ICAO

Doc 9880

Technical Specifications for ATN using ISO/OSI Standards and Protocols

Second edition, 2016

Part I – Air-Ground Applications

Approved by and published under the authority of the Secretary General

INTERNATIONAL CIVIL AVIATION ORGANIZATION

Doc 9880
AN/466

Manual on Detailed Technical Specifications for the Aeronautical Telecommunication Network (ATN) using ISO/OSI Standards and Protocols

Part I — Air-Ground Applications

Second Edition — 2016

International Civil Aviation Organization

Published in English only by the
INTERNATIONAL CIVIL AVIATION ORGANIZATION
999 Robert-Bourassa Boulevard, Montréal, Quebec, Canada H3C 5H7

For ordering information and for a complete listing of sales agents
and booksellers, please go to the ICAO website at www.icao.int

Doc 9880, ***Manual on Detailed Technical Specifications***
for the Aeronautical Telecommunication Network (ATN)
using ISO/OSI Standards and Protocols
Part I, ***Air-Ground Applications***
Order Number: 9880PI
ISBN 978-92-9258-140-4

AMENDMENTS

Amendments are announced in the supplements to the *Catalogue of ICAO Publications;* the Catalogue and its supplements are available on the ICAO website at www.icao.int. The space below is provided to keep a record of such amendments.

RECORD OF AMENDMENTS AND CORRIGENDA

AMENDMENTS		
No.	Date	Entered by

CORRIGENDA		
No.	Date	Entered by

TABLE OF CONTENTS

FOREWORD[1]

This manual contains the detailed technical specifications for the ATN based on relevant standards and protocols established for open systems interconnection (OSI) by the International Organization for Standardization (ISO) and the Telecommunication Standardization Sector of the International Telecommunication Union (ITU-T). A separate manual, the *Manual on the Aeronautical Telecommunication Network (ATN) using Internet Protocol Suite (IPS) Standards and Protocols* (Doc 9896), addresses detailed technical specifications for the ATN based on standards developed for the IPS by the Internet Society (ISOC). Standards and Recommended Practices (SARPs) for the ATN/IPS are contained in Annex 10 — *Aeronautical Telecommunications,* Volume III — *Communication Systems*. Where necessary and to avoid duplication of material, Doc 9896 refers to this manual.

Editorial practices in this document are as follows:

- The detailed technical specifications in this manual that include the operative verb “shall” are essential to be implemented to secure proper operation of the ATN.

- The detailed technical specifications in this manual that include the operative verb “should” are recommended for implementation in the ATN. However, particular implementations may not require this specification to be implemented.

- The detailed technical specifications in this manual that include the operative verb “may” are optional.

This manual is published in the following parts:

Part I: Air-Ground Applications (replaces Doc 9705, Sub-volume II)

Part II: Ground-Ground Applications — Air Traffic Services Message Handling Services (ATSMHS) (replaces Doc 9705, Sub-volume III)

Part III: Upper Layer Communications Service (ULCS) and Internet Communications Service (ICS) (replaces Doc 9705, Sub-volumes IV and V)

Part IV: Directory Services, Security and Identifier Registration (replaces Doc 9705, Sub-volumes I, VII, VIII and IX).

[1] The first edition of this manual amended and replaced the third edition of the *Manual of Technical Provisions for the Aeronautical Telecommunication Network (ATN) (*Doc 9705).

Structure of Part I:

This part of Doc 9880 contains technical provisions for air-ground applications. It is structured as follows:

Chapter 1: INTRODUCTION

Chapter 2: CONTEXT MANAGEMENT APPLICATION

Chapter 3: CONTROLLER-PILOT DATA LINK COMMUNICATIONS APPLICATION

Chapter 4: AUTOMATIC DEPENDENT SURVEILLANCE — CONTRACT (ADS-C)

Chapter 5: ATN MESSAGE INTEGRITY CHECK ALGORITHM

ACRONYMS AND ABBREVIATIONS

ACP	Aeronautical Communications Panel
ADM	Administration identifier
ADS	Automatic dependent surveillance
ADS-C	Automatic dependent surveillance — contract
AE	Application entity
AFI	Authority and format identifier
APDU	Application protocol data unit
ARS	Administration region selector
ASE	Application service element
ASN.1	Abstract Syntax Notation One
ATN	Aeronautical telecommunication network
ATS	Air traffic services
ATSC	Air traffic service communications
CF	Control function
CM	Context management
Cnf	Confirmation
CPDLC	Controller-pilot data link communications
DLIC	Data link initiation capability
DS	Dialogue service
DSC	Downstream clearance
FU	Functional unit
IC	Integrity check
ICAO	International Civil Aviation Organization
ICS	Internet communications service
IDI	Initial domain identifier
IDP	Initial domain part
IEC	International Electrotechnical Commission
IFR	Instrument flight rules
Ind	Indication
IPS	Internet Protocol Suite
ISO	International Organization for Standardization
ITU-T	International Telecommunication Union — Telecommunication Standardization Sector
Km	Kilometre
Nm	Nautical mile
NSEL	Network selector
OID	Object identifier
OSI	Open systems interconnection
P/OICS	Protocol/Operational Implementation Conformance Statements
PDU	Protocol data unit
PER	Packed Encoding Rules
QOS	Quality of Service
RDF	Routing domain format
RDP	Router domain part
Req	Request
RER	Residual error rate
RFC	Request for comments

Rsp	Response
SARPs	Standards and Recommended Practices
TSAP	Transport service access point
TSEL	TSAP selector
ULCS	Upper layer communications service
UTC	Coordinated universal time
VER	Version
VMC	Visual meteorological conditions

DEFINITIONS

Abstract service interface. The abstract interface between the application entity (AE) and the application-user.

Abstract Syntax Notation One (ASN.1). Abstract Syntax Notation One is defined in ISO/IEC 8824-1. The purpose of this notation is to enable data types to be defined, and values of those types specified, without determining their actual representation (encoding) for transfer by protocols.

Addressing plan. A plan that provides common address syntax and management of global addresses for the unambiguous identification of all end and intermediate systems in accordance with the rules prescribed in ISO/IEC 7498-3 and ISO/IEC TR 10730.

Aeronautical administrative communications (AAC). Communications necessary for the exchange of aeronautical administrative messages.

Aeronautical administrative messages. Messages regarding the operation or maintenance of facilities provided for the safety or regularity of aircraft operation. Messages concerning the functioning of the ATN and messages exchanged between government civil aviation authorities relating to aeronautical services.

Aeronautical operational control (AOC). Communication required for the exercise of authority over the initiation, continuation, diversion or termination of flight for safety, regularity and efficiency reasons.

Aeronautical passenger communication (APC). Communication relating to the non-safety voice and data services to passengers and crew members for personal communication.

Air application service element (air-ASE). An abstract part of the aircraft system that performs the communication-related functions of the application.

Air-ground application. An application that has one peer application on an aircraft and its other peer application on the ground. An air-ground application may require the use of ground-ground subnetworks.

Air traffic control (ATC) clearance. Authorization for an aircraft to proceed under conditions specified by an air traffic control unit.

Note 1.— For convenience, the term "air traffic control clearance" is frequently abbreviated to "clearance" when used in appropriate contexts.

Note 2.— The abbreviated term "clearance" may be prefixed by the words "taxi", take-off", "departure", "en-route", "approach" or "landing" to indicate the particular portion of flight to which the air traffic control clearance relates.

Air traffic control (ATC) instruction. Directives issued by air traffic control for the purpose of requiring a pilot to take specific action.

Air traffic control (ATC) service. A service provided for the purpose of:

a) preventing collisions:

1) between aircraft; and

2) on the manoeuvring area between aircraft and obstructions; and

b) expediting and maintaining an orderly flow of air traffic.

Air traffic services (ATS). A generic term meaning variously, flight information service, alerting service, air traffic advisory service, air traffic control service (area control service, approach control service or aerodrome control service).

Air user (air-user). The abstract part of the aircraft system that performs the non-communication-related functions of the application.

Aircraft address. A unique combination of twenty-four bits available for assignment to an aircraft for the purpose of air-ground communications, navigation and surveillance.

Aircraft flight identification. A group of letters, figures or a combination thereof which is either identical to, or the coded equivalent of, the aircraft call sign to be used in air-ground communication and which is used to identify the aircraft in ground-ground air traffic services communication.

Application. The ultimate use of an information system, as distinguished from the system itself.

Application entity (AE). Part of an application process that is concerned with communications within the OSI environment. The aspects of an application process that need to be taken into account for the purposes of OSI are represented by one or more AEs.

Application entity service interface. The interface between the application-users and the application service provider.

Application entity title. An unambiguous name for an application entity.

Application process (AP). A set of resources, including processing resources, within a real open system which may be used to perform a particular information processing activity.

Application protocol data unit (APDU). An application protocol data unit is an (N) PDU, where N refers to the application layer. An APDU is the basic unit of information exchanged between the airborne application and the ground application.

Application service. The abstract interface between the (N) service and the (N) service user, where N refers to the application layer; thus it is the boundary between the AE and the application user.

Application service element (ASE). The element in the communication system that executes the application specific protocol. In other words, it processes the application specific service primitive sequencing actions, message creation, timer management, and error and exception handling. The application's ASE interfaces only with the application's control function.

Application service element (ASE) service interface. The abstract interface through which the ASE service is accessed.

Note.— In version 1 of the ADS-C application, the ADS-ASE service interface coincides with the ADS-AE abstract service interface.

Application-user. That abstract part of the aircraft or ground system that performs the non-communication-related functions of the application.

ATN application. Refers to an application that is designed to operate over ATN communication services.

ATS communication (ATSC). Communication related to air traffic services including air traffic control, aeronautical and meteorological information, position reporting and services related to safety and regularity of flight. This communication involves one or more air traffic service administrations. This term is used for purposes of address administration.

ATS message. A unit of user-data, coded in binary form, which is conveyed from an originator of the data to one or more recipients of the data. It is possible to associate a unique message identifier and a priority with each ATS message.

ATS unit (ATSU). A generic term meaning variously, air traffic control unit, flight information centre or air traffic services reporting office.

ATSC class. The ATSC class parameter enables the ATSC-user to specify the quality of service expected for the offered data. The ATSC class value is specified in terms of ATN end-to-end transit delay at 95 per cent probability.

Automatic dependent surveillance (ADS). A surveillance technique in which aircraft automatically provide, via a data link, data derived from on-board navigation and position-fixing systems, including aircraft identification, four-dimensional position and additional data as appropriate.

Automatic dependent surveillance – Contract (ADS-C) application. An ATN application that provides ADS data from the aircraft to the ATS unit(s) for surveillance purposes.

Automatic dependent surveillance — contract (ADS-C). A means by which the terms of an ADS-C agreement will be exchanged between the ground system and the aircraft, via a data link, specifying under what conditions ADS-C reports would be initiated, and what data would be contained in the reports.

Note.— The abbreviated term "ADS contract" is commonly used to refer to ADS event contract, ADS demand contract, or an ADS periodic contract.

Context management (CM) application. An ATN application that provides a logon service allowing initial aircraft introduction into the ATN and a directory of all other data link applications on the aircraft. It also includes functionality to forward addresses between ATS units.

Note.— Context management is a recognized OSI presentation layer term. The OSI use and the ATN use have nothing in common.

Control function (CF). That abstract part of the AE that performs the mapping between the ASE service primitives, the association control service element (ACSE) service primitives and other elements within the application entity.

Controller pilot data link communication (CPDLC). A means of communication between controller and pilot, using data link for ATC communications.

Controller pilot data link communication (CPDLC) application. An ATN application that provides a means of ATC data communication between controlling, receiving or downstream ATS units and the aircraft, using air-ground and ground-ground subnetworks, and which is consistent with the ICAO phraseology for the current ATC voice communication.

Current data authority. The designated ground system through which a CPDLC dialogue between a pilot and a controller currently responsible for the flight is permitted to take place.

Data authority. A ground system that provides for the establishment and maintenance of a CPDLC transport connection with an aircraft. The transfer of communication from the current data authority to the next data authority is prepared prior to the actual data link switch by designating a next data authority in a specific CPDLC message.

Data link initiation capability (DLIC). A data link application that provides the ability to exchange addresses, names and version numbers necessary to initiate data link applications.

Dialogue service (DS). The lower service boundary of an ASE; the service allows two ASEs to communicate, e.g. a CM ground-ASE to communicate with a CM air-ASE.

Directory. A facility that supports on request the retrieval of address information or the resolution of application names.

Domain. A set of end systems and intermediate systems that operate according to the same routing procedures and that is wholly contained within a single administrative domain.

Downstream clearance (DSC). Specific clearance request by an aircraft to an ATSU that is not the controlling ATSU. The initiation of the DSC service can only be initiated by an aircraft.

Downstream data authority. A designated ground system, different from the current data authority through which the pilot can contact an appropriate ATC unit for the purposes of receiving a downstream clearance.

End system (ES). A system that contains the OSI seven layers and contains one or more end-user application processes.

End-to-end. Pertaining or relating to an entire communication path, typically from (1) the interface between the information source and the communication system at the transmitting end to (2) the interface between the communication system and the information user or processor or application at the receiving end.

End-user. An ultimate source and/or consumer of information.

Entity. An active element in any layer which can be either a software entity (such as a process) or a hardware entity (such as an intelligent I/O chip).

Flight information region (FIR). An airspace of defined dimensions within which flight information service and alerting service are provided.

Flight plan. Specified information provided to air traffic services units, relative to an intended flight or portion of a flight of an aircraft.

Note.— Specifications for flight plans are contained in Annex 2 — Rules of the Air. A model Flight Plan Form is contained in the Procedures for Air Navigation Services — Air Traffic Management (PANS-ATM, Doc 4444), Appendix 2.

General communication. A category of communications that includes APC, public correspondence and other non-operational and non-administrative communication.

Ground application service element (ground-ASE). An abstract part of the ground system that performs the communication-related functions of the application.

Ground forwarding function. The capability for a ground system to forward a CPDLC message to another ground system via a CPDLC message with an indication of success, failure or non-support from the receiving ground system. This function may be invoked by the current data authority in order to avoid retransmission of a request by an aircraft by forwarding the information to the next data authority. The downstream data authority may use this function in order to relay a message to the current data authority which then performs the actual transmission to the aircraft.

Ground user (ground-user). The abstract part of the ground system that performs the non-communication-related functions of the application.

International Alphabet No. 5 (IA5). International Alphabet Number 5 defined by ITU-T.

Note.— ATN uses the "6-bit ASCII" subset of 1A5, as used in SSR Mode S specifications.

Internet communications service (ICS). The Internet communications service is an internetwork architecture which allows ground, air-to-ground and avionics data subnetworks to interoperate by adopting common interface services and protocols based on the ISO/OSI reference model.

Internetwork. A set of interconnected, logically independent heterogeneous subnetworks. The constituent subnetworks are usually administrated separately and may employ different transmission media.

Internetworking protocol (IP). A protocol that performs the basic end-to-end mechanism for the transfer of data packets between network entities. In the ATN Internet communications service, the ISO/IEC 8473 internetwork protocol is used.

Interoperability. Describes the ability of the ATN to provide, as a minimum, a transparent data transfer service between end systems even though the ATN comprises various ground, air-to-ground and avionics subnetworks. The ability to interoperate between end systems can be extended to include commonality of upper layer protocols.

Long transport service access point (TSAP). Composed of the router domain part (RDP) and the short TSAP.

Lower layers. The physical, data link, network and transport layers of the OSI reference model.

Managed object. Data processing and data communication resources that may be managed through the use of the OSI management protocol.

Message. Basic unit of user information exchanged between an airborne application and its ground counterpart or between two ground applications. Messages are passed in one or more data blocks from one end-user to another through different subnetworks.

Message element. A component of a message used to define the context of the information exchanged.

Message element identifier. The ASN.1 tag of the ATCUplinkMsgElementId or the ATCDownlinkMsgElementId.

Mobile subnetwork. A subnetwork connecting a mobile system with another system not resident in the same mobile platform. These subnetworks tend to use free-radiating media (e.g. VHF/UHF radio, D-band satellite or D-band secondary surveillance radar) rather than contained media (e.g. wire or coaxial cable); thus they exhibit broadcast capabilities in the truest sense.

Naming plan. A plan that provides common naming conventions and designations for the unambiguous identification of all end and intermediate systems in accordance with the rules prescribed in ISO/IEC 7498-3, ISO/IEC TR 10730 and ISO/IEC 9545.

Network addressing domain. A subset of the global addressing domain consisting of all the NSAP addresses allocated by one or more addressing authorities.

Network entity (NE). A functional portion of an internetwork router or host computer that is responsible for the operation of internetwork data transfer, routing information exchange and network layer management protocols.

Network service access point (NSAP). Point within the ISO protocol architecture at which global end-users may be uniquely addressed on an end-to-end basis.

Network service access point (NSAP) address. A hierarchically organized global address supporting international, geographical and telephony-oriented formats by way of an address format identifier located within the protocol header. Although the top level of the NSAP address hierarchy is internationally administered by ISO, subordinate address domains are administered by appropriate local organizations.

Network service access point (NSAP) address prefix. Used to identify groups of systems that reside in a given routing domain or confederation. An NSAP prefix may have a length that is either smaller than or the same size as the base NSAP address.

Network topology map. Provides an overall view of the global network connectivity and is used in path computations by the operative routing algorithm.

Next data authority. The ground system so designated by the current data authority through which an onward transfer of communications and control can take place.

Open systems interconnection (OSI) protocol architecture. A set of protocols used to implement the OSI reference model.

Open systems interconnection (OSI) reference model. A model providing a standard approach to network design introducing modularity by dividing the complex set of functions into seven more manageable, self-contained, functional layers. By convention these are usually depicted as a vertical stack.

Note.— The OSI reference model is defined by ISO/IEC 7498-1.

Operational requirement. A statement of the operational attributes of a system needed for the effective and/or efficient provision of air traffic services to users.

Packed Encoding Rules (PER). Encoding rules, as defined in ISO/IEC 8825-2, that have been designed to minimize the number of bits transmitted.

Performance requirements. Requirements that define a function's characteristics, such as reliability, availability, response time, processing delay, integrity, that are necessary to meet the operational requirements for a specific application of the function.

Presentation data value (PDV). The unit of information specified in an abstract syntax, which is transferred by the OSI presentation service (ISO/IEC 8822).

Priority (P). The relative importance of a particular protocol data unit (PDU) relative to other PDUs in transit and used to allocate resources which become scarce during the transfer process.

Profile. Defines implementation conformance constraints on a set of reference specifications.

Protocol. A set of rules and formats (semantic and syntactic) that determines the communication behaviour between peer entities in the performance of functions at that layer.

Protocol data unit (PDU). (1) A unit of data transferred between peer entities within a protocol layer consisting of protocol control information and higher layer user data (i.e. service data units). (2) A unit of data specified in an (N) protocol and consisting of (N) protocol control information and possibly (N) user data, where N indicates the layer.

Protocol implementation conformance statement (PICS). A protocol implementation conformance statement enables conformance testing of protocols. As recommended by ISO/IEC 9646-2, PICS pro forma, tailored to ATN context, have been developed as ATN profile requirements list (APRLs) to provide an effective basis for conformance testing of implementations.

Quality of service (QOS). The information relating to data transfer characteristics used by various communication protocols to achieve various levels of performance for network users.

Runway visual range (RVR). The range over which the pilot of an aircraft on the centre line of a runway can see the runway surface markings or the lights delineating the runway or identifying its centre line.

Secondary surveillance radar (SSR). A surveillance radar system which uses transmitters/receivers (interrogators) and transponders.

Security label. May indicate requirements for protection of a protocol data unit (PDU) and provide information used by network layer access control functions.

Service data unit (SDU). A unit of data transferred between adjacent layer entities, which is encapsulated within a protocol data unit (PDU) for transfer to a peer layer.

Service primitive. A function of an application service element (ASE) that is not broken down further into subfunctions and is presented as part of the abstract service interface (i.e. request, indication, response or confirmation).

Service provider. An entity at a layer that provides services to the layer above. These services are provided at service access points through the use of service primitives.

Short transport service access point (TSAP). Composed of the administrative region selector (ARS), (Optional), the location identifier (LOC), the system identifier (SYS), the network selector (SEL) and the transport selector (TSAP selector).

System application. An application supports the operation of the air-ground applications, ground-ground applications or communication services. A system application can take the form of either an air-ground application or a ground-ground application.

System level requirement. The system level requirement is a high-level technical requirement that has been derived from operational requirements, technological constraints and regulatory constraints (administrative and institutional). The system level requirements are the basis for the functional requirements and lower-level requirements.

Traffic category. A subdivision of the operational communication traffic type used to distinguish between ATS communication and aeronautical operational control (AOC).

Traffic type. A means used to distinguish different types of message traffic for the purposes of establishing

communication paths to support operational and legal requirements. There are four traffic types:

a) the operational communication traffic type, which is subdivided into the following two categories representing safety and regularity of flight communication:

   1) ATS communication; and

   2) aeronautical operational control;

b) administrative communication, representing non-safety and regularity of flight communication sent by aircraft operating agencies and ATS administrations;

c) general communication, representing APC, public correspondence and other non-operational and non-administrative communication; and

d) systems management communication, representing systems management information that is critical for support of network operations.

Note.— The differentiation of traffic types is required because different data traffic may have different access to subnetworks. The traffic type is conveyed in the ATN security label of ISO/IEC 8473 and ISO/IEC 10747. It is used to qualify connectionless mode network protocol (CLNP) data packets and (inter-domain) routes according to the class of traffic that they carry. Based on this qualification, access of subnetworks is controlled by the ATN Internet communications service.

User requirements. Requirements that are allocated to the user to ensure the interoperability of the communication services and application entities.

Chapter 1

INTRODUCTION

1.1 OVERVIEW

1.1.1 General

Included in this part of Doc 9880 are technical provisions for:

a) context management (CM);

b) controller-pilot data link communications (CPDLC);

c) automatic dependent surveillance — contract (ADS-C); and

d) ATN message integrity check algorithm.

1.1.2 Context management (CM)

1.1.2.1 The CM application allows addressing capability for data link applications. It provides the capability to establish a logon between peer ATS ground systems and ATS ground and aircraft systems. Once an appropriate connection is established, CM provides data link application information, the capability to logon to another ground system and the capability to update logon information.

1.1.2.2 The CM application needs to be considered in the context of the guidance material for the data link initiation capability (DLIC), as provided in the *Manual of Air Traffic Services Data Link Applications* (Doc 9694, Part II). The DLIC process supports addressing requirements for ATS such as CPDLC and ADS-C.

1.1.3 Controller-pilot data link communications (CPDLC)

1.1.3.1 The CPDLC application allows data link communications between controllers and pilots.

1.1.3.2 The CPDLC application provides the capability to establish, manage and terminate CPDLC dialogues between ATS ground and aircraft system peers. Once a dialogue is established, CPDLC provides for message exchange between the controller and the pilot. The CPDLC application supports a variety of operational data link services, depending upon the CPDLC message and operational procedures applied.

1.1.3.3 Also provided by the CPDLC application is the capability to establish, manage and terminate CPDLC dialogues between two ATC ground system peers for the purpose of ground-ground forwarding of a CPDLC message.

1.1.3.4 The CPDLC application includes an end-to-end protection of message integrity by an application level integrity check that also provides assurance of correct delivery.

1.1.3.5 Depending upon the message subset and operational procedures applied, the CPDLC application supports a variety of operational data link services.

1.1.4 Automatic dependent surveillance — contract (ADS-C)

1.1.4.1 The ADS-C application allows ground systems to retrieve aircraft-generated data.

1.1.4.2 The ADS-C application provides the capability to establish, manage and terminate demand, periodic and event ADS-C contracts between ATS ground and aircraft system peers.

1.1.4.3 The ADS-C application includes an end-to-end protection of message integrity by an application level integrity check.

1.1.5 ATN message integrity check algorithm

The integrity check algorithm may optionally be invoked by the ATN application-user specification to provide a proof of message integrity, as well as proof of delivery to an intended recipient, etc. The default ATN message checksum algorithm is discussed in Chapter 5.

1.2 DIALOGUE SERVICE (DS)

Throughout Part I, references to dialogue services (D-START, D-DATA, D-END, D-ABORT, D-P-ABORT) are references to the DS specified in Part III of this manual.

Chapter 2

CONTEXT MANAGEMENT APPLICATION

2.1 INTRODUCTION

2.1.1 Overview

This chapter is designed as follows:

- Section 2.1 — INTRODUCTION: Besides outlining the material covered in Chapter 2, this section provides a description of the functions of the CM application.

- Section 2.2 — GENERAL REQUIREMENTS: This section focuses on the CM-ASE version number and error-processing requirements.

- Section 2.3 — THE ABSTRACT SERVICE: The description of the abstract service provided by the CM application service element (CM-ASE) is contained in this section.

- Section 2.4 — FORMAL DEFINITIONS OF MESSAGES: This section contains the formal definitions of messages exchanged by CM-ASEs using Abstract Syntax Notation One (ASN.1).

- Section 2.5 — PROTOCOL DEFINITION: This section describes the exchanges of messages allowed by the CM protocol, as well as time constraints. It also provides CM-ASE protocol descriptions and state tables.

- Section 2.6 — COMMUNICATION REQUIREMENTS: The requirements that the CM application imposes on the underlying communication system are covered in this section.

- Section 2.7 — CM-USER REQUIREMENTS: This section contains requirements imposed on the user of the CM-ASE service.

- Section 2.8 — SUBSETTING RULES: The conformance requirements that all implementations of the CM protocol obey are covered in this section.

2.1.2 Description of the functions of the CM application

2.1.2.1 Logon function

2.1.2.1.1 The logon function provides a means for an aircraft and a ground system to exchange ATN application information. The logon function has two variants — secure and unsecure. The security mechanism is assumed to be able to be inititated by indication from the dialog service.

Note. – The specific security mechanisms that may provide the security functionality are out of scope for this document.

2.1.2.1.2 The logon function can only be air-initiated. The aircraft system can use the logon function to provide an application name and version number for each air-only-initiated application, and an application name, address and version number for each application that the aircraft wishes to use that can be ground-initiated, along with flight plan information as required by the ground system. Additionally, if the aircraft can provide the *Security Requirements* parameter. This parameter will have different values depending on whether or not the logon will be requesting secure or unsecure services. The user data in the CM-logon request is the same for both secure and unsecure services. In response, the ground provides an application name for each ground-only-initiated requested application, and an application name, address and version number for each requested application that can be air-initiated and that the ground can support. If a secure logon service is requested and is able to be supported, the response to the CM-logon request will include the Security Requirements parameter and may result in additional security information processing. However, a successful secure logon function does not guarantee that other applications (such as ADS-C and CPDLC) will be able to be run in a secure mode — that will be dependent on local security policy.

2.1.2.1.3 Up to a maximum of 256 applications can be supported by the logon function.

2.1.2.1.4 Each time a logon is accomplished between a given aircraft and a ground system, the latest exchanged information replaces any previous information for each indicated application.

2.1.2.1.5 The CM-logon request message provides required flight plan information, the aircraft's CM application name and address, and information for each application for which data link services are desired. For each application that can be ground-initiated, the aircraft must provide the application name, version number and address. For each application that is only air-initiated, the aircraft must provide the application name and version number.

2.1.2.1.6 The CM-logon response message provides information for the logon-indicated air-initiated applications. For each desired air-initiated application, the ground provides the application name, version number and address. The CM-logon service also contains an indication of whether or not security is desired for the exchange.

Note. – The specific security mechanisms that may provide the security functionality are out of scope for this document.

2.1.2.3 Update function

2.1.2.3.1 The update function provides a method for the ground system to update application information. This function assumes that the logon function has been accomplished.

2.1.2.3.2 The CM-update service also contains an indication of whether or not security is desired for the exchange.

Note. – The specific security mechanisms that may provide the security functionality are out of scope of this document.

2.1.2.3.3 The CM update message can provide updated ground information for up to 256 applications. For each updated application, the ground provides the application's name, version number and address.

2.1.2.4 Contact function

2.1.2.4.1 The contact function provides a method for the ground system to request the aircraft system to initiate the logon function with a designated ground system. It is expected that the contact function will only be used when ground connectivity is not available between respective ground system applications. This function assumes that the logon function has been accomplished with the ground system initiating the contact function. The ground initiates this function with a contact request specifying the ground system with which to logon. The aircraft initiates a logon as specified in 2.1.2.1 and indicates the success or lack thereof of the logon. The contact function may be secure or unsecure. There are no differences in the user data provided by the CM-ground-user or CM-air-user; only the provision of the *Security*

Requirements parameter is different.

Note. – The specific security mechanisms that may provide the security functionality are out of scope of this document.

2.1.2.4.2 The CM contact request message provides the ground system CM application address that the initiating ground system is requesting the aircraft to logon with.

2.1.2.4.3 The CM contact response message provides the information indicating whether or not the requested contact was successful.

2.1.2.5 Forward function

2.1.2.5.1 The forward function provides a method for a ground system to forward aircraft information received from the CM-logon function to another ground system. This function is initiated by a ground system that supports ground-ground forwarding. After an aircraft has c
+ompleted a successful logon with that ground system, that ground system can then forward the aircraft CM-logon information to other ground systems. It is a one-way forwarding of information with an indication of success, failure or non-support from the receiving ground system. If the ground system receiving this CM information supports ground-ground forwarding, it can then initiate a CM update function to provide information to the aircraft for any air-initiated applications.

2.1.2.5.2 The CM-forward request message contains the information as provided in the initial logon.

2.1.2.5.3 The CM-forward service also contains an indication of whether or not security is desired for the exchange.

Note. – The specific security mechanisms that may provide the security functionality are out of scope of this document.

2.1.2.6 Registration function

2.1.2.6.1 The registration function provides a method for the air and ground CM applications to make available the application name, address and version number for each application exchanged in the logon, update or forward functions to other applications or communications systems in the aircraft or on the ground.

2.1.2.6.2 Registration may also include additional security functionality. This security functionality is beyond the scope of this document

2.1.2.6.3 There are no standard message exchanges for the registration function.

2.2 GENERAL REQUIREMENTS

2.2.1 CM-ASE version number

The CM-air-ASE and CM-ground-ASE version numbers shall both be set to one (basic CM functionality).

2.2.2 Error-processing requirements

2.2.2.1 In the event of information input by the CM-user being incompatible with that which is able to be processed by the system, the CM-user shall be notified.

2.2.2.2 In the event of a CM-user invoking a CM service primitive when the CM-ASE is not in a state specified in 2.5, the CM-user shall be notified.

2.3 THE ABSTRACT SERVICE

2.3.1 Service description

2.3.1.1 An implementation of either the CM ground-based service or the CM air-based service shall exhibit external behaviour consistent with having implemented a CM-ground-ASE or CM-air-ASE, respectively.

2.3.1.2 This section defines the abstract service interface for the CM service. The CM-ASE abstract service is described from the viewpoint of the CM-air-user, the CM-ground-user and the CM service provider.

2.3.1.3 This section defines the static behaviour (i.e. the format) of the CM abstract service. Its dynamic behaviour (i.e. how it is used) is described in 2.7.

2.3.1.4 Figure 2-1 shows the CM application. The functional modules identified in the CM application are the following:

a) the CM-user;

b) the CM application entity (CM-AE) service interface;

c) the CM-AE;

d) the CM control function (CM-CF);

e) the CM application service element (CM-ASE) service interface;

f) the CM-ASE; and

g) the DS interface.

2.3.1.5 The CM-user represents the operational part of the CM system. This user does not perform the communication functions but relies on a communication service provided to it via the CM-AE through the CM-AE service interface. The individual actions at this interface are called CM-AE service primitives. Similarly, individual actions at other interfaces in the communication system are called service primitives at these interfaces.

2.3.1.6 The CM-AE consists of several elements including the CM-ASE and the CM-CF. The DS interface is made available by the CM-CF to the CM-ASE for communication with the peer CM-ASE.

2.3.1.7 The CM-ASE is the element in the communication system that executes the CM-specific protocol. In other words, it takes care of the CM-specific service primitive sequencing actions, message creation, timer management, and error and exception handling.

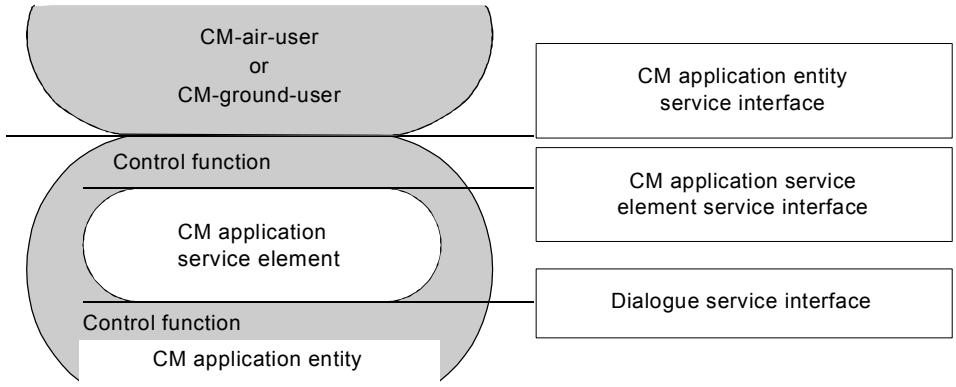


Figure 2-1. Functional model of the CM application

2.3.1.8 The CM-ASE interfaces only with the CM-CF. This CM-CF is responsible for mapping service primitives received from one element (such as the CM-ASE and the CM-user) to other elements that interface with it. The part of the CM-CF that is relevant from the point of view of the CM application, i.e. the part between the CM-user and the CM-ASE, will map CM-AE service primitives to CM-ASE service primitives transparently.

2.3.1.9 The DS interface is the interface between the CM-ASE and the part of the CM-CF underneath the CM-ASE, and provides the DS as defined in Part III of this manual.

2.3.2 The CM-ASE abstract service

2.3.2.1 There is no requirement to implement the CM-ASE abstract service in a CM product; however, it is necessary to implement the ground-based and air-based system in such a way that it will be impossible to detect (from the peer system) whether or not an interface has been built.

2.3.2.2 The CM-ASE abstract service shall consist of a subset of the following services as allowed by the subsetting rules in 2.8:

a) CM-logon service as defined in 2.3.3;

b) CM-update service as defined in 2.3.4;

c) CM-contact service as defined in 2.3.5;

d) CM-end service as defined in 2.3.6;

e) CM-forward service as defined in 2.3.7;

f) CM-user-abort service as defined in 2.3.8; and

g) CM-provider-abort service as defined in 2.3.9.

