (8 .. 256))
}

DeliveryMode             ::=   INTEGER
{ rtnTimelyOnline              (0)
, rtnCompleteOnline            (1)
, rtnOffline                   (2)
, fwdOnline                    (3)
, fwdOffline                   (4)
}

Diagnostics              ::=   INTEGER
{ duplicateInvokeId            (100)
, otherReason                  (127)
}

-- The Duration is expressed in microseconds
Duration                 ::=   IntUnsignedLong

ForwardDuStatus          ::=   INTEGER
{ radiated                     (0)
, expired                      (1)
, interrupted                  (2)
, acknowledged                 (3)   -- FSP
, productionStarted            (4)   -- CLTU: 'radiation started'
, productionNotStarted         (5)   -- CLTU: 'radiation not started'
, unsupportedTransmissionMode  (6)   -- FSP
}

-- 1 to (2^32)-1
IntPosLong               ::=   INTEGER (1 .. 4294967295)

-- 1 to (2^16)-1
IntPosShort              ::=   INTEGER (1 .. 65535)

-- 0 to (2^32)-1
IntUnsignedLong          ::=   INTEGER (0 .. 4294967295)

-- 0 to (2^16)-1
IntUnsignedShort         ::=   INTEGER (0 .. 65535)

InvokeId                 ::=   IntUnsignedShort
```

```
ParameterName                              ::=    INTEGER
{   acquisitionSequenceLength               (201)
,   apidList                                (2)
,   bitLockRequired                         (3)
,   blockEncode                             (1012)
,   blockingTimeoutPeriod                   (0)
,   blockingUsage                           (1)
,   bufferSize                              (4)
,   clcwGlobalVcid                          (202)
,   clcwPhysicalChannel                     (203)
,   convolutionallyEncode                   (1013)
,   deliveryMode                            (6)
,   directiveInvocation                     (7)
,   directiveInvocationOnline               (108)
,   expectedDirectiveIdentification         (8)
,   expectedEventInvocationIdentification   (9)
,   expectedSlduIdentification              (10)
,   fopSlidingWindow                        (11)
,   fopState                                (12)
,   latencyLimit                            (15)
,   mapList                                 (16)
,   mapMuxControl                           (17)
,   mapMuxScheme                            (18)
,   maximumFrameLength                      (19)
,   maximumPacketLength                     (20)
,   maximumSlduLength                       (21)
,   minimumDelayTime                        (204)
,   modulationFrequency                     (22)
,   modulationIndex                         (23)
,   notificationMode                        (205)
,   permittedControlWordTypeSet             (101)
,   permittedGvcidSet                       (24)
,   permittedTcVcidSet                      (102)
,   permittedTransmissionMode               (107)
,   permittedUpdateModeSet                  (103)
,   plop1IdleSequenceLength                 (206)
,   plopInEffect                            (25)
,   protocolAbortMode                       (207)
,   randomize                               (1015)
,   reportingCycle                          (26)
,   requestedControlWordType                (104)
,   requestedFrameQuality                   (27)
,   requestedGvcid                          (28)
,   requestedTcVcid                         (105)
,   requestedUpdateMode                     (106)
,   returnTimeoutPeriod                     (29)
,   rfAvailable                             (30)
,   rfAvailableRequired                     (31)
,   segmentHeader                           (32)
,   slPduLength                             (1014)
,   subcarrierToBitRateRatio                (34)
,   timeoutType                             (35)
,   timerInitial                            (36)
,   transmissionLimit                       (37)
,   transmitterFrameSequenceNumber          (38)
,   vcMuxControl                            (39)
,   vcMuxScheme                             (40)
,   virtualChannel                          (41)
}
```

```
SlduStatusNotification  ::=   INTEGER
{  produceNotification         (0)
,  doNotProduceNotification    (1)
}

SpaceLinkDataUnit        ::=   OCTET STRING (SIZE (1 .. 65536))

Time                    ::=   CHOICE
{  ccsdsFormat       [0]   TimeCCSDS
,  ccsdsPicoFormat   [1]   TimeCCSDSpico
}

TimeCCSDS               ::=   OCTET STRING (SIZE(8))
-- P-field is implicit (not present, defaulted to 41 hex
-- T-field:
-- 2 octets: number of days since 1958/01/01 00:00:00
-- 4 octets: number of milliseconds of the day
-- 2 octets: number of microseconds of the millisecond
--        (set to 0 if not used)
-- This definition reflects exactly the format of the CCSDS defined
-- time tag as used in spacelink data units (see Time Code Formats.
-- Recommendation for Space Data System Standards, CCSDS 301.0-B-3.
-- Blue Book.  Issue 3.  Washington, D.C.: CCSDS, January 2002).

TimeCCSDSpico           ::=   OCTET STRING (SIZE(10))
-- P-field is implicit (not present, defaulted to 42 hex
-- T-field:
-- 2 octets: number of days since 1958/01/01 00:00:00
-- 4 octets: number of milliseconds of the day
-- 4 octets: number of picoseconds of the millisecond
--        (set to 0 if not used)
-- This definition reflects exactly the format of the CCSDS defined
-- time tag as used in spacelink data units (see Time Code Formats.
-- Recommendation for Space Data System Standards, CCSDS 301.0-B-3.
-- Blue Book.  Issue 3.  Washington, D.C.: CCSDS, January 2002).

END
```

A2.2   SLE-TRANSFER SERVICE BIND TYPES

```
CCSDS-SLE-TRANSFER-SERVICE-BIND-TYPES

{  iso identified-organization(3)
   standards-producing-organization(112) ccsds(4)
   space-link-extension(3) sle-transfer-services(1)
   modules(1) common-modules(99) version- one-hundred-one(101)
   asn1-bind-types(2)
}

DEFINITIONS
IMPLICIT TAGS
::=   BEGIN
```

```
EXPORTS  SleBindInvocation
,        SleBindReturn
,        SlePeerAbort
,        SleUnbindInvocation
,        SleUnbindReturn
;


IMPORTS  Credentials
,        IntPosShort
    FROM CCSDS-SLE-TRANSFER-SERVICE-COMMON-TYPES

         ServiceInstanceIdentifier
    FROM CCSDS-SLE-TRANSFER-SERVICE-SERVICE-INSTANCE-ID
;


-- ============================================================
-- The first part of the module definition contains the SLE-PDUs
-- ============================================================

SleBindInvocation           ::=    SEQUENCE
{ invokerCredentials        Credentials
, initiatorIdentifier       AuthorityIdentifier
, responderPortIdentifier   PortId
, serviceType               ApplicationIdentifier
, versionNumber             VersionNumber
, serviceInstanceIdentifier ServiceInstanceIdentifier
}

SleBindReturn           ::=    SEQUENCE
{ performerCredentials   Credentials
, responderIdentifier    AuthorityIdentifier
, result                 CHOICE
   { positive          [0]   VersionNumber
   , negative          [1]   BindDiagnostic
   }
}

SlePeerAbort            ::=    PeerAbortDiagnostic

SleUnbindInvocation     ::=    SEQUENCE
{ invokerCredentials    Credentials
, unbindReason          UnbindReason
}

SleUnbindReturn         ::=    SEQUENCE
{ responderCredentials  Credentials
, result                CHOICE
   { positive         [0]   NULL
   }
}


-- ============================================================
-- The second part of the module definition contains the types
-- used by the SLE-PDUs declared in the first part.
-- ============================================================
```

```
ApplicationIdentifier        ::=    INTEGER
{  rtnAllFrames              (0)
,  rtnInsert                 (1)
,  rtnChFrames               (2)
-- rtnChFrames includes rtnMcFrames and rtnVcFrames
,  rtnChFsh                  (3)
-- rtnChFsh includes rtnMcFsh and rtnVcFsh
,  rtnChOcf                  (4)
-- rtnChOcf includes rtnMcOcf and rtnVcOcf
,  rtnBitstr                 (5)   -- AOS
,  rtnSpacePkt               (6)
,  fwdAosSpacePkt            (7)
,  fwdAosVca                 (8)
,  fwdBitstr                 (9)
,  fwdProtoVcdu              (10)
,  fwdInsert                 (11)
,  fwdCVcdu                  (12)
,  fwdTcSpacePkt             (13)  -- conventional telecommand
,  fwdTcVca                  (14)  -- conventional telecommand
,  fwdTcFrame                (15)
,  fwdCltu                   (16)
}

AuthorityIdentifier     ::=    IdentifierString (SIZE (3 .. 16))

BindDiagnostic                 ::=    INTEGER
{  accessDenied                 (0)
,  serviceTypeNotSupported      (1)
,  versionNotSupported          (2)
,  noSuchServiceInstance        (3)
,  alreadyBound                 (4)
,  siNotAccessibleToThisInitiator  (5)
,  inconsistentServiceType      (6)
,  invalidTime                  (7)
,  outOfService                 (8)
,  otherReason                  (127)
}

