CCSDS

The Consultative Committee for Space Data Systems

CCSDS Historical Document

This document's Historical status indicates that it is no longer current. It has either been replaced by a newer issue or withdrawn because it was deemed obsolete. Current CCSDS publications are maintained at the following location:

http://public.ccsds.org/publications/

Consultative Committee for Space Data Systems

**RECOMMENDATION FOR SPACE
DATA SYSTEM STANDARDS**

SPACE LINK EXTENSION— FORWARD CLTU SERVICE SPECIFICATION

CCSDS 912.1-B-1

BLUE BOOK

April 2002

AUTHORITY

Issue:	Blue Book, Issue 1
Date:	April 2002
Location:	Oberpfaffenhofen, Germany

This document has been approved for publication by the Management Council of the Consultative Committee for Space Data Systems (CCSDS) and represents the consensus technical agreement of the participating CCSDS Member Agencies. The procedure for review and authorization of CCSDS Recommendations is detailed in reference [E1], and the record of Agency participation in the authorization of this document can be obtained from the CCSDS Secretariat at the address below.

This Recommendation is published and maintained by:

CCSDS Secretariat
Program Integration Division (Code M-3)
National Aeronautics and Space Administration
Washington, DC 20546, USA

STATEMENT OF INTENT

The Consultative Committee for Space Data Systems (CCSDS) is an organization officially established by the management of member space Agencies. The Committee meets periodically to address data systems problems that are common to all participants, and to formulate sound technical solutions to these problems. Inasmuch as participation in the CCSDS is completely voluntary, the results of Committee actions are termed **Recommendations** and are not considered binding on any Agency.

This **Recommendation** is issued by, and represents the consensus of, the CCSDS Plenary body. Agency endorsement of this **Recommendation** is entirely voluntary. Endorsement, however, indicates the following understandings:

o   Whenever an Agency establishes a CCSDS-related **standard**, this **standard** will be in accord with the relevant **Recommendation**. Establishing such a **standard** does not preclude other provisions which an Agency may develop.

o   Whenever an Agency establishes a CCSDS-related **standard**, the Agency will provide other CCSDS member Agencies with the following information:

--   The **standard** itself.

--   The anticipated date of initial operational capability.

--   The anticipated duration of operational service.

o   Specific service arrangements shall be made via memoranda of agreement. Neither this **Recommendation** nor any ensuing **standard** is a substitute for a memorandum of agreement.

No later than five years from its date of issuance, this **Recommendation** will be reviewed by the CCSDS to determine whether it should: (1) remain in effect without change; (2) be changed to reflect the impact of new technologies, new requirements, or new directions; or, (3) be retired or canceled.

In those instances when a new version of a **Recommendation** is issued, existing CCSDS-related Agency standards and implementations are not negated or deemed to be non-CCSDS compatible. It is the responsibility of each Agency to determine when such standards or implementations are to be modified. Each Agency is, however, strongly encouraged to direct planning for its new standards and implementations towards the later version of the Recommendation.

FOREWORD

This document is a technical **Recommendation** for use in developing ground systems for space missions and has been prepared by the **Consultative Committee for Space Data Systems** (CCSDS). The Space Link Extension (SLE) Forward Command Link Transmission Unit (CLTU) Service described herein is intended for missions that are cross supported between Agencies of the CCSDS.

This **Recommendation** specifies a data service that extends certain of the space-to-ground communications services previously defined by CCSDS (references [2] and [3]) within the framework established by the CCSDS SLE Reference Model (reference [1]). It allows implementing organizations within each Agency to proceed with the development of compatible derived Standards for the ground systems that are within their cognizance. Derived Agency Standards may implement only a subset of the optional features allowed by the **Recommendation** and may incorporate features not addressed by the **Recommendation**.

Through the process of normal evolution, it is expected that expansion, deletion or modification to this document may occur. This **Recommendation** is therefore subject to CCSDS document management and change control procedures, as defined in reference [E1]. 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.

At time of publication, the active Member and Observer Agencies of the CCSDS were:

Member Agencies

–   Agenzia Spaziale Italiana (ASI)/Italy.
–   British National Space Centre (BNSC)/United Kingdom.
–   Canadian Space Agency (CSA)/Canada.
–   Centre National d'Etudes Spatiales (CNES)/France.
–   Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR)/Germany.
–   European Space Agency (ESA)/Europe.
–   Instituto Nacional de Pesquisas Espaciais (INPE)/Brazil.
–   National Aeronautics and Space Administration (NASA)/USA.
–   National Space Development Agency of Japan (NASDA)/Japan.
–   Russian Space Agency (RSA)/Russian Federation.

Observer Agencies

–   Austrian Space Agency (ASA)/Austria.
–   Central Research Institute of Machine Building (TsNIIMash)/Russian Federation.
–   Centro Tecnico Aeroespacial (CTA)/Brazil.
–   Chinese Academy of Space Technology (CAST)/China.
–   Commonwealth Scientific and Industrial Research Organization (CSIRO)/Australia.
–   Communications Research Centre (CRC)/Canada.
–   Communications Research Laboratory (CRL)/Japan.
–   Danish Space Research Institute (DSRI)/Denmark.
–   European Organization for the Exploitation of Meteorological Satellites (EUMETSAT)/Europe.
–   European Telecommunications Satellite Organization (EUTELSAT)/Europe.
–   Federal Service of Scientific, Technical & Cultural Affairs (FSST&CA)/Belgium.
–   Hellenic National Space Committee (HNSC)/Greece.
–   Indian Space Research Organization (ISRO)/India.
–   Institute of Space and Astronautical Science (ISAS)/Japan.
–   Institute of Space Research (IKI)/Russian Federation.
–   KFKI Research Institute for Particle & Nuclear Physics (KFKI)/Hungary.
–   MIKOMTEK: CSIR (CSIR)/Republic of South Africa.
–   Korea Aerospace Research Institute (KARI)/Korea.
–   Ministry of Communications (MOC)/Israel.
–   National Oceanic & Atmospheric Administration (NOAA)/USA.
–   National Space Program Office (NSPO)/Taipei.
–   Space and Upper Atmosphere Research Commission (SUPARCO)/Pakistan.
–   Swedish Space Corporation (SSC)/Sweden.
–   United States Geological Survey (USGS)/USA.

DOCUMENT CONTROL

Document	Title	Date	Status and Substantive Changes
CCSDS 912.1-B-1	Space Link Extension—Forward CLTU Service Specification	April 2002	Original Issue

CONTENTS

CONTENTS (continued)

CONTENTS (continued)

1 INTRODUCTION

1.1 PURPOSE OF THIS RECOMMENDATION

This Recommendation defines the Command Link Transmission Unit (CLTU) service in conformance with the transfer services specified in reference [1], *Cross Support Reference Model—Part 1: SLE Services*. The Forward CLTU service is a Space Link Extension (SLE) transfer service that enables a mission to send Command Link Transmission Units (CLTUs) to a spacecraft.

1.2 SCOPE

1.2.1 This Recommendation defines, in an abstract manner, the 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.

1.2.2 It 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 the spacecraft for extraction of the Operational Control Field;

d) the methods or technologies required for communications;

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

1.3 APPLICABILITY

1.3.1 APPLICABILITY OF THIS RECOMMENDATION

This Recommendation provides a basis for the development of real systems that implement the Forward CLTU service. Implementation of the Forward CLTU service in a real system additionally requires the availability of a communications service to convey invocations and returns of Forward CLTU service operations between service users and providers. This

Recommendation 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 the application layer in the event of congestion; and

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

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

1.3.2   LIMITS OF APPLICABILITY

This Recommendation specifies the 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 Recommendation 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 Recommendation is organized as follows:

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

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

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

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

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

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

g) annex C lists all acronyms used within this document;

h) annex D provides a conformance matrix that defines what capabilities must be provided for an implementation to be considered compliant with this Recommendation;

i) annex E contains a list of informative references;

j) annex F contains examples of usage of the CLTU-THROW-EVENT operation;

k) annex G explains the relationship of the Forward CLTU service to the status of the forward space link channel.

1.5.2 SLE SERVICES DOCUMENTATION TREE

This Recommendation is based on the architectural model for cross support defined in the *SLE Reference Model* (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 Forward CLTU transfer service.

This Recommendation is part of a 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.
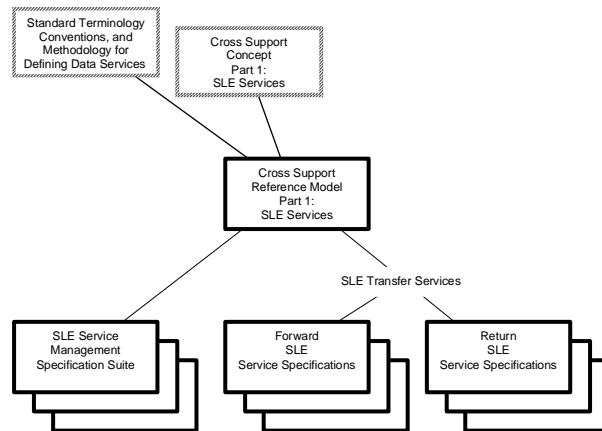
Figure 1-1: Space Link Extension (SLE) Services Documentation

a) *Standard Terminology, Conventions and Methodology (TCM) for Defining Data Services* (reference [E3]): a Report identifying selected international standards relevant for the definition of data services;

b) *Cross Support Concept – Part 1: Space Link Extension Services* (reference [E4]): a Report introducing the concepts of cross support and SLE services;

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

d) *SLE Service Management Specification Suite;* a set of Recommendations that establish the basis for SLE service management (reference [E5] is the primary Recommendation in that set);

e) *Forward SLE Transfer Service Specifications;* a set of Recommendations that will provide specification of all forward link SLE transfer services (this Recommendation is one of the specifications in that set);

f) *Return SLE Transfer Service Specifications;* a set of Recommendations that will provide specification of all return link SLE transfer services.

1.6 DEFINITIONS, NOMENCLATURE, AND CONVENTIONS

1.6.1 DEFINITIONS

1.6.1.1 Definitions from OSI Basic Reference Model

This Recommendation makes use of a number of terms defined in reference [7]. The use of those terms in this Recommendation shall 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).

1.6.1.2   Definitions from OSI Abstract Syntax Notation One

This Recommendation 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 Recommendation, ASN.1 is used for specifying the abstract syntax of the invocations and returns of the operations of the Forward CLTU service. The use of ASN.1 as a descriptive language is intended to support the specification of the abstract 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 Forward CLTU service.

1.6.1.3   Definitions from OSI Directory – Part 2: Models

This Recommendation makes use of the following terms defined in reference [8]:

a) distinguished name (DN);

b) relative distinguished name (RDN).

1.6.1.4   Definitions from CCSDS Telecommand Channel Service

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

   a) acquisition sequence;

   b) command link transmission unit (CLTU);

   c) carrier modulation mode (CMM);

   d) idle sequence;

   e) physical layer operations procedure (PLOP).

1.6.1.5   Definitions from CCSDS Telecommand Data Routing Service

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

command link control word (CLCW).

1.6.1.6   Definitions from CCSDS SLE Reference Model

This Recommendation 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 System;

z) SLE transfer service instance;

aa) SLE transfer service production;

bb) SLE transfer service provision;

cc) SLE Utilization Management;

dd) space link;

ee) space link data channel;

ff) space link data unit (SL-DU);

gg) space link session.

1.6.1.7   Additional Definitions

1.6.1.7.1   General

For the purposes of this Recommendation, the following definitions also apply.

1.6.1.7.2   Association

An association is 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.