2.3.2.3 For a given primitive, the presence of each parameter is described by one of the following values in the parameter tables in 2.3:

blank not present;

C conditional upon some predicate explained in the text;

C(=) conditional upon the value of the parameter to the immediate left being both present and equal;

M mandatory;

M(=) mandatory, and equal to the value of the parameter to the immediate left; or

U user option.

2.3.2.4 The following abbreviations are used in Chapter 2 of this manual:

Req request; data is input by the CM-user initiating the service to its respective ASE;

Ind indication; data is indicated by the receiving ASE to its respective CM-user;

Rsp response; data is input by the receiving CM-user to its respective ASE; and

Cnf confirmation; data is confirmed by the initiating ASE to its respective CM-user.

2.3.2.5 An unconfirmed service allows a message to be transmitted in one direction without providing a corresponding response.

2.3.2.6 A confirmed service provides end-to-end confirmation that a message sent by one user was received by its peer user.

2.3.2.7 An abstract syntax is a syntactical description of a parameter that does not imply a specific implementation. Only when the CM-ASE maps a parameter onto an application protocol data unit (APDU) field, or vice versa, is the abstract syntax of the parameter described by using the ASN.1 at 2.4 for this field.

2.3.3 CM-logon service

2.3.3.1 General

2.3.3.1.1 The CM-logon service allows the CM-air-user to initiate data link service. The CM-air-user provides information on each data link application for which it desires a data link service. The CM-ground-user responds indicating whether or not the CM-logon was successful, and if successful, includes information on each data link application it can support. It is a confirmed service.

2.3.3.1.2 The CM-air-user will set the Security Requirements parameter as required for the particular airspace. A CM-air-user may attempt either a secure or an unsecure CM-logon service.

Note. – The specific security mechanisms that may provide the security functionality are out of scope of this document.

2.3.3.1.3 If the CM-air-ASE version number is less than or equal to the CM-ground-ASE version number, then the CM-logon service shall contain the primitives and parameters as presented in Table 2-1.

Table 2-1. CM-logon service parameters — air-ASE version number ≤ ground-ASE version number

Parameter name	*Req*	*Ind*	*Rsp*	*Cnf*
Facility Designation	M			
Aircraft Address	M	M(=)		
CM-ASE Version Number		C		
Logon Request	M	M(=)		
Logon Response			M	M(=)
Class of Communication Service	U			
Maintain Dialogue			U	C(=)
Security Required	M	M(=)	C(=)	

2.3.3.1.4 If the CM-air-ASE version number is greater than the CM-ground-ASE version number, then the CM-logon service shall contain the primitives and parameters as presented in Table 2-2.

Table 2-2. CM-logon service parameters — air-ASE version number > ground-ASE version number

Parameter name	*Req*	*Cnf*
Facility Designation	M	
Aircraft Address	M	
CM-ASE Version Number		M
Logon Request	M	
Class of Communication Service	U	
Security Required	M	

2.3.3.2 Facility Designation parameter

2.3.3.2.1 This parameter contains the addressed ground system's facility designation.

2.3.3.2.2 The *Facility Designation* parameter value shall conform to the abstract syntax four- to eight-character facility designation.

2.3.3.3 Aircraft Address parameter

2.3.3.3.1 The *Aircraft Address* parameter contains the ICAO 24-bit aircraft address of the aircraft initiating the CM-logon service.

2.3.3.3.2 The *Aircraft Address* parameter value shall conform to the abstract syntax of the ICAO 24-bit aircraft address.

2.3.3.4 CM-ASE Version Number parameter

2.3.3.4.1 This parameter contains the version number of the CM-ASE.

2.3.3.4.2 When provided by the CM-ASE, the *CM-ASE Version Number* parameter shall conform to an abstract integer value from 1 to 255.

2.3.3.4.3 Only if the CM-air-ASE version number is less than the CM-ground-ASE version number shall the CM-air-ASE version number be indicated to the CM-ground-user.

2.3.3.4.4 Only if the CM-air-ASE version number is greater than the CM-ground-ASE version number shall the CM-ground-ASE version number be confirmed to the CM-air-user.

2.3.3.4.5 If the CM-air-ASE version number is the same as the CM-ground-ASE version number, the *CM-ASE Version Number* parameter is not present in the indication given to the CM-ground-user or in the confirmation to the CM-air-user.

2.3.3.5 Logon Request parameter

2.3.3.5.1 The *Logon Request* parameter contains the following data:

a) information for each data link application available on the aircraft, for which the aircraft requires data link service; and

b) aircraft flight plan information (e.g. flight identification, aircraft destination and departure airport and time) as required by the addressed ground system.

2.3.3.5.2 The *Logon Request* parameter value shall conform to the ASN.1 abstract syntax CMLogonRequest.

2.3.3.6 Logon Response parameter

2.3.3.6.1 The *Logon Response* parameter contains information for each requested data link application for which the ground is able to provide data link service.

2.3.3.6.2 The *Logon Response* parameter value shall conform to the ASN.1 abstract syntax CMLogonResponse.

2.3.3.7 Class of Communication Service parameter

2.3.3.7.1 This parameter contains the value of the required class of communication service if specified by the

CM-air-user.

2.3.3.7.2 When the *Class of Communication Service* parameter is specified by the CM-air-user, this parameter shall have one of the following abstract values: “A”, “B”, “C”, “D”, “E”, “F”, “G” or “H”.

2.3.3.7.3 If the *Class of Communication Service* parameter is not specified by the CM-air-user, it indicates that there is no routing preference.

2.3.3.8 Maintain Dialogue parameter

2.3.3.8.1 The *Maintain Dialogue* parameter is used to indicate whether or not the requested CM dialogue is to remain open after a Logon Response.

2.3.3.8.2 Whenever a CM dialogue is kept open by the CM-ground-user, it must later be explicitly closed by the CM-ground-user.

2.3.3.8.3 This parameter is only provided by the CM-ground-user when the CM-ground-user wishes to keep the CM dialogue open.

2.3.3.8.4 If provided by the CM-ground-user, this parameter shall have the abstract value “maintain”.

2.3.3.9 Security Required parameter

2.3.3.9.1 The *Security Required* parameter contains an indication of whether or not a CM requires security. It is used by the requester of the CM-logon service and is indicated to the responder.

Note. – The specific security mechanisms that may provide the security functionality are out of scope of this document.

2.3.3.9.2 If the received *Security Required* parameter is not as expected per the local security policy, the CM-ground-ASE will abort.

2.3.3.9.3 The value of the *Security Required* parameter provided by the CM-air-user shall be indicated to the CM-ground-user.

2.3.3.9.4 When the CM-air-user requires a secure CM-logon service, the *Security Required* parameter shall have the abstract value “secured dialogue supporting security information management”.

Note.— This is to give an indication to the CM-ground-user that security is to be used. However, the specific security mechanisms that may provide the security functionality are out of scope of this document

2.3.3.9.5 When the CM-air-user requires an unsecure CM-logon service, the *Security Required* parameter shall have the abstract value “no security”.

Note.— This is to give an indication to the CM-ground-user that security is not to be used. However, the CM-ground-user will have the ultimate decision of whether or not security is required. If security is required by the ground system but not requested by the aircraft, the aircraft may be denied data link services.

2.3.4 CM-update service

2.3.4.1 General

2.3.4.1.1 The CM-update service allows the CM-ground-user to transmit updated ground information for its applications to update previously coordinated CM-logon information. It is an unconfirmed service.

2.3.4.1.2 The CM-ground-user will set the Security Required parameter as appropriate in order to perform secure or unsecure services.

Note. – The specific security mechanisms that may provide the security functionality are out of scope of this document.

2.3.4.1.3 The CM-update service shall contain the primitives and parameters as presented Table 2-3.

Table 2-3. CM-update service parameters

Parameter name	*Req*	*Ind*
Aircraft Address	C	
Facility Designation	C	C(=)
Update Information	M	M(=)
Class of Communication Service	U	
Security Required	C	

2.3.4.2 Aircraft Address parameter

2.3.4.2.1 The *Aircraft Address* parameter contains the addressed aircraft's ICAO 24-bit aircraft address.

2.3.4.2.2 The *Aircraft Address* parameter value shall conform to the abstract syntax of the ICAO 24-bit aircraft address.

2.3.4.2.3 If a CM dialogue does not exist when a CM-ground-user invokes the CM-update service request, the CM-ground-user shall provide the *Aircraft Address* parameter value.

Note.— The CM-update service does not use this parameter when a CM dialogue exists.

2.3.4.3 Facility Designation parameter

2.3.4.3.1 This parameter contains the ground system's facility designation.

2.3.4.3.2 The *Facility Designation* parameter value shall conform to the abstract syntax four- to eight-character facility designation.

2.3.4.3.3 If a CM dialogue does not exist when a CM-ground-user invokes the CM-update service request, the CM-ground-user shall provide the *Facility Designation* parameter value.

Note.— The CM-update service does not use this parameter when a CM dialogue exists.

2.3.4.4 Update Information parameter

2.3.4.4.1 The *Update Information* parameter contains information on each updated data link application.

2.3.4.4.2 The *Update Information* parameter value shall conform to the ASN.1 abstract syntax CMUpdate.

2.3.4.5 Class of Communication Service parameter

2.3.4.5.1 This parameter contains the value of the required class of communication service if specified by the CM-ground-user.

2.3.4.5.2 The CM-update service does not use this parameter when a CM dialogue exists.

2.3.4.5.3 When the *Class of Communication Service* parameter is specified by the CM-ground-user, this parameter shall have one of the following abstract values: “A”, “B”, “C”, “D”, “E”, “F”, “G” or “H”.

2.3.4.5.4 If the *Class of Communication Service* parameter is not specified by the CM-ground-user, it indicates that there is no routing preference.

2.3.4.6 Security Required parameter

2.3.5.6.1 The *Security Required* parameter contains an indication of whether or not secure CM-update services will be used.

Note. – The specific security mechanisms that may provide the security functionality are out of scope of this document.

2.3.5 CM-contact service

2.3.5.1 General

2.3.5.1.1 The CM-contact service allows the CM-ground-user, after successful completion of a CM-logon, to request that an aircraft logon with another ground system. It is a confirmed service.

2.3.5.1.2 The CM-ground-user will set the Security Required parameter as appropriate in order to perform secure or unsecure services.

2.3.5.1.3 The CM-contact service shall contain the primitives and parameters as presented in Table 2-4.

Table 2-4. CM-contact service parameters

Parameter name	*Req*	*Ind*	*Rsp*	*Cnf*
Aircraft Address	C			
Facility Designation	C	C(=)		
Contact Request	M	M(=)		
Contact Response			M	M(=)
Class of Communication Service	U			
Security Required	C			

2.3.5.2 Aircraft Address parameter

2.3.5.2.1 The *Aircraft Address* parameter contains the addressed aircraft's ICAO 24-bit aircraft address.

2.3.5.2.2 The *Aircraft Address* parameter value shall conform to the abstract syntax of the ICAO 24-bit aircraft address.

2.3.5.2.3 If a CM dialogue does not exist when a CM-ground-user invokes the CM-contact service request, the CM-ground-user shall provide the *Aircraft Address* parameter value.

Note.— The CM-contact service does not use this parameter when a CM dialogue exists.

2.3.5.3 Facility Designation parameter

2.3.5.3.1 This parameter contains the ground system's facility designation.

2.3.5.3.2 The *Facility Designation* parameter value shall conform to the abstract syntax four- to eight-character facility designation.

2.3.5.3.3 If a CM dialogue does not exist when a CM-ground-user invokes the CM-contact service request, the CM-ground-user shall provide the *Facility Designation* parameter value.

Note.— The CM-contact service does not use this parameter when a CM dialogue exists.

2.3.5.4 Contact Request parameter

2.3.5.4.1 The *Contact Request* parameter contains the facility designation for the ground system that the CM-ground-user requests the aircraft to contact.

2.3.5.4.2 The *Contact Request* parameter value shall conform to the ASN.1 abstract syntax CMContactRequest.

2.3.5.5 Contact Response parameter

2.3.5.5.1 The *Contact Response* parameter indicates success, or lack thereof, of the requested contact.

2.3.5.5.2 The *Contact Response* parameter value shall conform to the ASN.1 abstract syntax CMContactResponse.

2.3.5.6 Class of Communication Service parameter

2.3.5.6.1 This parameter contains the value of the required class of communication service if specified by the CM-ground-user.

2.3.5.6.2 When the *Class of Communication Service* parameter is specified by the CM-ground-user, this parameter shall have one of the following abstract values: “A”, “B”, “C”, “D”, “E”, “F”, “G” or “H”.

2.3.5.6.3 If the *Class of Communication Service* parameter is not specified by the CM-ground-user, it indicates that there is no routing preference.

2.3.5.7 Security Required parameter

2.3.5.7.1 The *Security Required* parameter contains an indication of whether or not a CM-ground-user requires security. It is used by the requester of CM-contact service.

Note. – The specific security mechanisms that may provide the security functionality are out of scope of this document.

2.3.6 CM-end service

2.3.6.1 General

2.3.6.1.1 This service provides the capability for the CM-ground-user to terminate a CM dialogue. This service is only needed when the CM-ground-user maintains a CM dialogue during the logon process. It is an unconfirmed service.

2.3.6.1.2 Only the CM-ground-user will be capable of initiating the CM-end service.

2.3.6.1.3 The CM-end service shall contain the primitives as presented Table 2-5.

Table 2-5. CM-end service parameters

Parameter name	*Req*	*Ind*
None		

2.3.7 CM-forward service

2.3.7.1 General

2.3.7.1.1 The CM-forward service allows a CM-ground-user to forward data received in a CM-logon request to another CM-ground system. It is a confirmed service.

2.3.7.1.2 If security is supported, the CM-forward service also contains an indication of whether or not security is desired for the exchange.

Note. – The specific security mechanisms that may provide the security functionality are outside the scope of this document.

2.3.7.1.3 If the sending CM-ground-ASE version number is less than or equal to the receiving CM-ground-ASE version number, then the CM-forward service shall contain the primitives and parameters as presented in Table 2-6.

Table 2-6. CM-forward service parameters — sending ground-ASE version number ≤ receiving ground-ASE version number

Parameter name	*Req*	*Ind*	*Cnf*
Called Facility Designation	M		
Calling Facility Designation	M	M(=)	
CM-ASE Version Number		C	
Forward Request	M	M(=)	
Class of Communication Service	U		
Result			M
Security Required	M		

2.3.7.1.4 If the sending CM-ground-ASE version number is greater than the receiving CM-ground-ASE version number, then the CM-forward service shall contain the primitives and parameters as presented in Table 2-7.

Table 2-7. CM-forward service parameters — sending ground-ASE version number > receiving ground-ASE version number

Parameter name	*Req*	*Cnf*
Called Facility Designation	M	
Calling Facility Designation	M	
CM-ASE Version Number		M
Forward Request	M	
Class of Communication Service	U	
Result		M
Security Required	M	

2.3.7.2 Called Facility Designation parameter

2.3.7.2.1 The *Called Facility Designation* parameter contains the receiving ground system's facility designation.

2.3.7.2.2 The *Called Facility Designation* parameter value shall conform to the abstract syntax four- to eight-character facility designation.

2.3.7.3 Calling Facility Designation parameter

2.3.7.3.1 This parameter contains the sending ground system's facility designation.

2.3.7.3.2 The *Calling Facility Designation* parameter value shall conform to the abstract syntax four- to eight-character facility designation.

2.3.7.4 CM-ASE Version Number parameter

2.3.7.4.1 This parameter contains the version number of the CM-ground-ASE.

2.3.7.4.2 When provided by the CM-ASE, the *CM-ASE Version Number* parameter shall conform to an abstract integer value from 1 to 255.

2.3.7.4.3 Only if the sending CM-ground-ASE version number is less than the receiving CM-ground-ASE version number shall the sending CM-ground-ASE version number be indicated to the receiving CM-ground-user.

2.3.7.4.4 Only if the sending CM-ground-ASE version number is greater than the receiving CM-ground-ASE version number shall the receiving CM-ground-ASE version number be confirmed to the sending CM-ground-user.

2.3.7.4.5 If the sending CM-ground-ASE version number is the same as the receiving CM-ground-ASE version number, the *CM-ASE Version Number* parameter is not present in the indication given to the receiving CM-ground-user or in the confirmation given to the sending CM-ground-user.

2.3.7.5 Forward Request parameter

2.3.7.5.1 The *Forward Request* parameter contains information as provided in the CM-logon request.

2.3.7.5.2 The CM-forward service *Forward Request* parameter value shall conform to the ASN.1 abstract syntax CMForwardRequest.

2.3.7.6 Class of Communication Service parameter

2.3.7.6.1 This parameter contains the value of the required class of communication service if specified by the initiating CM-ground-user.

2.3.7.6.2 When the *Class of Communication Service* parameter is specified by the CM-ground-user, this parameter shall have one of the following abstract values: "A", "B", "C", "D", "E", "F", "G" or "H".

2.3.7.6.3 If the *Class of Communication Service* is not specified by the CM-ground-user, it indicates that there is no routing preference.

2.3.7.7 Result parameter

2.3.7.7.1 The *Result* parameter indicates whether or not the information was forwarded as requested.

2.3.7.7.2 The *Result* parameter shall conform to the ASN.1 abstract syntax CMForwardResponse.

2.3.7.7.3 When the sending CM-ground-ASE version number is less than or equal to the receiving CM-ground-ASE version number, the *Result* parameter takes the abstract value “success”. When the sending CM-ground-ASE version number is greater than the receiving CM-ground-ASE version number, the *Result* parameter takes the abstract value “incompatible version”. When the receiving CM-ground-ASE does not support ground-ground forwarding, the *Result* parameter takes the abstract value “service not supported”.

2.3.7.8 Security Required parameter

2.3.7.8.1 The *Security Required* parameter contains an indication of whether or not a CM-ground-user requires security. It is used by the requester of CM-forward service.

Note. – The specific security mechanisms that may provide the security functionality are out of scope for this document.

2.3.8 CM-user-abort service

2.3.8.1 This service provides the capability for either the CM-air-user or the CM-ground-user to abort communication with its peer. This can be used for operational or technical reasons. It can be invoked at any time by an active user. Messages in transit may be lost during this operation. It is an unconfirmed service.

2.3.8.2 If the service is invoked prior to complete establishment of the dialogue, the CM-user-abort indication may not be provided. A CM-provider-abort indication may result instead.

2.3.8.3 The CM-user-abort service shall contain the primitives as presented in Table 2-8.

Table 2-8. CM-user-abort service parameters

Parameter name	*Req*	*Ind*
None		

2.3.9 CM-provider-abort service

2.3.9.1 General

2.3.9.1.1 The CM-provider-abort service provides the capability for the CM service provider to inform its users that it can no longer provide the CM service. Messages in transit may be lost during this operation.

2.3.9.1.2 The CM-provider-abort service shall contain the primitives and parameters as presented in Table 2-9.

Table 2-9. CM-provider-abort service parameters

Parameter name	*Ind*
Reason	M

2.3.9.2 Reason parameter

2.3.9.2.1 This parameter identifies the reason for the abort.

2.3.9.2.2 The *Reason* parameter value shall conform to the ASN.1 abstract syntax CMAbortReason.

2.4 FORMAL DEFINITIONS OF MESSAGES

2.4.1 Encoding/decoding rules

2.4.1.1 A CM-air-ASE shall be capable of encoding CMAircraftMessage APDUs and decoding CMGroundMessage APDUs.

2.4.1.2 A CM-ground-ASE shall be capable of encoding and decoding CMGroundMessage APDUs and decoding CMAircraftMessage APDUs.

2.4.2 CM ASN.1 abstract syntax

The abstract syntax of the CM protocol data units (PDUs) shall comply with the description contained in the ASN.1 module CMMessageSetVersion1 (conforming to ISO/IEC 8824-1), as defined hereinafter. The ASN.1 module CMMessageSetVersion1 is used by CM version 1.

```
CMMessageSetVersion1 DEFINITIONS ::=

BEGIN


-- ------------------------------------------------------------------------------
-- CM Message Structure
-- ------------------------------------------------------------------------------
-- Aircraft-generated messages

CMAircraftMessage ::=CHOICE
{
cmLogonRequest                  [0]     CMLogonRequest,
cmContactResponse               [1]     CMContactResponse,
cmAbortReason                   [2]     CMAbortReason,
...
}
```

```
-- Ground-generated messages
CMGroundMessage ::=CHOICE
{
cmLogonResponse                 [0]     CMLogonResponse,
cmUpdate                        [1]     CMUpdate,
cmContactRequest                [2]     CMContactRequest,
cmForwardRequest                [3]     CMForwardRequest,
cmAbortReason                   [4]     CMAbortReason,
cmForwardResponse               [5]     CMForwardResponse,
...
}
-- ------------------------------------------------------------------------------
-- CM Message Components
-- ------------------------------------------------------------------------------

AircraftFlightIdentification ::= IA5String (SIZE(2..8))

Airport ::= IA5String (SIZE(4))

APAddress ::= CHOICE
{
        longTsap                [0]     LongTsap,
        shortTsap               [1]     ShortTsap
}

AEQualifier ::= INTEGER (0..255)        -- ATN AE-Qualifier Numeric Values are described in Doc 9880, Part IV

AEQualifierVersion ::= SEQUENCE
{
        aeQualifier             AEQualifier,
        apVersion               VersionNumber
}

AEQualifierVersionAddress ::= SEQUENCE
{
        aeQualifier             AEQualifier,
        apVersion               VersionNumber,
        apAddress               APAddress
}

CMAbortReason ::= ENUMERATED
{
        timer-expired                           (0),
        undefined-error                         (1),
        invalid-PDU                             (2),
        protocol-error                          (3),
        dialogue-acceptance-not-permitted       (4),
        dialogue-end-not-accepted               (5),
        communication-service-error             (6),
        communication-service-failure           (7),
        invalid-QOS-parameter                   (8),
```

```
        expected-PDU-missing                (9),
        ...
}

CMContactRequest ::= SEQUENCE
{
        facilityDesignation         FacilityDesignation,
        address                     LongTsap
}

CMContactResponse ::= Response

CMForwardRequest ::= CMLogonRequest

CMForwardResponse ::= ENUMERATED
{
        success                     (0),
        incompatible-version        (1),
        service-not-supported       (2)
}

CMLogonRequest ::= SEQUENCE
{
        aircraftFlightIdentification    [0]  AircraftFlightIdentification,
        cMLongTSAP                      [1]  LongTsap,
        groundInitiatedApplications     [2]  SEQUENCE SIZE (1..256) OF AEQualifierVersionAddress
        OPTIONAL,
        airOnlyInitiatedApplications    [3]  SEQUENCE SIZE (1..256) OF AEQualifierVersion
        OPTIONAL,
        facilityDesignation             [4]  FacilityDesignation     OPTIONAL,
        airportDeparture                [5]  Airport                 OPTIONAL,
        airportDestination              [6]  Airport                 OPTIONAL,
        dateTimeDepartureETD            [7]  DateTime                OPTIONAL
}

CMLogonResponse ::= SEQUENCE
{
        airInitiatedApplications        [0]  SEQUENCE SIZE (1..256) OF AEQualifierVersionAddress
                                             OPTIONAL,
        groundOnlyInitiatedApplications [1]  SEQUENCE SIZE (1..256) OF AEQualifierVersion
                                             OPTIONAL
}

CMUpdate ::= CMLogonResponse

Date ::= SEQUENCE
{
            year    Year,
            month   Month,
            day Day
}
-- The Date field does not have to correspond to the flight if the field is not to be used; the field's value can be assigned a
```

meaningless, but compliant, value locally. If operational use of the Date field is intended, there must be bilateral agreements in place to ensure its proper use. This is a local implementation issue.

```
DateTime ::= SEQUENCE
{
        dateDate,
        timeTime
}

Day ::= INTEGER (1..31)
        --unit = Day, Range (1..31), resolution = 1

FacilityDesignation ::= IA5String (SIZE(4..8))

LongTsap ::= SEQUENCE
{
rDP             OCTET STRING (SIZE(5)),
shortTsap   ShortTsap
}

Month ::= INTEGER (1..12)
        --unit = Month, Range (1..12), resolution = 1

Response ::= ENUMERATED
{
        contactSuccess                  (0),
        contactNotSuccessful            (1)
}


ShortTsap ::= SEQUENCE
{
        aRS                             [0]     OCTET STRING (SIZE(3)) OPTIONAL,
-- the aRS contains the ICAO 24-bit aircraft address when the ShortTsap belongs to an aircraft;
-- or a ground address when the Short Tsap belongs to a ground system
locSysNselTsel                          [1]     OCTET STRING (SIZE(10..11))
}

Time ::= SEQUENCE
{
        hours           Timehours,
        minutes         Timeminutes
}

Timehours ::= INTEGER (0..23)
        -- units = hour, range (0..23), resolution = 1 hour

Timeminutes ::= INTEGER (0..59)
        -- units = minute, range (0..59), resolution = 1 minute

VersionNumber ::= INTEGER (1..255)
        -- VersionNumber 0 is reserved for the dialogue service
```

Year ::= INTEGER (1996..2095)
--unit = Year, Range (1996..2095), resolution = 1

END

2.5 PROTOCOL DEFINITION

2.5.1 Sequence rules

2.5.1.1 With the exception of abort primitives, only the sequence of primitives illustrated in Figures 2-2 to 2-23 shall be permitted.

2.5.1.2 Figures 2-2 to 2-23 define the valid sequences of primitives that are possible to be invoked during the operation of the CM application. The figures show the relationship in time between the service request and the resulting indication and, if applicable, the subsequent response and resulting confirmation.

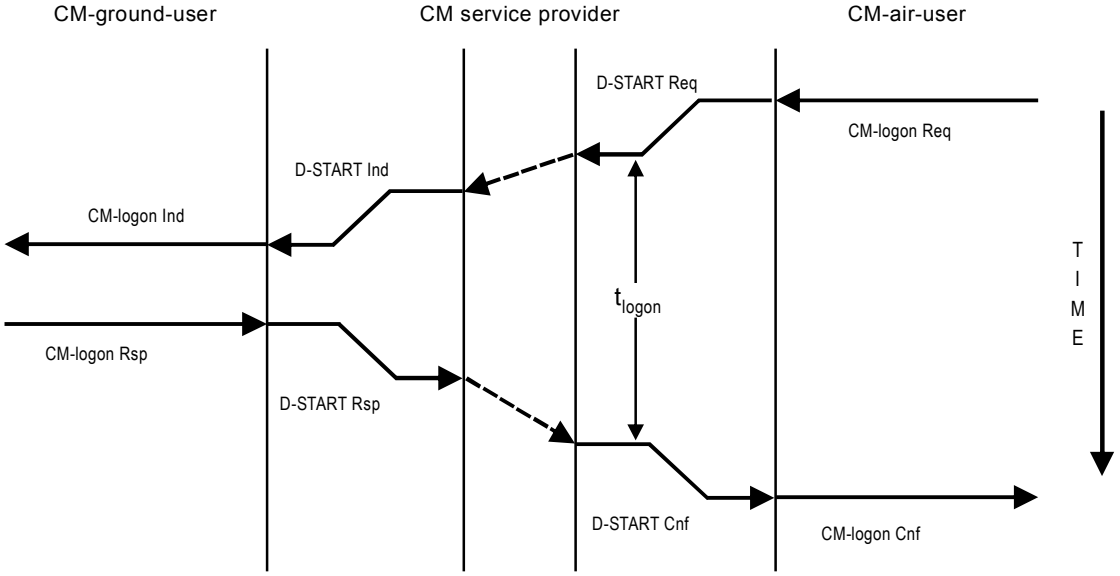


Figure 2-2. Sequence diagram for CM-logon service — CM-air-ASE version ≤ CM-ground-ASE version

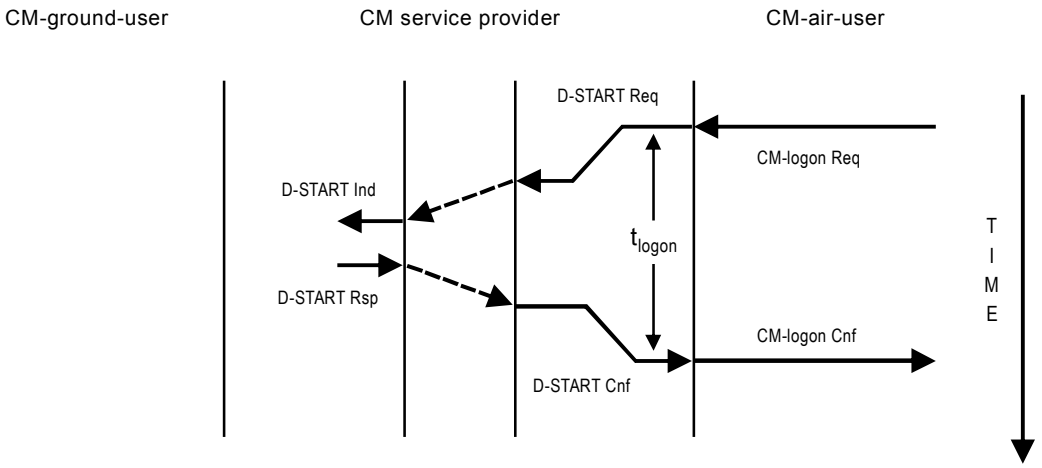


Figure 2-3. Sequence diagram for CM-logon service

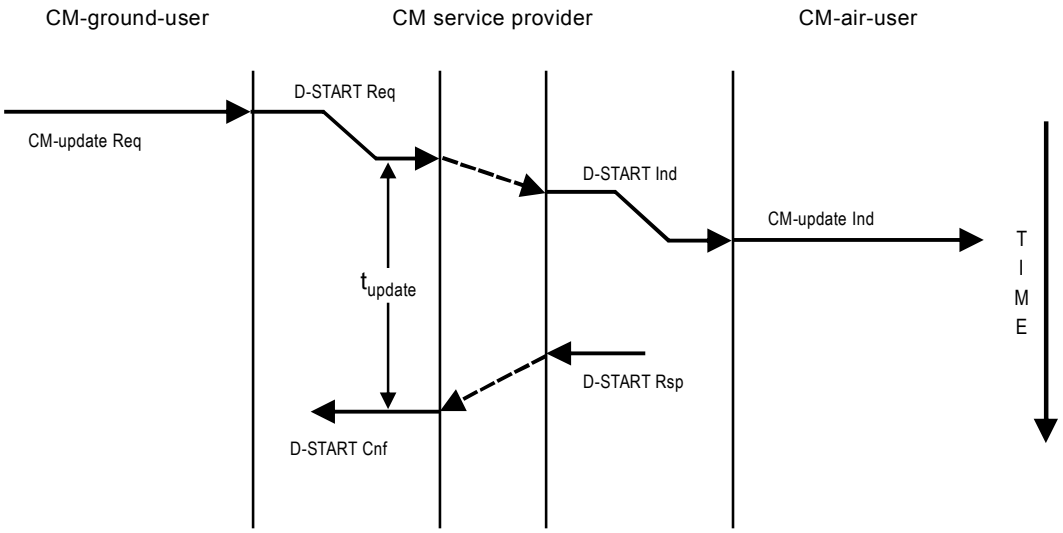


Figure 2-4. Sequence diagram for CM-update service — No existing CM dialogue

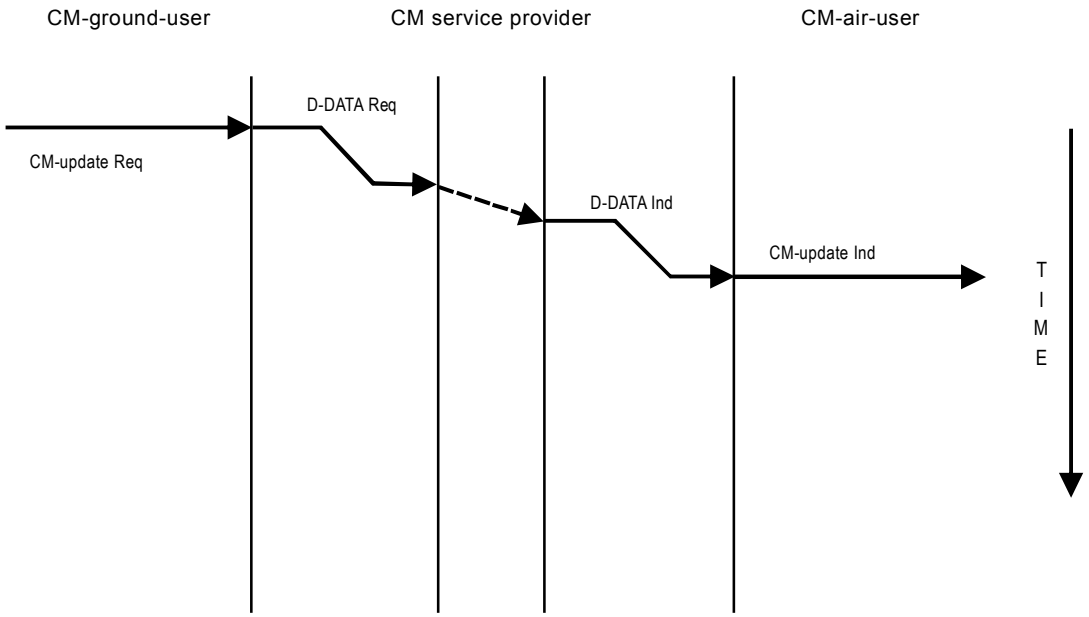


Figure 2-5. Sequence diagram for CM-update service — Existing CM dialogue

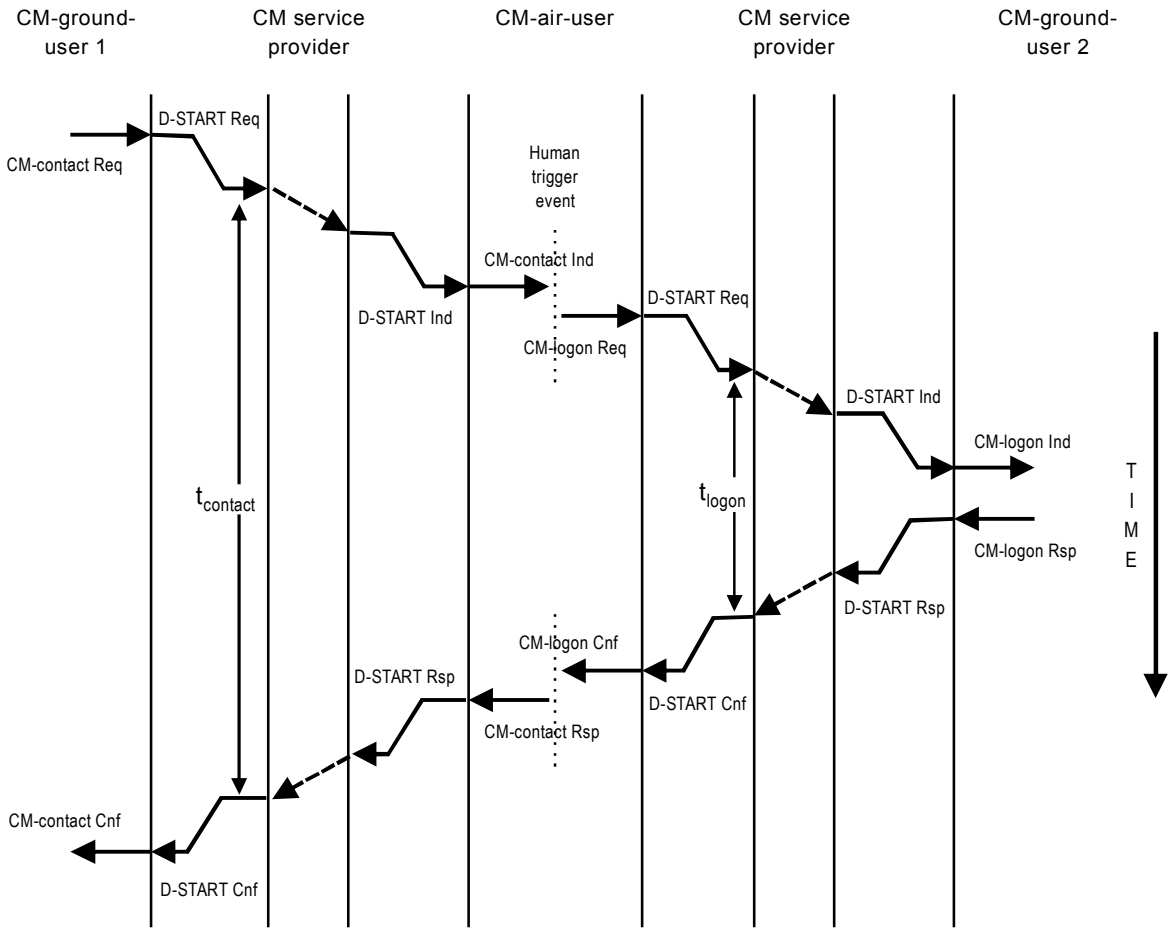


Figure 2-6. Sequence diagram for CM-contact service
— No existing CM dialogue
— With CM-logon service as in Figure 2-2