IdentifierString        ::=    VisibleString (FROM (ALL EXCEPT " "))

LogicalPortName         ::=    IdentifierString (SIZE (1 .. 128))

PeerAbortDiagnostic            ::=    INTEGER
{  accessDenied                (0)
,  unexpectedResponderId       (1)
,  operationalRequirement      (2)
,  protocolError               (3)
,  communicationsFailure       (4)
,  encodingError               (5)
,  returnTimeout               (6)
,  endOfServiceProvisionPeriod (7)
,  unsolicitedInvokeId         (8)
,  otherReason                 (127)
-- The range of this INTEGER shall be (0 .. 255).
-- (128 .. 255) is reserved for diagnostic codes
-- that are specific to the communications technology used.
}

PortId                  ::=    LogicalPortName
```

```
UnbindReason            ::=    INTEGER
{ end                   (0)
, suspend               (1)
, versionNotSupported   (2)
, other                 (127)
}


VersionNumber           ::=    IntPosShort

END
```

A2.3   SLE TRANSFER SERVICE COMMON PDUS

```
CCSDS-SLE-TRANSFER-SERVICE-COMMON-PDUS

{  iso identified-organization(3)
   standards-producing-organization(112) ccsds(4)
   space-link-extension(3) sle-transfer-services(1)
   modules(1) common-modules(99) version- one-hundred-one(101)
   asn1-common-pdu(3)
}

DEFINITIONS
IMPLICIT TAGS
::=   BEGIN

EXPORTS  ReportingCycle
,        SleAcknowledgement
,        SleScheduleStatusReportInvocation
,        SleScheduleStatusReportReturn
,        SleStopInvocation
;


IMPORTS  Credentials
,        Diagnostics
,        InvokeId
   FROM CCSDS-SLE-TRANSFER-SERVICE-COMMON-TYPES
;


-- ==============================================================
-- The first part of the module definition contains the SLE-PDUs
-- ==============================================================

SleAcknowledgement      ::=    SEQUENCE
{ credentials           Credentials
, invokeId              InvokeId
, result                CHOICE
   { positiveResult    [0]   NULL
   , negativeResult    [1]   Diagnostics
   }
}

SleScheduleStatusReportInvocation   ::=    SEQUENCE
{ invokerCredentials              Credentials
, invokeId                        InvokeId
, reportRequestType               ReportRequestType
}
```

```
SleScheduleStatusReportReturn ::=    SEQUENCE
{ performerCredentials        Credentials
, invokeId                    InvokeId
, result                      CHOICE
   { positiveResult       [0]   NULL
   , negativeResult       [1]   DiagnosticScheduleStatusReport
   }
}

SleStopInvocation        ::=    SEQUENCE
{ invokerCredentials      Credentials
, invokeId                InvokeId
}


-- ==============================================================
-- The second part of the module definition contains the types
-- used by the SLE-PDUs declared in the first part.
-- ==============================================================

DiagnosticScheduleStatusReport   ::=   CHOICE
{ common                       [0]   Diagnostics
, specific                     [1]   INTEGER
   { notSupportedInThisDeliveryMode   (0)   -- not used for
                                            -- version 1 forward transfer
                                            -- services
   ,      alreadyStopped            (1)
   ,      invalidReportingCycle     (2)
   }
}

-- The cycle duration is expressed in seconds
ReportingCycle           ::=    INTEGER (2 .. 600)

ReportRequestType        ::=    CHOICE
{ immediately       [0]   NULL
, periodically      [1]   ReportingCycle
, stop              [2]   NULL
}

END
```

A2.4   SLE TRANSFER SERVICE— SERVICE-INSTANCE-ID


```
CCSDS-SLE-TRANSFER-SERVICE-SERVICE-INSTANCE-ID

{ iso identified-organization (3) standards-producing-organization(112)
   ccsds(4) space-link-extension(3) sle-transfer-services(1)
   modules(1) common-modules(99) version- one-hundred-one(101)
   asn1-service-id-type(4)
}

DEFINITIONS
IMPLICIT TAGS
::= BEGIN
```

```
EXPORTS ServiceInstanceIdentifier
;

-- Object Identifiers definition
sagr      OBJECT IDENTIFIER     ::=     {iso 3 112 4 3 1 2 52}
spack     OBJECT IDENTIFIER     ::=     {iso 3 112 4 3 1 2 53}
fsl-fg    OBJECT IDENTIFIER     ::=     {iso 3 112 4 3 1 2 14}
rsl-fg    OBJECT IDENTIFIER     ::=     {iso 3 112 4 3 1 2 38}
cltu      OBJECT IDENTIFIER     ::=     {iso 3 112 4 3 1 2 7}
fsp       OBJECT IDENTIFIER     ::=     {iso 3 112 4 3 1 2 10}
raf       OBJECT IDENTIFIER     ::=     {iso 3 112 4 3 1 2 22}
rcf       OBJECT IDENTIFIER     ::=     {iso 3 112 4 3 1 2 46}
rcfsh     OBJECT IDENTIFIER     ::=     {iso 3 112 4 3 1 2 44}
rocf      OBJECT IDENTIFIER     ::=     {iso 3 112 4 3 1 2 49}
rsp       OBJECT IDENTIFIER     ::=     {iso 3 112 4 3 1 2 40}
tcf       OBJECT IDENTIFIER     ::=     {iso 3 112 4 3 1 2 12}
tcva      OBJECT IDENTIFIER     ::=     {iso 3 112 4 3 1 2 16}
```

```
-- +++++++++++++++++++++++++++++++++++++
-- +++++++++++++++++++++++++++++++++++++
-- The Service Instance Identifier is a concatenation of Attributes.
-- Each Attribute is composed of a name and a value.
-- Note that the Attribute names are not specified by the ASN.1 proper,
-- but by means of comments in this module.
-- The following Attributes identify the Service Instance:
-- Service agreement
--     the name of which is:    "sagr",
--     the value of which is to be agreed between the user and the provider;
-- Service package
--     the name of which is:    "spack",
--     the value of which is to be agreed between the user and the provider;
-- Forward/Return Service Functional Group
--     the name of which is:
--        "fsl-fg" for the forward services,
--     or:
--        "rsl-fg" for the return services,
--     the value of which is to be agreed between the user and the provider;
-- Service Name identifier,
--        the name of which is:    "cltu"
--        the value of which is:  "cltu" plus an instance number;
--     or:
--        the name of which is:    "fsp",
--        the value of which is:  "fsp" plus an instance number;
--     or:
--        the name of which is:    "tcva",
--        the value of which is:  "tcva" plus an instance number;
--     or:
--        the name of which is:    "tcf",
--        the value of which is:  "tcf" plus an instance number;
--     or:
--        the name of which is:    "raf",
--        the value ofwhich is:    "onlc" or "onlt" or "offl" plus an
--        instance number;
--     or:
--        the name of which is:    "rcf",
--        the value of which is:  "onlc" or "onlt" or "offl" plus an
--        instance number;
--     or:
--        the name of which is:    "rocf",
--        the value of which is:  "onlc" or "onlt" or "offl" plus an
--        instance number;
--     or:
--        the name of which is:    "rcfsh",
--        the value of which is:  "onlc" or "onlt" or "offl" plus an
--        instance number;
--     or:
--        the name of which is:    "rsp",
--        the value of which is:  "onlc" or "onlt" or "offl" plus an
--        instance number.
--
-- To obtain the string correponding to the instance identifier, one must
-- concatenate the Attributes values and Names as follows:
-- 'Atribute Name"="Attrbute value'.
-- Two Attributes must be separated by a ".".
--
-- Example: 'sagr=xyz.spack=abcdef.rsl-fg=gfjdy.raf=onlc2'
-- Note that the quotes are not part of the string.
```

```
ATTRIBUTE              ::=    CLASS
{ &id                   OBJECT IDENTIFIER UNIQUE
}
WITH SYNTAX { ID  &id }

GeneralAttributes ATTRIBUTE   ::=
{  serviceAgreement
|  servicePackage
|  forwardService
|  returnService
}

ServiceInstanceAttribute   ::=      SET SIZE(1) OF SEQUENCE
{  identifier                    ATTRIBUTE.&id
                                 ({ServiceInstanceAttributes})
,  siAttributeValue              VisibleString (SIZE (1 .. 256))
}

ServiceInstanceAttributes ATTRIBUTE ::=
{  GeneralAttributes
|  ServiceNames
}

ServiceInstanceIdentifier  ::=   SEQUENCE OF ServiceInstanceAttribute

ServiceNames ATTRIBUTE  ::=
{  rafService
|  rcfService
|  rcfshService
|  rocfService
|  rspService
|  cltuService
|  fspService
|  tcfService
|  tcvaService
}