1.6.1.7.3   Communications Service

A communications service is 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 Recommendation ensures, at least in principle, that it is possible to construct such a gateway.

1.6.1.7.4   Confirmed Operation

A confirmed operation is an operation that requires the performer to return a report of its outcome to the invoker.

1.6.1.7.5   Initiator

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

1.6.1.7.6   Invocation

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

1.6.1.7.7   Parameter

A parameter of an operation is 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.

1.6.1.7.8   Performance

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

1.6.1.7.9   Port Identifier

A port identifier identifies a source or a destination in a communications system.

NOTE –   See 2.6.4.6 for more information.

1.6.1.7.10 Responder

The responder is 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.

1.6.1.7.11 Return

The return of an operation is a report, from the performer to the invoker, of the outcome of the performance of the operation.

1.6.1.7.12 Service Instance Provision Period

The service instance provision period is 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.

1.6.1.7.13 Unconfirmed Operation

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

1.6.2 NOMENCLATURE

The following conventions apply throughout this Recommendation:

    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.

1.6.3 CONVENTIONS

1.6.3.1 Specification of Operations

1.6.3.1.1 General

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

1.6.3.1.2 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.

1.6.3.1.3 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 parameter `diagnostic` is present if and only if the value of the parameter `result` 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.

1.6.3.1.4 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 Recommendation.

1.6.3.2 Typographic Conventions

1.6.3.2.1 Operation Names

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

1.6.3.2.2   Parameter Names

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

1.6.3.2.3   Value Names

The values of many parameters discussed in this Recommendation 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 generally 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 Recommendation.

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'.

1.6.3.2.4   State Names

This Recommendation specifies the states of 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').

1.6.3.2.5   SLE-PDU Names

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

1.6.3.2.6   Data Type Definitions

Data type definitions for the 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 Recommendation, the text of the ASN.1 modules is typeset entirely in a fixed-width font.

1.6.3.3   Other Conventions

This Recommendation uses the conventions specified in reference [1].

1.7 REFERENCES

The following documents contain provisions which, through reference in this text, constitute provisions of this Recommendation. At the time of publication, the editions indicated were valid. All documents are subject to revision, and users of this Recommendation 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 Recommendations.

NOTE – A list of informative references forms annex E of this Recommendation.

[1] *Cross Support Reference Model — Part 1: Space Link Extension Services*. Recommendation for Space Data System Standards, CCSDS 910.4-B-1. Blue Book. Issue 1. Washington, D.C.: CCSDS, May 1996.

[2] *Telecommand Part 1—Channel Service*. Recommendation for Space Data System Standards, CCSDS 201.0-B-3. Blue Book. Issue 3. Washington, D.C.: CCSDS, June 2000.

[3] *Telecommand Part 2—Data Routing Service*. Recommendation for Space Data System Standards, CCSDS 202.0-B-3. Blue Book. Issue 3. Washington, D.C.: CCSDS, June 2001.

[4] *Telecommand Part 2.1—Command Operation Procedures*. Recommendation for Space Data System Standards, CCSDS 202.1-B-2. Blue Book. Issue 2. Washington, D.C.: CCSDS, June 2001.

[5] *Time Code Formats*. Recommendation for Space Data System Standards, CCSDS 301.0-B-3. Blue Book. Issue 3. Washington, D.C.: CCSDS, January 2002.

[6] *Information Technology—Open Systems Interconnection—Specification of Abstract Syntax Notation One (ASN.1)*. International Standard, ISO/IEC 8824:1990. 2nd ed. Geneva: ISO, 1990.

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

[8] *Information Technology—Open Systems Interconnection—The Directory: Models*. International Standard, ISO/IEC 9594-2:1998. 3rd ed. Geneva: ISO, 1998.

2   DESCRIPTION OF THE FORWARD CLTU SERVICE

2.1   OVERVIEW

The Forward CLTU service enables the user of the service to send Command Link Transmission Units (CLTUs) to a spacecraft via an established forward space link channel. A forward space link channel is a physical channel carrying an asynchronous stream of CLTUs (reference [1]).

The service user submits CLTUs, encapsulated in Space Link Extension (SLE) Service Data Units (SLE-SDUs), by means of the CLTU-TRANSFER-DATA operation. Production of the Forward CLTU service by the provider entails processing the CLTUs transferred by the user through the necessary transformations to modulate the Radio Frequency (RF) carrier channel providing uplink communications with the spacecraft.

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

CLTUs may or may not conform to the format defined in reference [2]. The Forward CLTU service may be used to uplink any octet-aligned bit pattern.

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

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

b)   send annotated CLTUs to the provider;

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

d)   temporarily stop and later re-start the sending of CLTUs;

e)   release an association.

The Sequence Controlled (AD) and Expedited (BD) Services, as defined in the Command Operation Procedures (COP) Recommendation (reference [4]) are accomplished by higher layer SLE services.

The 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 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 Forward CLTU service may exist per physical channel at a time.

2.2 SPACE LINK EXTENSION REFERENCE MODEL

2.2.1 INTRODUCTION

The Forward CLTU service is specified within the framework defined by the SLE Reference Model (reference [1]). The following paragraphs 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.[1]

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 Recommendation, when two objects are involved in provision of a service, the object

---

[1] 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.

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

2.3.1 INTRODUCTION

Service management determines the number and schedule of 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 Recommendations.

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 Forward CLTU data channel to the RF carrier channel.

2.3.2 CONFIGURATION

Service Management provides initial values of configuration parameters. Certain configuration parameters are associated with provision of 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 Forward CLTU service user may modify some provision configuration parameters through operations specified in this Recommendation.

2.4   ARCHITECTURE MODEL – FUNCTIONAL VIEW

2.4.1   FORWARD TC SPACE LINK PROCESSING SLE FUNCTIONAL GROUP

The Forward Telecommand (TC) Space Link Processing SLE Functional Group (SLE-FG) (shown in figure 2-1) produces the Forward CLTU service.

As described in reference [1], the Forward TC Space Link Processing SLE-FG consumes one 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 3.6). The Forward TC Space Link Processing SLE-FG uses these data to extract the CLTUs and inject them into the asynchronous physical channel.

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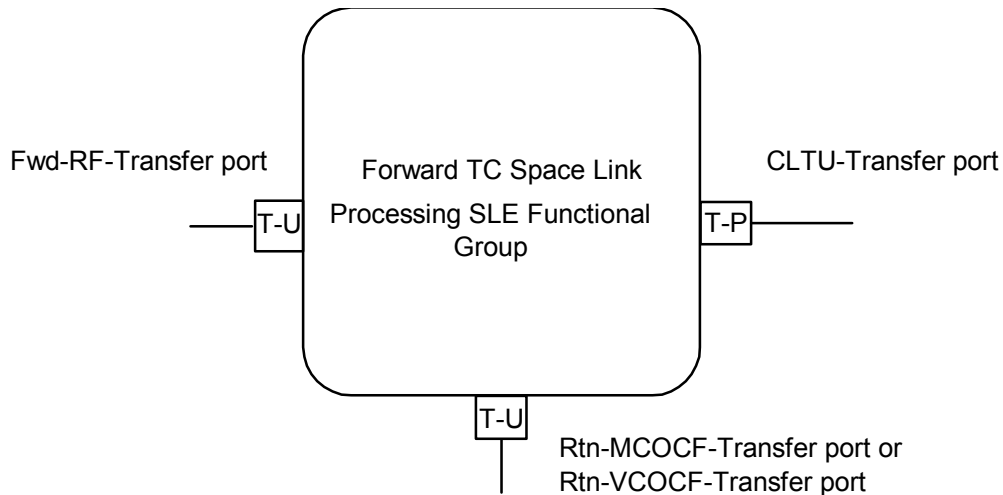
Figure 2-1: Forward TC Space Link Processing SLE-FG

The Forward TC Space Link Processing SLE-FG performs the following functions with respect to the Forward CLTU service:

a) consumes one CLTU data channel and extracts CLTUs encapsulated in SLE Service Data Units (SLE-SDUs);

b) consumes one Operational Control Field (OCF) data channel and extracts the Command Link Control Words (CLCWs). Based on the values in the CLCWs, the Forward CLTU service determines whether the physical channel is available;

NOTE – CLCWs may be ignored, as an option set by Service Management. See 3.1.10 and table 3-11.

   c) performs the following:

      1) generates the acquisition sequence and idle sequence on the physical channel in accordance with the PLOP in effect;

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

      3) modulates the CLTUs as a stream of bits on the ground-to-space RF channel; and

      4) radiates the signal to the spacecraft.

2.4.2 FORWARD CLTU SERVICE PRODUCTION AND PROVISION

Forward CLTU **production** is concerned with radiating CLTUs extracted from a stream of SLE-SDUs according to the CLTU control and annotation information in the SLE-SDU and according to the configuration set up by service management. Forward CLTU service **provision** is concerned with receiving a stream of SLE-SDUs from a 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 Forward CLTU service are sent by the service user by means of the 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 protocol data units (SLE-PDUs) which are exchanged between the Forward CLTU service provider and the 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.

NOTE – SLE-SDUs correspond to the parameters of the CLTU-TRANSFER-DATA and CLTU-ASYNC-NOTIFY operations defined in section 3. SL-DUs correspond to the radiated CLTUs.
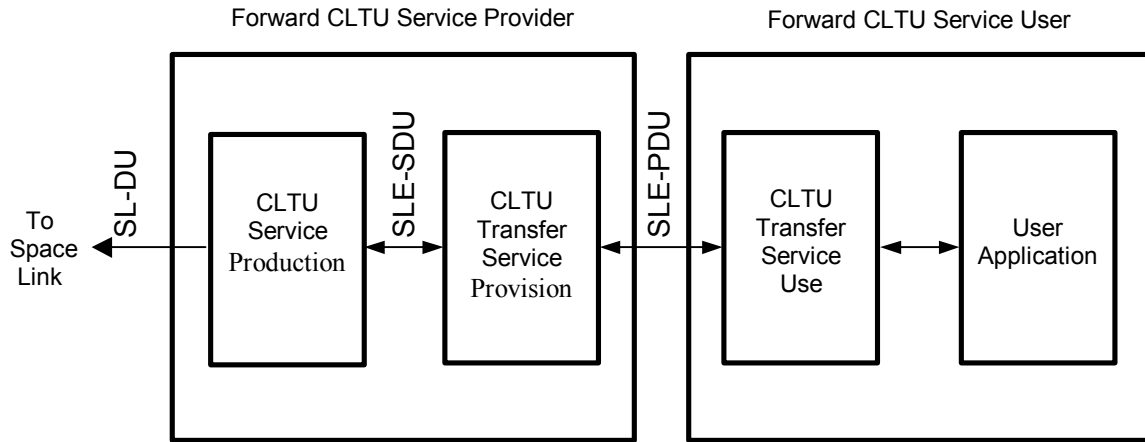
Figure 2-2: Forward CLTU Service Production and Provision

Production of the 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 G.