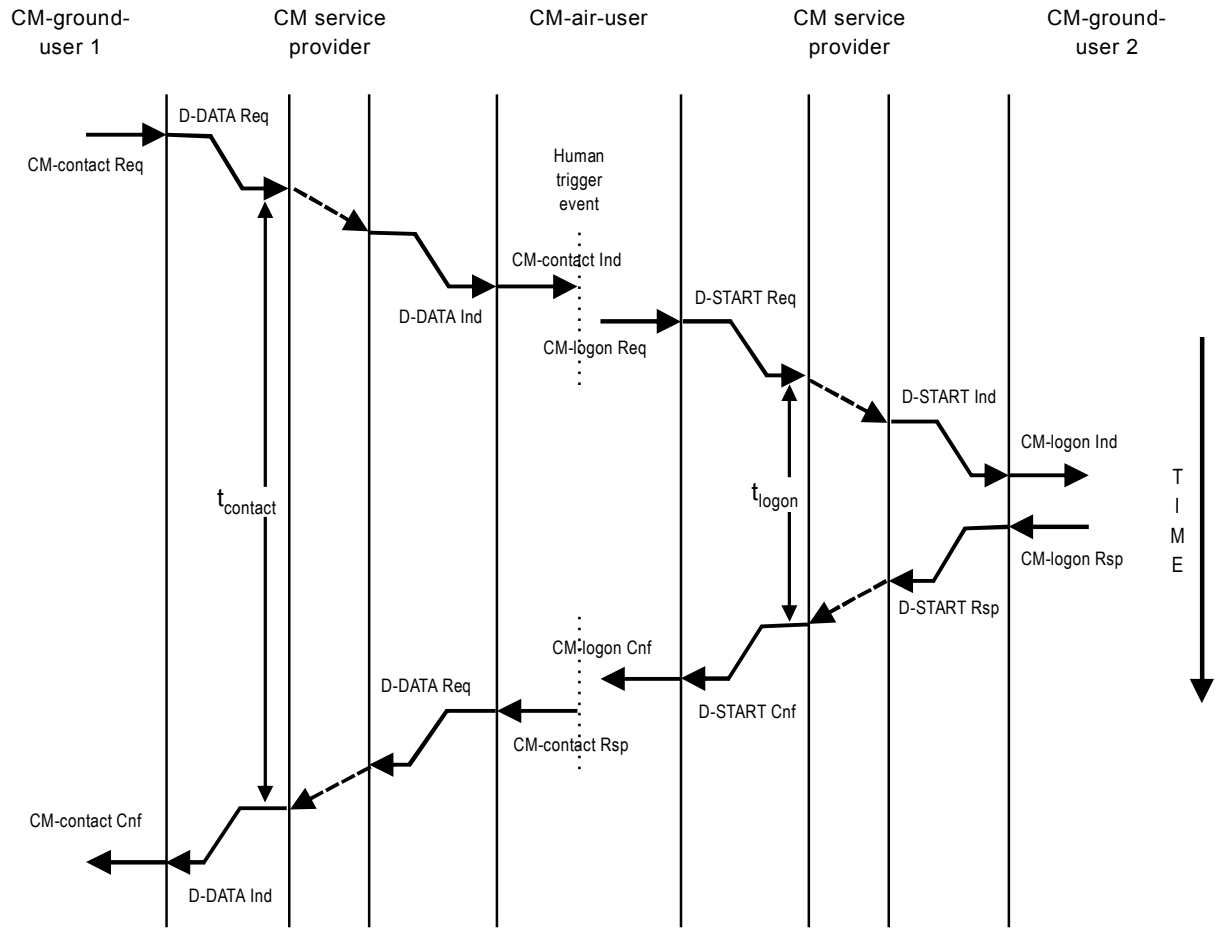


Figure 2-7. Sequence diagram for CM-contact Service
— With existing CM dialogue
— With CM-logon service as in Figure 2-2

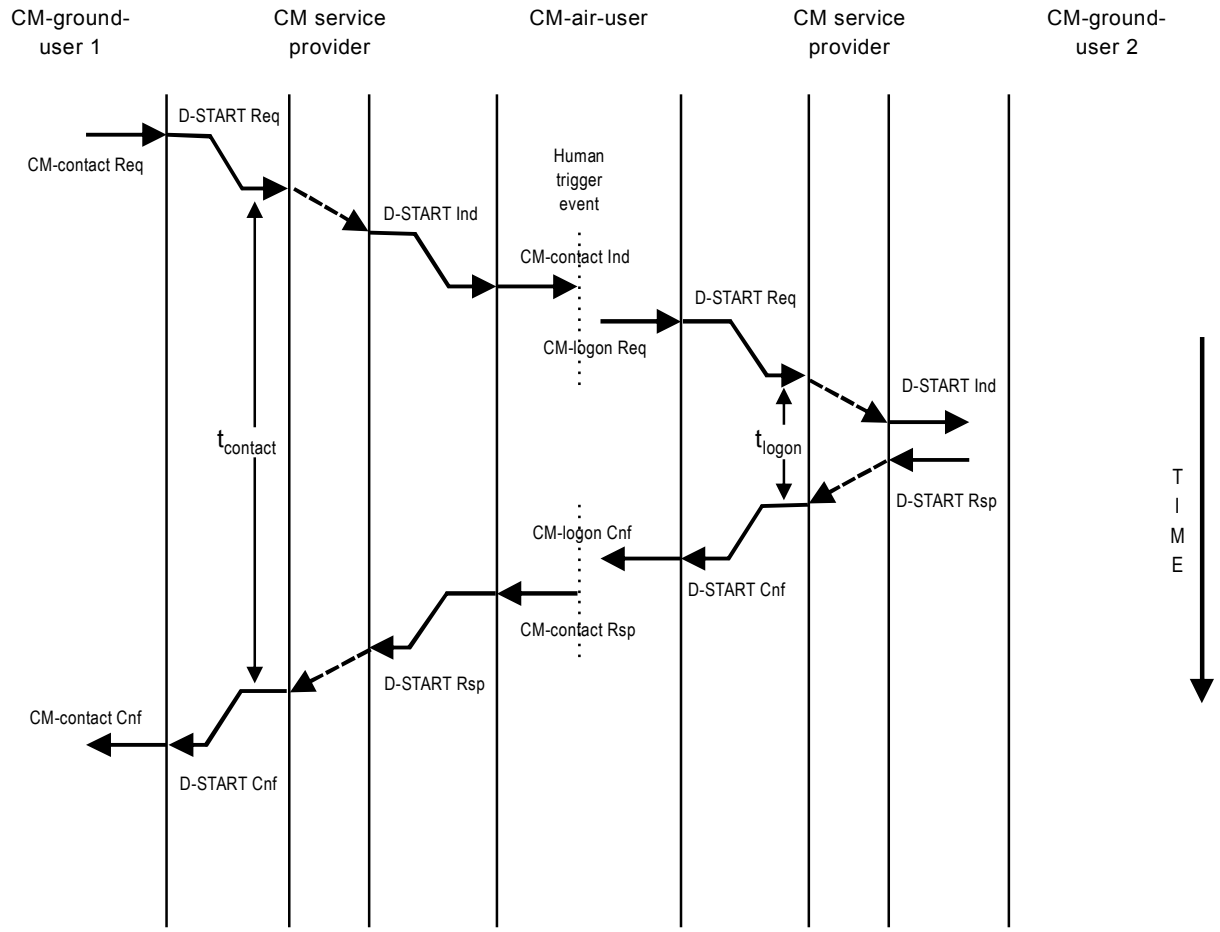


Figure 2-8. Sequence diagram for CM-contact service
— No existing CM dialogue
— With CM-logon service as in Figure 2-3

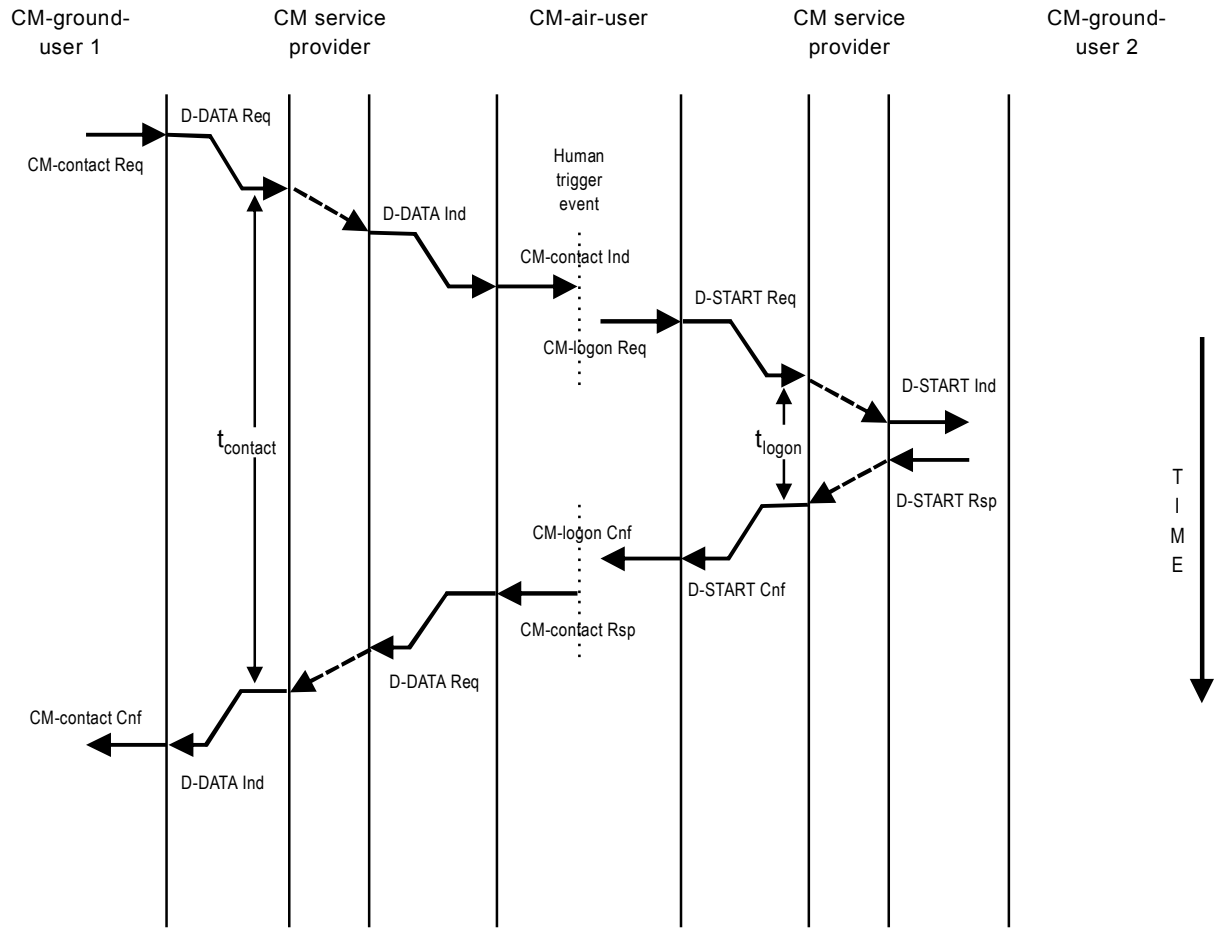


Figure 2-9. Sequence diagram for CM-contact service
— Existing CM dialogue
— With CM-logon service as in Figure 2-3

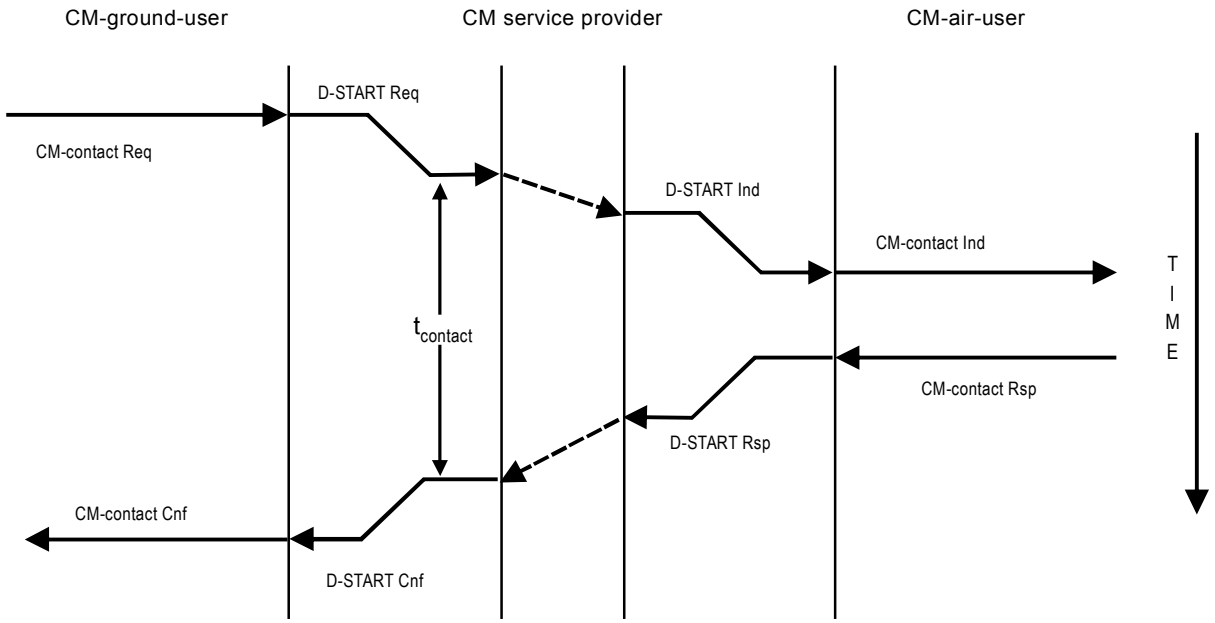


Figure 2-10. Sequence diagram for CM-contact service
— No existing CM dialogue
— Without CM-logon service as requested

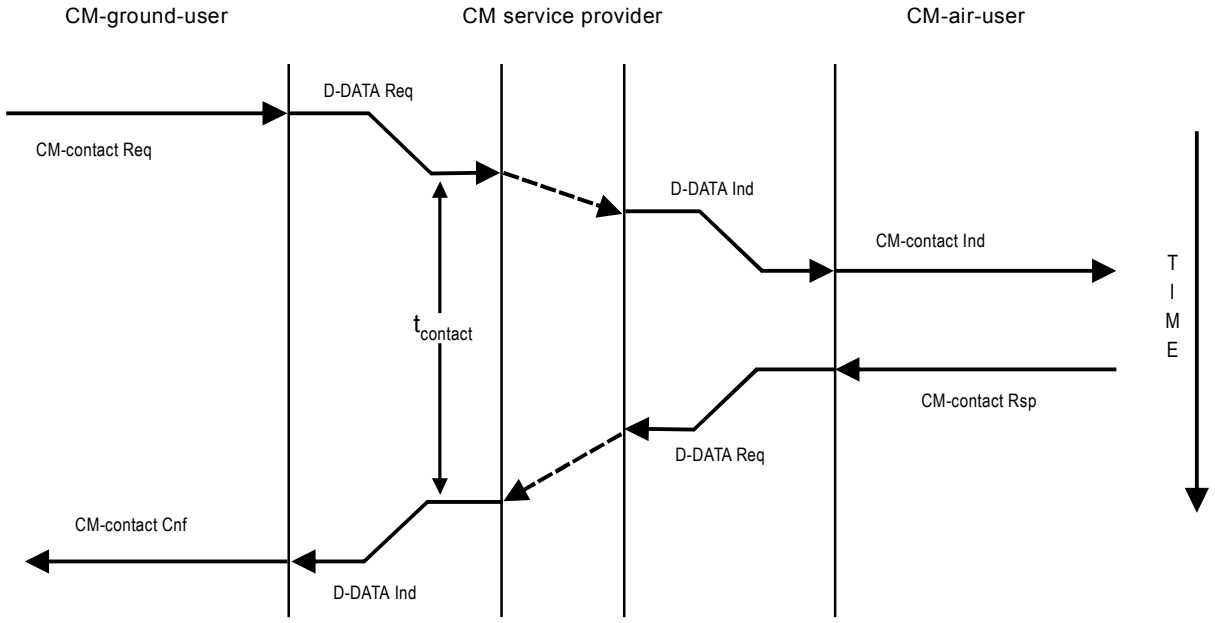


Figure 2-11. Sequence diagram for CM-contact service
— With existing CM dialogue
— Without CM-logon service as requested

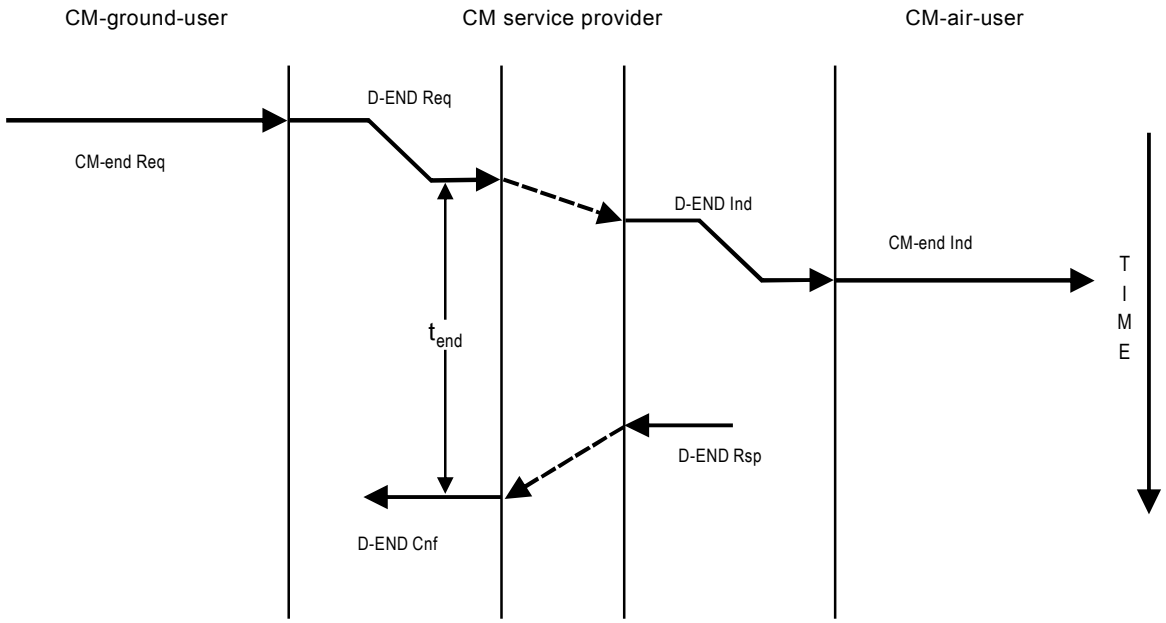


Figure 2-12. Sequence diagram for CM-end service

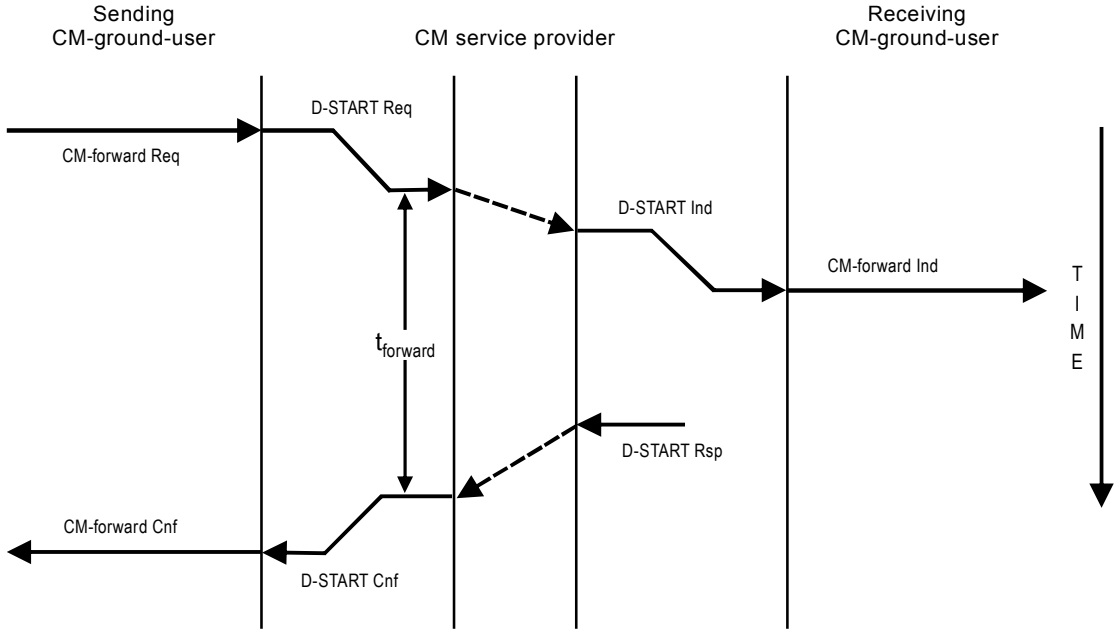


Figure 2-13. Sequence diagram for CM-forward service
— Sending CM-ground-ASE Version ≤ Receiving CM-ground-ASE Version,
— Ground-ground forwarding supported

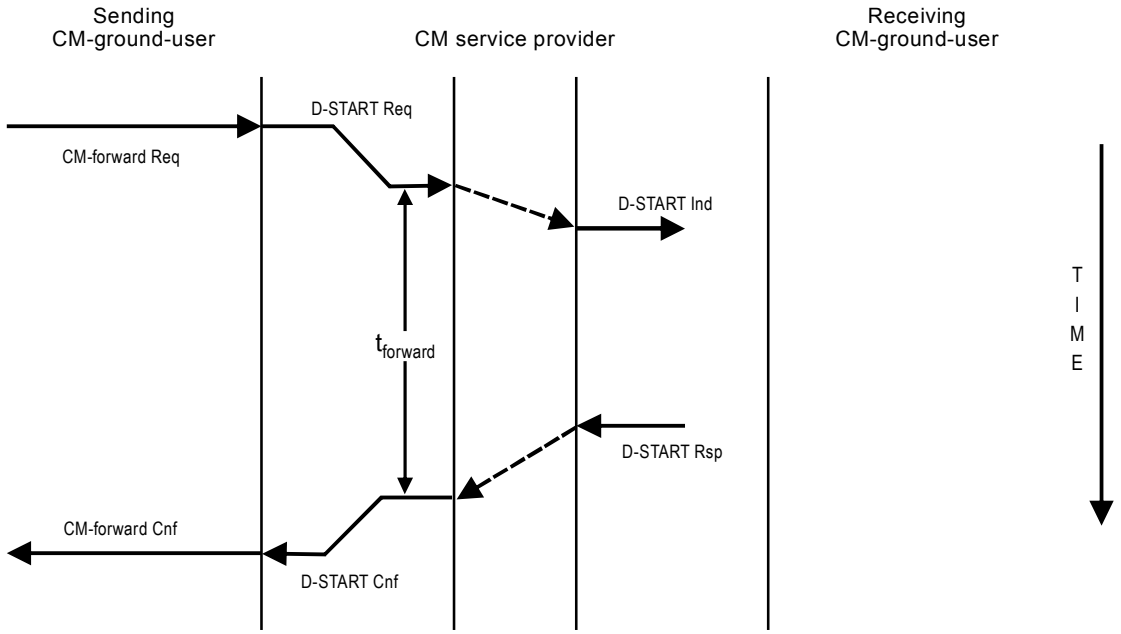


Figure 2-14. Sequence diagram for CM-forward service
— Sending CM-ground-ASE Version > Receiving CM-ground-ASE Version, or
— Receiving CM-ground-ASE does not support ground-ground forwarding

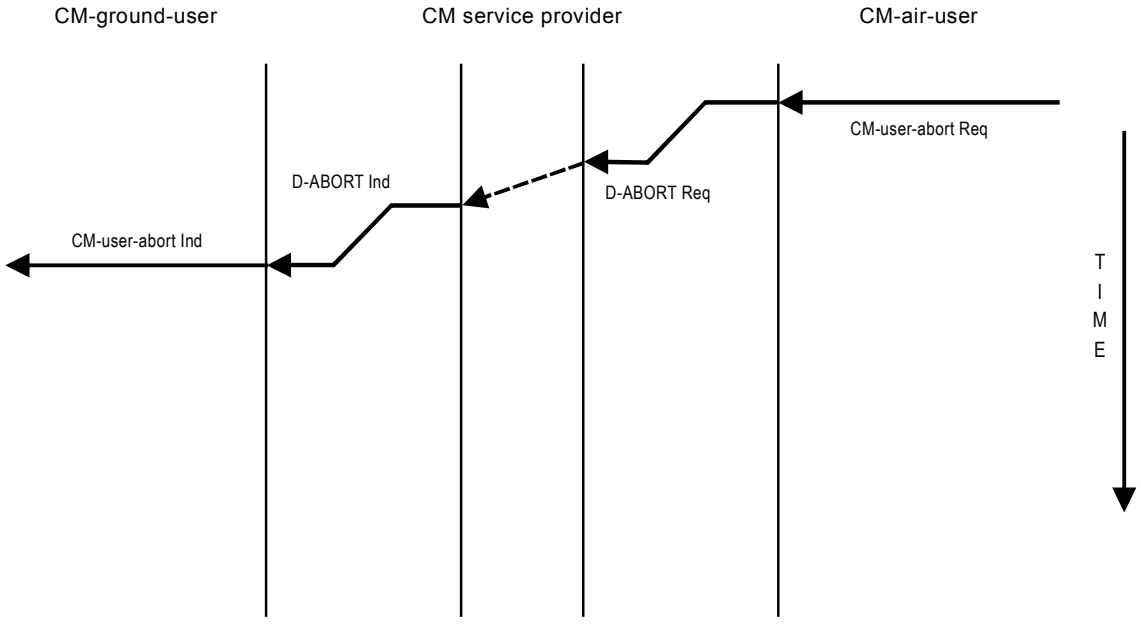


Figure 2-15. Sequence diagram for CM-user-abort service
— CM-air-user initiated

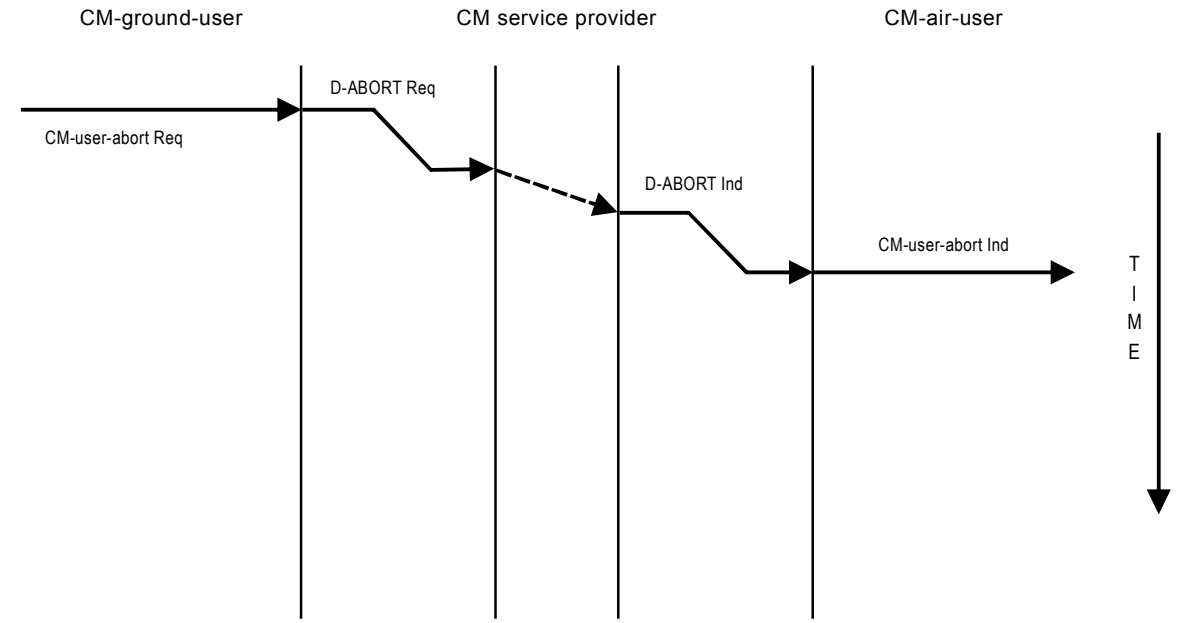


Figure 2-16. Sequence diagram for CM-user-abort service — CM-ground-user initiated

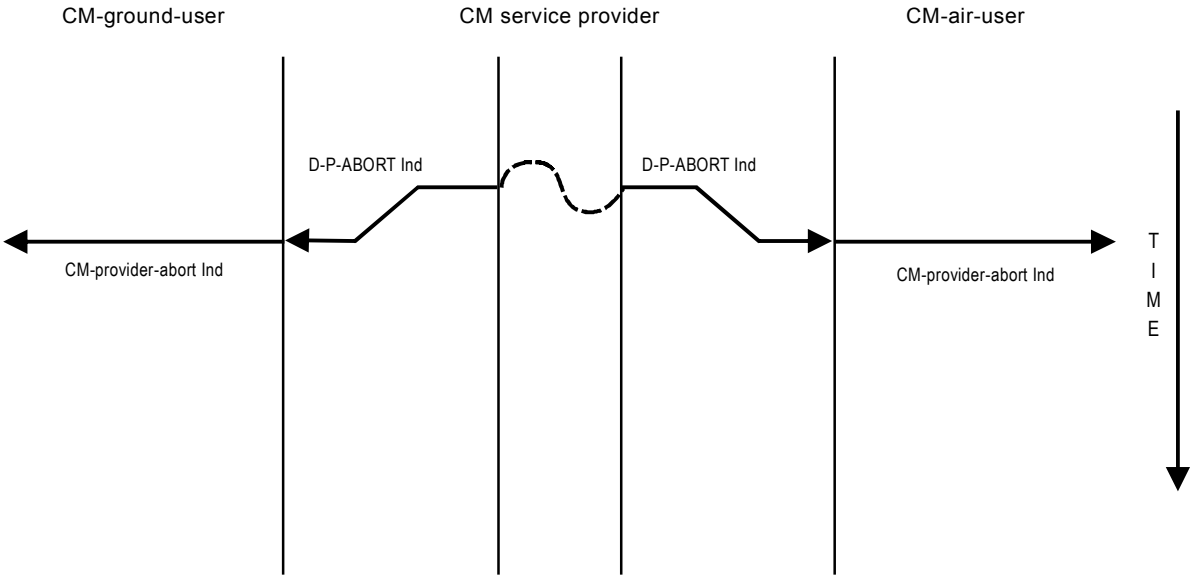


Figure 2-17. Sequence diagram for CM-provider-abort service — dialogue service abort