-- ++++++++++++++++++++++++++++++++++++
-- CLTU Service Definition
-- The cltu Service Id starts with "cltu" and is
-- followed by the instance number.
cltuService ATTRIBUTE      ::=   { ID cltu  }

-- ++++++++++++++++++++++++++++++++++++
-- Forward Service Definition
forwardService ATTRIBUTE   ::=   { ID fsl-fg  }

-- ++++++++++++++++++++++++++++++++++++
-- FSP Service Definition
-- The fsp Service Id starts with "fsp" and is
-- followed by the instance number.
fspService ATTRIBUTE       ::=   { ID fsp  }

-- ++++++++++++++++++++++++++++++++++++
-- RAF Service Definition
-- The raf Service Id starts with "onlc" or "onlt" or "offl" and is
-- followed by the instance number.
rafService ATTRIBUTE       ::=   { ID raf  }
```

```
-- +++++++++++++++++++++++++++++++++++++
-- RCF Service Definition
-- The rcf Service Id starts with "onlc" or "onlt" or "offl" and is
-- followed by the instance number.
rcfService ATTRIBUTE        ::=   {  ID rcf   }


-- +++++++++++++++++++++++++++++++++++++
-- RCFSH Service Definition
-- The rcfsh Service Id starts with "onlc" or "onlt" or "offl" and is
-- followed by the instance number.
rcfshService ATTRIBUTE      ::=   {  ID rcfsh }


-- +++++++++++++++++++++++++++++++++++++
-- Return Service Definition
returnService ATTRIBUTE     ::=   {  ID rsl-fg   }


-- +++++++++++++++++++++++++++++++++++++
-- ROCF Service Definition
-- The rocf Service Id starts with "onlc" or "onlt" or "offl" and is
-- followed by the instance number.
rocfService ATTRIBUTE       ::=   {  ID rocf  }


-- +++++++++++++++++++++++++++++++++++++
-- RSP Service Definition
-- The rocf Service Id starts with "onlc" or "onlt" or "offl" and is
-- followed by the instance number.
rspServiceId ATTRIBUTE      ::=   {  ID rsp   }


-- +++++++++++++++++++++++++++++++++++++
-- Service Agreement Definition
serviceAgreement ATTRIBUTE ::=   {  ID sagr  }


-- +++++++++++++++++++++++++++++++++++++
-- Service Package Definition
servicePackage ATTRIBUTE    ::=   {  ID spack }


-- +++++++++++++++++++++++++++++++++++++
-- TCF Service Definition
-- The tcf Service Id starts with "tcf" and is
-- followed by the instance number.
tcfServiceId ATTRIBUTE      ::=   {  ID tcf   }


-- +++++++++++++++++++++++++++++++++++++
-- TCVA Service Definition
-- The tcva Service Id starts with "tcva" and is
-- followed by the instance number.
tcvaServiceId ATTRIBUTE     ::=   {  ID tcva  }

END
```

A2.5   CLTU TRANSFER STRUCTURES

```
CCSDS-SLE-TRANSFER-SERVICE-CLTU-STRUCTURES

{ iso identified-organization (3) standards-producing-organization(112)
ccsds(4) space-link-extension(3) sle-transfer-services(1)
modules(1) forward-cltu-service(21) version-one-hundred-one(101)
asn1-transfer-structures(1)}

DEFINITIONS
IMPLICIT TAGS
::= BEGIN

EXPORTS  BufferSize
,         CltuData
,         CltuGetParameter
,         CltuIdentification
,         CltuLastProcessed
,         CltuLastOk
,         CltuNotification
,         CltuParameterName
,         CltuStatus
,         DiagnosticCltuGetParameter
,         DiagnosticCltuStart
,         DiagnosticCltuThrowEvent
,         DiagnosticCltuTransferData
,         EventInvocationId
,         NumberOfCltusProcessed
,         NumberOfCltusRadiated
,         NumberOfCltusReceived
,         ProductionStatus
,         UplinkStatus
;
IMPORTS  ConditionalTime
,         DeliveryMode
,         Diagnostics
,         Duration
,         ForwardDuStatus
,         IntPosLong
,         IntPosShort
,         IntUnsignedLong
,         IntUnsignedShort
,         ParameterName
,         SpaceLinkDataUnit
,         Time
   FROM CCSDS-SLE-TRANSFER-SERVICE-COMMON-TYPES
          ReportingCycle
   FROM CCSDS-SLE-TRANSFER-SERVICE-COMMON-PDUS
;

-- Available buffer in octets
BufferSize                            ::=    IntUnsignedLong

CltuData                              ::=    SpaceLinkDataUnit

CltuDeliveryMode                      ::=    DeliveryMode
(  fwdOnline
)
```

```
CltuGetParameter                   ::=   CHOICE
{ parAcquisitionSequenceLength     [0]   SEQUENCE
   { parameterName                 ParameterName
(acquisitionSequenceLength)
   , parameterValue                IntUnsignedShort
   }
,  parBitLockRequired             [1]    SEQUENCE
{ parameterName                    ParameterName  (bitLockRequired)
,  parameterValue                  INTEGER
   { yes                                 (0)
   , no                                  (1)
   }
}
,  parGlobalVcid                  [2]    SEQUENCE
{ parameterName                    ParameterName  (clcwGlobalVcid)
,  parameterValue                  GvcId
}
,  parClcwPhysicalChannel         [3]    SEQUENCE
{ parameterName                    ParameterName  (clcwPhysicalChannel)
,  parameterValue                  VisibleString (SIZE (1..32))
}
,  parDeliveryMode                [4]    SEQUENCE
   { parameterName                 ParameterName (deliveryMode)
   , parameterValue                CltuDeliveryMode
   }
,  parCltuIdentification          [5]    SEQUENCE
   { parameterName                 ParameterName
(expectedSlduIdentification)
   , parameterValue                CltuIdentification
   }
,  parEventInvocationIdentification [6]   SEQUENCE
   { parameterName                 ParameterName
(expectedEventInvocationIdentification)
   , parameterValue                EventInvocationId
   }
,  parMaximumCltuLength           [7]    SEQUENCE
   { parameterName                 ParameterName (maximumSlduLength)
   , parameterValue                INTEGER (12 .. 4096)
   }
,  parMinimumDelayTime            [8]    SEQUENCE
   { parameterName                 ParameterName (minimumDelayTime)
   , parameterValue                Duration
   }
,  parModulationFrequency         [9]    SEQUENCE
   { parameterName                 ParameterName (modulationFrequency)
   , parameterValue                ModulationFrequency
   }
,  parModulationIndex             [10]   SEQUENCE
   { parameterName                 ParameterName (modulationIndex)
   , parameterValue                ModulationIndex
   }
,  parNotificationMode            [11]   SEQUENCE
   { parameterName                 ParameterName (notificationMode)
   , parameterValue                INTEGER
{ deferred                              (0)
, immediate                             (1)
   }
}
,  parPlop1IdleSequenceLength     [12]   SEQUENCE
   { parameterName                 ParameterName
```

```
(plop1IdleSequenceLength)
    ,  parameterValue                 IntUnsignedShort
    }
,  parPlopInEffect          [13]  SEQUENCE
    {  parameterName                  ParameterName (plopInEffect)
    ,  parameterValue                 INTEGER
{  plop1                              (0)
,  plop2                              (1)
,  ascp                               (2)
    }
}
,  parProtocolAbortMode     [14]  SEQUENCE
    {  parameterName                  ParameterName (protocolAbortMode)
    ,  parameterValue                 INTEGER
{  abort                              (0)
,  continue                           (1)
    }
}
,  parReportingCycle        [15]  SEQUENCE
{  parameterName                      ParameterName (reportingCycle)
,  parameterValue                     CurrentReportingCycle
}
,  parReturnTimeout         [16]  SEQUENCE
    {  parameterName                  ParameterName (returnTimeoutPeriod)
    ,  parameterValue                 TimeoutPeriod
    }
,  parRfAvailableRequired   [17]  SEQUENCE
{  parameterName                      ParameterName  (rfAvailableRequired)
,  parameterValue                     INTEGER
    {  yes                                (0)
    ,  no                                 (1)
    }
}
,  parSubcarrierToBitRateRatio  [18]  SEQUENCE
    {  parameterName                  ParameterName
(subcarrierToBitRateRatio)
    ,  parameterValue                 SubcarrierDivisor
    }
,  parBlockEncode           [20]  SEQUENCE
    {  parameterName                  ParameterName (blockEncode)
    ,  parameterValue                 INTEGER
{  none                               (0)
,  rs                                 (1)
,  ldpc                               (2)
    }
}
,  parConvolutionallyEncode [21]  SEQUENCE
{  parameterName                      ParameterName  (convolutionallyEncode)
,  parameterValue                     INTEGER
    {  yes                                (0)
    ,  no                                 (1)
    }
}
,  parSlPduLength          [22]  SEQUENCE
    {  parameterName                  ParameterName (slPduLength)
    ,  parameterValue                 IntUnsignedShort
    }
,  parRandomize            [24]  SEQUENCE
{  parameterName                      ParameterName  (randomize)
,  parameterValue                     INTEGER
```