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 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 Forward CLTU service to a Mission Data Operations System (MDOS). As this figure shows, only a single Mission User Entity can use the Forward CLTU service provided by a single Forward Space Link Processing SLE-FG.
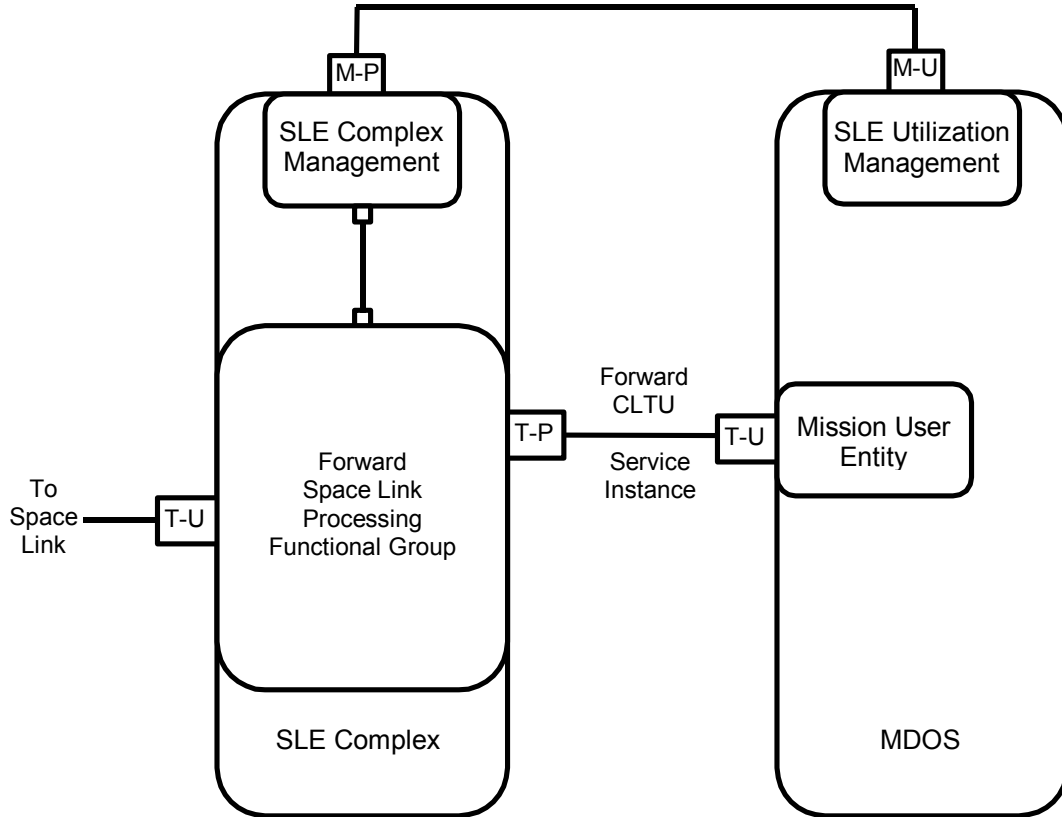
Figure 2-3: Example of Management and Provision of Forward CLTU Service

2.6 FUNCTIONAL DESCRIPTION

2.6.1 INTRODUCTION

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

2.6.2 SCHEDULING AND CONFIGURATION

The management of the SLE Complex that is providing the Forward CLTU service negotiates with SLE Utilization Management of the MDOS to establish mutually agreed upon SLE service packages. Among other things, SLE service packages specify the services to be provided, the schedules of service instances, and the resources needed to enable those services.

Service packages also specify the initial values of the mission-dependent parameters required for service production and provision. Forward CLTU service production parameters include such things as bit rate, modulation index, and subcarrier frequency. Provision parameters include such things as scheduled start and stop times of the 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 CLTU service does not depend on any other SLE transfer service. Provision of the CLTU Transfer service does depend on:

a) service management for scheduling, resources, and configuration; the schedule for a Forward CLTU service instance must be compliant with the schedule of the underlying equipment such as antennas, etc. The 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 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 Forward CLTU service are listed in table 2-1. Section 3 of this Recommendation contains the detailed specification of these operations.

Table 2-1: Forward CLTU Service Operations

Service Operation	Invoked By	Purpose	Con-firmed
CLTU-BIND	User	To establish an association with the provider	Yes
CLTU-UNBIND	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 a CLTU to the service provider	Yes
CLTU-ASYNC-NOTIFY	Provider	To notify the user of an event affecting production or provision of the 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 a Forward CLTU service instance is created, the 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 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 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 Recommendation. However, sending of CLTUs from the user to the provider (by means of the CLTU-TRANSFER-DATA operation) is not permitted. The user may enable the delivery of CLTUs 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 CLTUs and the provider is enabled to radiate CLTUs 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 and transition back to state 2.

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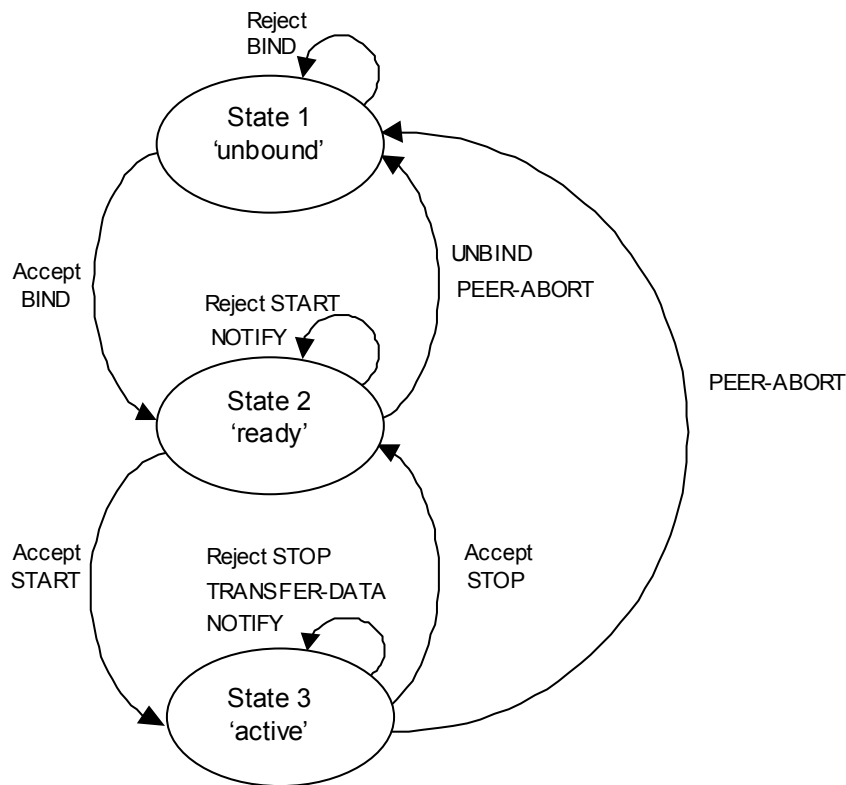
Figure 2-4: Forward CLTU Service Provider State Transition Diagram

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. 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 by the 'protocol abort' event described in 4.1.5.

2.6.4.4 Effects of Association Termination

The production of CLTUs stops immediately following the termination of an association, except for a CLTU in the process of being radiated. Any buffered CLTUs 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 of CLTUs continues, and buffered CLTUs 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 CLTUs processed (see 3.9), 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 Buffering

The 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.

2.6.4.6 Technology-specific Aspects—Interoperability and the Underlying Communications Service

This Recommendation defines the Forward CLTU service. Provision of the 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 Forward CLTU service operations can be exchanged between the user and the provider. In order not to restrict the applicability of this Recommendation 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 Recommendation 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 Recommendation conceptualizes such mapping in two parts:

a) 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 Forward CLTU service operation invocations and returns to SLE-PDUs is specified by this Recommendation. The mapping of SLE-PDUs to an underlying communications service is intentionally outside the scope of this Recommendation (e.g., so that the 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 Recommendation but also to an agreed upon specification of the mapping of the Forward CLTU service to the underlying communications service. The specification of a mapping of the 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 Forward CLTU service that results from such a mapping. The specification of such mappings is the subject of separate CCSDS Recommendations.
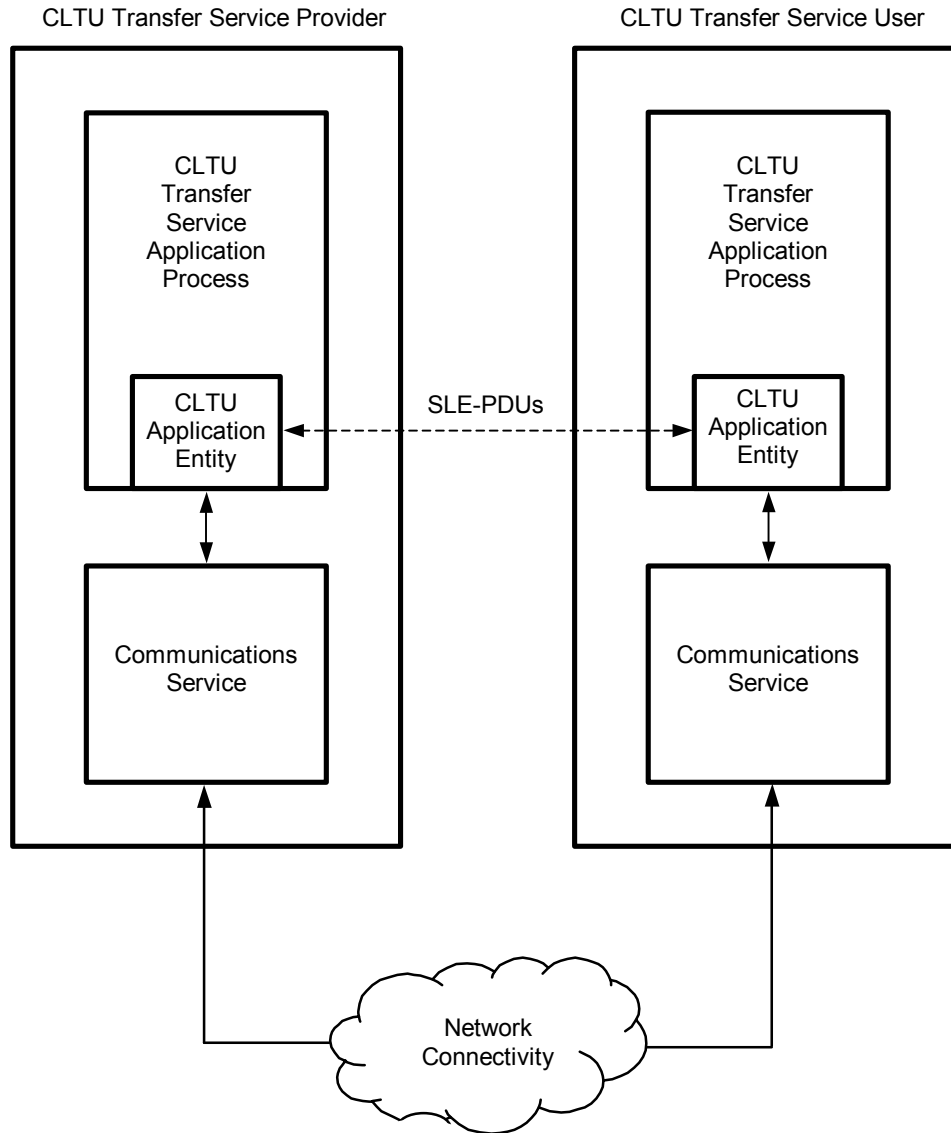
Figure 2-5: Communications Realization of Forward CLTU Service

Because the operations of the 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 Forward CLTU service to the underlying communications service.

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 Forward CLTU service to the underlying communications service.

The responder port identifier is included as a parameter of the CLTU-BIND operation. Generally speaking, that is unnecessary; it is only necessary that SLE applications 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 possibility that the implementation of a gateway might be simplified by the inclusion of this parameter in the CLTU-BIND operation.

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 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.

2.7   OPERATIONAL SCENARIO

Prior to the actual provision of service, start and stop times for both the space link session and the associated 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.