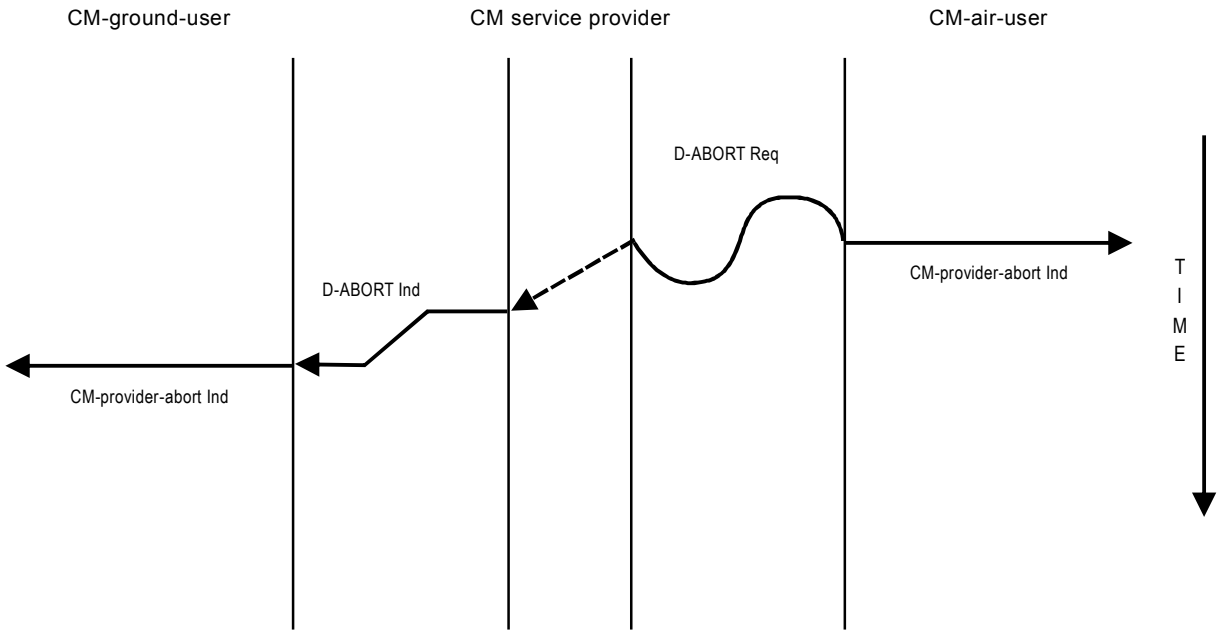


Figure 2-18. Sequence diagram for CM-provider-abort service — CM-air-ASE abort

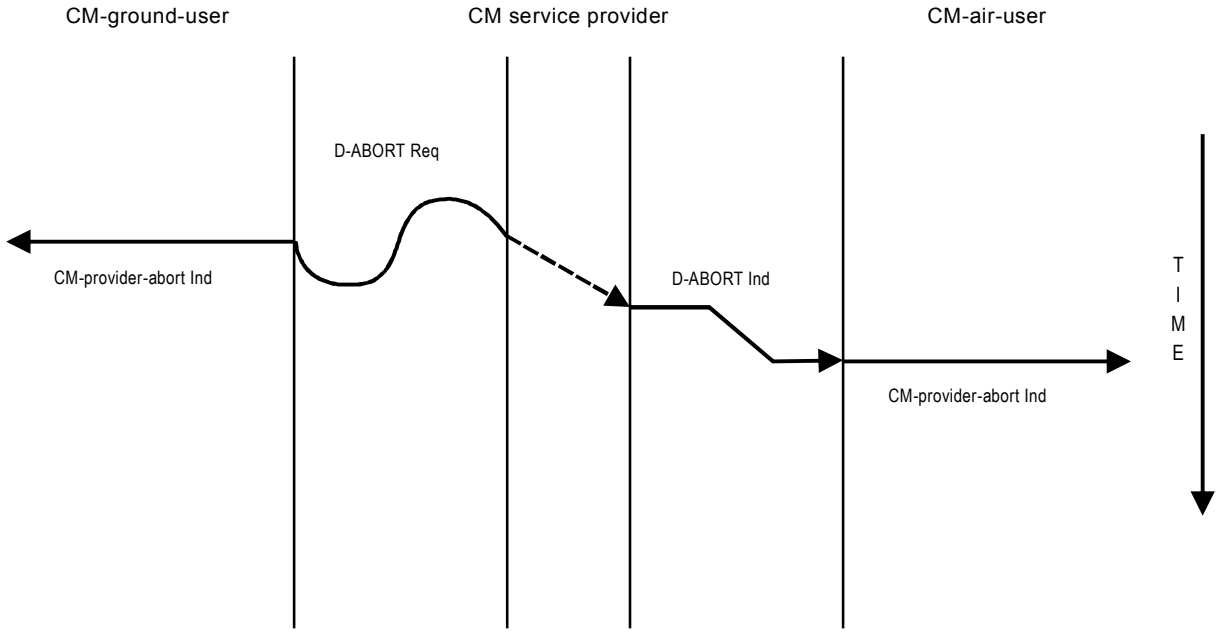


Figure 2-19. Sequence diagram for CM-provider-abort service — CM-ground-ASE abort

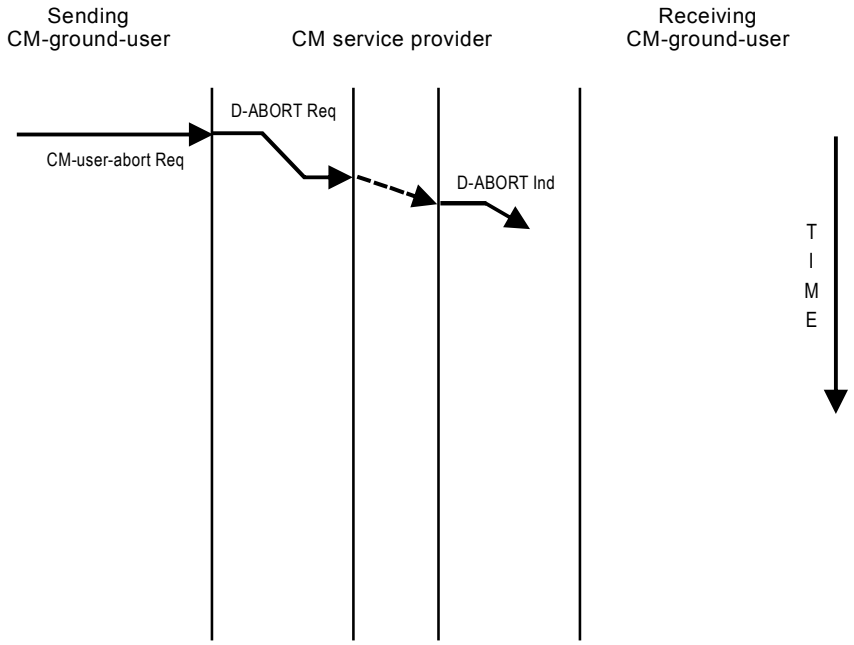


Figure 2-20. Sequence diagram for CM-user-abort service — Sending CM-ground-user initiated

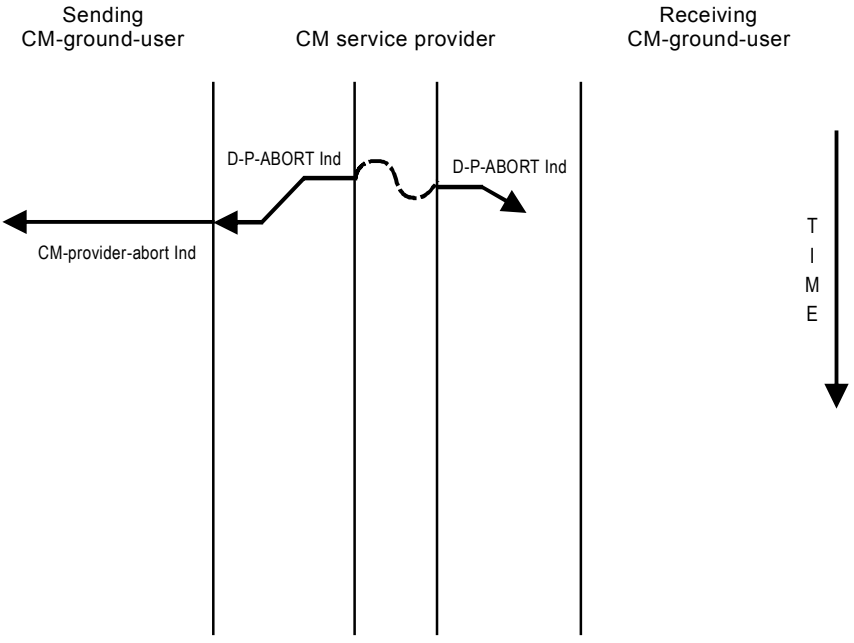


Figure 2-21. Sequence diagram for CM-provider-abort service — Dialogue service abort

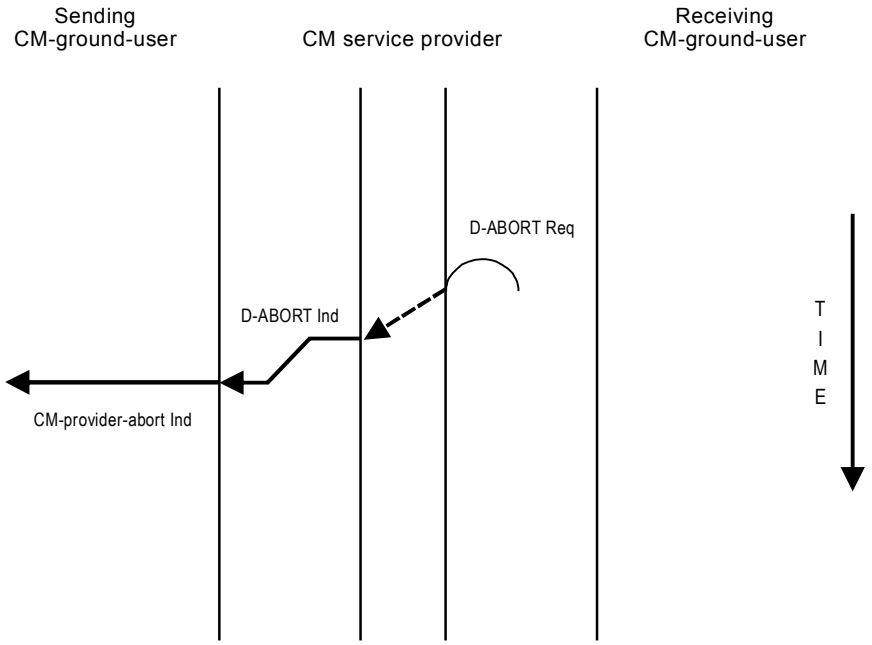


Figure 2-22. Sequence diagram for CM-provider-abort service — Receiving CM-ground-ASE abort

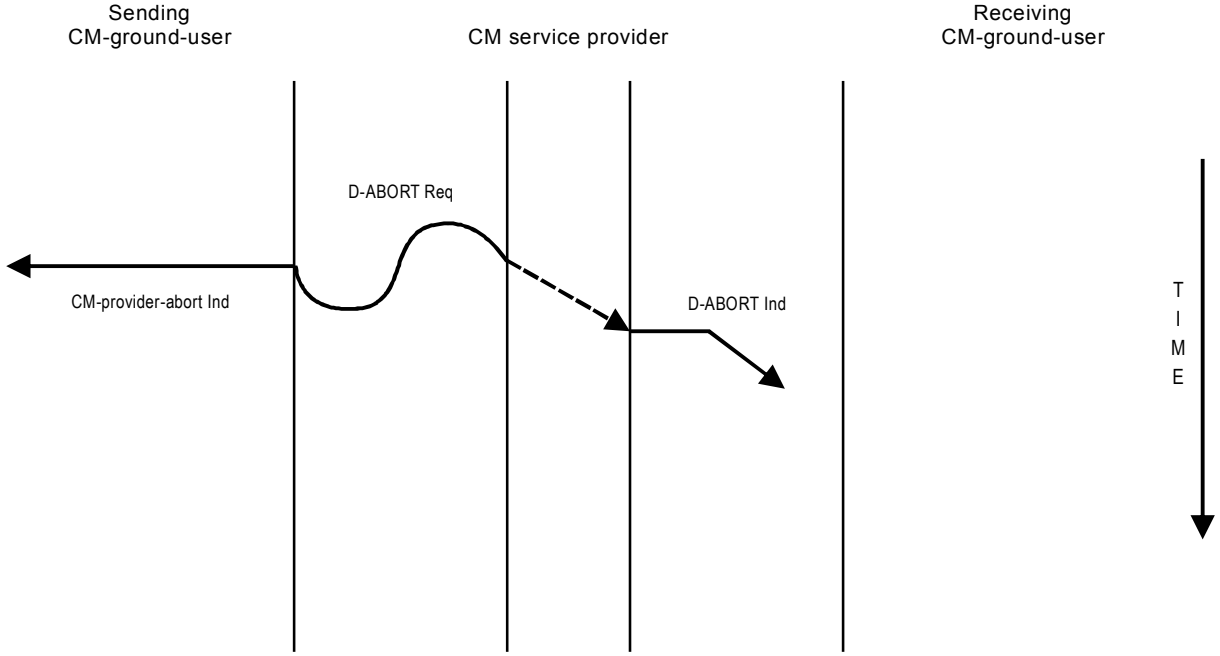


Figure 2-23. Sequence diagram for CM-provider-abort service — Sending CM-ground-ASE abort

2.5.1.3 Abort primitives may interrupt and terminate any of the normal message sequences outlined in the rest of Section 2.5.

2.5.1.4 More than one CM-logon attempt may be made for a given CM-contact request. The number of attempts may be determined by local procedures.

2.5.1.5 Primitives are processed in the order in which they are received.

2.5.2 CM service provider timer

2.5.2.1 A CM-ASE shall be capable of detecting when a timer expires.

2.5.2.2 Table 2-10 lists the time constraints related to the CM application. Each time constraint requires a timer to be set in the CM protocol machine.

2.5.2.3 If the timer expires before the final event has occurred, a CM-ASE takes the appropriate action as defined in 2.5.4.1.

2.5.2.4 The timer values should be as indicated in Table 2-10 for CM.

Table 2-10. CM service provider timers for CM

CM service	*Timer*	*Timer value*	*Timer start event*	*Timer stop event*
CM-logon	t_{logon}	4 minutes	D-START request	D-START confirmation
CM-update	t_{update}	4 minutes	D-START request	D-START confirmation
CM-contact	$t_{contact}$	8 minutes	D-START request, D-DATA request	D-START confirmation, D-DATA indication
CM-forward	$t_{forward}$	4 minutes	D-START request	D-START confirmation
CM-end	t_{end}	4 minutes	D-END request	D-END confirmation

Note — The receipts of CM-user-abort requests, D-ABORT indications or D-P-ABORT indications are also timer stop events.

2.5.3 CM-ASE protocol description

2.5.3.1 Introduction

Note.— Section 2.5.3 presents requirements for CM-ASEs in specific states. Section 2.5.5 contains state tables for the CM-ASEs.

2.5.3.1.1 If no actions are described for a CM service primitive when a CM-ASE is in a specific state, then the invocation of that service primitive shall be prohibited while the CM-ASE is in that state.

2.5.3.1.2 Upon receipt of a PDU or dialogue service primitive, if no actions are described for the arrival of that PDU or dialogue service primitive when a CM-ASE is in a specific state, then that PDU or dialogue service primitive is considered not permitted and exception handling procedures as described in 2.5.4.4 shall apply.

2.5.3.1.3 If a PDU is received that cannot be decoded, then that PDU is considered an invalid PDU and exception handling procedures as described in 2.5.4.3 shall apply.

2.5.3.1.4 If a PDU is not received when one is required, then that PDU is considered an invalid PDU and exception handling procedures as described in 2.5.4.3 shall apply.

2.5.3.2 CM-air-ASE protocol description

2.5.3.2.1 General

2.5.3.2.1.1 The states defined for the CM-air-ASE are the following:

a) IDLE;

b) LOGON;

c) CONTACT;

d) DIALOGUE; and

e) CONTACT DIALOGUE.

2.5.3.2.1.2 The CM-air-user is considered an active user from the time:

a) the CM-air-user invokes a CM-logon Req until it:

1) receives a CM-logon Cnf, if a dialogue is not maintained; or

2) receives a CM-end Ind, if a dialogue is maintained; or

3) receives a CM-user abort; or

4) receives a CM-provider abort; or

5) invokes a CM-user abort; or

b) the CM-air-user receives a CM-contact Ind until it:

1) invokes a CM-contact Rsp, if a dialogue is not maintained; or

2) receives a CM-user abort; or

3) receives a CM-provider abort; or

4) invokes a CM-user abort with the ground system that invoked the CM-contact service.

2.5.3.2.1.3 On initiation, the CM-air-ASE shall be in the *IDLE* state.

2.5.3.2.2 D-START indication

Upon receipt of a D-START indication, if the CM-air-ASE is in the *IDLE* state, and the D-START *Priority* Quality of Service (QOS) parameter has the abstract value "flight regularity communications", and the D-START *RER* QOS parameter has the abstract value of "low", and the D-START *Routing Class* QOS parameter identifies the traffic category "air traffic service communications (ATSC)", and the *Calling Peer ID* parameter is a valid four- to eight-character facility designation, and the D-START *Security Requirements* parameter is consistent with the local security policy, then:

a) if the APDU contained in the D-START *User Data* parameter is a CMGroundMesssage[CMUpdate] APDU, the CM-air-ASE shall:

   1) invoke CM-update service indication with:

      i) the D-START *Calling Peer ID* parameter value as the CM-update *Facility Designation* parameter value; and

      ii) the APDU contained in the D-START *User Data* parameter as the CM-update *Update Information* parameter value;

   2) invoke D-START response with the abstract value "rejected (permanent)" provided as the D-START *Result* parameter value; and

   3) enter the *IDLE* state; or

b) if the APDU contained in the D-START *User Data* parameter is a CMGroundMessage[CMContactRequest] APDU, the CM-air-ASE shall:

   1) invoke CM-contact service indication with:

      i) the D-START *Calling Peer ID* parameter value as the CM-contact *Facility Designation* parameter value; and

      ii) the APDU contained in the D-START *User Data* parameter as the CM-contact *Contact Request* parameter value; and

   2) enter the *CONTACT* state.

2.5.3.2.3 D-START confirmation

2.5.3.2.3.1 Upon receipt of a D-START confirmation, if the CM-air-ASE is in the *LOGON* state, and if the APDU contained in the D-START *User Data* parameter is either a CMGroundMessage[CMLogonResponse] APDU and the D-START confirmation Security Requirements parameter value is the same as was set for the D-START request, or if the D-START *User Data* parameter is not present but the D-START DS-User Version Number parameter is present, the CM-air-ASE shall:

a) stop timer t_{logon};

b) if the abstract value of the D-START *Result* parameter is "rejected (permanent)" and the abstract

value of the D-START *Reject Source* parameter is “DS-user”, then:

1) if the version number of the CM-air-ASE is greater than the D-START *DS-User Version Number* parameter value, invoke CM-logon service confirmation with the D-START *DS-User Version Number* parameter value as the CM-logon *CM-ASE Version Number* parameter value; or

2) if the version number of the CM-air-ASE is not greater than the D-START *DS-User Version Number* parameter value, invoke CM-logon service confirmation with the APDU contained in the D-START *User Data* parameter as the CM-logon *Logon Response* parameter; and

3) enter the *IDLE* state; and

c) if the abstract value of the D-START *Result* parameter is “accepted”, then:

1) invoke CM-logon service confirmation with:

i) the APDU contained in the D-START *User Data* parameter as the CM-logon *Logon Response* parameter value; and

ii) the abstract value of “maintain” as the CM-logon *Maintain Dialogue* parameter value; and

2) enter the *DIALOGUE* state.

Note.— If a secure D-START request was issued (i.e. the Security Requirements *parameter was set to “secured dialogue supporting security information management”), then the D-START confirmation* Security Requirements *parameter must be equal to that value. If an unsecure D-START request was issued (i.e. the* Security Requirements *parameter was set to “no security”), then the D-START confirmation* Security Requirements *parameter will be equal to that value. A peer CM may not set the* Security Requirements *parameter. However, the local upper layers will convert the absent parameter into a “no security” value. Therefore, the check to see that the received D-START confirmation* Security Requirements *parameter value is the same as the D-START request* Security Requirements *parameter value will work for all CM peers.*

2.5.3.2.4 D-DATA indication

2.5.3.2.4.1 Upon receipt of a D-DATA indication, if the CM-air-ASE is in the *DIALOGUE* state, then:

a) if the APDU contained in the D-DATA *User Data* parameter is a CMGroundMessage[CMUpdate] APDU, the CM-air-ASE shall:

1) invoke CM-update service indication with the APDU contained in the D-DATA *User Data* parameter as the CM-update service *Update Information* parameter value; and

2) remain in the *DIALOGUE* state; or

b) if the APDU contained in the D-DATA *User Data* parameter is a CMGroundMessage[CMContactRequest] APDU, the CM-air-ASE shall:

1) invoke CM-contact service indication with the APDU contained in the D-DATA *User Data* parameter as the CM-contact service *Contact Request* parameter value; and

2) enter the *CONTACT DIALOGUE* state.

2.5.3.2.5 D-END indication

2.5.3.2.5.1 Upon receipt of a D-END indication, if the CM-air-ASE is in the *DIALOGUE* state, then the CM-air-ASE shall:

a) invoke CM-end service indication;

b) invoke D-END response with the D-END *Result* parameter set to the abstract value “accepted”; and

c) enter the *IDLE* state.

2.5.3.2.6 CM-logon service request

2.5.3.2.6.1 Upon receipt of a CM-logon service request, if the CM-air-ASE is in the *IDLE* state, the CM-air-ASE shall:

a) create a CMAircraftMessage APDU with a cmLogonRequest APDU element based on the *Logon Request* parameter value;

b) invoke D-START request with:

1) the CM-logon *Facility Designation* parameter value as the D-START *Called Peer ID* parameter value;

2) the CM-logon *Aircraft Address* parameter value as the D-START *Calling Peer ID* parameter value;

3) the D-START *Quality of Service* parameters set as follows:

i) if provided, the CM-logon service *Class of Communication Service* parameter value as the D-START *Routing Class* parameter value;

ii) the abstract value of “flight regularity communications” as the D-START *Priority* parameter value; and

iii) the abstract value of “low” as the D-START *RER* parameter value;

5) the CM-logon *Security Required* parameter value as the D-START *Security Requirements* parameter value; and

6) the CMAircraftMessage APDU as the D-START *User Data* parameter value;

c) start timer t_{logon}; and

d) enter the *LOGON* state.

2.5.3.2.7 CM-contact service response

2.5.3.2.7.1 Upon receipt of a CM-contact service response, if the CM-air-ASE is in the *CONTACT* state, the CM-air-ASE shall:

a) create a CMAircraftMessage APDU with cmContactResponse APDU element based on the CM-contact *Contact Response* parameter value;

b) invoke D-START response with:

1) the abstract value “rejected (permanent)” as D-START *Result* parameter value;

2) the D-START *Security Requirements* parameter value set to the same value as was received in the D-START indication; and

3) the CMAircraftMessage APDU as the D-START *User Data* parameter value; and

c) enter the *IDLE* state.

2.5.3.2.7.2 Upon receipt of a CM-contact service response, if the CM-air-ASE is in the *CONTACT DIALOGUE* state, the CM-air-ASE shall:

a) create a CMAircraftMessage APDU with a cmContactResponse APDU element based on the CM-contact *Contact Response* parameter value;

b) invoke D-DATA request with the CMAircraftMessage APDU as the D-DATA *User Data* parameter value; and

c) enter the *DIALOGUE* state.

2.5.3.2.8 CM-user-abort service request

Upon receipt of a CM-user-abort service request, if the CM-air-ASE is not in the *IDLE* state, the CM-air-ASE shall:

a) stop timer t_{logon} if set;

b) invoke D-ABORT request with the D-ABORT *Originator* parameter set to the abstract value “user”; and

c) enter the *IDLE* state.

2.5.3.2.9 D-ABORT indication

Upon receipt of a D-ABORT indication, if the CM-air-ASE is not in the *IDLE* state, the CM-air-ASE shall:

a) stop timer t_{logon} if set;

b) if the CM-air-user is an active user, then:

1) if the D-ABORT *Originator* parameter contains the abstract value “user”, invoke CM-user-abort service indication; or

2) if the D-ABORT *Originator* parameter does not contain the abstract value “user”, invoke CM-provider-abort service indication with the APDU contained in the D-ABORT *User Data* parameter as the CM-provider-abort service *Reason* parameter value; and

c) enter *IDLE* state.

2.5.3.2.10 D-P-ABORT indication

Upon receipt of a D-P-ABORT indication, if the CM-air-ASE is not in the *IDLE* state, the CM-air-ASE shall:

a) stop timer t_{logon} if set;

b) if the CM-air-user is an active user, invoke CM-provider-abort service indication with the CM-provider-abort *Reason* parameter set to the abstract value “communication-service-failure”; and

c) enter the *IDLE* state.

2.5.3.3 CM-ground-ASE protocol description

2.5.3.3.1 General

2.5.3.3.1.1 The states defined for the CM-ground-ASE are the following:

a) IDLE;

b) LOGON;

c) UPDATE;

d) CONTACT;

e) DIALOGUE;

f) CONTACT DIALOGUE;

g) END; and

h) FORWARD.

2.5.3.3.1.2 The CM-ground-user is considered an active user from the time:

a) the CM-ground-user receives a CM-logon Ind until it:

1) invokes a CM-logon Rsp, if a dialogue is not maintained; or

2) invokes a CM-end Req, if a dialogue is maintained; or

3) receives a CM-user abort; or

4) receives a CM-provider abort; or

5) invokes a CM-user abort; or

b) the CM-ground-user invokes a CM-contact Req until it:

1) receives a CM-contact Cnf, if a dialogue is not maintained; or
2) receives a CM-user abort; or

3) receives a CM-provider abort; or

4) invokes a CM-user abort; or

c) the CM-ground-user invokes a CM-forward Req until it:

1) receives a CM-forward Cnf; or

2) receives a CM-provider abort; or

3) invokes a CM-user abort.

2.5.3.3.1.3 On initiation, the CM-ground-ASE shall be in the *IDLE* state.

2.5.3.3.2 D-START indication

2.5.3.3.2.1 Upon receipt of a D-START indication, if the CM-ground-ASE is in the *IDLE* state, and the abstract value of the D-START *Calling Peer ID* parameter is the ICAO 24-bit aircraft address, and the D-START *Priority* QOS parameter has the abstract value "flight regularity communications", and the D-START *RER* QOS parameter has the abstract value of "low", and the D-START *Routing Class* QOS parameter identifies the traffic category "air traffic service communications (ATSC)", and the D-START *Security Requirements* parameter is consistent with the local security policy, then:

a) if the D-START *DS-User Version Number* parameter value is greater than the CM-ground-ASE version number, the CM-ground-ASE shall:

1) invoke D-START response with:

i) the CM-ground-ASE version number as the D-START *DS-User Version Number* parameter value; and

ii) the abstract value of "rejected (permanent)" as the D-START *Result* parameter value; and

2) enter the *IDLE* state; or

Note.— The D-START Security Requirements *parameter is not explicitly set by the CM-ground-ASE in the case that the aircraft's CM version number is greater than the ground's CM version number.*

b) if the D-START *DS-User Version Number* parameter value is less than the CM-ground-ASE version number, and the APDU contained in the D-START *User Data* parameter is a CMAircraftMessage[CMLogonRequest] APDU, the CM-ground-ASE shall:

1) invoke CM-logon service indication with:

i) the D-START *Calling Peer ID* parameter value as the CM-logon service *Aircraft Address* parameter value;

ii) the D-START *Security Requirements* parameter value as the CM-logon service *Security Required* parameter value;

iii) the D-START *DS-User Version Number* parameter value as the CM-logon service *CM-ASE Version Number* parameter value; and

iv) the APDU in the D-START *User Data* parameter as the CM-logon service *Logon Request* parameter value; and

2) enter the *LOGON* state; or

c) if the D-START *DS-User Version Number* parameter value is equal to CM-ground-ASE version number, and the APDU contained in the D-START *User Data* parameter is a CMAircraftMessage[CMLogonRequest] APDU, the CM-ground-ASE shall:

1) invoke CM-logon service indication with:

i) the D-START *Calling Peer ID* parameter value as the CM-logon service *Aircraft Address* parameter value;

ii) the D-START *Security Requirements* parameter value as the CM-logon service *Security Required* parameter value; and

iii) the APDU in the D-START *User Data* parameter as the CM-logon service *Logon Request* parameter value; and

2) enter the *LOGON* state.

2.5.3.3.2.2 Upon receipt of a D-START indication, if the receiving CM-ground-ASE is in the *IDLE* state, and the APDU contained in the D-START *User Data* parameter is a CMGroundMessage[CMForwardRequest] APDU, and the abstract value of the D-START *Calling Peer ID* parameter is a four- to eight-character facility designation, and the D-START *Priority* QOS parameter has the abstract value “flight regularity communications”, and the D-START *RER* QOS parameter has the abstract value of “low”, and the D-START *Routing Class* QOS parameter identifies the traffic category “air traffic service communications (ATSC)”, then:

a) if the CM-ground-ASE does not support the CM-forward service, the CM-ground-ASE shall:

1) create a CMGroundMessage APDU with a cmForwardResponse [service-not-supported] APDU message element;

2) invoke D-START response with:

i) the APDU as the D-START *User Data* parameter value; and

ii) the abstract value “rejected (permanent)” as the D-START *Result* parameter value; and

3) remain in the *IDLE* state; or

Note.— The D-START Security Requirements *parameter is not explicitly set by the CM-ground-ASE if the CM-forward service is not supported.*

b) if the D-START *DS-User Version Number* parameter value is greater than the CM-ground-ASE version number and the D-START *Security Requirements* parameter is consistent with the local security policy and the CM-ground-ASE supports the CM-forward service, the CM-ground-ASE shall:

1) create a CMGroundMessage APDU with a cmForwardResponse [incompatible-version] APDU message element;

2) invoke D-START response with:

   i) the CM-ground-ASE version number as the D-START *DS-User Version Number* parameter value;

   ii) the APDU as the D-START *User Data* parameter value;

   iii) the abstract value of “rejected (permanent)” as the D-START *Result* parameter value; and

   iv) the D-START *Security Requirements* parameter value set to the same value as was received in the D-START indication; and

3) remain in the *IDLE* state; or

c) if the D-START *DS-User Version Number* parameter value is less than the CM-ground-ASE version number and the D-START *Security Requirements* parameter is consistent with the local security policy and the CM-ground-ASE supports the CM-forward service, the CM-ground-ASE shall:

1) invoke CM-forward service indication with:

   i) the D-START *Calling Peer ID* parameter value as the CM-forward service *Calling Facility Designation* parameter value;

   ii) the D-START *DS-User Version Number* parameter value as the CM-forward service *CM-ASE Version Number* parameter value; and

   iii) the APDU in the D-START *User Data* parameter as the CM-forward service *Forward Request* parameter value;

2) create a CMGroundMessage APDU with a cmForwardResponse [success] APDU message element;

3) invoke D-START response with:

   i) the APDU as the D-START *User Data* parameter value;

   ii) the D-START *Security Requirements* parameter value set to the same value as was received in the D-START indication; and

   iii) the abstract value of “rejected (permanent)” as the D-START *Result* parameter value; and

4) remain in the *IDLE* state; or

Note.— This assumes that CM-ASEs are backwards compatible.

d) if the D-START *DS-User Version Number* parameter value is equal to the CM-ground-ASE version number and the D-START *Security Requirements* parameter is consistent with the local security policy and the CM-ground-ASE supports the CM-forward service, the CM-ground-ASE shall:

1) invoke CM-forward service indication with:

   i) the D-START *Calling Peer ID* parameter value as the CM-forward service *Calling Facility Designation* parameter value; and

   ii) the APDU in the D-START *User Data* parameter as the CM-forward service *Forward Request* parameter value;

2) create a CMGroundMessage APDU with a cmForwardResponse [success] APDU message element;

3) invoke D-START response with:

   i) the APDU as the D-START *User Data* parameter value;

   ii) the D-START *Security Requirements* parameter value set to the same value as was received in the D-START indication; and

   iii) the abstract value of "rejected (permanent)" as the D-START *Result* parameter value; and

4) remain in the *IDLE* state.

2.5.3.3.3 D-START confirmation

2.5.3.3.3.1 Upon receipt of a D-START confirmation, if the CM-ground-ASE is in the *UPDATE* state and the D-START *User Data* parameter is not provided, the CM-ground-ASE shall:

a) stop timer t_{update}; and

b) if the abstract value of the D-START *Result* parameter is "rejected (permanent)", and the abstract value of the D-START *Reject Source* parameter is "DS-user" andthe abstract value of the D-START *Security Requirements* parameter is the same as was set in the D-START request, enter the *IDLE* state.

Note.— If a secure D-START request was issued (i.e. the Security Requirements *parameter was set to "secured dialogue supporting security information management" or "secured dialogue"), then the D-START confirmation* Security Requirements *parameter must be equal to that value. If an unsecure D-START request was issued (i.e. the* Security Requirements *parameter was set to "no security"), then the D-START confirmation* Security Requirements *parameter will be equal to that value. A peer CM may not set the* Security Requirements *parameter. However, the local upper layers will convert the absent parameter into a "no security" value. Therefore, the check to see that the received D-START confirmation* Security Requirements *parameter value is the same as the D-START request* Security Requirements *parameter value will work for all CM peers.*

2.5.3.3.3.2 Upon receipt of a D-START confirmation, if the CM-ground-ASE is in the *CONTACT* state, and the APDU contained in the D-START *User Data* parameter is a CMAircraftMessage[CMContactResponse] APDU, the CM-ground-ASE shall:

a) stop timer $t_{contact}$; and

b) if the abstract value of the D-START *Result* parameter is "rejected (permanent)", and the abstract value of the D-START *Reject Source* parameter is "DS-user" and the abstract value of the D-START *Security Requirements* parameter is the same as was set in the D-START request then:

1) invoke CM-contact service confirmation with the APDU in the D-START *User Data* parameter as the CM-contact *Contact Response* parameter value; and

2) enter the *IDLE* state.

Note.— If a secure D-START request was issued (i.e. the Security Requirements *parameter was set to "secured dialogue supporting security information management" or "secured dialogue"), then the D-START confirmation* Security Requirements *parameter must be equal to that value. If an unsecure D-START request was issued (i.e. the* Security Requirements *parameter was set to "no security"), then the D-START confirmation* Security Requirements *parameter will be equal to that value. A peer CM may not set the* Security Requirements *parameter. However, the local upper layers will convert the absent parameter into a "no security" value. Therefore, the check to see that the received D-START confirmation* Security Requirements *parameter value is the same as the D-START request* Security Requirements *parameter value will work for all CM peers.*

2.5.3.3.3.3 Upon receipt of a D-START confirmation, if the CM-ground-ASE is in the *FORWARD* state, and the D-START *User Data* parameter is a CMGroundMessage[CMForwardResponse] APDU, and the abstract value of the D-START *Result* parameter is "rejected (permanent)", and the abstract value of the D-START *Reject Source* parameter is "DS-user", the CM-ground-ASE shall:

a) stop timer $t_{forward}$;

b) if the abstract value of the D-START *User Data* parameter is "service-not-supported" and the abstract value of the D-START confirmation *Security Requirements* parameter is the same as was set in the D-START request, invoke a CM-forward service confirmation with the CM-forward *Result* parameter set to the abstract value "service-not-supported";

c) if the abstract value of the D-START *User Data* parameter is "incompatible-version" and the abstract value of the D-START confirmation *Security Requirements* parameter is the same as was set in the D-START request, invoke a CM-forward service confirmation with the *DS-User Version Number* parameter value as the CM-forward *CM-ASE Version Number* parameter and set the CM-forward *Result* parameter abstract value to "incompatible-version";

d) if the abstract value of the D-START *User Data* parameter is "success" and the abstract value of the D-START confirmation *Security Requirements* parameter is the same as was set in the D-START request, invoke a CM-forward service confirmation with the CM-forward *Result* parameter set to the abstract value "success"; and

e) enter the *IDLE* state.