```
    {  yes                                          (0)
    ,  no                                           (1)
    }
}
}

CltuIdentification                  ::=    IntUnsignedLong

CltuLastOk                              ::=    CHOICE
{ noCltuOk                              [0]      NULL
,  cltuOk                                [1]      SEQUENCE
{  cltuIdentification        CltuIdentification
,  radiationStopTime         Time
}
}

CltuLastProcessed              ::=   CHOICE
{  noCltuProcessed                  [0]   NULL
,  cltuProcessed                     [1]   SEQUENCE
{  cltuIdentification        CltuIdentification
,  radiationStartTime        ConditionalTime
,  cltuStatus                     CltuStatus
}
}

CltuNotification                 ::=   CHOICE
{  cltuRadiated                    [0]   NULL
,  slduExpired                     [1]   NULL
,  productionInterrupted           [2]   NULL
,  productionHalted              [3]   NULL
,  productionOperational           [4]   NULL
,  bufferEmpty                     [5]   NULL
,  actionListCompleted             [6]   EventInvocationId
,  actionListNotCompleted          [7]   EventInvocationId
,  eventConditionEvFalse           [8]   EventInvocationId
}


CltuParameterName            ::=    ParameterName
(  acquisitionSequenceLength
|  bitLockRequired
|  blockEncode
|  clcwGlobalVcid
|  clcwPhysicalChannel
|  convolutionallyEncode
|  deliveryMode
|  expectedSlduIdentification
|  expectedEventInvocationIdentification
|  slPduLength
|  maximumSlduLength
|  minimumDelayTime
|  modulationFrequency
|  modulationIndex
|  notificationMode
|  plop1IdleSequenceLength
|  plopInEffect
|  protocolAbortMode
|  reportingCycle
|  randomize
|  returnTimeoutPeriod
```

```
|  rfAvailableRequired
|  subcarrierToBitRateRatio
)

CltuStatus                              ::=     ForwardDuStatus
(  radiated
|  expired
|  interrupted
|  productionStarted                    -- radiation started
|  productionNotStarted                 -- radiation not started
)

CurrentReportingCycle           ::=     CHOICE
{  periodicReportingOff         [0]     NULL
,  periodicReportingOn          [1]     ReportingCycle
}

DiagnosticCltuGetParameter      ::=     CHOICE
{  common                               [0]     Diagnostics
,  specific                             [1]     INTEGER
   {  unknownParameter                  (0)
   }
}

DiagnosticCltuStart             ::=     CHOICE
{  common                               [0]     Diagnostics
,  specific                             [1]     INTEGER
   {  outOfService               (0)
   ,  unableToComply             (1)
   ,  productionTimeExpired      (2)
   ,  invalidCltuId              (3)
   }
}

DiagnosticCltuThrowEvent        ::=     CHOICE
{  common                               [0]     Diagnostics
,  specific                             [1]     INTEGER
   {  operationNotSupported      (0)
   ,  eventInvocIdOutOfSequence  (1)
   ,  noSuchEvent                (2)
   }
}

DiagnosticCltuTransferData      ::=     CHOICE
{  common                                    [0]     Diagnostics
,  specific                                  [1]     INTEGER
   {  unableToProcess                  (0)
   ,  unableToStore                    (1)
   ,  outOfSequence                    (2)
   ,  inconsistentTimeRange            (3)
   ,  invalidTime                      (4)
   ,  lateSldu                         (5)
   ,  invalidDelayTime                 (6)
   ,  cltuError                        (7)
   ,  invalidDataSize                  (8)
   }
}

EventInvocationId                       ::=     IntUnsignedLong
```

```
GvcId                          ::= SEQUENCE
{   spacecraftId                    INTEGER (0 .. 1023)
,   versionNumber                   INTEGER (0 .. 3)
,   vcId                            CHOICE
    {  masterChannel           [0]   NULL
  , virtualChannel             [1]   VcId
    }
}
-- Notes:
-- The permissible range of the spacecraftId depends on the versionNumber.
-- PTM and AOS frames are supported.
-- In case of AOS the permissible range is 0 to 255 (8 bits).
-- In case of PTM the permissible range is 0 to 1023 (10 bits).


-- either the subcarrier frequency, or,
-- in case of direct modulation, the bit rate, measured in 1/10 Hz
ModulationFrequency            ::=   IntPosLong

-- modulation index angle expressed in 1/1000 radians
ModulationIndex                ::=   IntPosShort

NumberOfCltusProcessed         ::=   IntUnsignedLong

NumberOfCltusRadiated          ::=   IntUnsignedLong

NumberOfCltusReceived          ::=   IntUnsignedLong


ProductionStatus               ::=   INTEGER
{   operational                     (0)
,   configured                      (1)
,   interrupted                     (2)
,   halted                          (3)
}

-- Divisor of the subcarrier frequency
-- If direct carrier modulation, the value is 1
SubcarrierDivisor              ::=   IntPosShort

-- measured in seconds
TimeoutPeriod                  ::=   INTEGER (1 .. 600)

UplinkStatus                   ::=   INTEGER
{   uplinkStatusNotAvailable        (0)
,   noRfAvailable                   (1)
,   noBitLock                       (2)
,   nominal                         (3)
}

VcId                           ::= INTEGER (0..63)
-- The actual permissible range depends on the version number
-- of the associated GVCID.

END
```

A2.6   CLTU INCOMING PDUS

```
CCSDS-SLE-TRANSFER-SERVICE-CLTU-INCOMING-PDUS

{ iso identified-organization(3) standards-producing-organization(112)
ccsds(4) space-link-extension(3) sle-transfer-services(1)
modules(1) forward-cltu-service(21) version- one-hundred-one(101)
asn1-incoming-pdus(2)}

DEFINITIONS
IMPLICIT TAGS
::= BEGIN

IMPORTS      ConditionalTime
,            Credentials
,            Duration
,            IntPosShort
,            InvokeId
,            SlduStatusNotification
FROM CCSDS-SLE-TRANSFER-SERVICE-COMMON-TYPES

             SleScheduleStatusReportInvocation
,            SleStopInvocation
FROM CCSDS-SLE-TRANSFER-SERVICE-COMMON-PDUS

             CltuData
,            CltuIdentification
,            CltuParameterName
,            EventInvocationId
FROM CCSDS-SLE-TRANSFER-SERVICE-CLTU-STRUCTURES

             SleBindInvocation
,            SlePeerAbort
,            SleUnbindInvocation
FROM CCSDS-SLE-TRANSFER-SERVICE-BIND-TYPES
;


-- ============================================================
-- The first part of the module definition contains the CLTU type
-- that contains all the possible PDUs the provider may receive.
-- ============================================================

CltuUserToProviderPdu         ::=   CHOICE

{ cltuBindInvocation                   [100] SleBindInvocation
, cltuUnbindInvocation                 [102] SleUnbindInvocation
, cltuStartInvocation                  [0]     CltuStartInvocation
, cltuStopInvocation                   [2]     SleStopInvocation
, cltuScheduleStatusReportInvocation   [4]

  SleScheduleStatusReportInvocation
, cltuGetParameterInvocation           [6]     CltuGetParameterInvocation
, cltuThrowEventInvocation             [8]     CltuThrowEventInvocation
, cltuTransferDataInvocation           [10]    CltuTransferDataInvocation
, cltuPeerAbortInvocation              [104]   SlePeerAbort
}


-- ============================================================
-- The second part of the module definition contains the types
```

```
-- used by the CLTU-PDUs declared in the first part.
-- ============================================================

CltuGetParameterInvocation     ::=    SEQUENCE
{ invokerCredentials                  Credentials
, invokeId                            InvokeId
, cltuParameter                       CltuParameterName
}

CltuStartInvocation            ::=    SEQUENCE
{ invokerCredentials                  Credentials
, invokeId                            InvokeId
, firstCltuIdentification             CltuIdentification
}

CltuThrowEventInvocation       ::=    SEQUENCE
{ invokerCredentials                  Credentials
, invokeId                            InvokeId
, eventInvocationIdentification       EventInvocationId
, eventIdentifier                     IntPosShort
, eventQualifier                      OCTET STRING (SIZE (1 .. 128))
}

CltuTransferDataInvocation     ::=    SEQUENCE
{ invokerCredentials                  Credentials
, invokeId                            InvokeId
, cltuIdentification                  CltuIdentification
, earliestTransmissionTime            ConditionalTime
, latestTransmissionTime              ConditionalTime
, delayTime                           Duration
, slduRadiationNotification           SlduStatusNotification
, cltuData                            CltuData
}