Some time before the scheduled start time of the 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 Forward CLTU service and, if applicable, of the underlying FTCF and Forward CLTU services. 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 the 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 Command 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, 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 operation to the provider. The provider verifies the invocation, and if acceptable, buffers the CLTU 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, if production status is operational, the first CLTU is injected into the physical channel and modulated onto the RF carrier. The signal is radiated to the spacecraft. If no start time was specified in the CLTU-TRANSFER-DATA operation, the CLTU is radiated as soon as received, or whenever the production status becomes operational, if that is later.

h) Successive CLTUs are processed in similar fashion after the delay period (if any) specified in the preceding CLTU is satisfied.

i) The user transfers the last CLTU to the provider.

j) The provider completes processing the buffered CLTUs. When the provider's CLTU buffer is empty, it sends CLTU-ASYNC-NOTIFY to inform the user.

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

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

3 OPERATIONS AND THEIR PARAMETERS

3.1 GENERAL CONSIDERATIONS

NOTES

1   This subsection specifies the processing of valid (i.e., recognized) SLE-PDUs. It defines common aspects of the operations of the Forward CLTU service.

2   Handling of invalid SLE-PDUs is specified in 0.

3.1.1 RESULT OF OPERATIONS

3.1.1.1   All confirmed operations shall report on the outcome of the operation in a return.

3.1.1.2   With the exception of CLTU-UNBIND, all returns shall include a `result` parameter that indicates whether the outcome of the operation was successful ('positive result') or unsuccessful ('negative result').

3.1.1.3   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.

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

3.1.2 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 so 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.

3.1.3 PARAMETER CHECKING

3.1.3.1   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 paragraphs.

3.1.3.2   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).

3.1.3.3   If a parameter value is not valid, the operation shall not be performed, and, for confirmed operations other than CLTU-UNBIND, a report of 'negative result' shall be returned to the invoker.

3.1.3.4   Except as noted in 3.2.2.11, 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.

3.1.3.5   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 checks are actually performed.

3.1.4   ACCESS CONTROL

3.1.4.1   The 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.

3.1.4.2   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.

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

3.1.4.4 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.

3.1.5 AUTHENTICATION

NOTE – Requirements for security depend on the application and on the SLE system environment (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. This Recommendation 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.

3.1.5.1 The Forward CLTU service shall provide the following options with respect to level of authentication:

   a) 'all'—all 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 Forward CLTU invocations or returns shall be authenticated.

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

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

NOTES

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

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 the algorithms necessary to generate and check credentials.

3.1.5.4 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.

3.1.5.5 For operations for which authentication of credentials 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.

3.1.6 THREADED APPLICATIONS

NOTES

1   Multi-threading has evolved to a widely used technique to optimize systems' performance. To support applications that may be implemented using multiple threads of execution, the parameter `invoke-ID` is specified for all confirmed operations except CLTU-BIND and CLTU-UNBIND.

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

3   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.

3.1.6.1   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 is not received in a timely manner, the invoker may invoke CLTU-PEER-ABORT to terminate the association.

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

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

3.1.6.4   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.

3.1.6.5   To ensure that the 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.

3.1.6.6   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.

3.1.6.7   Compliance with this Recommendation 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.

3.1.7   MANDATORY PARAMETERS

All providers shall be able to accept and respond to parameters specified as mandatory in annex D.

NOTE –   All parameters of operations are mandatory except where stated differently in annex D. The actual presence of a parameter in a particular invocation or return may be conditional based on the value of another parameter in the same invocation or return.   For some parameters, a value of 'null' is allowed, as discussed in the description of the relevant operations.

3.1.8   TIME

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

3.1.8.2   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.

3.1.8.3   All time values shall be expressed to a precision of at least one-tenth (0.1) of a second.

3.1.8.4   All time value shall be accurate to within one-tenth (0.1) of a second or better.

3.1.9   DELIVERY MODES

3.1.9.1   Forward Online Delivery

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

3.1.9.1.2   CLTUs supplied by the service user shall be buffered by the service provider until they are processed.

3.1.9.1.3   The timing of CLTU processing shall be determined by the order of CLTUs in the buffer and any annotation data provided with the CLTUs.

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

3.1.9.2   Forward Offline Delivery

3.1.9.2.1   Service provision and service production shall not overlap.

3.1.9.2.2   CLTUs 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 Recommendation.

3.1.10 SETTING OF PARAMETERS

3.1.10.1 A Forward CLTU provider shall permit setting of the service configuration parameters as specified in table 3-1.

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

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

3.1.11 PROVIDER BUFFERING REQUIREMENTS

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

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

3.1.11.3 The service provider shall buffer only complete CLTUs.

3.1.12 ACCOUNTING SUMMARY

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

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

Parameter	Service Management	CLTU-START Operation	CLTU-SCHEDULE-STATUS-REPORT Operation	CLTU-THROW-EVENT Operation (NOTE 3)
bit-lock-required	X			
delivery-mode	X			
expected-cltu-identification		X		
expected-event-invocation-identification				X
maximum-cltu-length	X			
modulation-frequency	X			
modulation-index	X			
notification-mode	X			
plop-in-effect	X			
protocol-abort-mode	X			
reporting-cycle			X	
return-timeout-period	X			
rf-available-required	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 3.7.2.3. Other parameters are presented and described in table 3-11. A complete list of parameters that may affect service production may be found in references [E5] and [E6].

2   The user can ascertain the current value of the parameters presented in table 3-11 by means of the CLTU-GET-PARAMETER operation.

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

3.2   CLTU-BIND

3.2.1   PURPOSE

3.2.1.1   The initiator shall invoke the CLTU-BIND operation to establish an association between the initiator and responder as defined in 1.6.1.7.2.

3.2.1.2   The responder shall confirm the CLTU-BIND operation.

3.2.1.3   Except as provided in 3.2.1.4, the initiator shall not invoke any further CLTU operations on this service instance until the bind is confirmed.

3.2.1.4   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 3.12) followed by another CLTU-BIND.

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

3.2.2   INVOCATION, RETURN, AND PARAMETERS

3.2.2.1   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

3.2.2.2 `invoker-credentials`

The **`invoker-credentials`** parameter shall provide information that enables the performer to authenticate the CLTU-BIND invocation (see 3.1.5).

3.2.2.3 `performer-credentials`

The **`performer-credentials`** parameter shall provide information that enables the invoker to authenticate the return from the performance of CLTU-BIND (see 3.1.5).

3.2.2.4 `initiator-identifier`

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

NOTES

1   The `initiator-identifier` parameter permits the responding SLE application to determine if this user is registered at this responder and which authentication scheme is to be applied.

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

3.2.2.5 `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.

3.2.2.6 `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 2.6.4.6 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.

3.2.2.7 `service-type`

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

3.2.2.8 `version-number`

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

3.2.2.8.2   `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.

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

---

[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. The CLTU-BIND `service-type` parameter is provided in an attempt to facilitate such a change.

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

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

3.2.2.8.4 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'.

3.2.2.8.5 If the responder proposes a lower version in the return than supported by the initiator, the initiator shall unbind the association.

3.2.2.8.6 The `version-number` value of the Forward CLTU service defined by this issue of this Recommendation shall be '1'.

NOTE – The version negotiation process as outlined above is only feasible as long as a future version of the Forward CLTU service retains the specification of the CLTU-BIND operation.

3.2.2.9 `service-instance-identifier`

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

NOTE – The `service-instance-identifier` parameter takes the form of a Distinguished Name (see reference [8]). The attributes of the Distinguished Name should include the SLE service agreement and the SLE service package applicable to this instance of service. More details on the way the `service-instance-identifier` is built can be found in reference [E5].

3.2.2.10 `result`

3.2.2.10.1 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 invocation is accepted by the responder and the association is established;

b) 'negative result'—the CLTU-BIND invocation is rejected by the responder for the reason specified in the `diagnostic` parameter.

3.2.2.11 `diagnostic`

3.2.2.11.1 If `result` is 'negative result', **diagnostic** shall be present and shall contain 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 `service-type` value in the CLTU-BIND invocation does not identify a service type supported by the responder;

c) 'version not supported'—

   1) the responder does not support the requested version, and

   2) 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 service instance identified by the value of the service-instance-identifier parameter has not been created for use by the initiator specified in the initiator-identifier parameter;

g) 'inconsistent service type'—the service-type value in the CLTU-BIND invocation is not consistent with the service type of this service instance; i.e., it is not 'fwdCltu;

h) 'invalid time'—the CLTU-BIND operation was invoked outside the service instance provision period agreed to in the service package;

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.

   NOTE – Initiators should consider that, under some conditions, CLTU-BIND might fail with no return, e.g., if `responder-port-identifier` has an incorrect value.

3.2.2.11.2 If `result` is 'positive result', `diagnostic` shall not be present.

3.2.3 EFFECTS

3.2.3.1 If `result` is 'positive result':

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 Forward CLTU service operations to initialize the service and enable data transfer (e.g., CLTU-SCHEDULE-STATUS-REPORT and CLTU-START).

3.2.3.2 If `result` is 'negative result':

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.

3.3 CLTU-UNBIND

3.3.1 PURPOSE

3.3.1.1 The initiator shall invoke the CLTU-UNBIND to release an association previously established by CLTU-BIND (see 0).

3.3.1.2 The responder shall confirm the CLTU-UNBIND operation.

3.3.1.3 Except as provided in 3.3.1.4, the initiator shall not invoke any further 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).

3.3.1.4 If the return from the CLTU-UNBIND invocation is not received after a sufficiently long time (to be determined by service management), the initiator shall invoke the CLTU-PEER-ABORT operation (see 3.12).

NOTE – The 'sufficiently long time' as determined by service management can be estimated by invoking the CLTU-GET-PARAMETER operation and selecting 'return-timeout-period' as the value of `cltu-parameter`.

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

3.3.2.2 invoker-credentials

The **invoker-credentials** parameter shall provide information that enables the performer to authenticate the CLTU-UNBIND invocation (see 3.1.5).

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-UNBIND (see 3.1.5).

3.3.2.4 unbind-reason

The **unbind-reason** parameter shall specify why 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 end the service instance and release its resources allocated to this service instance;

NOTE – Further invocation of the CLTU-BIND operation is not permitted 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 this service usage for an unspecified period of time: the initiator may attempt to re-bind to the responder to continue data transfer at any time prior to the end of the provision period scheduled for this service instance;

c) 'version not supported'—the initiator does not support the version of the Forward CLTU service proposed by the responder 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.

3.3.2.5 `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.

NOTE – 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.

3.3.3 EFFECTS

3.3.3.1 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').

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

3.3.3.3 If `unbind-reason` is not 'end':

a) the initiator may attempt to re-bind at any time prior to the end of the provision period scheduled for this service instance;

b) the provider shall maintain accumulated statistics such as are reported in CLTU-STATUS-REPORT for the duration of the service instance provision period;

c) parameters that serve to relate notifications on an activity to operations that triggered that activity shall not be altered when the association is released.

3.4   CLTU-START

3.4.1   PURPOSE

3.4.1.1   The user shall invoke the CLTU-START operation to request that the Forward CLTU service provider prepare to receive CLTU-TRANSFER-DATA invocations (see 3.6).

3.4.1.2   The Forward CLTU service provider shall confirm the CLTU-START operation.

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

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

3.4.2   INVOCATION, RETURN AND PARAMETERS

3.4.2.1   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

3.4.2.2 `invoker-credentials`

The **invoker-credentials** parameter shall provide information that enables the performer to authenticate the CLTU-START invocation (see 3.1.5).

3.4.2.3 `performer-credentials`

The **`performer-credentials`** parameter shall contain information that enables the invoker to authenticate the return from the performance of CLTU-START (see 3.1.5).