Note.— If a secure D-START request was issued (i.e. the Security Requirements *parameter was set to "secured dialogue supporting security information management"), then the D-START confirmation* Security Requirements *parameter must be equal to that value. If an unsecure D-START request was issued (i.e. the* Security Requirements *parameter was set to "no security"), then the D-START confirmation* Security Requirements *parameter will be equal to that value. A peer CM may not set the* Security Requirements *parameter. However, the local upper layers will convert the absent parameter into a "no security" value. Therefore, the check to see that the received D-START confirmation* Security Requirements *parameter value is the same as the D-START request* Security Requirements *parameter value will work for all CM peers.*

2.5.3.3.4 D-DATA indication

2.5.3.3.4.1 Upon receipt of a D-DATA indication, if the CM-ground-ASE is in the *CONTACT DIALOGUE* state, and the

APDU contained in the D-DATA *User Data* parameter is a CMAircraftMessage[CMContactResponse] APDU, the CM-ground-ASE shall:

a) stop timer $t_{contact}$;

b) invoke CM-contact service confirmation with the APDU contained in the D-DATA *User Data* parameter as the CM-contact *Contact Response* parameter value; and

c) enter the *DIALOGUE* state.

2.5.3.3.5 D-END confirmation

Upon receipt of a D-END confirmation, if the CM-ground-ASE is in the *END* state and the abstract value of the D-END *Result* is "accepted", the CM-ground-ASE shall:

a) stop timer t_{end}; and

b) enter the *IDLE* state.

2.5.3.3.6 CM-logon service response

2.5.3.3.6.1 Upon receipt of a CM-logon service response, if the CM-ground-ASE is in the *LOGON* state, the CM-ground-ASE shall:

a) create a CMGroundMessage APDU with a cmLogonResponse APDU element based on the CM-logon *Logon Response* parameter value; and

b) invoke D-START response with the CMGroundMessage APDU as the D-START User Data parameter value; and

c) if the CM-logon *Maintain Dialogue* parameter is provided by the CM-ground-user:

1) set the abstract value "accepted" as the D-START *Result* parameter value; and

2) enter the *DIALOGUE* state; or

d) if the CM-logon *Maintain Dialogue* parameter is not provided by the CM-ground-user:

1) set the abstract value "rejected (permanent)" as the D-START *Result* parameter value; and

2) enter the *IDLE* state.

2.5.3.3.7 CM-update service request

2.5.3.3.7.1 Upon receipt of a CM-update service request, if the CM-ground-ASE is in the *IDLE* state, the CM-ground-ASE shall:

a) create a CMGroundMessage APDU with a cmUpdate APDU element based on the CM-update *Update Information* parameter value;

b) invoke D-START request with:

1) the CM-update *Aircraft Address* parameter value as the D-START *Called Peer ID* parameter

value;

2) the CM-update *Facility Designation* parameter value as the D-START *Calling Peer ID* parameter value;
3) the D-START *Quality of Service* parameters set as follows:

i) if provided, the CM-update service *Class of Communication Service* parameter value as the D-START *Routing Class* parameter value;

ii) the abstract value of “flight regularity communications” as the D-START *Priority* parameter value; and

iii) the abstract value of “low” as the D-START *RER* parameter value;

4) the CMGroundMessage APDU as the D-START *User Data* parameter value; and

5) the CM-update *Security Required* parameter value as the D-START *Security Requirements* parameter value;

d) start timer t_{update}; and

e) enter the *UPDATE* state.

2.5.3.3.7.2 Upon receipt of a CM-update service request, if the CM-ground-ASE is in the *DIALOGUE* state, the CM-ground-ASE shall:

a) create a CMGroundMessage APDU with a cmUpdate APDU element based on the CM-update *Update Information* parameter value;

b) invoke D-DATA request with the CMGroundMessage APDU as the D-DATA *User Data* parameter value; and

c) remain in the *DIALOGUE* state.

2.5.3.3.8 CM-contact service request

2.5.3.3.8.1 Upon receipt of a CM-contact service request, if the CM-ground-ASE is in the *IDLE* state, the CM-ground-ASE shall:

a) create a CMGroundMessage APDU with a cmContactRequest APDU element based on the CM-contact *Contact Request* parameter value;

b) invoke D-START request with:

1) the CM-contact *Aircraft Address* parameter value as the D-START *Called Peer ID* parameter value;

2) the CM-contact *Facility Designation* parameter value as the D-START *Calling Peer ID* parameter value;

3) the D-START *Quality of Service* parameters set as follows:

i) if provided, the CM-contact service *Class of Communication Service* parameter value as the D-START *Routing Class* parameter value;

ii) the abstract value of “flight regularity communications” as the D-START *Priority* parameter value; and

iii) the abstract value of “low” as the D-START *RER* parameter value;

4) the CM-contact *Security Required* parameter as the D-START *Security Requirements* parameter value; and

5) the CMGroundMessage APDU as the D-START *User Data* parameter value;

c) start timer $t_{contact}$; and

d) enter the *CONTACT* state.

2.5.3.3.8.2 Upon receipt of a CM-contact service request, if the CM-ground-ASE is in the *DIALOGUE* state, the CM-ground-ASE shall:

a) create a CMGroundMessage APDU with a cmContactRequest APDU element based on the CM-contact *Contact Request* parameter value;

b) invoke D-DATA request with the CMGroundMessage APDU as the D-DATA *User Data* parameter value;

c) start timer $t_{contact}$; and

d) enter the *CONTACT DIALOGUE* state.

2.5.3.3.9 CM-end service request

Upon receipt of a CM-end service request, if the CM-ground-ASE is in the *DIALOGUE* state, the CM-ground-ASE shall:

a) invoke D-END request;

b) start timer t_{end}; and

c) enter the *END* state.

2.5.3.3.10 CM-forward service request

Upon receipt of a CM-forward service request, if the CM-ground-ASE is in the *IDLE* state, the CM-ground-ASE shall:

a) create a CMGroundMessage APDU with a cmForwardRequest APDU element based on the CM-forward service *Forward Request* parameter value;

b) invoke D-START request with:

1) the CM-forward *Called Facility Designation* parameter value as the D-START *Called Peer ID* parameter value;

2) the CM-forward *Calling Facility Designation* parameter value as the D-START *Calling Peer ID* parameter value;

3) the CM-forward *Security Required* parameter value as the D-START *Security Requirements* parameter value;

4) the D-START *Quality of Service* parameters set as follows:

i) if provided, the CM-forward service *Class of Communication Service* parameter value as the D-START *Routing Class* parameter value;

ii) the abstract value of "flight regularity communications" as the D-START *Priority* parameter value; and

iii) the abstract value of "low" as the D-START *RER* parameter value; and

5) the CMGroundMessage APDU as the D-START *User Data* parameter value;

c) start timer $t_{forward}$; and

d) enter the *FORWARD* state.

2.5.3.3.11 CM-user-abort service request

Upon receipt of a CM-user-abort service request, if the CM-ground-ASE is not in the *IDLE* state, the CM-ground-ASE shall:

a) stop any timer, if set;

b) invoke D-ABORT request with the D-ABORT *Originator* parameter set to the abstract value "user"; and

c) enter the *IDLE* state.

2.5.3.3.12 D-ABORT indication

Upon receipt of a D-ABORT indication, if the CM-ground-ASE is not in the *IDLE* state, the CM-ground-ASE shall:

a) stop any timer, if set;

b) if the CM-ground-user is an active user, then:

1) if the D-ABORT *Originator* parameter contains the abstract value "user", invoke CM-user-abort service indication; or

2) if the D-ABORT *Originator* parameter does not contain the abstract value "user", invoke CM-provider-abort service indication with the APDU contained in the D-ABORT *User Data* parameter as the CM-provider-abort service *Reason* parameter value; and

c) enter *IDLE* state.

2.5.3.3.13 D-P-ABORT indication

Upon receipt of a D-P-ABORT indication, if the CM-ground-ASE is not in the *IDLE* state, the CM-ground-ASE shall:

a) stop any timer, if set;

b) if the CM-ground-user is an active user, invoke CM-provider-abort service indication with the CM-provider-abort *Reason* parameter set to the abstract value “communication-service-failure”; and

c) enter the *IDLE* state.

2.5.4 Exception handling

2.5.4.1 Timer expiration

If a CM-ASE detects that a timer has expired, that CM-ASE shall:

a) stop all timers;

b) interrupt any current activity;

c) if the CM-ASE is a CM-air-ASE, create a CMAircraftMessage APDU with a cmAbortReason [timer-expired] APDU message element;

d) if the CM-ASE is a CM-ground-ASE, create a CMGroundMessage APDU with a cmAbortReason [timer-expired] APDU message element;

e) invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

f) if the CM-user is an active user, invoke CM-provider-abort service indication with the abstract value “timer-expired” as the CM-provider-abort *Reason* parameter value; and

g) enter the *IDLE* state.

2.5.4.2 Unrecoverable system error

If a CM-ASE has an unrecoverable system error, the CM-ASE should:

a) stop all timers;

b) if the CM-ASE is a CM-air-ASE, create a CMAircraftMessage APDU with a cmAbortReason

[undefined-error] APDU message element;

c) if the CM-ASE is a CM-ground-ASE, create a CMGroundMessage APDU with a cmAbortReason [undefined-error] APDU message element;
d) invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

e) if the CM-user is an active user, invoke CM-provider-abort service indication with the abstract value “undefined-error” as the CM-provider-abort Reason parameter value; and

f) enter the *IDLE* state.

2.5.4.3 Invalid PDU

2.5.4.3.1 If the *User Data* parameter of a D-START indication or D-DATA indication does not contain a valid PDU as defined in 2.5.3.1.3 and 2.5.3.1.4, the CM-ASE shall:

a) stop all timers;

b) if the CM-ASE is a CM-air-ASE, create a CMAircraftMessage APDU with a cmAbortReason [invalid-PDU] APDU message element;

c) if the CM-ASE is a CM-ground-ASE, create a CMGroundMessage APDU with a cmAbortReason [invalid-PDU] APDU message element;

d) invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

e) if the CM-user is an active user, invoke CM-provider-abort service indication with the abstract value “invalid-PDU” as the CM-provider-abort *Reason* parameter value; and

f) enter the *IDLE* state.

2.5.4.3.2 If the *User Data* parameter of a D-START confirmation does not contain a valid PDU as defined in 2.5.3.1.3 and 2.5.3.1.4, the CM-ASE shall:

a) stop all timers;

b) if the CM-ASE is a CM-air-ASE and if the D-START *Result* parameter is set to the abstract value “accepted”, then:

1) create a CMAircraftMessage APDU with a cmAbortReason [invalid-PDU] APDU message element; and

2) invoke D-ABORT request with:

i) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

ii) the APDU as the D-ABORT *User Data* parameter value;

c) if the CM-user is an active user, invoke CM-provider-abort service indication with the abstract value “invalid-PDU” as the CM-provider-abort *Reason* parameter value; and

d) enter the *IDLE* state.

2.5.4.4 Not permitted PDU or DS primitive

2.5.4.4.1 If the *User Data* parameter of a D-START indication or D-DATA indication is a valid PDU; however, it is not a permitted PDU as defined within 2.5.3.1.2, the CM-ASE shall:

a) stop all timers;

b) if the CM-ASE is a CM-air-ASE, create a CMAircraftMessage APDU with a cmAbortReason [protocol-error] APDU message element;

c) if the CM-ASE is a CM-ground-ASE, create a CMGroundMessage APDU with a cmAbortReason [protocol-error] APDU message element;

d) invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

e) if the CM-user is an active user, invoke CM-provider-abort service indication with the abstract value “protocol-error” as the CM-provider-abort *Reason* parameter value; and

f) enter the *IDLE* state.

2.5.4.4.2 If the *User Data* parameter of a D-START confirmation is a valid PDU; however, it is not a permitted PDU as defined within 2.5.3.1.2, the CM-ASE shall:

a) stop all timers;

b) if the D-START *Result* parameter is set to the abstract value “accepted”:

1) if the CM-ASE is a CM-air-ASE, create a CMAircraftMessage APDU with a cmAbortReason [protocol-error] APDU message element;

2) if the CM-ASE is a CM-ground-ASE, create a CMGroundMessage APDU with a cmAbortReason [protocol-error] APDU message element; and

3) invoke D-ABORT request with:

i) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

ii) the APDU as the D-ABORT *User Data* parameter value;

c) if the CM-user is an active user, invoke CM-provider-abort service indication with the abstract value “protocol-error” as the CM-provider-abort *Reason* parameter value; and

d) enter the *IDLE* state.

2.5.4.4.3 Upon receipt of a DS primitive for which there are no instructions in 2.5.3 (i.e. the primitive was not expected or was expected under other conditions or with other parameter values), the CM-ASE shall:

a) stop all timers;

b) if the CM-ASE is a CM-air-ASE, create a CMAircraftMessage APDU with a cmAbortReason [protocol-error] APDU message element;

c) if the CM-ASE is a CM-ground-ASE, create a CMGroundMessage APDU with a cmAbortReason [protocol-error] APDU message element;

d) if a dialogue exists, invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

e) if the CM-user is an active user, invoke CM-provider-abort service indication with the abstract value “protocol-error” as the CM-provider-abort *Reason* parameter value; and

f) enter the *IDLE* state.

2.5.4.5 D-START confirmation Result or Reject Source parameter values not as expected

2.5.4.5.1 If the CM-ground-ASE receives a D-START confirmation with the D-START *Result* parameter having the abstract value of “accepted”, the CM-ground-ASE shall:

a) stop all timers;

b) create a CMGroundMessage APDU with a cmAbortReason [dialogue-acceptance-not-permitted] APDU message element;

c) invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

d) if the CM-ground-user is an active user, invoke CM-provider-abort service indication with the abstract value “dialogue-acceptance-not-permitted” as the CM-provider-abort *Reason* parameter value; and

e) enter the *IDLE* state.

2.5.4.5.2 If the CM-ASE receives a D-START confirmation with the D-START *Result* parameter having the abstract value of “rejected (transient)”, or if the D-START *Reject Source* parameter has the abstract value of “DS-provider”, the

CM-ASE shall:

a) stop all timers;

b) if the CM-user is an active user, invoke CM-provider-abort service indication with the abstract value "communication-service-error" APDU as the CM-provider-abort *Reason* parameter value; and

c) enter the *IDLE* state.

2.5.4.6 D-END confirmation not as expected

If the CM-ground-ASE receives a D-END confirmation with the D-END *Result* parameter that does not have the abstract value of "accepted", the CM-ground-ASE shall:

a) stop all timers;

b) create a CMGroundMessage APDU with a cmAbortReason [dialogue-end-not-accepted] APDU message element;

c) invoke D-ABORT request with:

1) the abstract value "provider" as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value; and

d) enter the *IDLE* state.

2.5.4.7 D-START indication Quality of Service parameter not as expected

If the abstract value of the D-START *Priority* QOS parameter is not "flight regularity communications", or the abstract value of the D-START *RER* QOS parameter is not "low", or the abstract value of the D-START *Routing Class* QOS parameter does not identify the traffic category "air traffic service communications (ATSC)", the CM-ASE shall:

a) stop all timers;

b) if the CM-ASE is a CM-air-ASE, create a CMAircraftMessage APDU with a CMAbortReason [invalid-QOS-parameter] APDU message element;

c) if the CM-ASE is a CM-ground-ASE, create a CMGroundMessage APDU with a CMAbortReason [invalid-QOS-parameter] APDU message element;

d) invoke D-ABORT request with:

1) the abstract value "provider" as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value; and

e) enter the *IDLE* state.

2.5.4.8 Expected PDU not provided

2.5.4.8.1 If the *User Data* parameter of a D-START indication, D-START confirmation (with the *Result* parameter set to the abstract value “accepted”), or D-DATA indication is not provided where it is expected, the CM-ASE shall:

a) stop all timers;

b) if the CM-ASE is a CM-air-ASE, create a CMAircraftMessage APDU with a cmAbortReason [expected-PDU-missing] APDU message element;

c) if the CM-ASE is a CM-ground-ASE, create a CMGroundMessage APDU with a cmAbortReason [expected-PDU-missing] APDU message element;

d) invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

e) invoke a CM-provider-abort service indication with the abstract value “expected-PDU-missing”; and

f) enter the *IDLE* state.

2.5.4.8.2 If the *User Data* parameter of a D-START confirmation (with the *Result* parameter set to the abstract value “rejected (transient)” or “rejected (permanent)”) is not provided where it is expected, the CM-ASE shall:

a) stop all timers;

b) invoke a CM-provider-abort service indication with the abstract value “expected-PDU-missing”; and

c) enter the *IDLE* state.

2.5.4.9 D-START Security Requirements parameter not as expected

Note.— This is for the case when the D-START Security Requirements *parameter does not meet the local security policy.*

2.5.4.9.1 Upon receipt of a D-START indication with the *Security Requirements* parameter not consistent with the local security policy, or upon receipt of a D-START confirmation with the *Security Requirements* parameter not equal to the value that was set in the D-START request, the CM-ASE shall:

a) stop all timers;

b) if the dialogue with the peer is still open:

1) if the CM-ASE is a CM-air-ASE, create a CMAircraftMessage APDU with a cmAbortReason [communication-service-failure] APDU message element;

2) if the CM-ASE is a CM-ground-ASE, create a CMGroundMessage APDU with a cmAbortReason [communication-service-failure] APDU message element; and

3) invoke D-ABORT request with:

i) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

ii) the APDU as the D-ABORT *User Data* parameter value;

c) if the CM-user is active, invoke a CM-provider-abort service indication with the abstract value “communication-service-failure”; and

d) enter the *IDLE* state.

2.5.5 CM-ASE state tables

2.5.5.1 Priority

2.5.5.1.1 If the state tables for the CM-ground-ASE and the CM-air-ASE at Tables 2-11 and 2-12, respectively, conflict with textual statements made elsewhere in this manual, the textual statements shall take precedence.

2.5.5.1.2 In the state tables at Tables 2-11 and 2-12, the statement “cannot occur” means that if the implementation conforms to the SARPs, it is impossible for this event to occur. If the event does occur, this implies that there is an error in the implementation. If such a situation is detected, it is suggested that the ASE abort with the error “unrecoverable system error”.

2.5.5.1.3 In the state tables at Tables 2-11 and 2-12, the statement “not permitted” means that the implementation must prevent this event from occurring through some local means. If the event does occur, this implies that there is an error in the implementation. If such a situation is detected, it is suggested that the ASE perform a local rejection of the request rather than aborting the dialogue.

Table 2-11. CM-ground-ASE state table

STATE → *EVENT ↓*	*IDLE*	*LOGON*	*UPDATE*	*CONTACT*	*DIALOGUE*	*CONTACT DIALOGUE*	*END*	*FORWARD*		
DIALOGUE Service Events										
D-START Indication *Version Number* is greater than the CM-ground-ASE *version number*, ground-ground forwarding supported	●D-START response →*IDLE*	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur		
D-START Indication *Version Number* is less than or equal to the CM-ground-ASE version number, *User Data* = CMLogonRequest, ground-ground forwarding supported	●CM-logon indication →*LOGON*	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur		
D-START Indication *Version Number* is less than or equal to the CM-ground-ASE version number, *User Data* = CMForwardRequest ground-ground forwarding supported	●CM-forward indication ●D-START response →*IDLE*	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur		
D-START Indication, ground-ground forwarding is not supported, *User Data* = CMForwardRequest	●D-START response →*IDLE*	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur		
D-START Confirmation *Result* "rejected (permanent)" and *Reject Source* "DS-user", D-START *User Data* parameter not provided	cannot occur	cannot occur	●Stop timer t_{update} →*IDLE*	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur		

STATE → *EVENT ↓*	*IDLE*	*LOGON*	*UPDATE*	*CONTACT*	*DIALOGUE*	*CONTACT DIALOGUE*	*END*	*FORWARD*		
D-START Confirmation *Result* "rejected (permanent)" and *Reject Source* "DS-user", D-START User *Data* = CMContactResponse	cannot occur	cannot occur	cannot occur	●Stop timer $t_{contact}$ ●CM-contact confirmation →*IDLE*	cannot occur	cannot occur	cannot occur	cannot occur		
D-START Confirmation *Result* "rejected (permanent)" and *Reject Source* "DS-user", D-START User *Data* = CMForwardResponse	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	●Stop timer $t_{forward}$ ●CM-forward confirmation →*IDLE*		
D-DATA Indication, D-DATA *User Data* = CMContactResponse	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	●CM contact confirmation ●Stop timer $t_{contact}$ →*DIALOGUE*	cannot occur	cannot occur		
D-END Confirmation *Result* "accepted"	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	●stop timer t_{end} →*IDLE*	cannot occur		
CM-User Events										
CM-update Request	●D-START request ●Start timer t_{update} →*UPDATE*	not permitted	not permitted	not permitted	●D-DATA request →*DIALOGUE*	not permitted	not permitted	not permitted		
CM-contact Request	●D-START request ●Start timer $t_{contact}$ →*CONTACT*	not permitted	not permitted	not permitted	●D-DATA request ●Stop timer $t_{contact}$ →*CONTACT DIALOGUE*	not permitted	not permitted	not permitted		
CM-forward Request	●D-START request ●Start timer $t_{forward}$ →*FORWARD*	not permitted	not permitted	not permitted	not permitted	not permitted	not permitted	not permitted		
CM-logon Response *Maintain Dialogue* not supplied by CM-ground-user	not permitted	●D-START response →*IDLE*	not permitted	not permitted	not permitted	not permitted	not permitted	not permitted		
CM-logon Response *Maintain Dialogue* "accepted"	not permitted	●D-START response →*DIALOGUE*	not permitted	not permitted	not permitted	not permitted	not permitted	not permitted		

STATE → *EVENT ↓*	*IDLE*	*LOGON*	*UPDATE*	*CONTACT*	*DIALOGUE*	*CONTACT DIALOGUE*	*END*	*FORWARD*		
CM-end Request	not permitted	not permitted	not permitted	not permitted	●D-END request ●Start timer t_{end} →*END*	not permitted	not permitted	not permitted		
ABORT Events										
CM-user-abort Request	not permitted	●D-ABORT request →*IDLE*	●stop timer t_{update} ●D-ABORT request →*IDLE*	●stop timer $t_{contact}$ ●D-ABORT request →*IDLE*	●D-ABORT request →*IDLE*	●D-ABORT request ●Stop timer $t_{contact}$ →*IDLE*	not permitted	●stop timer $t_{forward}$ ●D-ABORT request →*IDLE*		
D-ABORT Indication *Originator* is "provider"	cannot occur	●CM-provider-abort indication →*IDLE*	●stop timer t_{update} ●CM-provider-abort indication →*IDLE*	●stop timer $t_{contact}$ ●CM-provider-abort indication →*IDLE*	●CM-provider-abort indication →*IDLE*	●CM-provider-abort indication ●Stop timer $t_{contact}$ →*IDLE*	●stop timer t_{end} →*IDLE*	●stop timer $t_{forward}$ ●CM-provider-abort indication →*IDLE*		
D-ABORT Indication *Originator* is "user"	cannot occur	●CM-user-abort indication →*IDLE*	●stop timer t_{update} ●CM-user-abort indication →*IDLE*	●stop timer $t_{contact}$ ●CM-user-abort indication →*IDLE*	●CM-user-abort indication →*IDLE*	●CM-user-abort indication ●Stop timer $t_{contact}$ →*IDLE*	●stop timer t_{end} ⇒*IDLE*	cannot occur		
D-P-ABORT Indication	cannot occur	●CM-provider-abort indication →*IDLE*	●stop timer t_{update} ●CM-provider-abort indication →*IDLE*	●stop timer $t_{contact}$ ●CM-provider-abort indication →*IDLE*	●CM-provider-abort indication →*IDLE*	●CM-provider-abort indication ●Stop timer $t_{contact}$ →*IDLE*	●stop timer t_{end} →*IDLE*	●stop timer $t_{forward}$ ●CM-provider-abort indication →*IDLE*		
T_{update} Expires	cannot occur	cannot occur	●D-ABORT request ●CM-provider-abort indication →*IDLE*	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur		
$T_{contact}$ Expires	cannot occur	cannot occur	cannot occur	●D-ABORT request ●CM-provider-abort indication →*IDLE*	cannot occur	●D-ABORT request ●CM-provider-abort indication →*IDLE*	cannot occur	cannot occur		
T_{end} Expires	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	●D-ABORT request →*IDLE*	cannot occur		

STATE → *EVENT ↓*	*IDLE*	*LOGON*	*UPDATE*	*CONTACT*	*DIALOGUE*	*CONTACT DIALOGUE*	*END*	*FORWARD*		
$T_{forward}$ Expires	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur	●D-ABORT request ●CM-provider-abort indication *→IDLE*		

Table 2-12. CM-air-ASE state table

STATE → *EVENT ↓*	*IDLE*	*LOGON*	*CONTACT*	*DIALOGUE*	*CONTACT DIALOGUE*		
DIALOGUE Service Events							
D-START Indication *User Data* CMUpdate	●CM-update indication ●D-START response →*IDLE*	cannot occur	cannot occur	cannot occur	cannot occur		
D-START Indication *User Data* CMContactRequest	●CM-contact indication →*CONTACT*	cannot occur	cannot occur	cannot occur	cannot occur		
D-START Confirmation *Result* "rejected (permanent)" and *Reject Source* "DS-user"	cannot occur	●Stop timer t_{logon} ●CM-logon confirmation →*IDLE*	cannot occur	cannot occur	cannot occur		
D-START Confirmation *Result* "accepted" *User Data* CMLogonResponse	cannot occur	●Stop timer t_{logon} ●CM-logon confirmation →*DIALOGUE*	cannot occur	cannot occur	cannot occur		
D-DATA Indication *User Data* CMUpdate	cannot occur	cannot occur	cannot occur	●CM-update indication →*DIALOGUE*	cannot occur		
D-DATA Indication *User Data* CMContactRequest	cannot occur	cannot occur	cannot occur	●CM-contact indication →*CONTACT DIALOGUE*	cannot occur		
D-END Indication	cannot occur	cannot occur	cannot occur	●CM-end indication ●D-END response →*IDLE*	cannot occur		
CM-User Events							
CM-contact Response	not permitted	not permitted	●D-START response →*IDLE*	not permitted	●D-DATA request →*DIALOGUE*		
CM-logon Request	●D-START request ●Start timer t_{logon} →*LOGON*	not permitted	not permitted	not permitted	not permitted		
ABORT Events							
CM-user-abort Request	not permitted	●stop timer t_{logon} ●D-ABORT request →*IDLE*	●D-ABORT request →*IDLE*	●D-ABORT request →*IDLE*	●D-ABORT request →*IDLE*		
D- ABORT Indication *Originator* is "provider"	cannot occur	●stop timer t_{logon} ●CM-provider-abort indication →*IDLE*	●CM-provider-abort indication →*IDLE*	●CM-provider-abort indication →*IDLE*	●CM-provider-abort indication →*IDLE*		

STATE → *EVENT ↓*	*IDLE*	*LOGON*	*CONTACT*	*DIALOGUE*	*CONTACT DIALOGUE*		
D-ABORT Indication *Originator* is "user"	cannot occur	●stop timer t_{logon} ●CM-user-abort indication →*IDLE*	●CM-user-abort indication →*IDLE*	●CM-user-abort indication →*IDLE*	●CM-user-abort indication →*IDLE*		
D-P-ABORT Indication	cannot occur	●stop timer t_{logon} ●CM-provider-abort indication →*IDLE*	●CM-provider-abort indication →*IDLE*	●CM-provider-abort indication →*IDLE*	●CM-provider-abort indication ⇒*IDLE*		
T_{logon} Expires	cannot occur	●D-ABORT request ●CM-provider-abort indication →*IDLE*	cannot occur	cannot occur	cannot occur		

2.6 COMMUNICATION REQUIREMENTS

2.6.1 Encoding rules

2.6.1.1 The CM application shall apply the Packed Encoding Rules (PER) encoding as defined in ISO/IEC 8825-2, using the basic unaligned variant to encode/decode the ASN.1 message structure and content specified in 2.4.

2.6.1.2 When encoded CM APDUs are treated as bit-oriented values that are not padded to an integral number of octets, the length determinant includes only the significant bits of the encoding corresponding to the ASN.1 type.

2.6.2 Dialogue service requirements

2.6.2.1 Primitive requirements

Where dialogue service primitives, i.e. D-START, D-DATA, D-END, D-ABORT and D-P-ABORT, are described as being invoked in 2.5, the CM-ground-ASE and the CM-air-ASE shall exhibit external behaviour consistent with the dialogue service, as described in Part III of this manual, having been implemented and its primitives invoked.

2.6.2.2 ATN Quality of Service requirements

2.6.2.2.1 The *Priority* Quality of Service parameter of the D-START for CM shall be the abstract value of "flight regularity communications".

2.6.2.2.2 The *RER* Quality of Service parameter of the D-START for CM shall be set to the abstract value of "low".

2.6.2.2.3 The CM-ASE shall map the Class of Communication Service parameter abstract values to the ATSC routing class abstract value part of the D-START Quality of Service parameter as presented in Table 2-13.

Table 2-13. Mapping between class of communication and routing class abstract values

Class of communication abstract value	*Routing class abstract value*
A	Traffic follows Class A ATSC route(s)
B	Traffic follows Class B ATSC route(s)
C	Traffic follows Class C ATSC route(s)
D	Traffic follows Class D ATSC route(s)
E	Traffic follows Class E ATSC route(s)
F	Traffic follows Class F ATSC route(s)
G	Traffic follows Class G ATSC route(s)
H	Traffic follows Class H ATSC route(s)

Note.— ATSC class values are defined in Part IV of this manual.

2.6.2.3 ATN Security Requirements parameter

Note. – The specific security mechanisms that may provide the security functionality are out of scope for this document. The Security Requirements parameter is provided so that additional security provisions may be invoked if local constraints require security.

2.6.2.3.1 The Security Requirements parameter of the D-START shall be set to either “secured dialogue supporting security information management” or “no security” for the CM-logon service.

Note.— The value “secured dialogue supporting security information management” is intended to be used if there has not been a previous CM-logon performed with the peer CM application. This is in order to establish the session key. The value “secured dialogue” is used if there has been a previous CM-logon performed with the peer CM application. However, the specific security provisions to provide this functionality are outside the scope of this document.

2.6.2.3.2 The Security Requirements parameter of the D-START shall be set to either “secured dialogue” or “no security” for the CM-contact service.

2.6.2.3.3 The Security Requirements parameter of the D-START shall be set to either “secured dialogue” or “no security” for the CM-forward service.

2.7 CM-USER REQUIREMENTS

Note.— Requirements imposed on CM-users concerning CM messages and interfacing with the CM-ASEs are presented in 2.7.

2.7.1 CM-air-user requirements

2.7.1.1 General CM-air-user requirements

2.7.1.1.1 When a CM-air-user invokes the CM-logon and requires a particular class of communication service, it sets the *Class of Communication Service* parameter to be the class of communication it requires.

2.7.1.1.2 When the CM-air-user invokes a CM-logon and has no preference for the class of communication service to be used, the *Class of Communication Service* parameter does not need to be provided.

2.7.1.1.3 When the CM-air-user invokes the CM-logon service, it sets the *Security Required* parameter as appropriate to the local security policy.

Note. – The specific security mechanisms that may provide the security functionality are outside the scope of this document.

2.7.1.1.4 For each CM-air-ASE invocation, the CM-air-user establishes a correlation between a CM-air-ASE invocation and the facility designation.

2.7.1.1.5 Upon the initiation of a CM-logon or upon receipt of a CM-update or a CM-contact service indication, the ASE invocation correlation is based on the facility designation in the *Facility Designation* parameter of the respective CM service.

2.7.1.1.6 The correlation is maintained for the duration of the ASE invocation.

2.7.1.2 CM-logon service requirements

2.7.1.2.1 Only the CM-air-user is permitted to initiate the CM-logon service.

2.7.1.2.2 Either secure or unsecure CM-logon services may be performed.

Note. – The specific security mechanisms that may provide the security functionality are out of scope for this document.

2.7.1.2.3 When invoking the CM-logon service request, the CM-air-user shall provide the following as part of the CMLogonRequest:

a) its CM long TSAP;

b) the aircraft's flight identification;

c) information on each application for which it requires a data link service as follows:

1) for air-only-initiated services: application name and version number for all the versions that can be supported; and
2) for applications that can be ground-initiated: application name, version number and address for all the versions that can be supported;

d) the facility designation of the facility with which the CM-air-user wishes to exchange application information (if required); and

e) flight information data as required by the ground system.

Note.— The facility designation is only used if the CM-air-user wants to explicitly identify a facility other than the one contained in the Facility Designation *parameter of the CM-logon service for which it requires application information.*

2.7.1.2.4 The CM-air-user should only use the facilityDesignation field of the CMLogonRequest if requesting information for a facility other than the one specified in the Facility Designation parameter of the CM-logon service.

2.7.1.2.5 When invoking the CM-logon service request, if any router domain part (RDP) for a given application address is different than the CM RDP, the CM-air-user shall use the long TSAP for each application address provided.

Note.— The long TSAP = RDP + short TSAP. The short TSAP = ARS + LOC + SYS + NSEL + TSEL. The RDP = VER + ADM + RDF.

2.7.1.2.6 When invoking the CM-logon service request, the ARS component of the short TSAP shall contain the ICAO 24-bit aircraft address.

Note.— If there is more than one routing domain on the aircraft, the LOC field is used to differentiate one from the other(s).

2.7.1.2.7 Upon receipt of a CM-logon service confirmation, the CM-air-user shall create the actual TSAP for each type of ground application information contained in the *Logon Response* based on the IDP and long TSAP for each application as defined in 2.4.

Note 1.— The actual TSAP = IDP + long TSAP. The IDP = AFI + IDI.

Note 2.— If the Version Number *parameter is not provided and the* Logon Response *parameter is provided but is empty, this means that the CM-ground-user has rejected the logon. This may be for operational reasons (e.g. optional information that is required for a particular airspace was missing in the CM-logon request) or for technical reasons (e.g. there is a problem with the ground system that prevents it from providing the information). This case will NOT be due to a version number incompatibility, since in that case only the* Version Number *is provided. If a CM-logon service is rejected, the CM-air-user may evaluate possible reasons for the rejection and retry the CM-logon service.*

2.7.1.2.11 Upon receipt of a CM-logon service confirmation, the CM-air-user shall make the information contained in the CMLogonResponse available to the other applications (i.e. ADS-C and CPDLC), as well as to the DS-provider.