END
```

A2.7   CLTU OUTGOING PDUS

```
CCSDS-SLE-TRANSFER-SERVICE-CLTU-OUTGOING-PDUS

{ iso identified-organization (3) standards-producing-organization(112)
ccsds(4) space-link-extension(3) sle-transfer-services(1)
modules(1) forward-cltu-service(21) version- one-hundred-one(101)
asn1-outgoing-pdus(3) }

DEFINITIONS
IMPLICIT TAGS
::= BEGIN

IMPORTS      ConditionalTime
,                    Credentials
,                    InvokeId
,                    Time
FROM CCSDS-SLE-TRANSFER-SERVICE-COMMON-TYPES

                     SleAcknowledgement
,                    SleScheduleStatusReportReturn
FROM CCSDS-SLE-TRANSFER-SERVICE-COMMON-PDUS

                     SleBindReturn
,                    SleUnbindReturn
,                    SlePeerAbort
FROM CCSDS-SLE-TRANSFER-SERVICE-BIND-TYPES

                     BufferSize
,                    CltuGetParameter
,                    CltuIdentification
,                    CltuLastProcessed
,                    CltuLastOk
,                    CltuNotification
,                    CltuStatus
,                    DiagnosticCltuGetParameter
,                    DiagnosticCltuStart
,                    DiagnosticCltuThrowEvent
,                    DiagnosticCltuTransferData
,                    EventInvocationId
,                    NumberOfCltusProcessed
,                    NumberOfCltusRadiated
,                    NumberOfCltusReceived
,                    ProductionStatus
,                    UplinkStatus
FROM CCSDS-SLE-TRANSFER-SERVICE-CLTU-STRUCTURES
;

-- ===============================================================
-- The first part of the module definition contains the CLTU type
-- that contains all the possible PDUs the provider may send.
-- ===============================================================

CltuProviderToUserPdu          ::=    CHOICE
{ cltuBindReturn                      [101] SleBindReturn
, cltuUnbindReturn                    [103] SleUnbindReturn
, cltuStartReturn                     [1]   CltuStartReturn
, cltuStopReturn                      [3]   SleAcknowledgement
```

```
, cltuScheduleStatusReportReturn         [5]
   SleScheduleStatusReportReturn
, cltuGetParameterReturn                 [7]    CltuGetParameterReturn
, cltuThrowEventReturn                   [9]    CltuThrowEventReturn
, cltuTransferDataReturn                 [11]   CltuTransferDataReturn
, cltuAsyncNotifyInvocation              [12]   CltuAsyncNotifyInvocation
, cltuStatusReportInvocation             [13]   CltuStatusReportInvocation
, cltuPeerAbortInvocation                [104]  SlePeerAbort

}


-- =============================================================
-- The second part of the module definition contains the types
-- used by the CLTU-PDUs declared in the first part.
-- =============================================================

CltuAsyncNotifyInvocation      ::=    SEQUENCE
{ invokerCredentials                   Credentials
, cltuNotification                     CltuNotification
, cltuLastProcessed                    CltuLastProcessed
, cltuLastOk                           CltuLastOk
, productionStatus                     ProductionStatus
, uplinkStatus                         UplinkStatus
}

CltuGetParameterReturn         ::=    SEQUENCE
{ performerCredentials                 Credentials
, invokeId                             InvokeId
, result                               CHOICE
   { positiveResult                    [0]   CltuGetParameter
   , negativeResult                    [1]   DiagnosticCltuGetParameter
   }
}

CltuStartReturn            ::=    SEQUENCE
{ performerCredentials                 Credentials
, invokeId                             InvokeId
, result                               CHOICE
{ positiveResult                       [0]   SEQUENCE
     { startRadiationTime      Time
     , stopRadiationTime       ConditionalTime
     }
   , negativeResult                    [1]   DiagnosticCltuStart
   }
}

CltuStatusReportInvocation     ::=    SEQUENCE
{ invokerCredentials                   Credentials
, cltuLastProcessed                    CltuLastProcessed
, cltuLastOk                           CltuLastOk
, cltuProductionStatus                 ProductionStatus
, uplinkStatus                         UplinkStatus
, numberOfCltusReceived                NumberOfCltusReceived
, numberOfCltusProcessed               NumberOfCltusProcessed
, numberOfCltusRadiated                NumberOfCltusRadiated
, cltuBufferAvailable                  BufferSize
}

CltuThrowEventReturn           ::=    SEQUENCE
{ performerCredentials                 Credentials
```

```
, invokeId                         InvokeId
, eventInvocationIdentification    EventInvocationId
, result                               CHOICE
  {  positiveResult                    [0]   NULL
     , negativeResult                  [1]   DiagnosticCltuThrowEvent
  }
}

CltuTransferDataReturn      ::=    SEQUENCE
{ performerCredentials             Credentials
, invokeId                         InvokeId
, cltuIdentification               CltuIdentification
, cltuBufferAvailable              BufferSize
, result                               CHOICE
{ positiveResult                       [0]   NULL
, negativeResult                       [1]   DiagnosticCltuTransferData
}
}