3.4.2.4 `invoke-ID`

The Forward CLTU service provider shall return unchanged the user-supplied value of the **`invoke-ID`** parameter (see 3.1.6).

3.4.2.5 `first-cltu-identification`

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

3.4.2.5.2 If the user invokes a CLTU-START after re-binding to the provider following a protocol abort when the protocol-abort-mode parameter specified 'continue' (see 4.1.5.5), the value of the `first-cltu-identification` parameter must be greater than the `cltu-identification` of any CLTU remaining in the buffer. Otherwise, the provider returns 'negative result' with a diagnostic of 'invalid cltu-ID'.

NOTE – Following certain exceptions (e.g. 'sldu expired', see 3.7.2.3), 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 a CLTU for which processing did not yet complete can be uniquely identified.

3.4.2.6 `start-production-time`

3.4.2.6.1 The **`start-production-time`** parameter shall be returned when the value of the `result` parameter is 'positive result'.

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

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

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

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

3.4.2.6.4   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'.

3.4.2.7   `stop-production-time`

3.4.2.7.1   The **`stop-production-time`** parameter shall be returned when the value of `result` is 'positive result'.

3.4.2.7.2   `stop-production-time` shall be the estimated time at which the provider will stop radiating CLTUs.

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

3.4.2.8   `result`

The **`result`** parameter shall be used by the Forward CLTU service provider to specify the result of the CLTU-START invocation and shall contain one of the following values:

   a) 'positive result'—the user may start invoking CLTU-TRANSFER-DATA operations;

   b) 'negative result'—the CLTU-START invocation is rejected by the provider for the reason specified in the `diagnostic` parameter; the provider cannot accept CLTU-TRANSFER-DATA invocations.

3.4.2.9   `diagnostic`

3.4.2.9.1   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) '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 CLTUs is already past;

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

f) 'other reason'—the provider cannot accept CLTU-TRANSFER-DATA operations for unspecified reasons.

3.4.2.9.2 If `result` is 'positive result', `diagnostic` shall not be present.

3.4.3 EFFECTS

3.4.3.1 If `result` is 'positive result':

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

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

3.4.3.2 If `result` is 'negative result':

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, it should set the `production-status` parameter to 'halted'.

3.5 CLTU-STOP

3.5.1 PURPOSE

3.5.1.1   The user shall invoke the CLTU-STOP operation to inform the provider that the user is stopping the sending of CLTUs and to request that the provider stop production, i.e., stop radiation, of CLTUs 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 CLTUs on the provider side by invoking the CLTU-START operation.

3.5.1.2   The provider shall confirm the CLTU-STOP operation.

3.5.1.3   The CLTU-STOP operation is valid only in state 3 ('active') 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-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

3.5.2.2   **invoker-credentials**

The **invoker-credentials** parameter shall provide information that enables the performer to authenticate the CLTU-STOP invocation (see 3.1.5).

3.5.2.3   **performer-credentials**

The **performer-credentials** parameter shall provide information that enables the invoker to authenticate the return from the performance of CLTU-STOP (see 3.1.5).

3.5.2.4 `invoke-ID`

The provider shall return unchanged the user-supplied value of the `invoke-ID` parameter (see 3.1.6).

3.5.2.5 `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 invocation is accepted, and the user may invoke a CLTU-UNBIND operation or a new CLTU-START operation;

b) 'negative result'—the CLTU-STOP invocation is rejected by the provider for the reason specified in `diagnostic`.

3.5.2.6 `diagnostic`

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

3.5.2.6.2 If `result` is 'positive result', `diagnostic` shall not be present.

3.5.3 EFFECTS

3.5.3.1 If `result` is 'positive result':

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

b) for CLTUs already buffered:

　1) the provider shall discard any CLTUs for which radiation has not yet begun;

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

　NOTE – An additional CLTU 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 a CLTU 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.

3.5.3.2   If `result` is 'negative result':

a) processing shall continue;

b) the provider state shall remain unchanged;

c) any buffered CLTUs shall not be discarded.

3.6 CLTU-TRANSFER-DATA

3.6.1 PURPOSE

3.6.1.1 The user shall invoke the CLTU-TRANSFER-DATA operation to transmit data to the provider.

3.6.1.2 The provider shall confirm the CLTU-TRANSFER-DATA operation.

3.6.1.3 The CLTU-TRANSFER-DATA 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-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

3.6.2.2 **invoker-credentials**

The **invoker-credentials** parameter shall provide information that enables the performer to authenticate the CLTU-TRANSFER-DATA invocation (see 3.1.5).

3.6.2.3 `performer-credentials`

The `performer-credentials` parameter shall contain information that enables the invoker to authenticate the return from the performance of CLTU-TRANSFER-DATA (see 3.1.5).

3.6.2.4 `invoke-ID`

The provider shall return unchanged the user-supplied value of the `invoke-ID` parameter (see 3.1.6).

3.6.2.5 `cltu-identification`

3.6.2.5.1 The `cltu-identification` parameter shall contain a monotonically increasing sequence number placed in all CLTU-TRANSFER-DATA operations.

NOTE – The number is used to reference the CLTU in all subsequent notifications relating to this CLTU.

3.6.2.5.2 The value of `cltu-identification` returned shall be the sequence number expected in the next CLTU-TRANSFER-DATA operation:

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

b) if the transfer invocation is rejected, the `cltu-identification` in the return shall contain the value expected by the 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`.

3.6.2.6 `earliest-radiation-time`

3.6.2.6.1 The `earliest-radiation-time` parameter shall be used to specify the earliest time that the provider shall start processing this CLTU.

3.6.2.6.2 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 may begin to be radiated.

3.6.2.6.3 If the `earliest-radiation-time` parameter is unspecified, the provider shall begin processing immediately after any delay associated with the previous CLTU has timed out, provided the `production-status` parameter is 'operational'.

3.6.2.7 `latest-radiation-time`

3.6.2.7.1 The **`latest-radiation-time`** parameter shall either contain a value of 'null' or specify the latest time at which the leading edge of the first bit of the CLTU may begin to be radiated.

3.6.2.7.2 If the parameter contains a value of 'null', the CLTU shall be radiated, as long as radiation occurs before the production process is halted.

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

3.6.2.7.4 If radiation has not begun at or before `latest-radiation-time`, an 'sldu expired' exception shall be notified to the user, and the CLTU shall not be radiated.

3.6.2.8 `delay-time`

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

3.6.2.8.2 The delay shall be measured from the trailing edge of the last radiated bit of this CLTU to the leading edge of the first bit of the next CLTU.

3.6.2.8.3 If `delay-time` contains a value of zero, radiation of the next CLTU may occur immediately.

3.6.2.8.4 The effect of `delay-time` varies depending on the setting of the `plop-in-effect` parameter (see 3.1.10).

    a) when the PLOP in effect is PLOP-1 (reference [2]),

        1) command modulation shall be dropped between CLTUs for the period of time specified by `delay-time`,

        2) the acquisition sequence shall be transmitted prior to radiation of the next CLTU,

        3) time for radiation of the acquisition sequence shall be in addition to the time specified by `delay-time`;

b) when the PLOP in effect is PLOP-2 (reference [2]),

    1) an idle sequence shall be radiated between CLTUs, for the period of time specified by `delay-time`,

    2) the time specified by `delay-time` shall be increased to an integral number of bit times.

3.6.2.9 `report`

The **`report`** parameter shall specify whether the provider shall invoke the CLTU-ASYNC-NOTIFY operation upon completion of the radiation of the CLTU, 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.

3.6.2.10 data

The **`data`** parameter shall contain a CLTU for radiation to the space element of a mission.

3.6.2.11 `cltu-buffer-available`

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

3.6.2.12 `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 invocation has been accepted by the provider and the data will be buffered until processing is due;

b) 'negative result'—the CLTU-TRANSFER-DATA invocation is rejected by the provider, for the reason returned in the `diagnostic` parameter.

3.6.2.13 `diagnostic`

3.6.2.13.1 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) 'unable to process'—the provider cannot process CLTUs 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 a CLTU becomes 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 G.

   3) the `production-status` is 'operational', but the provider is blocked due to an expired CLTU: the provider has reported the fault condition to the user via a CLTU-ASYNC-NOTIFY operation containing a `notification-type` of 'sldu expired';

   4) 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 CLTU;

d) 'out of sequence'—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 invocation;

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

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 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 radiation service is scheduled;

2) the service instance does not allow the user to specify a radiation window, but the `earliest-radiation-time` and/or `latest-radiation-time` parameters contain values other than null;

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

h) 'invalid delay time'—the value of the delay-time is outside the range set in the service package;

i) '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 CLTU exceeds the maximum size allowed for this service instance;

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

3.6.2.13.2 If `result` is 'positive result', `diagnostic` shall not be present.

3.6.3 EFFECTS

3.6.3.1 If `result` is 'positive result':

a) the provider shall buffer the CLTU until it is due to be processed;

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

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

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

3.7 CLTU-ASYNC-NOTIFY

3.7.1 PURPOSE

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

3.7.1.2 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.

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

3.7.2 INVOCATION AND PARAMETERS

3.7.2.1 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

3.7.2.2 invoker-credentials

The **invoker-credentials** parameter shall provide information that enables the performer to authenticate the CLTU-ASYNC-NOTIFY invocation (see 3.1.5).

3.7.2.3 `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 CLTU 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'.

b) 'sldu expired'—radiation of the CLTU 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. No further CLTUs shall be radiated; buffered CLTUs 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, 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.

c) 'production interrupted'—the production process has stopped due to a condition that may be temporary. No further CLTUs shall be radiated; buffered CLTUs 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   After the 'production interrupted' event, the service user has to clear the blocking 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.

2   When the value of the notification-mode parameter is 'immediate', the provider sends the 'production interrupted' notification immediately on occurrence of a production fault. When the value of the notification-mode parameter is 'deferred', the provider waits until a CLTU is ready to be radiated before sending the 'production interrupted' notification. The notification-mode parameter (see annex G) documents the notification method supported by the provider implementation.

d) 'production halted'—the production process has been stopped by management action. No further CLTUs shall be radiated; buffered CLTUs 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 by successful invocation of CLTU-START.

e) 'production operational'—the production process is ready to radiate CLTUs 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 CLTUs 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.

3.7.2.4 `event-thrown-identification`

3.7.2.4.1 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.

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

3.7.2.5 `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) no CLTUs 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 CLTU that the provider most recently processed or attempted to process, whether the CLTU was successfully radiated or an exception occurred.

3.7.2.6 `cltu-last-OK`

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

a) if no CLTUs 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 CLTU that most recently was successfully radiated.

3.7.2.7 `cltu-status`

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

3.7.2.7.2 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 CLTU identified by `cltu-last-processed`:

a) 'radiated'—radiation of the CLTU completed without aborting: the CLTU is guaranteed to have been radiated nominally;

b) 'expired'—the `latest-radiation-time` of the CLTU was reached before beginning radiation: the CLTU is guaranteed not to have been radiated;

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

d) 'radiation started'—radiation of the CLTU started but processing is not yet complete;

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

3.7.2.8 `radiation-start-time`

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

3.7.2.8.2 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 CLTU 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;

    b) if the `cltu-status` value is 'expired' or 'undefined', the value of `radiation-start-time` shall be set to 'null'.

3.7.2.9 `radiation-stop-time`

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

3.7.2.9.2 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 identified by `cltu-last-OK` completed, i.e., the time of the trailing edge of the last radiated bit.

3.7.2.10 `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;

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

    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.

3.7.2.11 `uplink-status`

3.7.2.11.1 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.

3.7.2.11.2 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'.