2.7.1.2.12 Upon receipt of a *Logon Response* from a CM-logon service confirmation from a ground facility for which CM information has previously been received, the CM-air-user shall only replace the previous information for which new logon information has been received.

Note 1.— If the facility designation field of the Logon Request was provided, then the information contained in the Logon Response parameter corresponds to that facility designation. If the facility designation field of the Logon Request was not provided, then the information contained in the Logon Response parameter corresponds to the Facility

Designation parameter of the CM-logon service.

Note 2.— If the facility designation field of the Logon Request was provided and no application information is returned in the Logon Response parameter, this means that the CM-ground-user does not have access to the information for the requested facility, or that the requested facility does not support any of the proposed applications.

2.7.1.3 CM-update service requirements

2.7.1.3.1 If a CM-update service indication is received, then the information contained in the Update Information parameter corresponds to the facility designation contained in the Facility Designation parameter or, if a dialogue exists, the facility with which the dialogue is in place.

2.7.1.3.2 Upon receipt of *Update Information* from a CM-update service indication from a ground facility designation for which CM information has previously been received, the CM-air-user shall only replace the previous information for which updated information has been received.

2.7.1.3.3 Upon receipt of a CM-update service indication, the CM-air-user shall create the actual TSAP for each type of ground application information contained in the *Update Information* based on the IDP and long TSAP for each application as defined in 2.4.

Note.— The actual TSAP = IDP + long TSAP. The IDP = AFI + IDI.

2.7.1.3.4 The CM-air-user shall make the updated information contained in the *Update Information* available to the other applications (i.e. ADS-C and CPDLC), as well as to the DS-provider.

2.7.1.4 CM-contact service requirements

2.7.1.4.1 Upon receipt of a CM-contact indication, the CM-air-user should invoke the CM-logon request with the indicated ground system within 0.5 seconds.

2.7.1.4.2 Upon receipt of a CM-logon confirmation when performing the CM-contact service, the CM-air-user shall invoke a CM-contact response.

2.7.1.4.3 Upon receipt of a CM-logon confirmation when performing the CM-contact service, the CM-air-user should invoke a CM-contact response within 0.5 seconds.

2.7.1.4.4 Upon receipt of a CM-contact service indication, the CM-air-user shall attempt to initiate a CM-logon service request with the indicated ground system.

Note.— If a CM-logon service request is initiated, the CM-air-user will comply with the CM-logon requirements as stated in 2.7.1.1. However, the facility designation will not be provided as part of the CMLogonRequest in this case.

2.7.1.4.5 In addition to the above CM-logon service requirements, upon receipt of a CM-logon service response from the indicated facility designation, or if no CM-logon service request can be initiated, the CM-air-user shall invoke the CM-contact service response indicating the success or lack thereof of the CM-logon service request.

2.7.1.5 CM-user-abort service requirements

2.7.1.5.1 When it is required that a CM-air-user abort the current service or maintained dialogue, the CM-air-user

initiates a CM-user-abort request.

2.7.1.5.2 A CM-air-user may need to abort the current service or maintained dialogue for several reasons, including the detection of an unrecoverable error situation and the local policies. The user abort request and indication primitives do not indicate the reason of the abort.

2.7.2 CM-ground-user requirements

2.7.2.1 General CM-ground-user requirements

2.7.2.1.1 A CM-ground-user shall invoke the CM-logon service, CM-update service, CM-contact service, and CM-end service only when communicating with a CM-air-user.

2.7.2.1.2 A CM-ground-user shall invoke the CM-forward service only when communicating with another CM-ground-user.

2.7.2.1.3 For services requiring secuirty , the CM-ground-user shall perform the necessary security functions for each application that requires security.

Note 1. – The specific security mechanisms that may provide the security functionality are out of scope for this document.

Note 2. – The Directory Service, if available, may also be utilized for security provisions. The Directory Service implementation and policy is local implementation.

Note 3.— When a CM-ground-user invokes the CM-update service, CM-contact service, or CM-forward service and requires a particular class of communication service, it will set the Class of Communication Service parameter to be the class of communication it requires.

Note 4.— When the CM-ground-user invokes a CM-update service, CM-contact service, or CM-forward service and has no preference for the class of communication service to be used, the Class of Communication Service parameter does not need to be provided.

Note 5.— When a CM-ground-user specifies the Class of Communication Service parameter and the dialogue is in place, the Class of Communication parameter is ignored.

Note 6.— When the CM-ground-user invokes the CM-update service, CM-contact service or CM-forward request service, it sets the Security Required parameter as appropriate to the local security policy.

Note 7.— For each CM-ground-ASE invocation, the CM-ground-user establishes a correlation between a CM-ground-ASE invocation and the ICAO 24-bit aircraft address.

Note 8.— Upon the initiation of a CM-update or CM-contact service request, or upon receipt of a CM-logon service indication, the ASE invocation correlation is based on the ICAO 24-bit aircraft address in the Aircraft Address parameter of the respective CM service.

Note 9.— The correlation is maintained for the duration of the ASE invocation.

2.7.2.2 CM-logon service requirements

2.7.2.2.1 Upon receipt of a CM-logon indication, the CM-ground-user should invoke the CM-logon response within 0.5 seconds.

2.7.2.2.2 Upon receipt of a CM-logon service indication, the CM-ground-user shall make the aircraft application information contained in the Logon Request available to the other applications (i.e. ADS-C and CPDLC), as well as to the DS-provider.

2.7.2.2.3 Upon receipt of a CM-logon service indication, the CM-ground-user shall create the actual TSAP for each type of aircraft application information contained in the Logon Request based on the IDP and long TSAP for each application, as defined in 2.4.

Note.— The actual TSAP = IDP + long TSAP. The IDP = AFI + IDI.

2.7.2.2.4 Upon receipt of a *Logon Request* from a CM-logon service indication from an aircraft for which CM information has previously been received and is still being maintained, the CM-ground-user shall update the aircraft information accordingly.

2.7.2.2.5 Upon receipt of a CM-logon service indication, the CM-ground-user shall invoke a CM-logon service response with a CMLogonResponse containing:

a) application names, addresses and version numbers for the requested applications that can be air-initiated for all versions that the ground and aircraft systems can support; and

b) application names and version numbers for the requested ground-only-initiated applications that the ground system can support.

2.7.2.2.6 Upon receipt of a CM-logon service indication with the *Security Required* parameter provided with the value “secured dialogue supporting security information management”, the CM-ground-user shall invoke the necessary security mechanisms to provide the secured dialogue service.*Note. – The specific security mechanisms that may provide the security functionality are out of scope for this document.*

2.7.2.2.7 If the facility designation is present in the *Logon Request* parameter, then the application information contained in the *Logon Response* shall correspond to that facility designation.

Note 1.— The information for the requested facility may be obtained through the use of the ATN directory service (see Part IV of this manual).

Note 2.— If a CM-ground-user does not have access to the information for the requested facility, no application information is returned.

2.7.2.2.8 When If the facility designation is not present in the *Logon Request* parameter, then the application information contained in the *Logon Response* shall correspond to the responding CM-ground-user’s facility designation.

2.7.2.2.9 When invoking the CM-logon service response, if any RDP for a given application address is different than the CM RDP, the CM-ground-user shall use the long TSAP for each application address provided.

Note 1.— The long TSAP = RDP + short TSAP. The short TSAP = ARS (optional) + LOC + SYS + NSEL + TSEL. The RDP = VER + ADM + RDF.

Note 2.— If there is more than one routing domain on the ground, the ARS field is used to differentiate one from the other(s). If there is not more than one routing domain on the ground, the ARS field need not be used.

Note 3.— The value of the ARS field is a 24-bit unsigned binary number that uniquely identifies the addressed system in a single routing domain and is assigned by the state or organization identified in the ADM field.

2.7.2.2.10 When the CM-ground-user requires a CM dialogue to be maintained, the CM-ground-user shall set the CM-logon service response *Maintain Dialogue* parameter if, and only if, the dialogue maintain service is supported.

2.7.2.2.11 If a CM-ground-user wishes to reject a CM-logon service indication for any reason, the CM-ground-user shall invoke a CM-logon service response with a *Logon Response* containing no information.

Note 1.— This may be done for either operational reasons (e.g. optional information that is required for a particular airspace is missing or information for the requested facility is not available) or technical reasons other than version incompatibility (e.g. there is a problem with the ground system and the service is not available). (These example reasons are not meant to constitute an exhaustive list.)

Note 2.— This will not be done if the CM-air-ASE and CM-ground-ASE versions are incompatible. In that case, the CM-ground-ASE will reject the D-START indication, as detailed in 2.5, before it reaches the CM-ground-user.

2.7.2.3 CM-update service requirements

2.7.2.3.1 Only the CM-ground-user is permitted to initiate the CM-update-service.

2.7.2.3.2 Either secure or unsecure CM-update services may be performed.

Note. – Indication of security is conveyed through the use of the Security Requirements parameter. The specific security mechanisms that may provide the security functionality are out of scope for this document.

2.7.2.3.3 When invoking the CM-update service request, the CM-ground-user shall provide a CMUpdate containing application names, addresses and version numbers for each of the data link applications being updated.

Note.— The CM-update service only corresponds to a ground facility's local applications.

2.7.2.3.4 When invoking the CM-update service request, the CM-ground-user shall use the long TSAP for each application address provided.

2.7.2.5 CM-contact service requirements

2.7.2.5.1 Only the CM-ground-user is permitted to initiate the CM-contact-service.

2.7.2.5.2 Either secure or unsecure CM-contact services may be performed.

Note. – Indication of security is conveyed through the use of the Security Requirements parameter. The specific security mechanisms that may provide the security functionality are out of scope for this document.

2.7.2.5.3 When invoking the CM-contact service request, the CM-ground-user shall provide a CMContactRequest containing the facility designation of the ground facility that the ground requests the aircraft to contact.

2.7.2.6 CM-end service requirements

2.7.2.6.1 Only the CM-ground-user is permitted to initiate the CM-end-service.

2.7.2.6.2 If the CM-ground-user establishes a CM dialogue with the CM-logon Maintain Dialogue parameter set, the CM-ground-user is responsible for closing the CM dialogue with the CM-end service.

2.7.2.7 CM-forward service requirements

2.7.2.7.1 Only the CM-ground-user is permitted to initiate the CM-forward service.

2.7.2.7.2 Either secure or unsecure CM-forward services may be performed.

Note. – Indication of security is conveyed through the use of the Security Requirements parameter. The specific security mechanisms that may provide the security functionality are out of scope for this document.

2.7.2.7.3 When requesting the CM-forward service, the CM-ground-user shall provide a CMForwardRequest containing all of the information from either a CM-logon request message or a CM-forward request message, whichever is more recent.

2.7.2.7.4 Upon receipt of a CM-forward service indication, the CM-ground-user shall make the aircraft application information contained in the *Forward Request* available to the other applications (i.e. ADS-C and CPDLC), as well as to the DS-provider.

2.7.2.7.5 Upon receipt of a CM-forward service indication, the CM-ground-user shall create the actual TSAP for each type of aircraft application information contained in the *Forward Request* based on the IDP and long TSAP for each application as defined in 2.4.

Note.— The actual TSAP = IDP + long TSAP. The IDP = AFI + IDI.

2.7.2.7.6 Upon receipt of a forward request from a CM-forward service indication concerning an aircraft identifier for which CM information has previously been received and is still being maintained, the CM-ground-user shall update the aircraft information accordingly.

2.7.2.7.7 Upon receipt of a forward request from a CM-forward service indication, the receiving CM-ground-user should invoke a CM-update service request with the indicated aircraft, if the update service is supported.

2.7.2.8 CM-user-abort service requirements

2.7.2.8.1 When it is required that a CM-ground-user abort the current service or maintained dialogue, the CM-ground-user initiates a CM-user-abort request.

2.7.2.8.2 A CM-ground-user may need to abort the current service or maintained dialogue for several reasons, including the detection of an unrecoverable error situation and the local policies. The user abort request and indication primitives do not indicate the reason of the abort.

2.7.3 Parameter value unit, range and resolution

A CM-user shall interpret parameter value unit, range and resolution as defined in 2.4.

2.8 SUBSETTING RULES

2.8.1 General

Note.— This section specifies conformance requirements that all implementations of the CM protocol obey.

2.8.1.1 An implementation of either the CM ground-based service or the CM air-based service claiming conformance shall support the CM protocol features as shown in Tables 2-14 and 2-15.

2.8.1.2 The "status" column indicates the level of support required for conformance to the CM-ASE protocol. The values are as follows:

"M" mandatory support is required;

"O" optional support is permitted for conformance to the CM protocol;

"N/A" the item is not applicable; and

"C.n" the item is conditional where n is the number which identifies the condition which is applicable.

Table 2-14. CM protocol versions implemented

Version	*Status*	*Associated predicate*
Version 1	M	V1

Table 2-15. CM Operational functional units

CM state	*Status*	*Associated predicate*
The CM system acts as an airborne system	C.1	CM/air
The CM system acts as a ground system	C.1	CM/ground
The CM ground system can update application information	If (CM/ground), then O, else N/A	G-UP-FU
The CM ground system can request the aircraft to initiate a logon with a specified CM ground system	If (CM/ground), then O, else N/A	G-CO-FU
The CM-ground-user can process forwarded aircraft information	If (CM/ground), then O, else N/A	G-FO-FU
The CM ground system can forward aircraft information to another CM ground system	If (CM/ground), then O, else N/A	G-FO-IN
The CM-air-user can process an update request	If (CM/air), then O, else N/A	A-UP-FU

CM state	*Status*	*Associated predicate*
The CM-air-user can process a contact request	If (CM/air), then O, else N/A	A-CO-FU

C.1: A conformant implementation will support one, and only one, of these two options.

Note 1.— A conformant CM-air implementation will consist of at least the core functionality (CM/air) with, optionally, any combination of the A-UP-FU and A-CO-FU functional units. If only the core functionality is supported, then logon exchanges can be initiated with ground systems, received contact requests are rejected with the reason "contact not successful", received update requests are processed by the CM-air-ASE but ignored by the CM-air-user, and the capability to maintain dialogue is supported.

Note 2.— A conformant CM-ground implementation will consists of at least the core functionality (CM/ground) with, optionally, any combination of the G-FO-IN, G-FO-FU, G-CO-FU, and G-UP-FU. If only the core functionality is supported, then logon exchanges are supported with an aircraft, received forward requests are rejected with reason "service not supported",and the capability to maintain dialogue optionally is supported.

Note 3.— The templates of the Protocol/Operational Implementation Conformance Statements (P/OICS) provide the capability to detail all implementation particulars to a much deeper level than can be given by the functional unit descriptions of 2.8. All details for both air and ground implementations can be found in completed P/OICS for particular States' or industries' implementations.

Chapter 3

CONTROLLER-PILOT DATA LINK COMMUNICATIONS APPLICATION

3.1 INTRODUCTION

3.1.1 Overview

3.1.1.1 This chapter is designed as follows:

- Section 3.1 — INTRODUCTION: Besides outlining the material covered in Chapter 3, this section provides a description of the functions of the CPDLC application.

- Section 3.2 — GENERAL REQUIREMENTS: This section focuses on the CPDLC version number and error-processing requirements.

- Section 3.3 — THE ABSTRACT SERVICE: The description of the abstract service provided by the CPDLC application service element (CPDLC-ASE) is contained in this section.

- Section 3.4 — FORMAL DEFINITIONS OF MESSAGES: This section contains the formal definitions of messages exchanged by CPDLC-ASEs using Abstract Syntax Notation One (ASN.1).

- Section 3.5 — PROTOCOL DEFINITION: This section describes the exchanges of messages allowed by the CPDLC protocol, as well as time constraints. It also provides CPDLC-ASE protocol descriptions and state tables.

- Section 3.6 — COMMUNICATION REQUIREMENTS: The requirements that the CPDLC application imposes on the underlying communication system are covered in this section.

- Section 3.7 — CPDLC-USER REQUIREMENTS: This section contains requirements imposed on the user of the CPDLC-ASE service, and message description tables.

- Section 3.8 — SUBSETTING RULES: This section defines the conformant subsets for the CPDLC-ASE.

3.1.1.2 Throughout this manual, references to the CPDLC message refer to the actual CPDLC message as generated by, and delivered to, the CPDLC-users (operational content of the CPDLC message). References to CPDLC/IC data refer to CPDLC data to which an integrity check (IC) (checksum) has been added before sending the CPDLC message.

3.1.2 Description of the functions of the CPDLC application

3.1.2.1 Controller-pilot message exchange function

The controller-pilot message exchange function defines a method for a controller and a pilot to exchange CPDLC messages via data link.

3.1.2.2 Downstream clearance function

The downstream clearance function provides the capability for an aircraft to contact an air traffic service unit that is not the current data authority for the purpose of receiving a downstream clearance. This information is exchanged using CPDLC message(s).

3.1.2.3 Ground forward function

The ground forward function provides the capability for a ground system to forward information received in a CPDLC message to another ground system. The ground forward function can be used by the current data authority to forward an aircraft request to the next data authority so that the aircraft does not need to issue the same request again. This function can also be used by a downstream data authority to pass a message to a current data authority for transmission by the current data authority to an aircraft. This information is exchanged using a CPDLC message or messages. It is a one-way forwarding of information with an indication of success, failure or non-support from the receiving ground system. The CPDLC ground forward function does not use the CPDLC integrity check.

3.2 GENERAL REQUIREMENTS

3.2.1 CPDLC-ASE version number

The CPDLC-air-ASE and CPDLC-ground-ASE version numbers shall both be set to one.

3.2.2 Error-processing requirements

3.2.2.1 In the event of information input by the CPDLC-user being incompatible with that which is able to be processed by the system, the CPDLC-user shall be notified.

3.2.2.2 In the event of a CPDLC-user invoking a CPDLC service primitive when the CPDLC-ASE is not in a state specified in 3.5, the CPDLC-user shall be notified.

3.3 THE ABSTRACT SERVICE

3.3.1 Service description

3.3.1.1 An implementation of either the CPDLC ground-based service or the CPDLC air-based service shall exhibit external behaviour consistent with having implemented a CPDLC-ground-ASE or CPDLC-air-ASE, respectively, with the abstract service interface primitives defined in this section, making them available to the CPDLC-ground-user or CPDLC-air-user, respectively.

3.3.1.2 There is no requirement to implement the CPDLC-ASE abstract service in a CPDLC product; however, it is necessary to implement the ground-based and air-based system in such a way that it will be impossible to detect (from the peer system) whether or not the interface has been built.

3.3.1.3 This section defines the abstract service interface for the CPDLC service. The CPDLC-ASE abstract service is described in this section from the viewpoint of the CPDLC-air-user, the CPDLC-ground-user and the CPDLC service provider.

3.3.1.4 This section defines the static behaviour (i.e. the format) of the CPDLC abstract service. Its dynamic behaviour (i.e. how it is used) is described in 3.7.

3.3.1.5 Figure 3-1 shows the functional model of the CPDLC application. The various elements identified in this model are the following:

a) the CPDLC-user;

b) the CPDLC application entity (CPDLC-AE) service interface;

c) the CPDLC-AE;

d) the CPDLC control function (CPDLC-CF);

e) the CPDLC application service element (CPDLC-ASE) service interface;

f) the CPDLC-ASE; and

g) the DS interface.

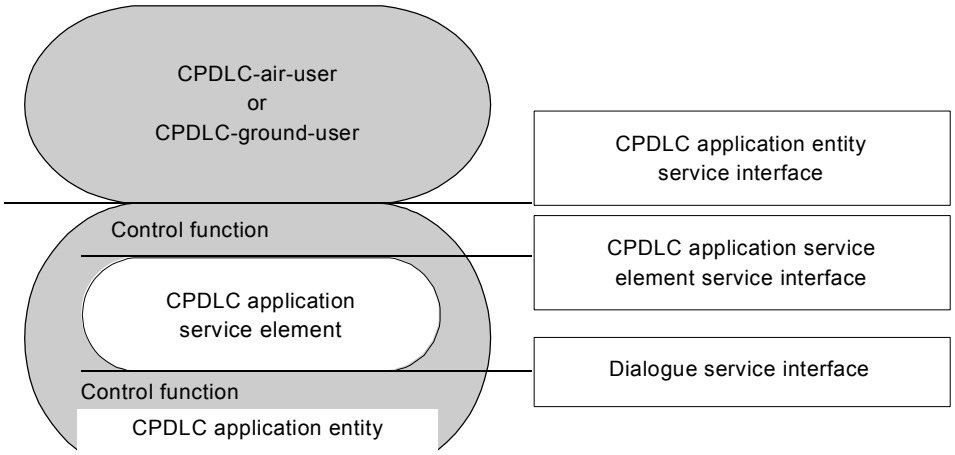


Figure 3-1. Functional model of the CPDLC application

3.3.1.6 The CPDLC-user represents the operational part of the CPDLC system. This user does not perform the communication functions but relies on a communication service provided to it via the CPDLC-AE through the CPDLC-AE service interface. The individual actions possible through the CPDLC-AE service interface are called service primitives.

3.3.1.7 The CPDLC-AE consists of several elements including the CPDLC-ASE and the CPDLC-CF.

3.3.1.8 The CPDLC-ASE is the element in the communication system that executes the CPDLC-specific protocol. In other words, it takes care of the CPDLC-specific service primitive sequencing actions, message creation, timer management, and error and exception handling. The actual encoding and decoding of each CPDLC message is handled by the CPDLC-users.

3.3.1.9 The CPDLC-CF is responsible for mapping service primitives received from one element (such as the CPDLC-ASE and the CPDLC-user) to service primitives of other abstract elements.

3.3.1.10 The CPDLC-ASE has two abstract boundaries with the CPDLC-CF: the CPDLC-ASE service and the DS. The CPDLC-CF maps CPDLC-AE service primitives to other abstract elements in the CPDLC-AE and the underlying communication service and vice versa.

3.3.2 The CPDLC-ASE abstract service

3.3.2.1 The CPDLC-ASE abstract service shall consist of a subset of the following services as allowed by the subsetting rules in 3.8:

a) CPDLC-start service as defined in 3.3.3;

b) DSC-start service as defined in 3.3.4;

c) CPDLC-message service as defined in 3.3.5;

d) CPDLC-end service as defined in 3.3.6;

e) DSC-end service as defined in 3.3.7;

f) CPDLC-forward service as defined in 3.3.8;

g) CPDLC-user-abort service as defined in 3.3.9; and

h) CPDLC-provider-abort service as defined in 3.3.10.

3.3.2.2 For a given primitive, the presence of each parameter is described by one of the following values in the parameter tables in 3.3:

blank	not present;
C	conditional upon some predicate explained in the text;
C(=)	conditional upon the value of the parameter to the immediate left being both present and equal;
M	mandatory;
M(=)	mandatory, and equal to the value of the parameter to the immediate left; or
U	user option.

3.3.2.3 The following abbreviations are used in Chapter 3 of this manual:

Req	request; data is input by the CPDLC-user initiating the service to its respective ASE;
Ind	indication; data is indicated by the receiving ASE to its respective CPDLC-user;
Rsp	response; data is input by the receiving CPDLC-user to its respective ASE; and
Cnf	confirmation; data is confirmed by the initiating ASE to its respective CPDLC-user.

3.3.2.4 An unconfirmed service allows a message to be transmitted in one direction without providing a corresponding response.

3.3.2.5 A confirmed service provides end-to-end confirmation that a message sent by one user was received by its peer user.

3.3.2.6 An abstract syntax is a syntactical description of a parameter that does not imply a specific implementation. Only when the CPDLC-ASE maps a parameter onto an APDU field, or vice versa, is the abstract syntax of the parameter described by using the ASN.1 at 3.4 for this field.

3.3.3 CPDLC-start service

3.3.3.1 General

3.3.3.1.1 The CPDLC-start service is used by the CPDLC-air-user or CPDLC-ground-user to establish a CPDLC dialogue. It is a confirmed service.

3.3.3.1.2 Once a CPDLC dialogue is established, it remains open until explicitly closed (see CPDLC-end service at 3.3.6 and CPDLC-abort services at 3.3.9 and 3.3.10).

3.3.3.1.3 The CPDLC-start service shall contain the primitives and parameters as presented in Table 3-1.

Table 3-1. CPDLC-start service parameters

Parameter name	*Req*	*Ind*	*Rsp*	*Cnf*
Called Peer Identifier	M			
Calling Peer Identifier	M	M(=)		
CPDLC Message Set Version Number	U	C(=)		
CPDLC/IC Data	M	M(=)		
Response CPDLC/IC Data			M	M(=)
Result			M	M(=)
Class of Communication Service	U	M		
Security Required	U			

3.3.3.2 Called Peer Identifier parameter

3.3.3.2.1 If the service is ground-initiated, the *Called Peer Identifier* parameter contains the addressed ICAO 24-bit aircraft address.

3.3.3.2.2 If the service is air-initiated, the *Called Peer Identifier* parameter contains the addressed ground system's facility designation.

3.3.3.2.3 If the service is ground-initiated, the *Called Peer Identifier* parameter value shall conform to the abstract syntax of the ICAO 24-bit aircraft address.

3.3.3.2.4 If the service is air-initiated, the *Called Peer Identifier* parameter value shall conform to the abstract syntax four- to eight-character facility designation.

3.3.3.3 Calling Peer Identifier parameter

3.3.3.3.1 If the service is ground-initiated, the *Calling Peer Identifier* parameter contains the sending ground system's facility designation.

3.3.3.3.2 If the service is air-initiated, the *Calling Peer Identifier* parameter contains the sending ICAO 24-bit aircraft address.

3.3.3.3.3 If the service is ground-initiated, the *Calling Peer Identifier* parameter value shall conform to the abstract syntax four- to eight-character facility designation.

3.3.3.3.4 If the service is air-initiated, the *Calling Peer Identifier* parameter value shall conform to the abstract syntax of the ICAO 24-bit aircraft address.

3.3.3.4 CPDLC Message Set Version Number parameter

Note. – The CPDLC Message Set Version Number parameter contains the version number of the CPDLC message set. This is not the CDPLC ASE Version Number, nor is it the PM CPDLC Version Number.

The *CPDLC Message Set Version Number* parameter shall conform to an abstract integer value from 1 to 255.

3.3.3.5 CPDLC/IC Data parameter

3.3.3.5.1 The CPDLC-user uses this parameter to send CPDLC/IC Data to its peer user.

3.3.3.5.2 If the CPDLC-start request primitive is invoked by the CPDLC-ground-user, the *CPDLC/IC Data* parameter value shall conform to the ASN.1 abstract syntax ICUplinkMessage.

3.3.3.5.3 If the CPDLC-start request primitive is invoked by the CPDLC-air-user, the *CPDLC/IC Data* parameter value shall conform to the ASN.1 abstract syntax ICDownlinkMessage.

Note.— The CPDLC/IC Data parameter usually contains an embedded operational CPDLC message, but this is not present in some cases. It always contains an IC field.

3.3.3.6 Response CPDLC/IC Data parameter

3.3.3.6.1 The *Response CPDLC/IC Data* parameter is used either to provide a reason for rejecting a CPDLC dialogue or to provide a means of verifying that a CPDLC dialogue establishment request has been received and accepted by the intended peer. This is to ensure a rapid response to a CPDLC-start. However, the CPDLC-ASE is not required to police this requirement. The air or ground CPDLC-user is instead required to ensure that the Response parameter includes either an allowed CPDLC message or no CPDLC message.

3.3.3.6.2 If the CPDLC-start response primitive is invoked by the CPDLC-ground-user, the *Response CPDLC/IC Data* parameter value shall conform to the ASN.1 abstract syntax ICUplinkMessage.

3.3.3.6.3 If the CPDLC-start response primitive is invoked by the CPDLC-air-user, the *Response CPDLC/IC Data* parameter value shall conform to the ASN.1 abstract syntax ICDownlinkMessage.

Note.— A response may, and often does, contain no CPDLC message. It always includes an IC field.

3.3.3.7 Result parameter

3.3.3.7.1 The *Result* parameter is used to indicate whether or not a requested CPDLC dialogue is accepted.

3.3.3.7.2 This parameter shall have one of two abstract values: “accepted” or “rejected”.

3.3.3.8 Class of Communication Service parameter

3.3.3.8.1 This parameter contains the value of the required class of communication service. If not specified by the CPDLC-user, this indicates that there is no routing preference.

3.3.3.8.2 This parameter is used by the CPDLC-ground-user to determine if the Class of Communication value is acceptable for the establishment of a CPDLC dialogue.

3.3.3.8.3 The parameter indicated to the peer CPDLC-user is that provided by the CPDLC-user if specified by the user; otherwise, it indicates that no routing preference was requested by the CPDLC dialogue initiator.

3.3.3.8.4 Where specified by the CPDLC-user, the *Class of Communication Service* parameter shall have one of the following abstract values: “A”, “B”, “C”, “D”, “E”, “F”, “G” or “H”.

3.3.3.8.5 When this parameter is provided by the CPDLC-user, the same value shall be indicated to the peer CPDLC-user; otherwise, the abstract value “ATSC - no traffic type policy preference” is indicated.

3.3.3.9 Security Required parameter

Note. – Indication of security is conveyed through the use of the Security Required parameter. The specific security mechanisms that may provide the security functionality are outside the scope of this document.

3.3.3.9.1 The *Security Required* parameter contains the value of the required level of security, if specified by the CPDLC-user.

3.3.3.9.2 If the received *Security Required* parameter is not as expected per the local security policy, the receiving CPDLC-ASE will abort.

3.3.3.9.3 Where specified by the CPDLC-user, the *Security Required* parameter shall have one of the following abstract values: “no security” or “secured exchange”.

Note.— Where not specified by the CPDLC-user, this indicates that there is no security required.

3.3.4 DSC-start service

3.3.4.1 General

3.3.4.1.1 The DSC-start service is used by the CPDLC-air-user to establish a DSC dialogue for the purpose of providing downstream clearances. It is a confirmed service.

3.3.4.1.2 Once a DSC dialogue is established, it remains open until explicitly closed (see DSC-end service at 3.3.7 and CPDLC-abort services at 3.3.9 and 3.3.10).

3.3.4.1.3 The DSC-start service shall contain the primitives and parameters as presented in Table 3-2.

Table 3-2. DSC-start service parameters

Parameter name	*Req*	*Ind*	*Rsp*	*Cnf*
Facility Designation	M			
Aircraft Address	M	M(=)		
CPDLC Message Set Version Number	U	C(=)		
CPDLC/IC Data	M	M(=)		
Response CPDLC/IC Data			M	M(=)
Result			M	M(=)
Class of Communication Service	U	M		
Security Required	U			

3.3.4.2 Facility Designation parameter

3.3.4.2.1 This parameter contains the addressed ground system's facility designation.

3.3.4.2.2 The *Facility Designation* parameter value shall conform to the abstract syntax four- to eight-character facility designation.

3.3.4.3 Aircraft Address parameter

3.3.4.3.1 The *Aircraft Address* parameter value shall conform to the abstract syntax of the ICAO 24-bit aircraft address.

3.3.4.3.2 This parameter contains the ICAO 24-bit aircraft address.

3.3.4.4 CPDLC Message Set Version Number parameter

Note.— The CPDLC Message Set Version Number parameter contains the version number of the CPDLC-ASE and identifies the user data abstract syntax.

When provided by the CPDLC-user, the *CPDLC Message Set Version Number* parameter shall conform to an abstract integer value from 1 to 255.

3.3.4.5 CPDLC/IC Data parameter

3.3.4.5.1 The CPDLC-air-user uses this parameter to send CPDLC/IC Data to a CPDLC-ground-user.

3.3.4.5.2 The *CPDLC/IC Data* parameter value shall conform to the ASN.1 abstract syntax ICDownlinkMessage.

Note.— The CPDLC/IC Data parameter usually contains an embedded operational CPDLC message, but this is not present in some cases. It always contains an IC field.

3.3.4.6 Response CPDLC/IC Data parameter

3.3.4.6.1 The *Response CPDLC/IC Data* parameter is used either to provide a reason for rejecting a DSC dialogue or to provide a means of verifying that a DSC dialogue establishment has been received and accepted by the intended peer.

3.3.4.6.2 The *Response CPDLC/IC Data* parameter shall conform to the ASN.1 abstract syntax ICUplinkMessage.

Note.— A response may, and often does, contain no CPDLC message. It always includes an IC field.

3.3.4.7 Result parameter

3.3.4.7.1 The *Result* parameter is used to indicate whether or not a requested DSC dialogue is accepted.

3.3.4.7.2 The *Result* parameter value shall have one of two abstract values: “accepted” or “rejected”.

3.3.4.8 Class of Communication Service parameter

3.3.4.8.1 This parameter contains the value of the required class of communication service. If not specified by the CPDLC-air-user, this indicates that there is no routing preference.

3.3.4.8.2 This parameter is used by the CPDLC-ground-user to determine if the Class of Communication value is acceptable for the establishment of a DSC dialogue.

3.3.4.8.3 If provided by the user, the parameter indicated to the peer user is that provided by the user; otherwise, it indicates that no routing preference was requested by the CPDLC dialogue initiator.

3.3.4.8.4 Where specified by the CPDLC-air-user, the *Class of Communication Service* parameter shall have one of the following abstract values: “A”, “B”, “C”, “D”, “E”, “F”, “G” or “H”.

3.3.4.8.5 When this parameter is provided by the CPDLC-user, the same value shall be indicated to the peer CPDLC-user; otherwise, the abstract value “ATSC - no traffic type policy preference” is indicated.

3.3.4.9 Security Required parameter

Note. – Indication of security is conveyed through the use of the Security Required parameter. The specific security mechanisms that may provide the security functionality are out of scope for this document.

3.3.4.9.1 The *Security Required* parameter contains the value of the required level of security, if specified by the CPDLC-air-user.

3.3.4.9.2 If the received *Security Required* parameter is not as expected per the local security policy, the receiving

CPDLC-ground-ASE will abort.

3.3.4.9.3 Where specified by the CPDLC-air-user, the *Security Required* parameter shall have one of the following abstract values: “no security” or “secured exchange”.

Note.— Where not specified by the CPDLC-air-user, this indicates that there is no security required.

3.3.5 CPDLC-message service

3.3.5.1 General

3.3.5.1.1 The CPDLC-message service can be used for pilot/controller message exchange once a dialogue is established. It is an unconfirmed service.

3.3.5.1.2 The CPDLC-message service shall contain the primitives and parameters as presented in Table 3-3.

Table 3-3. CPDLC-message service parameters

Parameter name	*Req*	*Ind*
CPDLC/IC Data	M	M(=)

3.3.5.2 CPDLC/IC Data parameter

3.3.5.2.1 This parameter contains a CPDLC/IC Data value.

3.3.5.2.2 If the CPDLC-message service is invoked by the CPDLC-ground-user, the *CPDLC/IC Data* parameter value shall conform to the ASN.1 abstract syntax ICUplinkMessage.