END
```

ANNEX B

PRODUCTION STATUS

(NORMATIVE)

This annex describes the states and transitions of the forward physical channel, and presents in tabular form the effect of production status on the processing of invocations of the Enhanced Forward CLTU transfer service.

The possible transitions of production status of the Enhanced Forward CLTU service are shown in figure B-1 below.
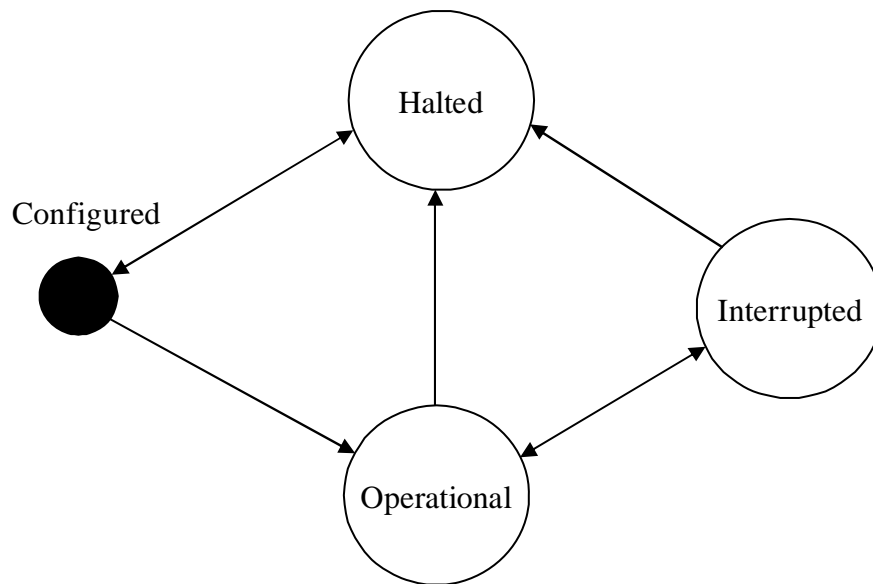

Figure B-1: CLTU Production Status Transitions

Changes in production status occur for reasons outlined in table B-1.

Table B-1: Production Status Changes and Notifications

Start Status	End Status	Cause of Status Change	Notification
Configured	Operational	Management action to make the production process operational; typically includes: completion of Uplink Sweep, and radiation of the Acquisition Sequence. Optionally, bit-lock and/or rf-available may be required before `production-status` can change to 'operational'. (See table 3-11 and NOTE 1.) If any error occurs that prevents the transition to production status 'operational' the production status reported will still stay 'configured'. If the status remains 'configured' for an extended period the user will have to check for the reasons by means outside the scope of this document.	'production operational'
Operational	Interrupted	Occurrence of a production fault detected by the provider.	'production interrupted' (but see NOTEs 2 and 3)
Interrupted	Operational	Maintenance action typically is required to correct a production fault. The CLTU `production-status` returns to 'operational' when the provider detects that the fault is corrected.	'production operational'
[Any]	Halted	Direct management action is required, such as an operator directive causing the provider to halt production.	'production halted'
Halted	Configured	Direct management action is required, such as an operator directive restoring the desired configuration and setting the `production-status` to 'configured'.	none

NOTES

1   The initial `production-status` value is 'configured'. When requested by the supported agency (either by schedule or via voice communication), the supporting agency starts CMM-1 by turning on the uplink and performing the nominal uplink sweep in accordance with the mission specific parameters. The `production-status` changes from 'configured' to 'operational':

a) at completion of the uplink sweep, if `plop-in-effect` is 'PLOP-1' and `rf-available-required` is set to 'no';

b) at completion of the uplink sweep, if `plop-in-effect` is 'PLOP-1', `rf-available-required` is set to 'yes' and the CLCW indicates 'RF available';

c) at completion of the radiation of the acquisition sequence, if `plop-in-effect` is 'PLOP-2', `rf-available-required` is set to 'no' and `bit-lock-required` is set to 'no';

d) at completion of the radiation of the acquisition sequence, if `plop-in-effect` is 'PLOP-2', `rf-available-required` is set to 'yes', the CLCW indicates 'RF available', and `bit-lock-required` is set to 'no';

e) at completion of the radiation of the acquisition sequence, if `plop-in-effect` is 'PLOP-2', `rf-available-required` is set to 'no', `bit-lock-required` is set to 'yes', and the CLCW flags 'bit lock';

f) at completion of the radiation of the acquisition sequence, if `plop-in-effect` is 'PLOP-2', `rf-available-required` is set to 'yes', `bit-lock-required` is set to 'yes', and the CLCW indicates both 'RF available' and 'bit lock';

g) at completion of the uplink sweep, if `plop-in-effect` is 'ASCP'.

2   The `production-status` changes to 'interrupted' when the checking of `uplink-status` is required (`rf-available-required` and/or `bit-lock required` is set to 'yes') and `uplink-status` has no longer the required value.

3   This Specification does not require that a production interruption be notified immediately to the user (see 3.8.2.3). If a fault affecting production can be corrected before the provider attempts to process a CLTU, the 'production interrupted' notification need not be sent. It is an implementation choice whether always to send the 'production interrupted' notification immediately on occurrence of a production fault or to wait until a CLTU is ready to be radiated before sending the notification. The implementation choice is documented in the service configuration via the `notification-mode` parameter, which may have a value of 'immediate' or 'deferred', corresponding to immediate or deferred notification in event of a production interruption.

The effect of `production-status` on the processing of Enhanced Forward CLTU transfer service invocations is specified in 3.2.10 through 3.13 and in table 4-1. The effects are summarized in table B-2 for convenient reference.

Table B-2: Effect of Production Status on Operations

Production Status	Operation	Effect	Diagnostic
Halted	CLTU-BIND	Rejected	'out of service'
	CLTU-START	Rejected	'out of service'
	CLTU-TRANSFER-DATA	Rejected	'unable to process'
	Other operations	None specified	N/A
Configured	All operations	None specified	N/A
Operational	All operations	None specified	N/A
Interrupted	CLTU-START	Rejected	'unable to comply'
	CLTU-TRANSFER-DATA	Rejected (but see NOTE)	'unable to process'
	Other operations	None specified	N/A

NOTE   –   When the value of the `notification-mode` parameter is 'deferred', CLTU-TRANSFER-DATA invocations are rejected only after the provider has notified the user that a production interruption has occurred. (See 3.8.2.3.)

ANNEX C

CONFORMANCE OPTIONS MATRIX

(NORMATIVE)

This annex provides the Conformance Matrix for implementations of SLE Enhanced Forward CLTU Service systems. An SLE Enhanced Forward CLTU Service system will be considered to be 'conformant' if the mandatory elements identified in the matrix are implemented as described in this Experimental Specification.

Table C-1: Conformance Matrix for CLTU Service (Operations)

CLTU Operations	Optional / Mandatory

CLTU-BIND
CLTU-UNBIND
	These operations are mandatory. All parameters are also mandatory with the exception of:
	
	`invoker-credentials`
	
	`performer-credentials`
	
	Support of the `version-number` negotiation as part of the CLTU-BIND operation is optional.

CLTU-START
CLTU-STOP
CLTU-TRANSFER-DATA
CLTU-ASYNC-NOTIFY
CLTU-SCHEDULE-STATUS-REPORT
CLTU-STATUS-REPORT
CLTU-GET-PARAMETER
	These operations are mandatory. All parameters are also mandatory with the exception of:
	
	`invoker-credentials`
	
	`performer-credential`
	
	`stop-production-time`

CLTU-THROW-EVENT	This operation is optional.
CLTU-PEER-ABORT	This operation is mandatory. The `diagnostic` parameter is also mandatory.

Table C-2: Conformance Matrix for CLTU Service (Other Requirements)

Other Sections	Optional / Mandatory
Protocol Abort Mode Parameter (see table 3-1 and 4.1.5.5)	The parameter is optional. The default behavior of the provider in event of protocol abort shall be as described for the 'flush' value of the `protocol-abort-mode` parameter.
`notification-mode` Parameter (see table 3-1 and 3.8.2.3)	This parameter is optional; its purpose is only to document the behavior of the provider.

Other Sections	Optional / Mandatory
Provider Buffering Requirements (see 3.2.10)	Mandatory minimum size is 4,194,304 octets. NOTE – This number is equivalent to 1024 CLTUs each of length of 4096 octets.
State transition table (see table 4-1)	Mandatory.
ASN.1 Protocol Specification (annex A)	All data types are mandatory. Implementations may vary as described in the annex.
AOS space link protocol support (see 3.7.2.5.3, 3.7.2.5.5, 3.7.2.8.2c), 3.7.2.10.2, 3.7.2.10.3, 3.8.2.3b)2), 3.8.3.3, and table 3-11).	Support for AOS space link protocol (i.e., ASCP) is mandatory for implementations that do not support Telecommand space link protocols (i.e., PLOP-1 and/or PLOP-2). Support for AOS space link protocol is optional for implementations that support Telecommand space link protocol.
Telecommand space link protocol support (see 3.7.2.5.2, 3.7.2.5.4, 3.7.2.8.2a), 3.7.2.8.2b), 3.7.2.10.1, 3.8.2.3b)1), 3.8.3.2, and table 3-11).	Support for Telecommand space link protocol (i.e., PLOP-1 and PLOP-2) is mandatory for implementations that do not support AOS space link protocols (i.e., ASCP). Support for Telecommand space link protocol is optional for implementations that support AOS space link protocol.

ANNEX D

AOS FORWARD SPACE LINK SYNCHRONIZATION AND CHANNEL CODING PRODUCTION SPECIFICATION

(NORMATIVE)

D1   INTRODUCTION

As of the writing of this Orange Book, recommended standards for synchronization and channel coding for space links in which the data units being transferred have uniform length and are transmitted back-to-back are currently formally specified by CCSDS for return (a.k.a. telemetry) space links only (see reference [10]).

For the Enhanced Forward CLTU service, these return link ('telemetry') space link synchronization and channel coding functions must be performed for forward link data. These space link synchronization and channel coding functions are performed by the EFCLTU provider as part of service production. For the purposes of this Experimental Specification, when applied to AOS forward space links, telemetry space link synchronization and channel coding functions are referred to as *AOS forward space link synchronization and channel coding functions*.