3.7.3 EFFECTS

3.7.3.1 If `notification-type` indicates 'sldu expired', 'production interrupted', or 'production halted':

a) no further CLTUs shall be radiated;

b) buffered CLTUs shall be discarded; and

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

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

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

3.8 CLTU-SCHEDULE-STATUS-REPORT

3.8.1 PURPOSE

3.8.1.1 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 3.9);

b) send periodic status reports (see 3.9); or

c) stop sending periodic status reports.

3.8.1.2 The provider shall confirm the CLTU-SCHEDULE-STATUS-REPORT operation.

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

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

3.8.2 INVOCATION, RETURN, AND PARAMETERS

3.8.2.1 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

3.8.2.2 **invoker-credentials**

The **invoker-credentials** parameter shall provide information that enables the performer to authenticate the CLTU-SCHEDULE-STATUS-REPORT invocation (see 3.1.5).

3.8.2.3 `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 3.1.5).

3.8.2.4 `invoke-ID`

The performer shall return unchanged the invoker-supplied value of the **invoke-ID** parameter (see 3.1.6).

3.8.2.5 `report-request-type`

3.8.2.5.1 The **report-request-type** parameter shall specify how reporting shall be done and shall contain one of the following values:

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

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

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

3.8.2.5.2 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.

3.8.2.6 `reporting-cycle`

3.8.2.6.1 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.

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

3.8.2.7 `result`

The **result** parameter shall specify the result of the CLTU-SCHEDULE-STATUS-REPORT invocation and shall contain one of the following values:

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

b) 'negative result'—the CLTU-SCHEDULE-STATUS-REPORT invocation is rejected by the provider for the reason specified in the `diagnostic` parameter; the previous setting for status reporting remains in effect.

3.8.2.8 `diagnostic`

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

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

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

3.8.2.8.2 If `result` is 'positive result', `diagnostic` shall not be present.

3.8.3 EFFECTS

3.8.3.1 If `result` is 'positive result':

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.

3.8.3.2 If `result` is 'negative result', the previous setting for reporting shall remain in effect.

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

3.9 CLTU-STATUS-REPORT

3.9.1 PURPOSE

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

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

3.9.1.3 Status reports shall be sent (or not sent) in accordance with user requests conveyed by means of the CLTU-SCHEDULE-STATUS-REPORT operation (see 3.8).

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

3.9.2 INVOCATION AND PARAMETERS

3.9.2.1 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

3.9.2.2 invoker-credentials

The **invoker-credentials** parameter shall provide information that enables the performer to authenticate the CLTU-STATUS-REPORT invocation (see 3.1.5).

3.9.2.3 `cltu-last-processed`

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

3.9.2.3.2 Otherwise, the `cltu-last-processed` parameter shall specify the sequence number of the CLTU that the provider most recently processed or attempted to process, whether the CLTU was successfully radiated or an exception occurred.

3.9.2.4 `cltu-last-OK`

3.9.2.4.1 If no CLTUs 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'.

3.9.2.4.2 Otherwise, the `cltu-last-OK` parameter shall specify the sequence number of the CLTU that most recently was successfully radiated.

3.9.2.5 `cltu-status`

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

3.9.2.5.2 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 CLTU identified by `cltu-last-processed`:

a) 'radiated'—radiation of the CLTU completed without aborting: the CLTU is guaranteed to have been radiated nominally;

b) 'expired'—the `latest-radiation-time` of the CLTU was reached before beginning radiation: the CLTU is guaranteed not to have been radiated;

c) 'interrupted'—radiation of the CLTU was aborted when the production status changed from 'operational' to 'interrupted': status information available to the provider indicates that the CLTU started radiating and aborted prior to nominal completion;

d) 'radiation started'—radiation of the CLTU started but processing is not yet complete;

e) 'undefined'— the reported CLTU did not start radiation: the CLTU was blocked because of the 'interrupted' status of the production process.

3.9.2.6 `radiation-start-time`

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

3.9.2.6.2 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 CLTU 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;

    b) if the `cltu-status` value is 'expired' or 'undefined', the value of `radiation-start-time` shall be set to 'null'.

3.9.2.7 `radiation-stop-time`

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

3.9.2.7.2 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 CLTU identified by `cltu-last-OK` completed, i.e., the time of the trailing edge of the last radiated bit.

3.9.2.8 `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;

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

    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.

3.9.2.9 `uplink-status`

3.9.2.9.1 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.

3.9.2.9.2 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'.

3.9.2.10 `number-of-cltus-received`

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

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

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

3.9.2.11 `number-of-cltus-processed`

3.9.2.11.1 The `number-of-cltus-processed` parameter shall specify the number of CLTUs that the provider attempted to radiate during this service instance, including

– CLTUs that radiated successfully;

– CLTUs that expired;

– CLTUs that aborted;

– a CLTU in the process of being radiated.

3.9.2.11.2 If no CLTUs have been processed, the value of this parameter shall be zero.

3.9.2.12 `number-of-cltus-radiated`

3.9.2.12.1 The `number-of-cltus-radiated` shall contain the count of CLTUs successfully radiated during this service instance.

3.9.2.12.2 A CLTU in the process of being radiated shall not be included in this count.

3.9.2.12.3 If no CLTUs have been radiated, the value of this parameter shall be zero.

3.9.2.13 `cltu-buffer-available`

3.9.2.13.1 The **`cltu-buffer-available`** parameter shall specify the remaining number of octets available for buffering CLTUs.

3.9.2.13.2 Once the provider starts to process a CLTU, it shall be considered removed from the buffer.

3.9.3   EFFECTS

3.9.3.1   Status information shall be delivered to the user.

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

3.10 CLTU-GET-PARAMETER

3.10.1 PURPOSE

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

3.10.1.2 The provider shall confirm the CLTU-GET-PARAMETER operation.

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

3.10.2 INVOCATION, RETURN, AND PARAMETERS

3.10.2.1 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

3.10.2.2 **invoker-credentials**

The **invoker-credentials** parameter shall provide information that enables the performer to authenticate the CLTU-GET-PARAMETER invocation (see 3.1.5).

3.10.2.3 **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 3.1.5).

3.10.2.4 invoke-ID

The performer shall return unchanged the invoker-supplied value of the **invoke-ID** parameter (see 3.1.6).

3.10.2.5 cltu-parameter

3.10.2.5.1 The **cltu-parameter** parameter shall specify the 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.

3.10.2.5.2 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.

3.10.2.6 parameter-value

3.10.2.6.1 The **parameter-value** parameter shall contain the value for the parameter specified by cltu-parameter as described in 3.10.2.5.

3.10.2.6.2 parameter-value 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.

3.10.2.7 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 invocation is accepted by the provider, and the value of the specified CLTU parameter is provided in the positive return;

b) 'negative result'—the CLTU-GET-PARAMETER invocation is rejected by the provider for the reason specified in the diagnostic parameter.

Table 3-11: Forward CLTU Parameters

Parameter	Description
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'.
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.
maximum-cltu-length	The size, in octets, of the maximum-length CLTU that will be accepted by the provider for this service instance.
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).
plop-in-effect	The physical layer operation procedure (PLOP) being used: 'PLOP-1' or 'PLOP-2'.
reporting-cycle	'null' if cyclic reporting is off, otherwise the time (in seconds) between successive CLTU-STATUS-REPORT invocations (see 3.8.2.6).
return-timeout-period	Maximum time period (in seconds) permitted between the invocation of a Forward CLTU operation and the receipt by the invoker of the return (for confirmed operations). (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 the production status to 'operational'.
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.

3.10.2.8 `diagnostic`

3.10.2.8.1 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 parameter specified by the value of `cltu-parameter` is not known to the service provider;

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

3.10.2.8.2 If `result` is 'positive result', `diagnostic` shall not be present.

3.10.3 EFFECTS

3.10.3.1 If `result` is 'positive result', the value of the parameter specified by the `cltu-parameter` parameter shall be returned to the user in the positive return.

3.10.3.2 If `result` is 'negative result', the value of the parameter specified by the `cltu-parameter` parameter shall not be returned to the user.

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

3.11 CLTU-THROW-EVENT

3.11.1 PURPOSE

3.11.1.1 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.

NOTE – 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 changes by throwing the associated event.

3.11.1.2 The performer shall confirm the CLTU-THROW-EVENT operation.

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 as to the effect the event.

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

3.11.2 INVOCATION, RETURN, AND PARAMETERS

3.11.2.1 General

The parameters of the CLTU-THROW-EVENT operation shall be present in the invocation and return as specified in table 3-12.

3.11.2.2 `invoker-credentials`

The `invoker-credentials` parameter shall provide information that enables the performer to authenticate the CLTU-THROW-EVENT invocation (see 3.1.5).

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-THROW-EVENT (see 3.1.5).

3.11.2.4 `invoke-ID`

The performer shall return unchanged the invoker-supplied value of the `invoke-ID` parameter (see 3.1.6).

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

3.11.2.5 `event-invocation-identification`

3.11.2.5.1 The **`event-invocation-identification`** parameter shall contain a monotonically increasing sequence number and shall be

   a) set to zero 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.

3.11.2.5.2 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 parameter`;

   b) if the invocation is rejected, the `event-invocation-identification` in the return shall contain the value expected by the 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.

3.11.2.5.3 The user shall set the value of `event-invocation-identification` in the next CLTU-THROW-EVENT invocation to the value returned by the provider.

3.11.2.5.4 The value expected by the provider shall not be altered when the association between user and provider is reestablished after being released or aborted.

3.11.2.5.5 The provider shall use the user-supplied value of `event-invocation-identification` to reference the respective event in all subsequent reports.

3.11.2.6 `event-identifier`

The **event-identifier** parameter shall identify the event to be forwarded to SLE Complex Management by the service provider.

3.11.2.7 `event-qualifier`

The **event-qualifier** parameter may be used to provide additional information about the event identified by this CLTU-THROW-EVENT invocation. The provider does not verify the contents of the `event-qualifier` parameter, but forwards the contents to SLE Complex Management together with the event.

3.11.2.8 `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 invocation is accepted by the provider; the event and the event qualifier will be forwarded to SLE Complex Management;

b) 'negative result'—the CLTU-THROW-EVENT invocation is rejected by the provider for the reason specified in the `diagnostic` parameter. The provider will not forward the event.

3.11.2.9 `diagnostic`

3.11.2.9.1 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) '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 `event-identifier` specified by the user through this CLTU-THROW-EVENT invocation is not in the set of `event-identifiers` recognized by SLE Complex Management;

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

3.11.2.9.2 If `result` is 'positive result', `diagnostic` shall not be present.

3.11.3 EFFECTS

3.11.3.1 If `result` is 'positive result':

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 CLTU-THROW-EVENT operation is asynchronous with respect to the radiation of CLTUs. If synchronism is required between the CLTU-THROW-EVENT and the radiation of a particular CLTU, it is the user's responsibility to ensure this by means of the radiation control and notification capabilities provided by the service.

3 The effects of the event thrown may consist of preparations made for a change to be triggered by a later CLTU-THROW-EVENT invocation.

4 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'.

3.11.3.2 If `result` is 'negative result', the event shall not be forwarded to SLE Complex Management.

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

3.12 CLTU-PEER-ABORT

3.12.1 PURPOSE

3.12.1.1 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.

3.12.1.2 CLTU-PEER-ABORT shall be an unconfirmed operation.

3.12.1.3 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.

3.12.1.4 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).