3.3.5.2.3 If the CPDLC-message service is invoked by the CPDLC-air-user, the *CPDLC/IC Data* parameter value shall conform to the ASN.1 abstract syntax ICDownlinkMessage.

3.3.5.2.4 The CPDLC/IC Data parameter provided by the CPDLC-user always contains both an embedded operational CPDLC message and an IC field.

3.3.6 CPDLC-end service

3.3.6.1 General

3.3.6.1.1 The CPDLC-end service is used by the CPDLC-ground-user to end a CPDLC dialogue with a CPDLC-air-user. It is a confirmed service.

3.3.6.1.2 The CPDLC-end service shall contain the primitives and parameters as presented in Table 3-4.

Table 3-4. CPDLC-end service parameters

Parameter name	*Req*	*Ind*	*Rsp*	*Cnf*
CPDLC/IC Data	M	M(=)	M	M(=)
Result			M	M(=)

3.3.6.2 CPDLC/IC Data parameter

3.3.6.2.1 This parameter contains a CPDLC/IC Data value.

3.3.6.2.2 The *CPDLC/IC Data* parameter value shall conform to the ASN.1 abstract syntax ICUplinkMessage, if provided by the CPDLC-ground-user.

3.3.6.2.3 The *CPDLC/IC Data* parameter value shall conform to the ASN.1 abstract syntax ICDownlinkMessage, if provided by the CPDLC-air-user.

3.3.6.2.4 The CPDLC/IC Data parameter value does not necessarily contain an embedded operational CPDLC message. It always includes an IC field.

3.3.6.3 Result parameter

3.3.6.3.1 This parameter is used to indicate whether or not a request to terminate a CPDLC dialogue is accepted.

3.3.6.3.2 The *Result* parameter shall have one of two abstract values: “accepted” or “rejected”.

3.3.7 DSC-end service

3.3.7.1 General

3.3.7.1.1 The DSC-end service is used by the CPDLC-air-user to end a DSC dialogue with a CPDLC-ground-user. It is a confirmed service.

3.3.7.1.2 The DSC-end service shall contain the primitives and parameters as presented in Table 3-5.

Table 3-5. DSC-end service parameters

Parameter name	*Req*	*Ind*	*Rsp*	*Cnf*
CPDLC/IC Data	M	M(=)	M	M(=)
Result			M	M(=)

3.3.7.2 CPDLC/IC Data parameter

3.3.7.2.1 This parameter contains a CPDLC/IC Data value.

3.3.7.2.2 The *CPDLC/IC Data* parameter value shall conform to the ASN.1 abstract syntax ICUplinkMessage, if provided by the CPDLC-ground-user.

3.3.7.2.3 The *CPDLC/IC Data* parameter value shall conform to the ASN.1 abstract syntax ICDownlinkMessage, if provided by the CPDLC-air-user.

3.3.7.2.4 The *CPDLC/IC Data* parameter value does not necessarily contain an embedded operational CPDLC message. It always includes an IC field.

3.3.7.3 Result parameter

3.3.7.3.1 This parameter is used to indicate whether or not a request to terminate a DSC dialogue is accepted.

3.3.7.3.2 The *Result* parameter shall have one of two abstract values: “accepted” or “rejected”.

3.3.8 CPDLC-forward service

3.3.8.1 General

3.3.8.1.1 The CPDLC-forward service is used by a CPDLC-ground-user to send a CPDLC message to another CPDLC-ground-user. Its primary use is for the forwarding of aircraft requests.

3.3.8.1.2 If the CPDLC-forward service is supported by the receiving ground system, and the sending CPDLC-ground-ASE and receiving CPDLC-ground-ASE version numbers are equal, the CPDLC-forward service shall contain the primitives and parameters as presented in Table 3-6.

Table 3-6. CPDLC-forward service parameters (service supported, versions equal)

Parameter name	*Req*	*Ind*	*Cnf*
Called Facility Designation	M		
Calling Facility Designation	M	M(=)	
CPDLC Message	M	M(=)	
Class of Communication Service	U		
Security Required	U		
Result			M

3.3.8.1.3 If the CPDLC-forward service is not supported by the receiving ground system, or if the CPDLC-forward service is supported by the receiving ground system but the sending CPDLC-ground-ASE and receiving CPDLC-ground-ASE version numbers are not equal, the CPDLC-forward service shall contain the primitives and parameters as presented in Table 3-7.

Table 3-7. CPDLC-forward service parameters (service not supported or versions not equal)

Parameter name	*Req*	*Cnf*
Called Facility Designation	M	
Calling Facility Designation	M	
ASE Version Number		C
CPDLC Message	M	
Class of Communication Service	U	
Security Required	U	
Result		M

3.3.8.2 Called Facility Designation parameter

3.3.8.2.1 This parameter contains the addressed ground system's facility designation.

3.3.8.2.2 The *Called Facility Designation* parameter value shall conform to the abstract syntax four- to eight-character facility designation.

3.3.8.3 Calling Facility Designation parameter

3.3.8.3.1 This parameter contains the sending ground system's facility designation.

3.3.8.3.2 The *Calling Facility Designation* parameter value shall conform to the abstract syntax four- to eight-character facility designation.

3.3.8.4 ASE Version Number parameter

3.3.8.4.1 This parameter contains the version number of the CPDLC-ASE.

3.3.8.4.2 When provided by the CPDLC-ground-ASE, the *ASE Version Number* parameter shall conform to the abstract integer value in the range of 1-255.

3.3.8.4.3 Only if the sending CPDLC-ground-ASE version number is not equal to the receiving CPDLC-ground-ASE version number shall the receiving CPDLC-ground-ASE version number be confirmed to the sending CPDLC-ground-user.

Note.— If the sending CPDLC-ground-ASE version number is the same as the receiving CPDLC-ground-ASE version number, the Version Number parameter is not present in the indication given to the receiving CPDLC-ground-user or in the confirmation given to the sending CPDLC-ground-user.

3.3.8.5 CPDLC Message parameter

3.3.8.5.1 The sending CPDLC-ground-user uses this parameter to forward a CPDLC message to another CPDLC-ground-user.

3.3.8.5.2 The *CPDLC Message* parameter value shall conform to the ASN.1 abstract syntax ATCForwardMessage.

3.3.8.6 Class of Communication Service parameter

3.3.8.6.1 This parameter contains the value of the required class of communication service. If not specified by the CPDLC-ground-user, this indicates that there is no routing preference.

3.3.8.6.2 Where specified by the CPDLC-ground-user, the *Class of Communication Service* parameter shall have one of the following abstract values: “A”, “B”, “C”, “D”, “E”, “F”, “G” or “H”.

3.3.8.7 Security Required parameter

Note. – Indication of security is conveyed through the use of the Security Required parameter. The specific security mechanisms that may provide the security functionality are out of scope for this document.

3.3.8.7.1 The *Security Required* parameter contains the value of the required level of security, if specified by the CPDLC-ground-user.

3.3.8.7.2 If the received *Security Required* parameter is not as expected per the local security policy, the receiving CPDLC-ground-ASE will abort.

3.3.8.7.3 Where specified by the CPDLC-ground-user, the *Security Required* parameter shall have one of the following abstract values: “no security” or “secured exchange”.

Note.— Where not specified by the CPDLC-ground-user, this indicates that there is no security required.

3.3.8.8 Result parameter

3.3.8.8.1 The *Result* parameter contains the result of the CPDLC-forward service. It will indicate success (service supported and matching versions), service unsupported, or version number incompatibility.

3.3.8.8.2 The *Result* parameter value shall conform to the ASN.1 abstract syntax ATCForwardResponse.

3.3.9 CPDLC-user-abort service

3.3.9.1 General

3.3.9.1.1 This service provides the capability for either the CPDLC-air-user or the CPDLC-ground-user to abort communication with its peer. It can be invoked at any time the CPDLC-user is aware that the CPDLC service is in operation. The CPDLC-user-abort service can be used for operational or technical reasons. It is an unconfirmed service. Messages in transit may be lost during this operation.

3.3.9.1.2 If the service is invoked prior to complete establishment of the dialogue, the CPDLC-user-abort indication may not be provided. A CPDLC-provider-abort indication may result instead.

3.3.9.1.3 The CPDLC-user-abort service shall contain the primitives and parameters as presented in Table 3-8.

Table 3-8. CPDLC-user-abort service parameters

Parameter name	*Req*	*Ind*
Reason	U	M

3.3.9.2 Reason parameter

3.3.9.2.1 The *Reason* parameter is used to indicate a reason for aborting the CPDLC or DSC dialogue.

3.3.9.2.2 If provided by the CPDLC-user, the parameter indicated to the peer CPDLC-user is that provided by the CPDLC-user; otherwise, it is what the ASE supplies.

3.3.9.2.3 The *Reason* parameter value shall conform to the ASN.1 abstract syntax CPDLCUserAbortReason.

3.3.9.2.4 When this parameter is provided by the CPDLC-user, the same value shall be indicated to the peer CPDLC-user.

3.3.10 CPDLC-provider-abort service

3.3.10.1 General

3.3.10.1.1 This service provides the capability for the CPDLC service provider to inform its active users that it can no longer provide the CPDLC service. Messages in transit may be lost during this operation.

3.3.10.1.2 The CPDLC-provider-abort service shall contain the primitives and parameters as presented in Table 3-9.

Table 3-9. CPDLC-provider-abort service parameters

Parameter name	*Ind*
Reason	M

3.3.10.2 Reason parameter

3.3.10.2.1 This parameter identifies the reason for the abort.

3.3.10.2.2 The *Reason* parameter shall conform to the ASN.1 abstract syntax CPDLCProviderAbortReason.

3.4 FORMAL DEFINITIONS OF MESSAGES

3.4.1 Encoding/decoding rules

3.4.1.1 A CPDLC-air-ASE shall be capable of encoding AircraftPDUs APDUs and decoding GroundPDUs APDUs as defined in the ASN.1 module CPDLCAPDUsVersion1 specified in 3.4.2.

Note 1.— The ICUplinkMessage and ICDownlinkMessage ASN.1 types both contain an IC field and the EncodedCPDLCMessage type, which both resolve to a BIT STRING type. Even though this BIT STRING is specified as containing a PER-encoded ATCUplinkMessage or a PER-encoded ATCDownlinkMessage, there is no requirement on the CPDLC-air-ASE to verify this. The CPDLC-air-user is responsible for encoding a valid ATCDownlinkMessage and for verifying the correctness of a received ATCUplinkMessage.

3.4.1.2 A CPDLC-ground-ASE shall be capable of encoding GroundPDUs APDUs and decoding AircraftPDUs APDUs as defined in the ASN.1 module CPDLCAPDUsVersion1 specified in 3.4.2.

Note 1.— The ICUplinkMessage and ICDownlinkMessage ASN.1 types both contain an IC field and the CPDLCMessage type, which both resolve to a BIT STRING type. Even though this BIT STRING is specified as containing a PER-encoded ATCUplinkMessage or a PER-encoded ATCDownlinkMessage, there is no requirement on the CPDLC-ground-ASE to verify this. The CPDLC-ground-user is responsible for encoding a valid ATCUplinkMessage and for verifying the correctness of a received ATCDownlinkMessage.

Note 2.— The ForwardMessage ASN.1 type contains the uplink or downlink CPDLC message element data type, which both resolve to a BIT STRING type. Even though this BIT STRING is specified as containing a PER-encoded ATCUplinkMessageData or a PER-encoded ATCDownlinkMessageData, there is no requirement on the CPDLC-ground-ASE to verify this. The CPDLC-ground-user is responsible for encoding and verifying the correctness of valid ATCDownlinkMessageData and ATCUplinkMessageData.

3.4.2 CPDLC ASN.1 abstract syntax

The abstract syntax of the CPDLC protocol data units shall comply with the description contained in the ASN.1 module CPDLCAPDUsVersion1 (conforming to ISO/IEC 8824), as defined hereinafter:

```
CPDLCAPDUsVersion1 DEFINITIONS::=

BEGIN

-- ------------------------------------------------------------------------------
-- Ground Generated Messages - Top level
-- ------------------------------------------------------------------------------
GroundPDUs ::= CHOICE
        {
        abortUser               [0]     CPDLCUserAbortReason,
        abortProvider           [1]     CPDLCProviderAbortReason,
        startup                 [2]     ICUplinkMessage,
        send                    [3]     ICUplinkMessage,
        forward                 [4]     ATCForwardMessage,
        forwardresponse         [5]     ATCForwardResponse,
```

```
        ...
        }


ICUplinkMessage  ::= SEQUENCE
        {
        algorithmIdentifier         [0]     AlgorithmIdentifier OPTIONAL,
        embeddedMessage             [1]     EncodedCPDLCMessage OPTIONAL,
                                            -- PER encoded ATCUplinkMessage
                                            -- (see Module CPDLCMessageSetVersion1)
        integrityCheck      [2]     BIT STRING,
        ...
        }

ATCForwardMessage ::= SEQUENCE
        {
        forwardHeader               ForwardHeader,
        forwardMessage              ForwardMessage
        }

ForwardHeader ::= SEQUENCE
        {
        dateTime                    DateTimeGroup,
        aircraftID                  AircraftFlightIdentification,
        aircraftAddress             AircraftAddress
        }

ForwardMessage ::= CHOICE
        {
        upElementIDs                [0]     BIT STRING,
                                            --PER encoded ATCUplinkMessageData,
                                            -- (see Module CPDLCMessageSetVersion1)
        downElementIDs              [1]     BIT STRING
                                            --PER encoded ATCDownlinkMessageData,
                                            -- (see Module CPDLCMessageSetVersion1)
        }

ATCForwardResponse ::= ENUMERATED
        {
        success                     (0),
        service-not-supported       (1),
        version-not-equal           (2),
        ...
        }


-- ------------------------------------------------------------------------------
-- Aircraft Generated Messages - Top level
-- ------------------------------------------------------------------------------

AircraftPDUs::= CHOICE
        {
```

```
        abortUser               [0]     CPDLCUserAbortReason,
        abortProvider           [1]     CPDLCProviderAbortReason,
        startdown               [2]     StartDownMessage,
        send                    [3]     ICDownlinkMessage,
        ...
        }

StartDownMessage ::= SEQUENCE
        {
        mode                            Mode DEFAULT cpdlc,
        startDownlinkMessage            ICDownlinkMessage
        }

Mode ::= ENUMERATED
        {
        cpdlc                   (0),
        dsc                     (1)
        }


ICDownlinkMessage::= SEQUENCE
        {
        algorithmIdentifier     [0]     AlgorithmIdentifier OPTIONAL,
        embeddedMessage         [1]     EncodedCPDLCMessage OPTIONAL,
                                        --PER encoded ATCDownlinkMessage,
                                        -- (see Module CPDLCMessageSetVersion1)
        integrityCheck          [2]     BIT STRING,
        ...
        }

-- ------------------------------------------------------------------------------
-- Uplink and Downlink messages - Common Elements
-- ------------------------------------------------------------------------------

AircraftAddress ::= BIT STRING (SIZE(24))

AircraftFlightIdentification ::= IA5String (SIZE (2..8))

AlgorithmIdentifier ::= RELATIVE-OID --root is {icao-arc atn-algorithms(9)}
                                -- see Doc 9880, Part IV, for OID root assignment

EncodedCPDLCMessage ::= BIT STRING

CPDLCUserAbortReason ::= ENUMERATED
        {
        undefined                                       (0),
        no-message-identification-numbers-available     (1),
        duplicate-message-identification-numbers        (2),
        no-longer-next-data-authority                   (3),
        current-data-authority-abort                    (4),
        commanded-termination                           (5),
        invalid-response                                (6),
```

```
        time-out-of-synchronization             (7),
        unknown-integrity-check                 (8),
        validation-failure                      (9),
        unable-to-decode-message                (10),
        invalid-pdu                             (11),
        invalid-CPDLC-message                   (12),
        ...
        }

CPDLCProviderAbortReason ::= ENUMERATED
        {
        timer-expired                   (0),
        undefined-error                 (1),
        invalid-PDU                     (2),
        protocol-error                  (3),
        communication-service-error     (4),
        communication-service-failure   (5),
        invalid-QOS-parameter           (6),
        expected-PDU-missing            (7),
        ...
        }

Date ::= SEQUENCE
        {
        year            Year,
        month           Month,
        day             Day
        }

DateTimeGroup ::= SEQUENCE
        {
        date            Date,
        timehhmmss      Timehhmmss
        }

Day ::= INTEGER (1..31)
        --unit = Day, Range (1..31), resolution = 1

Month ::= INTEGER (1..12)
                --unit = Month, Range (1..12), resolution = 1

Time ::= SEQUENCE
        {
        hours           TimeHours,
        minutes         TimeMinutes
        }

TimeHours ::= INTEGER (0..23)
        -- unit = Hour, Range (0..23), resolution = 1

Timehhmmss ::= SEQUENCE
        {
```

```
    hoursminutes        Time,
    seconds             TimeSeconds
    }

TimeMinutes ::= INTEGER (0..59)
    -- unit = Minute, Range (0..59), resolution = 1

TimeSeconds ::= INTEGER (0..59)
    -- unit = Second, Range (0..59), resolution = 1

Year ::= INTEGER (1996..2095)
    -- unit = Year, Range (1996..2095), resolution = 1


END
```

3.5 PROTOCOL DEFINITION

3.5.1 Sequence rules

3.5.1.1 With the exception of abort primitives, only the sequence of primitives illustrated in Figures 3-2 to 3-19 shall be permitted.

3.5.1.2 Figures 3-2 to 3-19 define the valid sequences of primitives that are possible to be invoked during the operation of the CPDLC application. The figures show the relationship in time between the service request and the resulting indication and, if applicable, the subsequent response and resulting confirmation.

3.5.1.3 Abort primitives may interrupt and terminate any of the normal message sequences outlined in the rest of Section 3.5.

3.5.1.4 Primitives are processed in the order in which they are received.

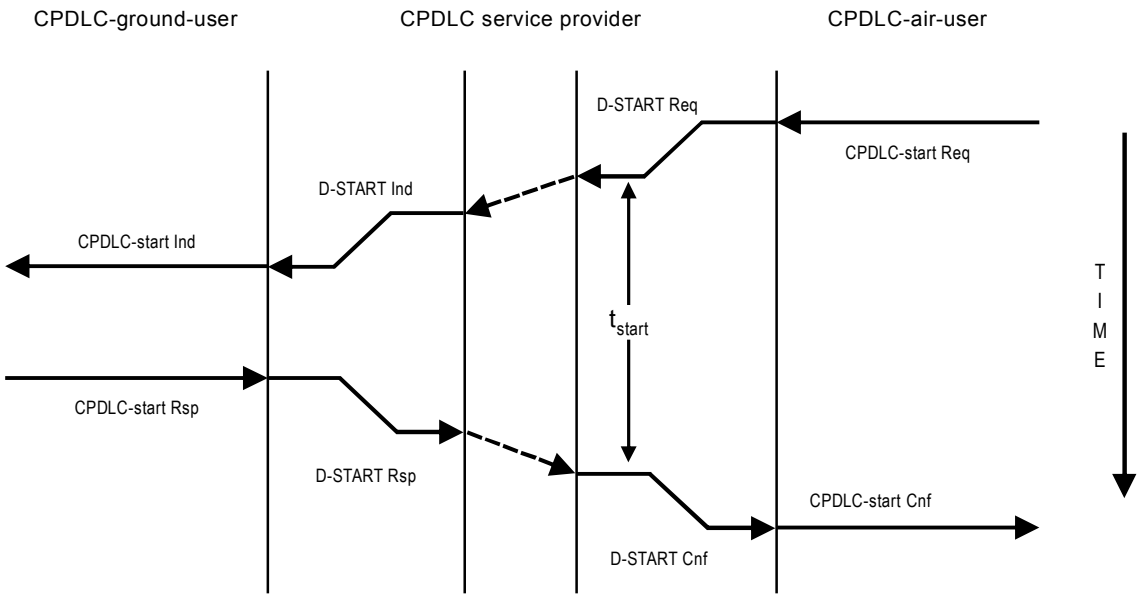


Figure 3-2. Sequence diagram for CPDLC-start service — Air-initiated

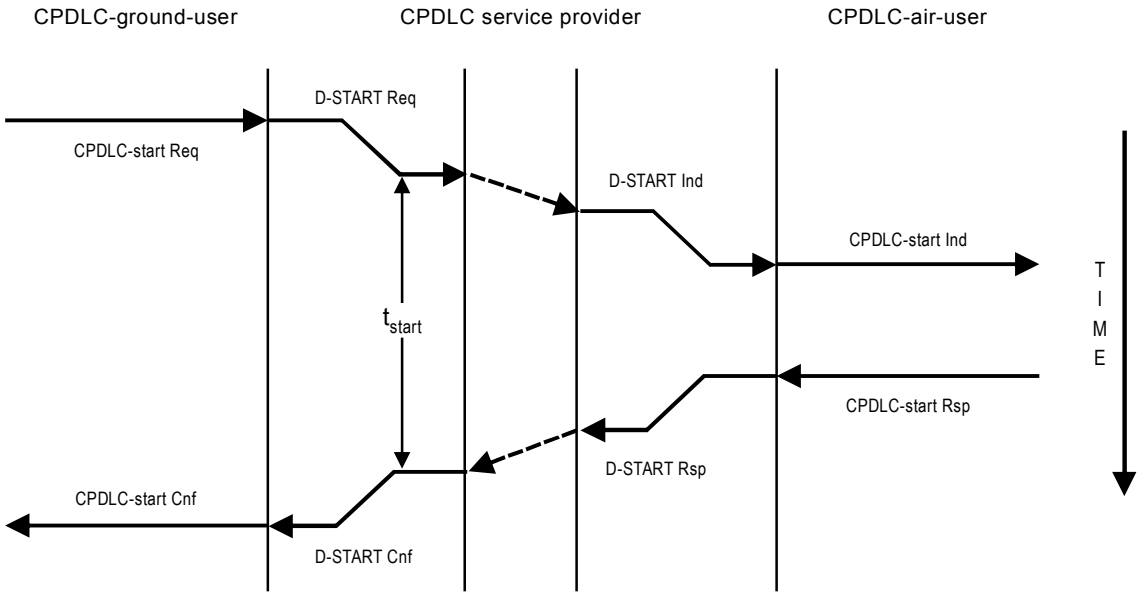


Figure 3-3. Sequence diagram for CPDLC-start service — Ground-initiated

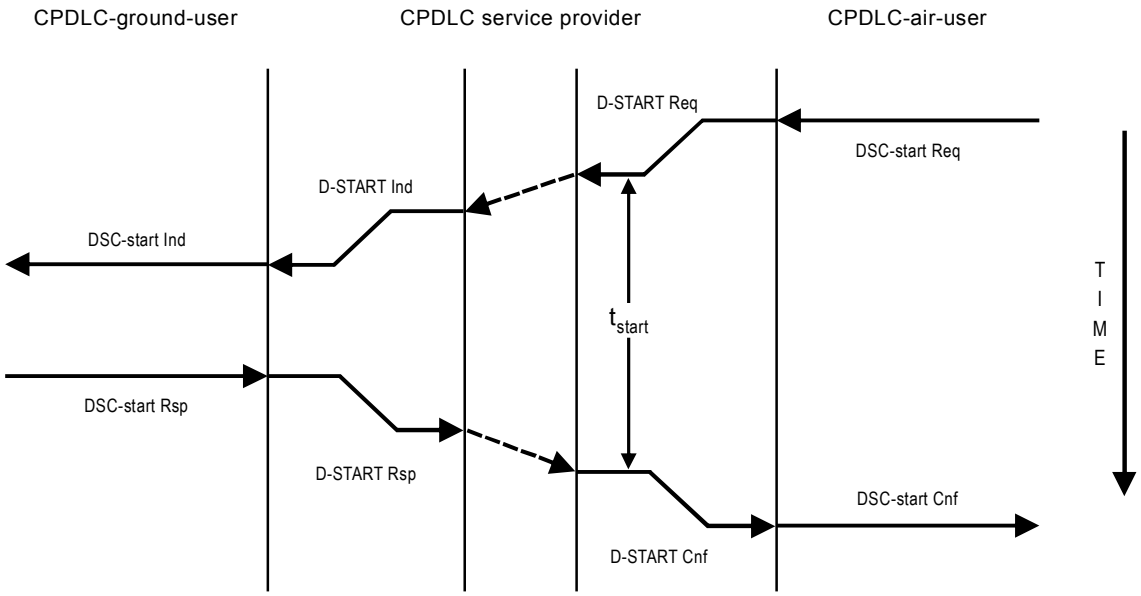


Figure 3-4. Sequence diagram for DSC-start service

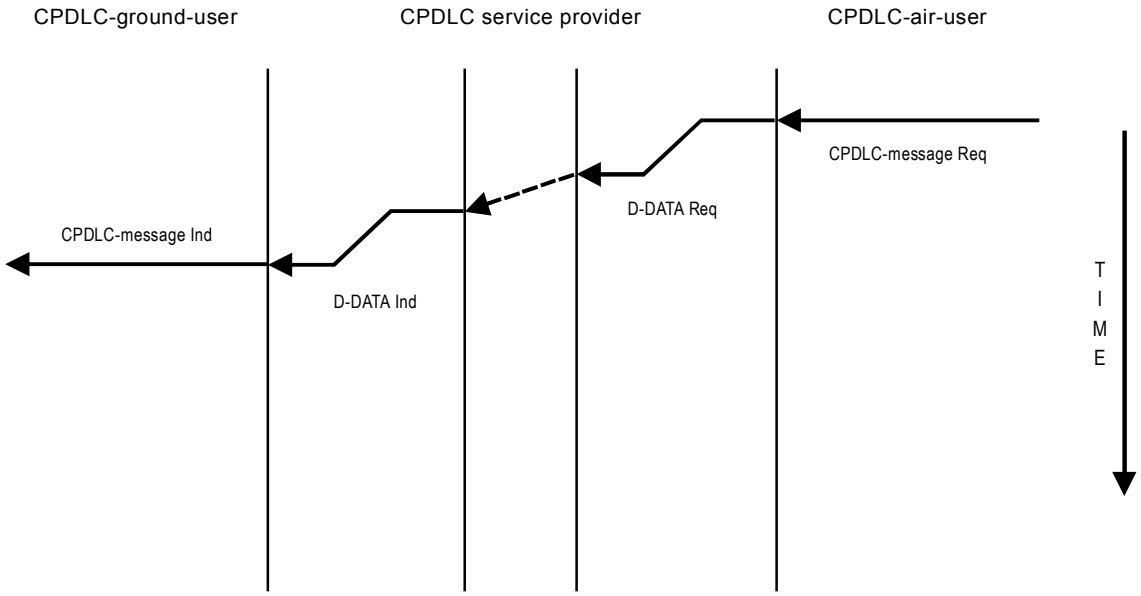


Figure 3-5. Sequence diagram for CPDLC-message service — Air-initiated

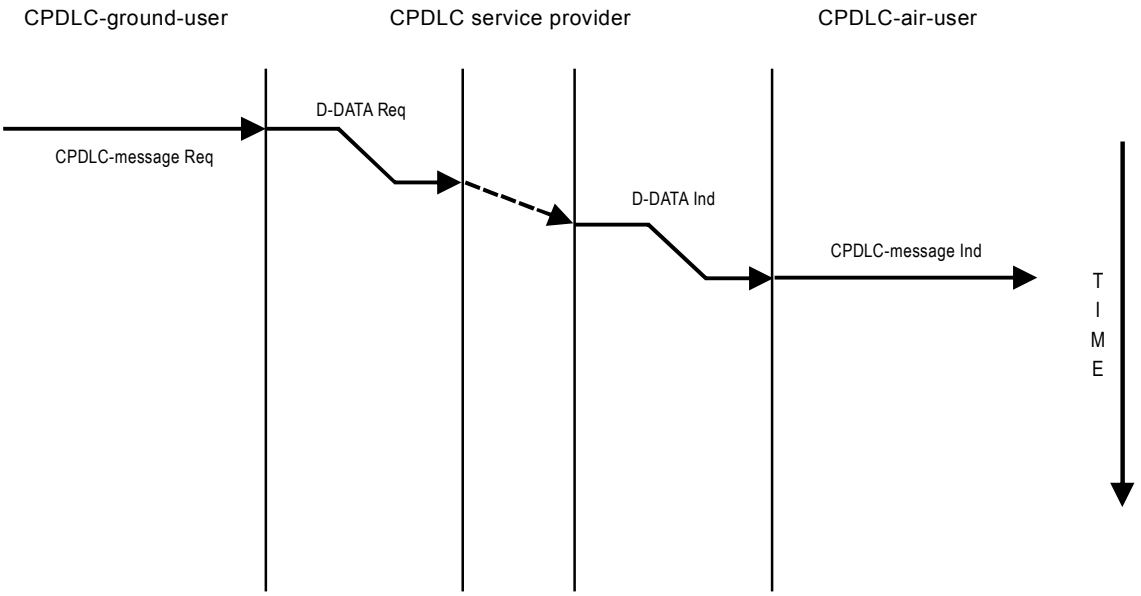


Figure 3-6. Sequence diagram for CPDLC-message service — Ground-initiated

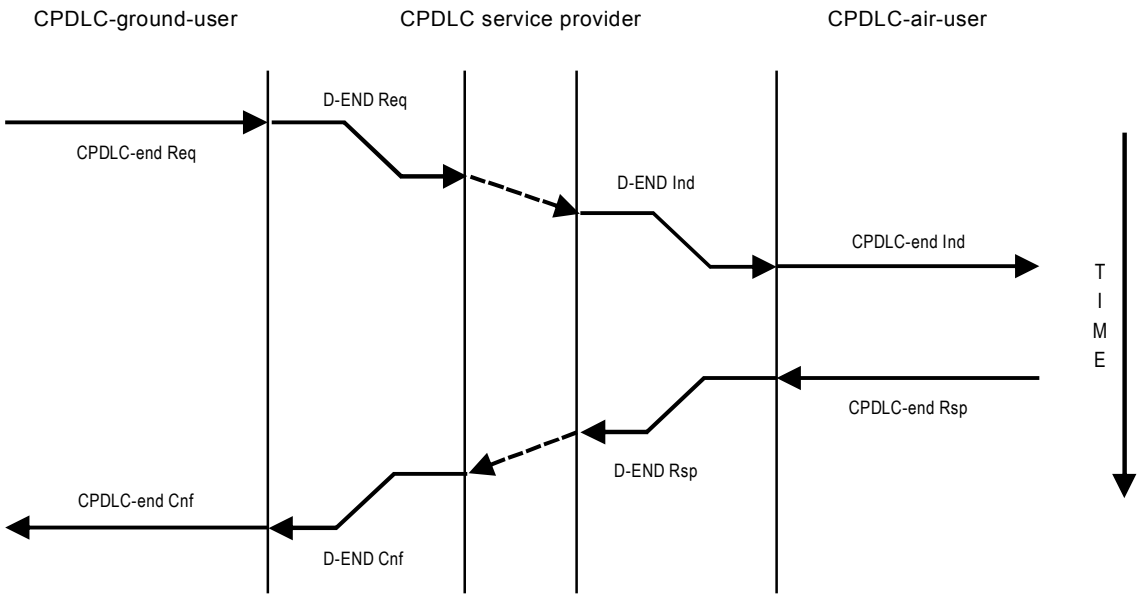


Figure 3-7. Sequence diagram for CPDLC-end service

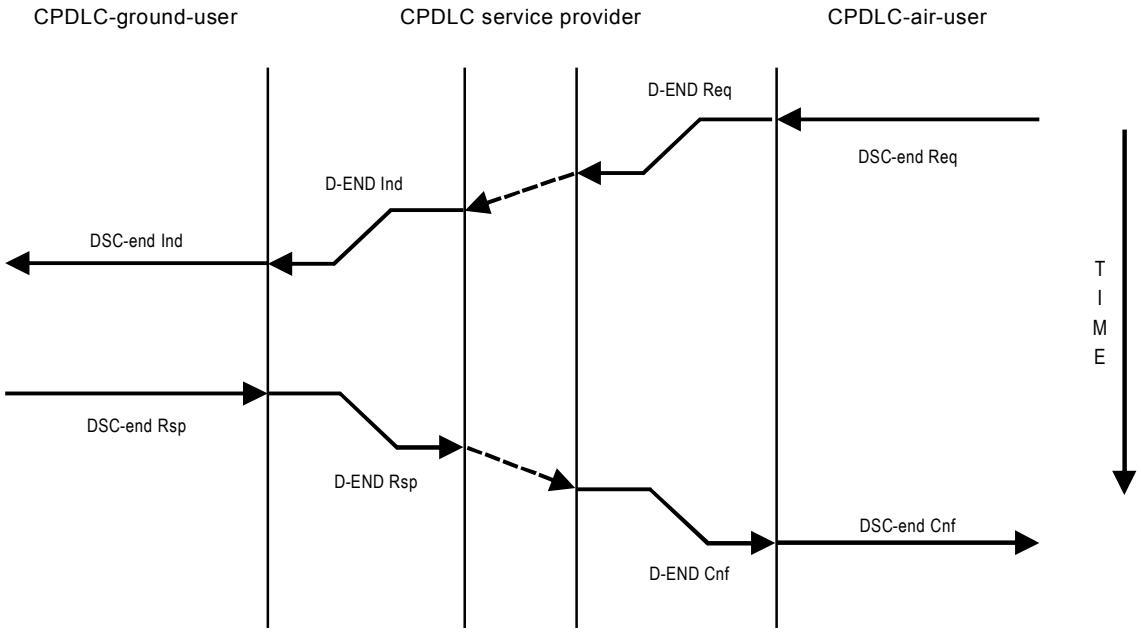


Figure 3-8. Sequence diagram for DSC-end service

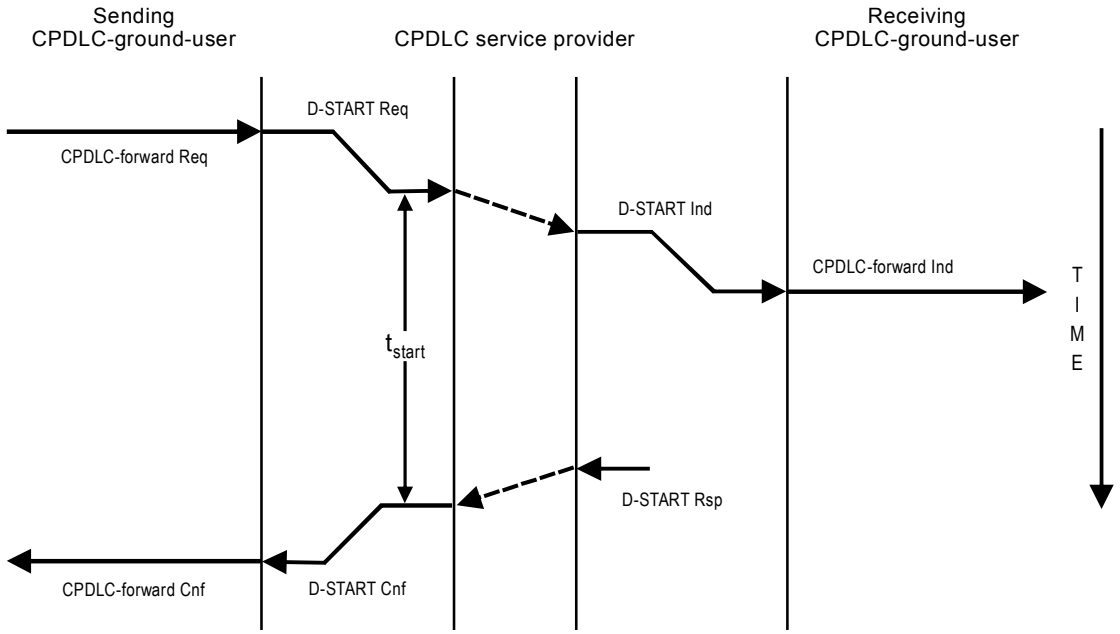


Figure 3-9. Sequence diagram for CPDLC-forward service
— Ground forwarding supported, ASE version numbers the same