The AOS forward space link synchronization and channel coding production functions are specified in the context of the model illustrated in figure D-1. These AOS forward space link synchronization and channel coding functions are: Block Forward Error Correction (FEC) Encoding, Randomization, Sync Marker Attachment, and Convolutional Encoding. In addition, the production processing includes the SL-PDU Extraction and Idle Data Insertion function. Collectively, these AOS forward space link synchronization and channel coding functions and the SL-PDU Extraction and Idle Data Insertion function are referred to as ASCP. The ASCP production model establishes the framework for describing the relationships among the various service provision and production functions. The model itself is not normative; an implementation of EFCLTU service production is only required to produce the same resulting symbol stream going into the forward link physical channel.

When ASCP is configured to support AOS transfer frames, the combination of the Block FEC Encoding, Randomization, Sync Marker Attachment, and Convolutional Encoding functions provides the service as specified in annex A of the TM Synchronization and Channel Coding Blue Book (reference [10]). However, when ASCP is configured to transfer AOS CADUs, the Randomization and Sync Marker Attachment are bypassed, providing only the optional Convolutional Encoding function.

NOTE – The bypass mode used when CADUs are transferred by the Enhanced FCLTU service is not currently supported by CCSDS Space Link Recommended Standards (see references [9] and [10]). The bypassing of these functions is strictly local to this Experimental Specification and not intended to replace or redefine the Space Link Recommended Standards. However, if CADU multiplexing is to be included in a future real CCSDS Recommended Standard, modifications to the Space Link Recommended Standards may be necessary.

Figure D-1 shows the CLTU Buffer and CLTU release functions of EFCLTU Service Provision. The CLTU Buffer holds each SLE FTDU until it satisfies its release criteria or it expires and is discarded, as specified in the main text of this Experimental Specification. SLE FTDUs that are successfully released to ASCP production are processed as specified in the subsections of this annex.
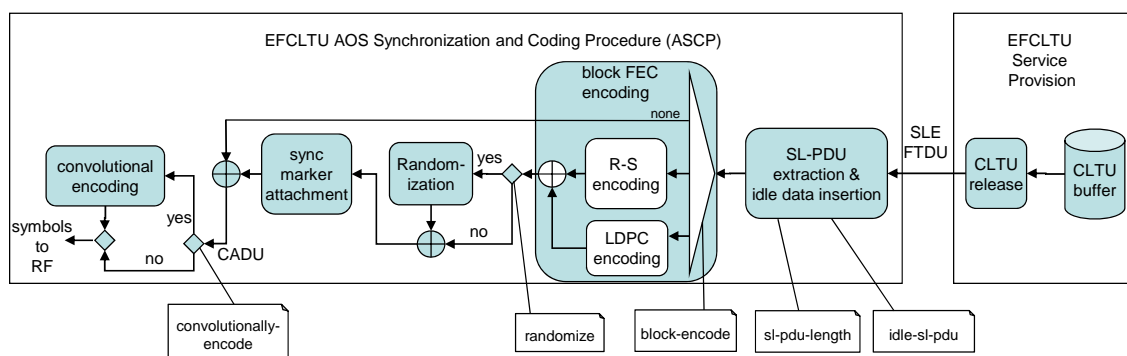

Figure D-1: EFCLTU ASCP Production Model

D2   SL-PDU EXTRACTION AND IDLE DATA INSERTION

D2.1   The SL-PDU Extraction and Idle Data Insertion function determines when the Block Forward Error Correction (FEC) Encoding function is ready to process an SL-PDU. In accordance with the AOS Space Data Link Protocol (reference [9]) requirement to release Idle Transfer Frames when a user-data-bearing Transfer Frames are not available, if there is not an SLE CLTU available from EFCLTU Service Provision, the SL-PDU Extraction and Idle Data Insertion function supplies an idle SL-PDU to the Block FEC Encoding function. If an SLE-FTDU is available, the SL-PDU Extraction and Idle Data Insertion function extracts the SL-PDU(s) from the SLE-FTDU and supplies it (them) to the Block FEC Encoding function.

NOTES

1   The SLE FTDUs are assumed to contain one or more SL-PDUs, all of the same fixed length, `sl-pdu-length` (see 3.7.2.13.1i)). The idle SL-PDU must be configured to be of that same length.

2   The EFCLTU transfer service rejects any SLE-FTDU with a length that in not an integral multiple of `sl-pdu-length`, ensuring that only full SL-PDUs are extracted from the SLE-FTDUs.

3   The SL-PDU Extraction and Idle Data Insertion function must be able to determine when the Block FEC Encoding function is ready to process the next SL-PDU. The method by which the SL-PDU Extraction and Idle Data Insertion function determines that the Block FEC Encoding function is ready to process an SL-PDU is implementation-specific and outside the scope of this Experimental Specification.

D2.2   When the Block FEC Encoding function is ready to receive an SL-PDU, if an SLE-FTDU is available from the CLTU release function of EFCLTU Service Provision:

a) then each of N SL-PDUs shall be extracted from the SLE-FTDU and supplied to the Block FEC Encoding function, where N is the ratio of the length of the SLE-FTDU to the value of the `sl-pdu-length` managed parameter;

b) otherwise an idle SL-PDU shall be supplied to the Block FEC Encoding function.

NOTES

1   Determination of availability of an SLE-FTDU for transmission is the responsibility of the transfer service instance that supplies the SLE-FTDUs to service production. For example, the Enhanced Forward CLTU service will not release an SLE-FTDU until the earliest-radiation-time for that SLE-FTDU is reached. Also, the Enhanced Forward CLTU service will reject or discard any SLE-FTDU if it cannot be transmitted prior to its latest-radiation-time, removing that SLE-FTDU from availability for transmission.

2   When the Enhanced Forward CLTU service instance is used to transfer AOS transfer frames, the `idle-sl-pdu` managed parameter should be formatted as an AOS Idle Transfer Frame as specified in reference [9] in order to ensure proper execution of the AOS space data link protocol. However, the Enhanced Forward CLTU service does not enforce this formatting.

3   When the Enhanced Forward CLTU service instance is used to transfer CADUs, the `idle-sl-pdu` managed parameter should be formatted as an AOS Idle Transfer Frame as specified in reference [9] with the same synchronization and channel coding functions as applied to the AOS transfer frames in the user-generated CADUs, in order to ensure proper execution of the AOS space data link protocol. However, the Enhanced Forward CLTU service does not enforce this formatting.

4   The ability to generate idle CADUs and multiplex then with user-generated CADUs is not currently supported by CCSDS Space Link Recommended Standards (see references [9] and [10]). The multiplexing of CADUs is strictly local to this Experimental Specification and not intended to replace or redefine the Space Link Recommended Standards. However, if CADU multiplexing is to be included in a future real CCSDS Recommended Standard, modifications to the Space Link Recommended Standards may be necessary.

D2.3   If multiple SL-PDUs are contained within the SLE-FTDU, the SL-PDUs shall be supplied to the Block FEC Encoding function in the order in which the SL-PDUs appear in the SLE-FTDU.

D2.4   D2.2 and D2.3 shall be repeated as long as production processing is operational.

D3   BLOCK FORWARD ERROR CORRECTION ENCODING

D3.1   If the SL-PDUs are transfer frames that are to be Reed-Solomon-encoded by the service provider (as indicated by the 'rs' value of the `block-encode` managed parameter), each transfer frame shall be encoded into an RS codeblock, in accordance with reference [10]).

D3.2   If the SL-PDUs are transfer frames that are to be LDPC-encoded by the service provider (as indicated by the 'ldpc' value of the `block-encode` managed parameter), each transfer frame shall be encoded into an LDPC codeblock, in accordance with reference [10]).

D3.3   If the SL-PDUs are already-formed CADUs, no block forward error correction coding is to be applied by the service provider (as indicated by the 'none' value of the `block-encode` managed parameter).

NOTE – The setting of other configuration parameters associated with R-S and LDPC encoding as specified in reference [10] (e.g., the interleave depth for R-S) must be configured via service management but is otherwise outside the scope of the Enhanced Forward CLTU service.

D4   RANDOMIZATION

If the codeblocks are to be pseudo-randomized by the service provider (as indicated by the `randomize` managed parameter), the provider shall randomize each codeblock as specified in reference [10].

NOTE – If the SL-PDUs received via the EFCLTU service instance are already fully formed CADUs, then the Randomization function is not performed and the `randomize` managed parameter is not applicable.

D5   SYNC MARKER ATTACHMENT

The provider shall attach a sync marker to each (randomized) codeblock as specified in reference [10].

NOTES

1   As specified in reference [10], the kind of sync marker to be attached depends on the block encoding option selected (i.e., the value of the `block-encode` managed parameter).

2   If the SL-PDUs received via the EFCLTU service instance are already fully formed CADUs, then the Synch Marker Attachment function is not performed and the `attach-sync-marker` managed parameter is not applicable.

D6   CONVOLUTIONALLY ENCODE

If the CADUs are to be convolutionally encoded by the service provider (as indicated by the `convolutionally-encode` managed parameter's having a value of 'yes'), the bitstream resulting from the application of ASCP shall be convolutionally encoded in accordance with reference [10]).

NOTE –   The setting of other configuration parameters associated with convolutional coding as specified in reference [10] must be configured via service management but is otherwise outside scope of the Enhanced Forward CLTU service.

D7   SUMMARY OF MANAGED PARAMETER CONFIGURATIONS

Table D-1 summarizes the proposed possible configurations of managed parameters for AOS forward link production processing for the EFCLTU SLE experimental service. Only those combinations of SLE CLTU data content and processing that appear in the table are required to be supported.

Table D-1: Configurations of EFCLTU Managed Parameters

SLE CLTU content	Required Processing	`idle-sl-pdu`	`sl-pdu-length`	`block-encode`	`random-ize`	`convolutionally-encode`
Single transfer frame/ block of transfer frames	R-S encode, optionally randomize, ASM mandatory, optionally conv-encode	Block of octets of length of transfer frame	Length of transfer frame	'rs'	yes/no	yes/no