3.12.2 INVOCATION AND PARAMETERS

3.12.2.1 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

3.12.2.2 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 identify of the authorized responder for this service instance as specified by service management;

c) 'operational requirement'—the local system had to abort the association to accommodate some other operational need;

   d) 'protocol error'—the local application detected an error in the sequencing of CLTU operations;

   e) 'communications failure'—the communications service on the other side of the gateway was disrupted;

NOTE – This `diagnostic` value is only applicable when the SLE applications are communicating via a gateway.

   f) 'encoding error'—the local application detected an error in the encoding of one or more operation parameters or did not recognize the operation;

   g) 'return timeout'—the local application detected that a return of 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 detected that the period scheduled for provision of service for this service instance has ended, and the initiator has not invoked an UNBIND operation;

   i) 'unsolicited invoke-ID'—the local application received a return with an invoke-ID that does not match with 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. Although it is conceivable that an intruder might use the CLTU-PEER-ABORT operation for a denial-of-service attack, such an attack would require that the intruder succeed in breaking into an existing connection. If an intruder has this capability, 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.

3.12.3 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 CLTUS;

   d) the provider shall stop production, except for a CLTU 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) the provider shall maintain accumulated statistics such as are reported in CLTU-STATUS-REPORT for the duration of 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.

NOTE – In a well-behaved implementation of the protocol, the CLTU-PEER-ABORT operation is truly abortive. The transfer of the SLE-PDU carrying this invocation will not be delayed by other invocations submitted earlier and possibly being buffered by the underlying communications service. This objective needs to be considered when the mapping onto the underlying communications services is decided. For instance, when mapping to TCP-IP, it would be appropriate to use the Urgent Data mechanism to send the CLTU-PEER-ABORT invocation.

4   CLTU PROTOCOL

NOTE – This section specifies the handling of invalid SLE-PDUs and other failures affecting the protocol.

4.1   GENERIC PROTOCOL CHARACTERISTICS

4.1.1   UNEXPECTED PROTOCOL DATA UNIT

If the peer application invokes an operation 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 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 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 a 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 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 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 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 receives a return with an `invoke-ID` parameter value that does not correspond to any invocation for which the return is still pending, the association shall be

aborted by means of 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 Every SLE implementation shall provide that the user and the provider have a consistent view of the health of the communications interface.

4.1.5.3 If an SLE entity determines that communications with the peer SLE entity have been disrupted (e.g., due to a communications service fault), then the SLE entity shall consider that the association with the peer has been aborted.

NOTE –   The exact criteria for determining when communications have been disrupted may depend on the characteristics of the underlying communications service and may be specific to a given implementation. However, every RAF user and provider implementation shall provide for monitoring the health of the communications interface and for ensuring that the user and the provider have a consistent view of the health of the communications interface. If the underlying communications service does not intrinsically provide such a capability, the transmission of a periodic 'heartbeat' indicator or equivalent may need to be implemented.

4.1.5.4 Occurrence of the communications failure event described above shall be referred to as a 'protocol abort'.

4.1.5.5 Following a 'protocol abort' event:

a) the provider shall transition to state 1 ('unbound');

b) the provider shall complete processing a CLTU in the process of being radiated;

c) if the value of the protocol-abort-mode parameter (see 3.1.10) is 'flush', all buffered CLTUs shall be discarded;

d) if the value of the protocol-abort-mode parameter is 'continue', buffered CLTUs shall not be discarded, and CLTUs 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 CLTUs 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 CLTU that remains in the buffer.

4.1.5.6   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.

4.1.5.7   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 3.1.5) 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 3.1.5) 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'.

NOTE –   If authentication fails, the initiator shall behave as specified in 4.1.7. 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 be neither confirmed nor performed.

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.

4.1.8   REDUNDANT BIND INVOCATION

Once the association is established, if a CLTU-BIND invocation for a new association with the same value of the `service-instance-identifier` parameter is received, it shall be rejected with a CLTU-BIND return where the `result` parameter is 'negative result' and the `diagnostic` parameter is 'already bound'. This event shall not affect the association already in place.

NOTE –   Once the association is established, if a CLTU-BIND invocation for a different service instance is received, the performer shall abort the association by invoking the CLTU-PEER-ABORT operation with the `diagnostic` parameter set to 'protocol error'.

4.2   CLTU SERVICE PROVIDER BEHAVIOR

4.2.1   GENERAL REQUIREMENTS

4.2.1.1   The behavior of the 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 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.

4.2.2   STATE TABLE

NOTES

1   The state table specifies operation interactions and state transitions for the service provider.

2   The leftmost 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 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   The actions to be taken and the state transition are considered to be one atomic action. The sequence shown in the table is irrelevant except that SLE-PDUs shall be sent in the sequence stated in the table.

11   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: Behavior of Provider

	Incoming Event	Unbound (State 1)	Ready (State 2)	Active (State 3)
1.	(cltuBindInvocation)	IF "positive result" THEN (+cltuBindReturn) → 2 ELSE (-cltuBindReturn)	IF "same service instance" THEN (-cltuBindReturn ('already bound')) ELSE {peer abort ('protocol error')} → 1	IF "same service instance" THEN (-cltuBindReturn ('already bound')) ELSE {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 CLTU (+cltuTransferDataReturn) ELSE discard CLTU (-cltuTransferDataReturn)

CCSDS 912.1-B-1

CCSDS RECOMMENDATION FOR SLE FORWARD CLTU SERVICE

CCSDS HISTORICAL DOCUMENT

CCSDS HISTORICAL DOCUMENT

Page 4-8

April 2002

	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	{notify('sldu expired') and block}
14.	'production interrupted'	IF "continue" THEN clear CLTU buffer ELSE [ignore]	{notify('production interrupted') and clear}	{notify('production interrupted' and block}

	Incoming Event	Unbound (State 1)	Ready (State 2)	Active (State 3)
15.	'production halted'	IF "continue" THEN clear CLTU buffer ELSE [ignore]	{notify('production halted') and clear}	{notify('production halted') and block}
16.	'production operational'	[ignore]	{notify('production operational')}	{notify('production operational')}
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'	3.7.2.3
'action list not completed'	3.7.2.3
'buffer empty'	3.7.2.3
'cltu radiated'	3.7.2.3
'end of service instance provision period'	3.12.2.2
'event condition evaluated to false'	3.7.2.3
'invalid SLE-PDU'	4.1.2
'not authenticated SLE-PDU'	4.1.7
'production halted'	3.7.2.3
'production interrupted'	3.7.2.3
'production operational'	3.7.2.3
'protocol abort'	4.1.5.4
'reporting-cycle timer expired'	3.8.2.6
'return-timeout-period timer <n> expired'	4.1.3
'sldu expired'	3.7.2.3
'unexpected SLE-PDU'	4.1.1
'unsolicited invoke-ID'	3.12.2.2

Table 4-3: Predicate Definitions

Predicate	Evaluates to TRUE if
"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'
"same service instance"	the `service-instance-identifier` value in the (cltuBindInvocation) PDU denotes a service instance that is already bound

Table 4-4: Boolean Flags

Flag Name	Initial Value
"service instance blocked"	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 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

(This annex **is** part of the Recommendation)

A1   INTRODUCTION

A1.1   GENERAL

This annex defines the data types that are used by the 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 Forward CLTU service.

The data type definitions are presented in six ASN.1 modules.

A1.2   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.

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.

Subsection A2.3 specifies SLE-PDUs related to invocations and returns that are common to SLE transfer service types.

Subsection A2.4 specifies data types specific to the Forward CLTU service. In part, these specific types are derived from types specified in A2.1 by means of subtyping.

Subsection A2.5 specifies all incoming SLE-PDUs of the 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.

Subsection A2.6 specifies in the same way all outgoing SLE-PDUs.

Although A2.2, A2.3, A2.5 and A2.6 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 org(3) standards-producing-organization(112) ccsds(4)
      space-link-extension(3) sle-transfer-services(1)
      forward-cltu-service(2) version-one(1) 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'
,       undefined                                       (5)    --
CLTU
,       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
{       aggregationTimeoutPeriod                        (0)
,       aggregationUsage
        (1)
,       apidList
                (2)
,       bitLockRequired
        (3)
,       bufferSize                                      (4)
,       deliveryMode
             (6)
,       directiveInvocation                             (7)
,       expectedDirectiveIdentification         (8)
,       expectedEventInvocationIdentification   (9)
,       expectedSlduIdentification              (10)
,       fopSlidingWindow                        (11)
,       fopState
                (12)
,       idlePattern
            (13)
,       latencyLimit                                    (15)
,       mapList
                (16)
```

```
,     mapMuxControl
      (17)
,     mapMuxScheme
      (18)
,     maximumFrameLength                                          (19)
,     maximumPacketLength                                         (20)
,     maximumSlduLength
      (21)
,     modulationFrequency
      (22)
,     modulationIndex
      (23)
,     permittedGvcidSet
      (24)
,     plopInEffect
            (25)
,     reportingCycle
            (26)
,     requestedFrameQuality                             (27)
,     requestedGvcid
            (28)
,     returnTimeoutPeriod
      (29)
,     rfAvailable
                  (30)
,     rfAvailableRequired
      (31)
,     segmentHeader
      (32)
,     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 (7 ..
65536))

Time                                                      ::=
      CHOICE
{     ccsdsFormat                                    [0]   TimeCCSDS
}
```

```
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 reference [5]).


END
```

A2.2 SLE-TRANSFER SERVICE BIND TYPES

```
CCSDS-SLE-TRANSFER-SERVICE-BIND-TYPES

{ iso org(3) standards-producing-organization(112) ccsds(4)
      space-link-extension(3) sle-transfer-services(1)
      forward-cltu-service(2) version-one(1) asn1-bind-types(2)
}

DEFINITIONS
IMPLICIT TAGS
::=    BEGIN

EXPORTS     SleBindInvocation
,                      SleBindReturn
,                      SlePeerAbort
,                      SleUnbindInvocation
,                      SleUnbindReturn
;


IMPORTS     Credentials
,                      IntPosShort
      FROM CCSDS-SLE-TRANSFER-SERVICE-COMMON-TYPES
;


-- ===============================================================
-- 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)
}

ATTRIBUTE                                               ::=   CLASS
{     &attributeId                             OBJECT
IDENTIFIER UNIQUE
,     &AttributeType
}

AttributeTypeAndValue                   ::=   SEQUENCE
{ attributeIdentifier                          ATTRIBUTE.&attributeId
                                     ({ServManagementAttributeSet})
,     attributeValue
      ATTRIBUTE.&AttributeType

                ({ServManagementAttributeSet}

                {@attributeIdentifier})
}

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

RDNSequence                             ::=  SEQUENCE OF
        RelativeDistinguishedName

RelativeDistinguishedName    ::=  SET SIZE (1) OF AttributeTypeAndValue

ServiceInstanceIdentifier    ::=  RDNSequence

ServManagementAttributeSet    ATTRIBUTE  ::=  {...}
-- For a real implementation, the above type definition must
-- be replaced by importing the corresponding information object set
-- specification from the applicable service management specification.
-- It is assumed that all attributes are of the type VisibleString
-- See also [F4].

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 org(3) standards-producing-organization(112) ccsds(4)
      space-link-extension(3) sle-transfer-services(1)
      forward-cltu-service(2) version-one(1) asn1-common-pdus(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 CLTU TRANSFER STRUCTURES

```
CCSDS-SLE-TRANSFER-SERVICE-CLTU-STRUCTURES

{ iso org(3) standards-producing-organization(112) ccsds(4)
      space-link-extension(3) sle-transfer-services(1)
      forward-cltu-service(2) version-one(1) asn1-transfer-structures(4)
}

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
,          ForwardDuStatus
,          IntPosLong
,          IntPosShort
,          IntUnsignedLong
,          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
{ parBitLockRequired          [0]  SEQUENCE
{ parameterName                 ParameterName (bitLockRequired)
, parameterValue                INTEGER
  { yes                           (0)
  , no                            (1)
  }
}
, parCltuIdentification       [1]  SEQUENCE
  { parameterName                 ParameterName (expectedSlduIdentification)
  , parameterValue                CltuIdentification
  }
, parEventInvocationIdentification [2]  SEQUENCE
  { parameterName                 ParameterName
                                  (expectedEventInvocationIdentification)
  , parameterValue                EventInvocationId
  }
, parSubcarrierToBitRateRatio [3]  SEQUENCE
  { parameterName                 ParameterName (subcarrierToBitRateRatio)
  , parameterValue                SubcarrierDivisor
  }
, parDeliveryMode             [4]  SEQUENCE
  { parameterName                 ParameterName (deliveryMode)
  , parameterValue                CltuDeliveryMode
  }
, parMaximumCltuLength        [5]  SEQUENCE
  { parameterName                 ParameterName (maximumSlduLength)
  , parameterValue                INTEGER (12 .. 4098)
  }
, parModulationFrequency      [6]  SEQUENCE
  { parameterName                 ParameterName (modulationFrequency)
  , parameterValue                ModulationFrequency
  }
, parModulationIndex          [7]  SEQUENCE
  { parameterName                 ParameterName (modulationIndex)
  , parameterValue                ModulationIndex
  }
, parPlopInEffect             [8]  SEQUENCE
  { parameterName                 ParameterName (plopInEffect)
  , parameterValue                INTEGER
{ plop1                         (0)
, plop2                         (1)
  }
}
, parReportingCycle           [9]  SEQUENCE
{ parameterName                 ParameterName (reportingCycle)
, parameterValue                CurrentReportingCycle
}
, parReturnTimeout            [10] SEQUENCE
  { parameterName                 ParameterName (returnTimeoutPeriod)
  , parameterValue                TimeoutPeriod
  }
, parRfAvailableRequired      [11] SEQUENCE
{ parameterName                 ParameterName (rfAvailableRequired)
, 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
, productionConfigured          [9]  NULL
}


CltuParameterName               ::=  ParameterName
( bitLockRequired
| deliveryMode
| expectedSlduIdentification
| expectedEventInvocationIdentification
| maximumSlduLength
| modulationFrequency
| modulationIndex
| plopInEffect
| reportingCycle
| returnTimeoutPeriod
| rfAvailableRequired
| subcarrierToBitRateRatio
)

CltuStatus                      ::=  ForwardDuStatus
( radiated
| expired
| interrupted
| productionStarted
| undefined
)

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)
  }
}

EventInvocationId                 ::=  IntUnsignedLong


-- 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)
}

END
```

A2.5   CLTU INCOMING PDUS

```
CCSDS-SLE-TRANSFER-SERVICE-CLTU-INCOMING-PDUS

{ iso org(3) standards-producing-organization(112) ccsds(4)
      space-link-extension(3) sle-transfer-services(1)
      forward-cltu-service(2) version-one(1) asn1-incoming-pdus(5)
}

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                  [1]    SleStopInvocation

, cltuTransferDataInvocation          [2]    CltuTransferDataInvocation

, cltuScheduleStatusReportInvocation  [3]
                                      SleScheduleStatusReportInvocation

, cltuGetParameterInvocation          [4]    CltuGetParameterInvocation
```

```
, cltuThrowEventInvocation              [5]     CltuThrowEventInvocation

, 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
```

A2.6 CLTU OUTGOING PDUS

CCSDS-SLE-TRANSFER-SERVICE-CLTU-OUTGOING-PDUS

```
{ iso org(3) standards-producing-organization(112) ccsds(4)
      space-link-extension(3) sle-transfer-services(1)
      forward-cltu-service(2) version-one(1) asn1-outgoing-pdus(6)
}

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              [0]    CltuStartReturn

, cltuStopReturn               [1]    SleAcknowledgement

, cltuTransferDataReturn       [2]    CltuTransferDataReturn

, cltuAsyncNotifyInvocation    [3]    CltuAsyncNotifyInvocation

, cltuScheduleStatusReportReturn [4]  SleScheduleStatusReportReturn

, cltuStatusReportInvocation   [5]    CltuStatusReportInvocation

, cltuGetParameterReturn       [7]    CltuGetParameterReturn

, cltuThrowEventReturn          [8]    CltuThrowEventReturn

, 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
   { radiationStartTime           Time
   , radiationStopTime            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

INDEX TO DEFINITIONS

(This annex **is not** part of the Recommendation)

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	1.6.1.7.2
Carrier Modulation Mode (CMM)	reference [2]
CLTU channel	reference [1]
Command Link Control Word (CLCW)	reference [3]
Command Link Transmission Unit (CLTU)	reference [2]
communications service	1.6.1.7.3
confirmed operation	1.6.1.7.4
distinguished name (DN)	reference [8]
flow control	reference [7]
Forward CLTU service	reference [1]
idle sequence	reference [2]
initiator	1.6.1.7.5
invocation	1.6.1.7.6
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	1.6.1.7.7
performance	1.6.1.7.8
performer	reference [1]
physical channel	reference [1]
Physical Layer Operations Procedure (PLOP)	reference [2]
port identifier	1.6.1.7.9
real system	reference [7]
relative distinguished name (RDN)	reference [8]
responder	1.6.1.7.10
return	1.6.1.7.10
service access point (SAP)	reference [7]
service instance provision period	1.6.1.7.12
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 protocol data unit (SLE-PDU)	reference [1]
SLE service data unit (SLE-SDU)	reference [1]
SLE service package	reference [1]
SLE System	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 session	reference [1]
unconfirmed operation	1.6.1.7.13

ANNEX C

ACRONYMS

(This annex **is not** part of the Recommendation)

This annex expands the acronyms used throughout this Recommendation.

ASDC	Abstract Service Definition Conventions
ASN.1	Abstract Syntax Notation One
C	Conditional
CCSDS	Consultative Committee for Space Data Systems
CDS	CCSDS Day Segmented (time code)
CLCW	Command Link Control Word
CLTU	Command Link Transmission Unit
CLTU-	Forward CLTU service (operation prefix)
CMM	Carrier Modulation Mode
COP	Command Operation Procedure
DN	Distinguished Name
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
RDN	Relative Distinguished Name
RF	Radio Frequency
SAP	Service Access Point
SDU	Service Data Unit
SLE	Space Link Extension
SLE-FG	SLE Functional Group
SLE-PDU	SLE Protocol Data Unit
SLE-SDU	SLE Service Data Unit
SL-DU	Space Link 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

ANNEX D

CONFORMANCE OPTIONS MATRIX

(This annex **is** part of the Recommendation)

This annex provides the Conformance Matrix for implementations of SLE Forward CLTU Service systems. An SLE Forward CLTU Service system will be considered to be 'conformant' if the mandatory elements identified in the matrix are implemented as described in this Recommendation.

Table D-1: Conformance Matrix for CLTU Service (Operations)

Service Operations and their Parameters	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 D-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 3.7.2.3)	This parameter is optional; its purpose is only to document the behavior of the provider.
Provider Buffering Requirements (see 3.1.11)	Mandatory minimum size is 1024 maximum sized CLTUs.
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.

ANNEX E

INFORMATIVE REFERENCES

(This annex **is not** part of the Recommendation)

[E1] *Procedures Manual for the Consultative Committee for Space Data Systems*. CCSDS A00.0-Y-7. Yellow Book. Issue 7. Washington, D.C.: CCSDS, November 1996.

[E2] *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.

[E3] *Standard Terminology, Conventions, and Methodology (TCM) for Defining Data Services*. Report Concerning Space Data Systems Standards, CCSDS 910.2-G-1. Green Book. Issue 1. Washington, D.C.: CCSDS, November 1994.

[E4] *Cross Support Concept — Part 1: Space Link Extension Services*. Report Concerning Space Data System Standards, CCSDS 910.3-G-2. Green Book. Issue 2. Washington, D.C.: CCSDS, April 2002.

[E5] *Space Link Extension—Service Management Specification*. Draft Recommendation for Space Data System Standards, CCSDS 910.5-R-2. Red Book. Issue 2. Washington, D.C.: CCSDS, September 2001.

[E6] *Space Link Extension—Service Management—Space Link Physical Layer Managed Object Specification*. Draft Recommendation for Space Data System Standards, CCSDS 910.7-R-1. Red Book. Issue 1. Washington, D.C.: CCSDS, October 2001.

ANNEX F

THROW EVENT DEFINITIONS

(This annex **is not** part of the Recommendation)

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 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 F-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 G

PRODUCTION STATUS

(This annex **is not** part of the Recommendation)

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 Forward CLTU transfer service.

The possible transitions of production status of the Forward CLTU service are shown in figure G-1 below.
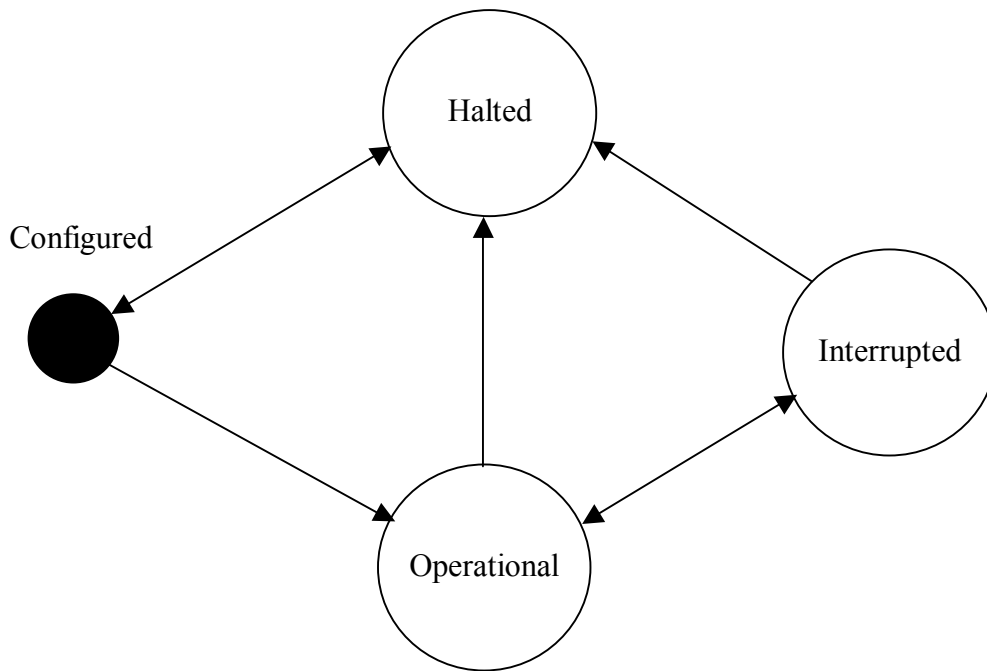

Figure G-1: CLTU Production Status Transitions

Changes in production status occur for reasons outlined in table G-1.

Table G-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-lock may be required before `production-status` can change to 'operational'. See table 3-11.	'production operational'
Operational	Interrupted	Occurrence of a production fault detected by the provider.	'production interrupted'; but see NOTE
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'.	'production configured'

NOTE – This Recommendation does not require that a production interruption be notified immediately to the user (see 3.7.2.3). 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 table 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 Forward CLTU transfer service invocations is specified in 0 through 3.12 and in table 4-1. The effects are summarized in table G-2 for convenient reference.

Table G-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', only after the provider notifies the user that a production interruption has occurred are CLTU-TRANSFER-DATA invocations rejected. See 3.7.2.3.