	LDPC encode, optionally randomize, ASM mandatory, no convolutional-encoding			'ldpc'		no
Single CADU/ block of CADUs	No block encode, no randomization, no ASM, optionally convolutionally encode	Block of octets of length of CADU	Length of CADU	'none'	N/A	yes/no

The first column of the table specifies the content of the SLE-FTDU as it is transferred via EFCLTU service provision.

NOTE – The difference in processing of a single SL-PDU (e.g., transfer frame) and a block of that SL-PDU is that the SL-PDU Extraction and Idle Data Insertion function extracts multiple SL-PDUs of length `sl-pdu-length` from the block of SL-PDUs. The size of the SLE-FTDU is required to be an integral multiple of `sl-pdu-length`).

The second column identifies the processing to be performed for this particular combination. If a processing step is optional for a given SLE-FTDU content type, it is so indicated. If a processing step is not to be applied for a given SLE-FTDU content type, it is so indicated.

The `idle-sl-pdu`, `sl-pdu-length`, `block-encode`, `randomize`, and `convolutionally-encode` columns specify the values of those managed configuration parameters to perform the required processing. If `block-encode` is either 'rs' or 'ldpc', the sync marker attachment is always performed. If `block-encode` is 'none', then block encoding, randomization, and synch marker attachment are not performed and the `randomize` configuration parameter is not applicable.

ANNEX E

SECURITY, SANA, AND PATENT CONSIDERATIONS

(INFORMATIVE)

E1 SECURITY CONSIDERATIONS

Security considerations are discussed in 2.8.

E2 SANA CONSIDERATION

The recommendations of this document do not require any action from SANA.

E3 PATENT CONSIDERATIONS

There are no known patent issues related to this Experimental Specification.

ANNEX F

INDEX TO DEFINITIONS

(INFORMATIVE)

This annex provides an index to the terminology defined in the references.

Term	Reference
(data) type	reference [6]
(data) value	reference [6]
abstract binding	reference [1]
abstract object	reference [1]
abstract port	reference [1]
abstract service	reference [1]
abstract syntax	reference [7]
Abstract Syntax Notation One (ASN.1)	reference [6]
acquisition sequence	reference [2]
application entity	reference [7]
Application layer	reference [7]
association	subsection 1.6.8
Carrier Modulation Mode (CMM)	reference [2]
channel access data unit (CADU)	reference [10]
CLTU (CLTU)	subsection 1.6.3
CLTU channel	reference [1]
Communications Link Control Word (CLCW)	reference [3]
Communications Link Transmission Unit (CLTU)	reference [2]
communications service	subsection 1.6.8
confirmed operation	subsection 1.6.8
flow control	reference [7]
Forward CLTU service	reference [1]
idle sequence	reference [2]
initiator	subsection 1.6.8
invocation	subsection 1.6.8
invoker	reference [1]
Mission Data Operation System (MDOS)	reference [1]
Mission User Entity (MUE)	reference [1]
object identifier	reference [6]

Term	**Reference**
offline delivery mode	reference [1]
online delivery mode	reference [1]
Open System Interconnection (OSI)	reference [7]
operation	reference [1]
parameter	subsection 1.6.8
performance	subsection 1.6.8
performer	reference [1]
physical channel	reference [1]
Physical Layer Operations Procedure (PLOP)	reference [2]
port identifier	subsection 1.6.8
real system	reference [7]
responder	subsection 1.6.8
return	subsection 1.6.8
Service Access Point (SAP)	reference [7]
service instance provision period	subsection 1.6.8
service provider (provider)	reference [1]
service user (user)	reference [1]
SLE Complex	reference [1]
SLE Complex Management	reference [1]
SLE data channel	reference [1]
SLE functional group (SLE-FG)	reference [1]
SLE-FTDU	subsection 1.6.8
SLE protocol data unit (SLE-PDU)	reference [1]
SLE service data unit (SLE-SDU)	reference [1]
SLE service package	reference [1]
SLE transfer service instance	reference [1]
SLE transfer service production	reference [1]
SLE transfer service provision	reference [1]
SLE Utilization Management	reference [1]
space link	reference [1]
space link data channel	reference [1]
space link data unit (SL-DU)	reference [1]
space link protocol data unit (SL-PDU)	reference [1]
space link session	reference [1]
unconfirmed operation	subsection 1.6.8

ANNEX G

ACRONYMS

(INFORMATIVE)

This annex expands the acronyms used throughout this Experimental Specification.

ASCP	AOS Synchronization and Coding Procedure
ASN.1	Abstract Syntax Notation One
C	Conditional
CCSDS	Consultative Committee for Space Data Systems
CDS	CCSDS Day Segmented (time code)
CLCW	Communications Link Control Word
CLTU	Communications Link Transmission Unit
CLTU-	Enhanced Forward CLTU service (operation prefix)
CMM	Carrier Modulation Mode
COP	Communications Operation Procedure
FG	Functional Group
FWD	Forward
IEC	International Electrotechnical Commission
IP	Internet Protocol
ISO	International Organization for Standardization
M	Mandatory
MDOS	Mission Data Operation System
M-P	Management Provider Service Port
M-U	Management User Service Port
MUE	Mission User Entity

OCF	Operational Control Field
OSI	Open System Interconnection
PDU	Protocol Data Unit
PLOP	Physical Layer Operations Procedure
RF	Radio Frequency
SANA	Space Assigned Numbers Authority
SAP	Service Access Point
SDU	Service Data Unit
SL-DU	Space Link Data Unit
SLE	Space Link Extension
SLE-FTDU	SLE Forward Transfer Data Unit (EFCLTU SDU)
SLE-FG	SLE Functional Group
SLE-PDU	SLE Protocol Data Unit
SLE-SDU	SLE Service Data Unit
SL-PDU	Space Link Protocol Data Unit
TC	Telecommand
TCM	Terminology, Conventions and Methodology
TCP	Transmission Control Protocol
T-P	Transfer Provider Service Port
T-U	Transfer User Service Port
UTC	Universal Time Coordinated
WAN	Wide Area Network

ANNEX H

THROW EVENT DEFINITIONS

(INFORMATIVE)

This annex provides several examples of `event-identification` and `event-qualifier` for the CLTU-THROW-EVENT operation. These examples are non-binding on implementations of the SLE Enhanced Forward CLTU Service system; however, they may serve as models for implementations when the matching functionality is supplied.

In the following table the event-qualifier Contents column defines the string passed in the `event-qualifier` parameter. Values such as 'on' and 'mi' are literal. Values such as <BR> represent an alphanumeric quantity. At least one white space must separate values in the `event-qualifier` string.

Table H-1: Throw Event Examples

event-identification		event-qualifier Contents [Range]	Event Description
Name	**ID**		
Set Mod	1	on|off	Enable or disable command modulation of the carrier. (See NOTE 1.)
Change Rate	2	br [7.8125 - 4000.0]	Change bit rate to new bit rate defined by (requested new bit rate is the value of in bits/second). (See NOTE 2.)
Change Index	3	mi <MI> [1 - 1570]	Change modulation index to new modulation index angle defined by <MI> (requested new modulation index is the value of <MI> in milli-radians). (See NOTE 2.)
Change Rate/Index	4	br mi <MI>	Change both bit rate and modulation index as described above. (See NOTE 2.)

NOTES

1   The following condition is required for the Set Mod event with the `event-qualifier` parameter value set to 'on': production status must be operational. If this condition is not met, CLTU-ASYNC-NOTIFY is invoked with the notification-type 'event condition evaluated to false'.

2   The following conditions are required for the Change Rate, Change Index, and Change Rate/Index events: no CLTUs may currently be radiating, and the CLTU buffer must be empty. If these conditions are not met, CLTU-ASYNC-NOTIFY is invoked with the notification-type 'event condition evaluated to false'.

ANNEX I

INFORMATIVE REFERENCES

(INFORMATIVE)

[I1] *Organization and Processes for the Consultative Committee for Space Data Systems*. CCSDS A02.1-Y-3. Yellow Book. Issue 3. Washington, D.C.: CCSDS, July 2011.

[I2] *Telecommand Summary of Concept and Rationale*. Report Concerning Space Data System Standards, CCSDS 200.0-G-6. Green Book. Issue 6. Washington, D.C.: CCSDS, January 1987.

[I3] *Cross Support Concept—Part 1: Space Link Extension Services*. Report Concerning Space Data System Standards, CCSDS 910.3-G-3. Green Book. Issue 3. Washington, D.C.: CCSDS, March 2006.

[I4] *Telecommand Part 1—Channel Service*. Recommendation for Space Data System Standards, CCSDS 201.0-B-3-S. Historical Recommendation. Issue 3-S. Washington, D.C.: CCSDS, (June 2000) August 2005.

[I5] *Telecommand Part 2—Data Routing Service*. Recommendation for Space Data System Standards, CCSDS 202.0-B-3-S. Historical Recommendation. Issue 3-S. Washington, D.C.: CCSDS, (June 2001) August 2005.

[I6] *Telecommand Part 2.1—Command Operation Procedures*. Recommendation for Space Data System Standards, CCSDS 202.1-B-2-S. Historical Recommendation. Issue 2-S. Washington, D.C.: CCSDS, (June 2001) August 2005.

[I7] *Telecommand Part 3—Data Management Service*. Recommendation for Space Data System Standards, CCSDS 203.0-B-2-S. Historical Recommendation. Issue 2-S. Washington, D.C.: CCSDS, (June 2001) August 2005.

[I8] *Space Link Extension—Internet Protocol for Transfer Services*. Recommendation for Space Data System Standards, CCSDS 913.1-B-1. Blue Book. Issue 1. Washington, D.C.: CCSDS, September 2008.

[I9] *Cross Support Transfer Service—Specification Framework*. Draft Recommendation for Space Data System Standards, CCSDS 921.1-R-1. Red Book. Issue 1. Washington, D.C.: CCSDS, August 2010.

[I10] *IOAG Service Catalog 1*. IOAG. March 2010.

[I11] *IOAG Service Catalog 2*. IOAG.T.SC2.2011.V1.0. IOAG. February 2011.