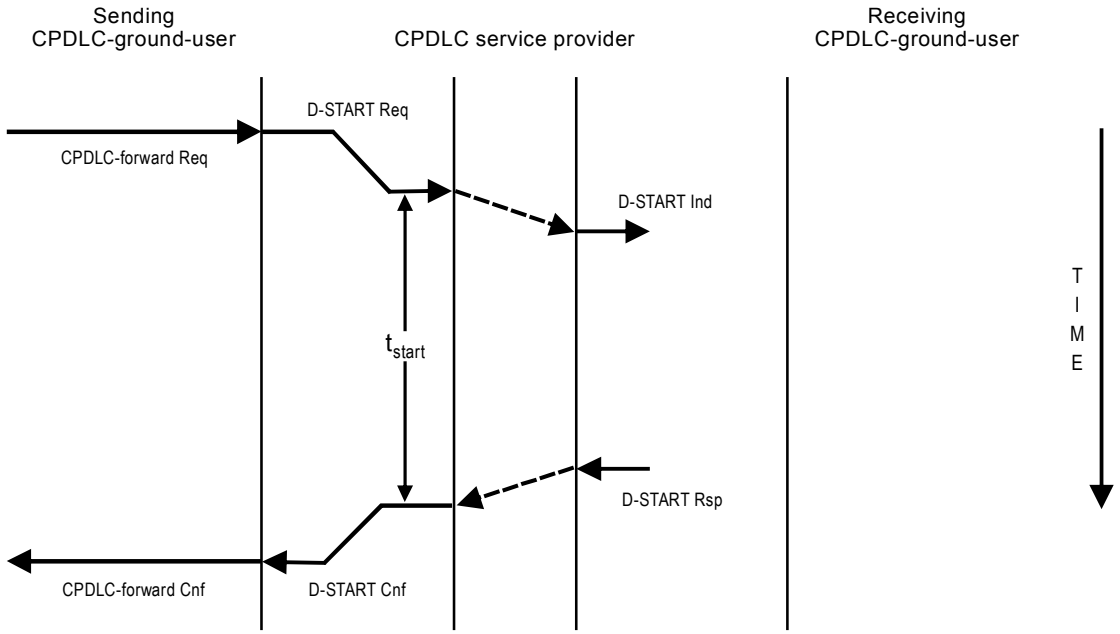


Figure 3-10. Sequence diagram for CPDLC-forward service
— Ground forwarding not supported, or
— Ground forwarding supported and ASE version numbers not the same

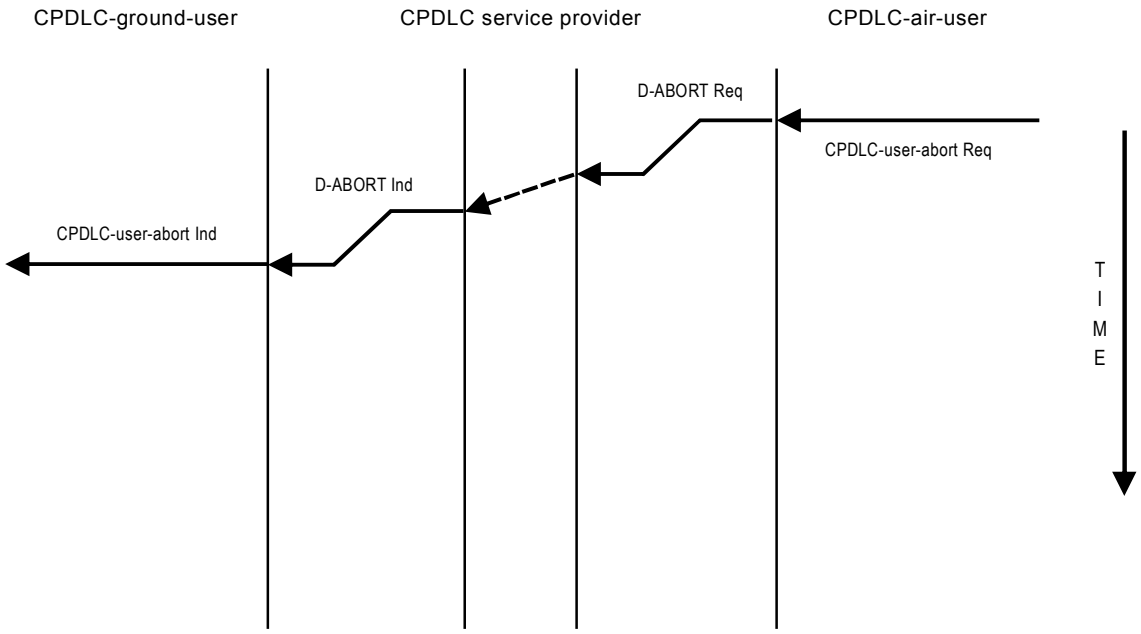


Figure 3-11. Sequence diagram for CPDLC-user-abort service — CPDLC-air-user initiated

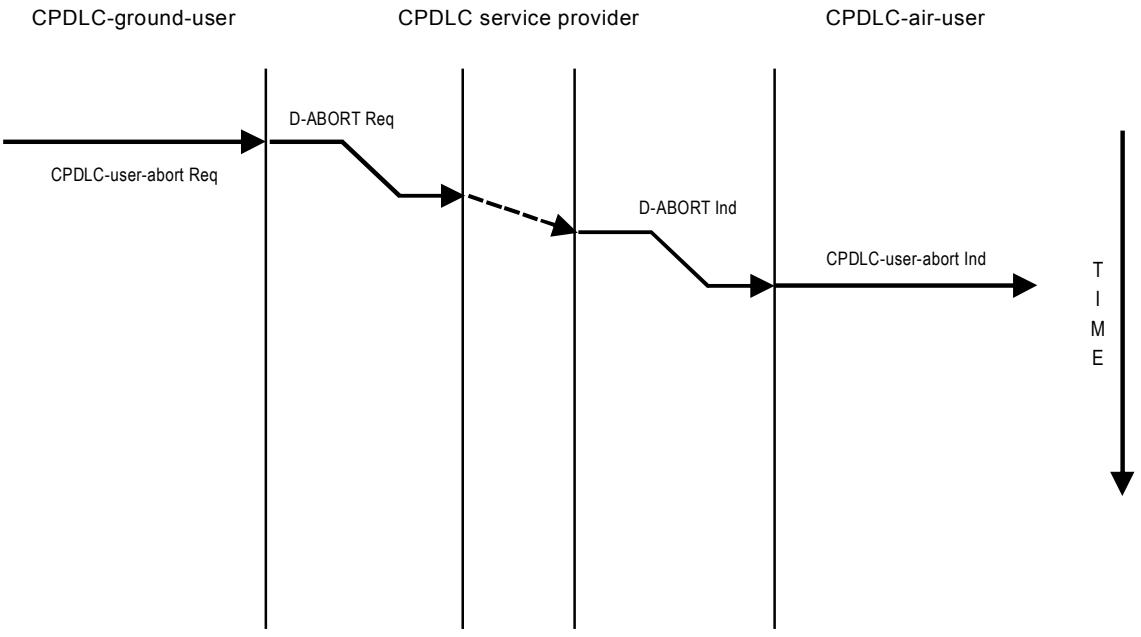


Figure 3-12. Sequence diagram for CPDLC-user-abort service — CPDLC-ground-user initiated

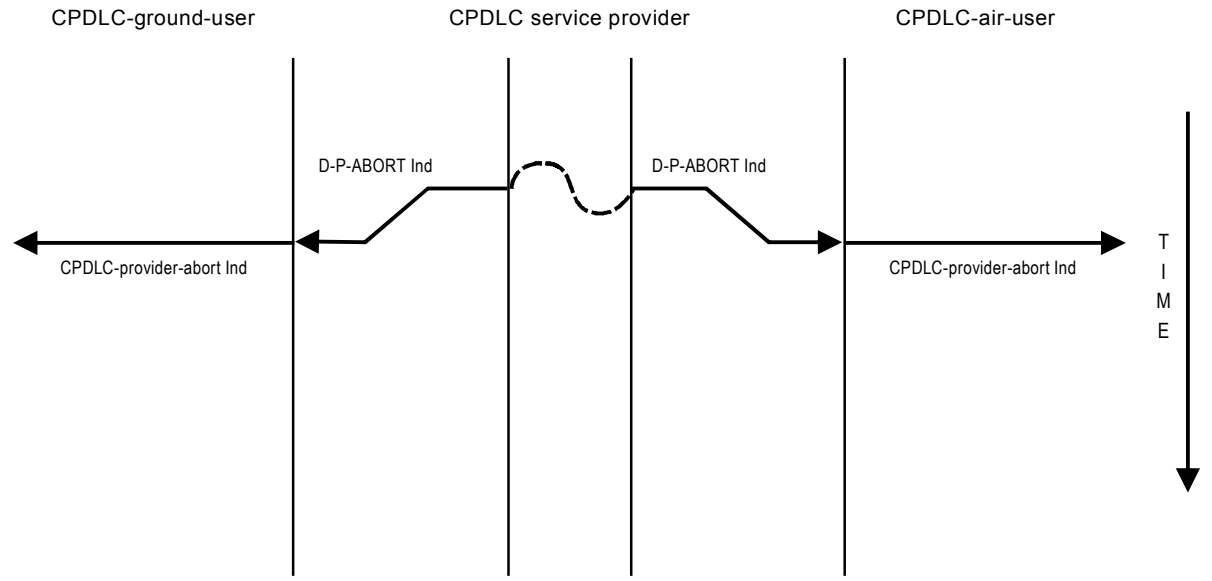


Figure 3-13. Sequence diagram for CPDLC-provider-abort service — dialogue service abort

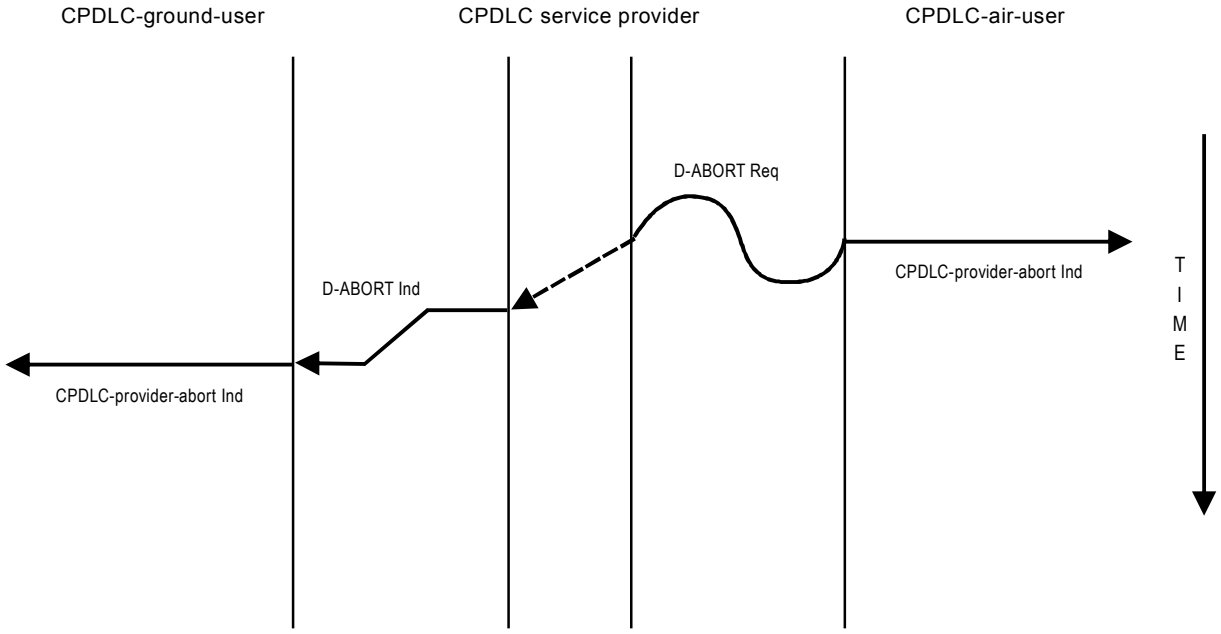


Figure 3-14. Sequence diagram for CPDLC-provider-abort service — CPDLC-air-ASE abort

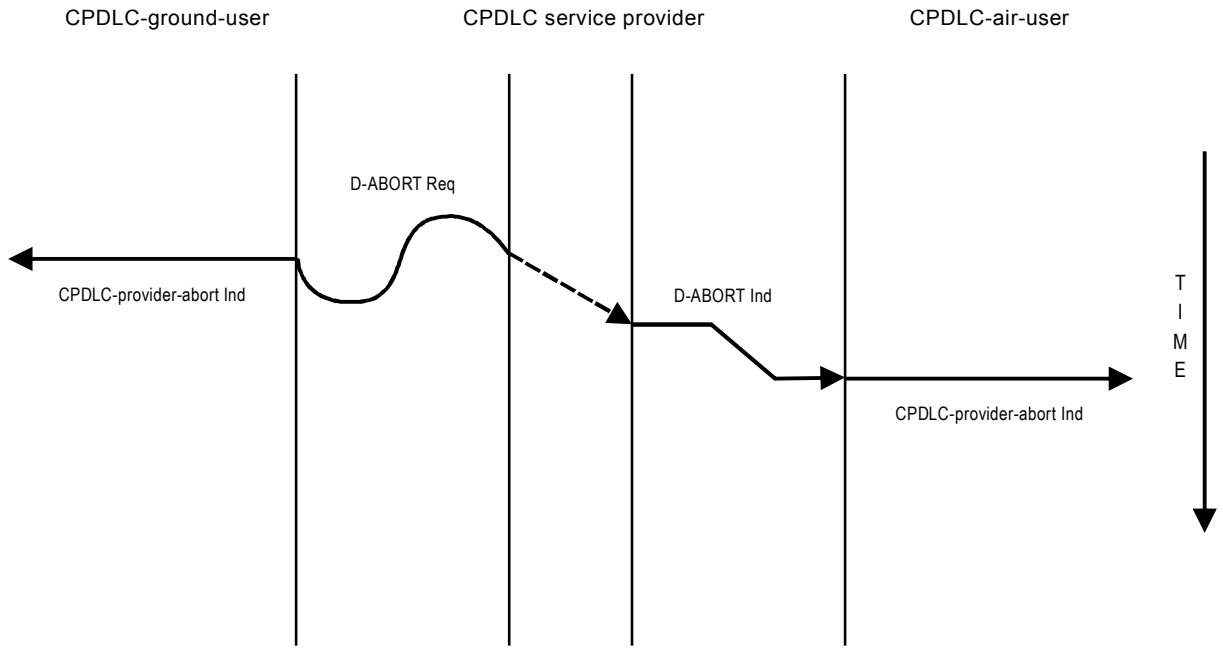


Figure 3-15. Sequence diagram for CPDLC-provider-abort service — CPDLC-ground-ASE abort

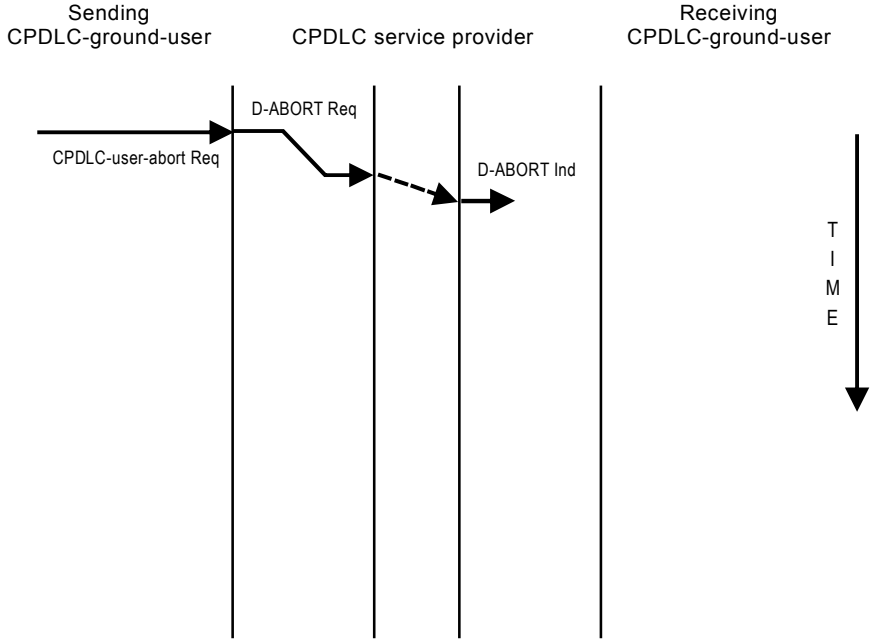


Figure 3-16. Sequence diagram for CPDLC-user-abort service — Sending CPDLC-ground-user initiated

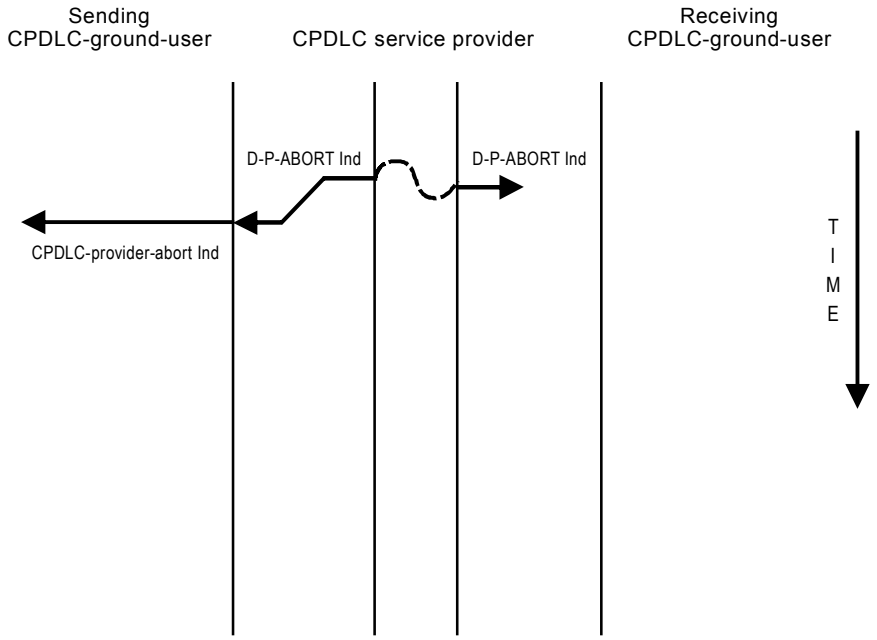


Figure 3-17. Sequence diagram for CPDLC-provider-abort service — Dialogue service abort

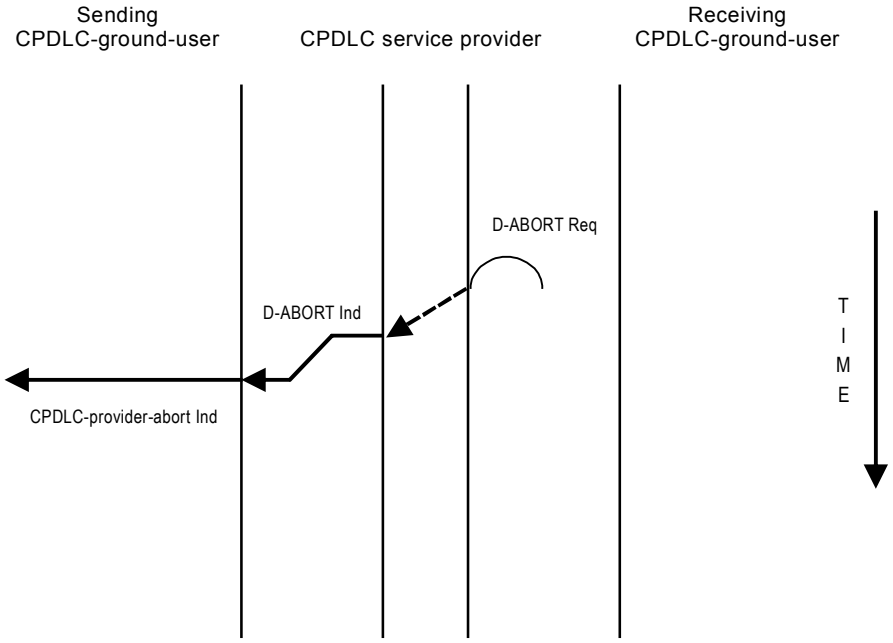


Figure 3-18. Sequence diagram for CPDLC-provider-abort service — Receiving CPDLC-ground-ASE abort

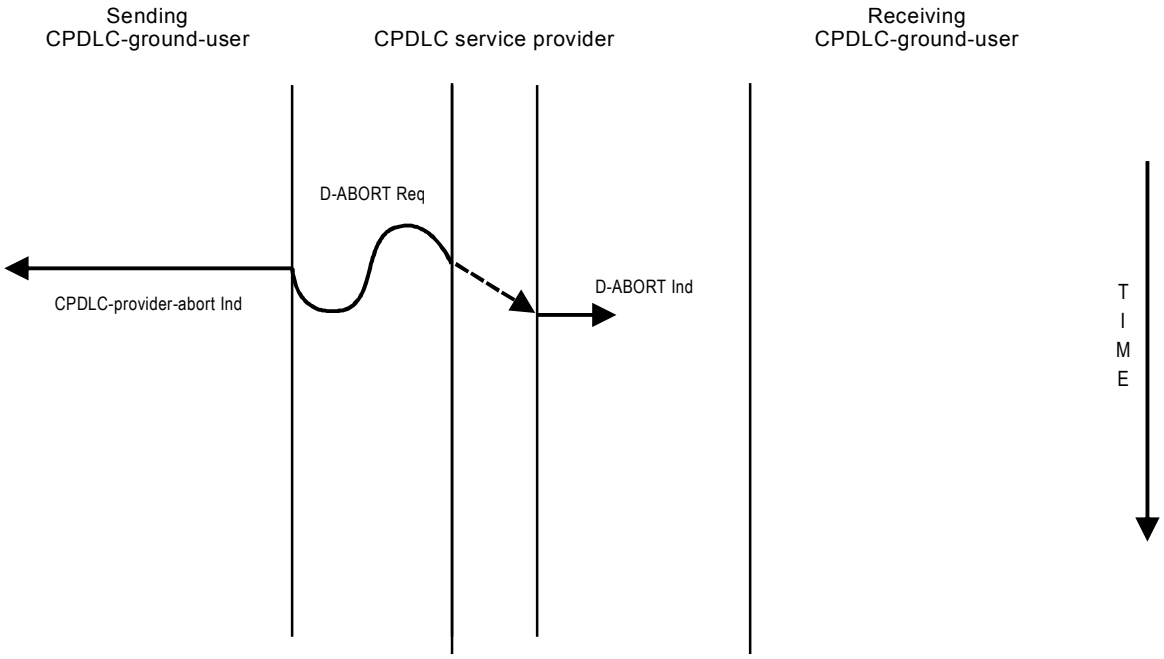


Figure 3-19. Sequence diagram for CPDLC-provider-abort service — Sending CPDLC-ground-ASE abort

3.5.2 CPDLC service provider timers

3.5.2.1 A CPDLC-ASE shall be capable of detecting when a timer expires.

3.5.2.2 Table 3-10 lists the time constraints related to the CPDLC application. Each time constraint requires a timer to be set in the CPDLC protocol machine.

3.5.2.3 If the timer expires before the final event has occurred, a CPDLC-ASE takes appropriate action as defined in 3.5.4.1.

3.5.2.4 The timer values should be as indicated in Table 3-10.

Table 3-10. CPDLC service provider timers

CPDLC service	*Timer*	*Timer value*	*Timer start event*	*Timer stop event*
CPDLC-start	t_{start}	6 minutes	D-START request	D-START confirmation
DSC-start	t_{start}	6 minutes	D-START request	D-START confirmation
CPDLC-forward	t_{start}	6 minutes	D-START request	D-START confirmation

Note.— The receipts of CPDLC-user-abort requests, D-ABORT indications or D-P-ABORT indications are also timer stop events.

3.5.3 CPDLC-air-ASE protocol description

3.5.3.1 Introduction

3.5.3.1.1 If no actions are described for a CPDLC service primitive when a CPDLC-air-ASE is in a specific state, then the invocation of that service primitive shall be prohibited while the CPDLC-air-ASE is in that state.

3.5.3.1.2 Upon receipt of a PDU, if no actions are described for the arrival of that PDU when a CPDLC-air-ASE is in a specific state, then that PDU is considered not permitted and exception handling procedures as described in 3.5.4.4 shall apply.

3.5.3.1.3 If a PDU is received that cannot be decoded, then exception handling procedures for an invalid PDU as described in 3.5.4.3 shall apply.

3.5.3.1.4 If a PDU is not received when one is required, then exception handling procedures as described in 3.5.4.3 shall apply.

3.5.3.1.5 The states defined for the CPDLC-air-ASE are the following:

a) IDLE;

b) START-REQ;

c) START-IND;

d) DIALOGUE; and

e) END.

3.5.3.1.6 The CPDLC-air-user is an active user from the time:

a) the CPDLC-air-user invokes the CPDLC-start service request until it:

1) receives a CPDLC-start service confirmation with the Result parameter equal to the abstract value “rejected”; or

2) invokes a CPDLC-end service response with the Result parameter set to the abstract value “accepted”; or

3) invokes a CPDLC-user-abort service request; or

4) receives a CPDLC-user-abort service indication; or

5) receives a CPDLC-provider-abort service indication; or

b) the CPDLC-air-user receives the CPDLC-start service indication until it:

1) invokes a CPDLC-start service response with the Result parameter equal to the abstract value “rejected”; or

2) invokes a CPDLC-end service response with the Result parameter set to the abstract value “accepted”; or

3) invokes a CPDLC-user-abort service request; or

4) receives a CPDLC-user-abort service indication; or

5) receives a CPDLC-provider-abort service indication; or

c) the CPDLC-air-user invokes the DSC-start service request until it:

1) receives a DSC-start service confirmation with the Result parameter equal to the abstract value "rejected"; or

2) receives a DSC-end service confirmation with the Result parameter equal to the abstract value "accepted"; or

3) invokes a CPDLC-user-abort service request; or

4) receives a CPDLC-user-abort service indication; or

5) receives a CPDLC-provider-abort service indication.

3.5.3.1.7 On initiation, the CPDLC-air-ASE shall be in the *IDLE* state.

3.5.3.1.8 The CPDLC-air-ASE contains a Boolean called DSC. DSC has the abstract value "true" when the dialogue is a DSC dialogue and has the abstract value "false" otherwise.

3.5.3.1.9 On the initiation of a CPDLC-air-ASE, DSC shall be set to the abstract value "false".

3.5.3.2 D-START indication

Upon receipt of a D-START indication, if the CPDLC-air-ASE is in the *IDLE* state, and the D-START *User Data* parameter contains a GroundPDUs [ICUplinkMessage(startup)] APDU, and the D-START *QOS Priority* parameter has the abstract value "high priority flight safety messages", and the D-START *QOS Residual Error Rate* parameter has the abstract value "low", and the D-START *QOS Routing Class* parameter has one of the abstract values specified in Table 3-13 (see 3.6.2.2), and the D-START *Calling Peer ID* parameter is a valid four- to eight-character facility designation, and the D-START *Security Requirements* parameter is consistent with the local security policy, the CPDLC-air-ASE shall:

a) invoke CPDLC-start service indication with:

1) the D-START *Calling Peer ID* parameter value as the CPDLC-start service Calling Peer Identifier parameter value;

2) the D-START *QOS Routing Class* parameter value as the CPDLC-start service Class of Communication parameter value;

b) enter the *START-IND* state.

3.5.3.3 D-START confirmation

3.5.3.3.1 Upon receipt of a D-START confirmation, if the CPDLC-air-ASE is in the *START-REQ* state, and the

D-START *Result* parameter has the abstract value "accepted", and DSC has the abstract value "false", and the D-START *User Data* parameter contains a GroundPDUs [ICUplinkMessage(send)] APDU, and the D-START *Security Requirements* parameter value is the same as the one set for the D-START request, the CPDLC-air-ASE shall:

a) stop timer t_{start};

b) invoke CPDLC-start service confirmation with:

1) the APDU contained in the D-START *User Data* parameter as the CPDLC-start service *Response CPDLC/IC Data* parameter value; and

2) the abstract value "accepted" as the CPDLC-start service *Result* parameter value; and

c) enter the *DIALOGUE* state.

3.5.3.3.2 Upon receipt of a D-START confirmation, if the CPDLC-air-ASE is in the *START-REQ* state, and the D-START *Result* parameter has the abstract value "rejected (permanent)", and the D-START *Reject Source* parameter has the abstract value "DS-user", and DSC has the abstract value "false", and the D-START *User Data* parameter contains a GroundPDUs [ICUplinkMessage(send)] APDU, and the D-START *Security Requirements* parameter value is the same as the one set for the D-START request, the CPDLC-air-ASE shall:

a) stop timer t_{start};

b) invoke CPDLC-start service confirmation with:

1) the APDU contained in the D-START *User Data* parameter as the CPDLC-start service *Response CPDLC/IC Data* parameter value; and

2) the abstract value "rejected" as the CPDLC-start service *Result* parameter value; and

c) enter the *IDLE* state.

3.5.3.3.3 Upon receipt of a D-START confirmation, if the CPDLC-air-ASE is in the *START-REQ* state, and the D-START *Result* parameter has the abstract value "accepted", and DSC has the abstract value "true", and the D-START *User Data* parameter contains a GroundPDUs [ICUplinkMessage(send)] APDU, and the D-START *Security Requirements* parameter value is the same as the one set for the D-START request, the CPDLC-air-ASE shall:

a) stop timer t_{start};

b) invoke DSC-start service confirmation with:

1) the APDU contained in the D-START *User Data* parameter as the DSC-start service *Response CPDLC/IC Data* parameter value; and

2) the abstract value "accepted" as the DSC-start service *Result* parameter value; and

c) enter the *DIALOGUE* state.

3.5.3.3.4 Upon receipt of a D-START confirmation, if the CPDLC-air-ASE is in the *START-REQ* state, and the D-START *Result* parameter has the abstract value "rejected (permanent)", and the D-START *Reject Source* parameter has the abstract value "DS-user", and DSC has the abstract value "true", and the D-START *User Data* parameter contains a GroundPDUs [ICUplinkMessage(send)] APDU, the CPDLC-air-ASE shall:

a) stop timer t_{start};

b) invoke DSC-start service confirmation with:

1) the APDU contained in the D-START *User Data* parameter as the DSC-start service *Response CPDLC/IC Data* parameter value; and

2) the abstract value “rejected” as the DSC-start service *Result* parameter value;

c) set DSC to the abstract value “false”; and

d) enter the *IDLE* state.

3.5.3.4 D-DATA indication

3.5.3.4.1 Upon receipt of a D-DATA indication, if the CPDLC-air-ASE is in the *DIALOGUE* state, and the APDU contained in the D-DATA *User Data* parameter is a GroundPDUs [ICUplinkMessage(send)] APDU, the CPDLC-air-ASE shall:

a) invoke CPDLC-message service indication with the APDU contained in the D-DATA *User Data* parameter as the CPDLC-message service *CPDLC/IC Data* parameter value; and

b) remain in the *DIALOGUE* state.

3.5.3.4.2 Upon receipt of a D-DATA indication, if the CPDLC-air-ASE is in the *END* state, and DSC has the abstract value “true”, and the APDU contained in the D-DATA *User Data* parameter is a GroundPDUs [ICUplinkMessage(send)] APDU, the CPDLC-air-ASE shall:

a) invoke CPDLC-message service indication with the APDU contained in the D-DATA *User Data* parameter as the CPDLC-message service *CPDLC/IC Data* parameter value; and

b) remain in the *END* state.

3.5.3.5 D-END indication

Upon receipt of a D-END indication, if the CPDLC-air-ASE is in the *DIALOGUE* state, and DSC has the abstract value “false”, and the D-END *User Data* parameter contains a GroundPDUs [ICUplinkMessage(send)] APDU, the CPDLC-air-ASE shall:

a) invoke CPDLC-end service indication with the APDU contained in the D-END *User Data* parameter as the CPDLC-end service *CPDLC/IC Data* parameter value; and

b) enter the *END* state.

3.5.3.6 D-END confirmation

3.5.3.6.1 Upon receipt of a D-END confirmation, if the CPDLC-air-ASE is in the *END* state, and the D-END *Result* parameter has the abstract value “accepted”, and DSC has the abstract value “true”, and the D-END *User Data*

parameter contains a GroundPDUs [ICUplinkMessage(send)] APDU, the CPDLC-air-ASE shall:

a) invoke DSC-end service confirmation with:

1) the APDU contained in the D-END *User Data* parameter as the DSC-end service *CPDLC/IC Data* parameter value; and

2) the abstract value “accepted” as the CPDLC-end service *Result* parameter value;

b) set DSC to the abstract value “false”; and

c) enter the *IDLE* state.

3.5.3.6.2 Upon receipt of a D-END confirmation, if the CPDLC-air-ASE is in the *END* state, and the D-END *Result* parameter has the abstract value “rejected”, and DSC has the abstract value “true”, and the D-END *User Data* parameter contains a GroundPDUs [ICUplinkMessage(send)] APDU, the CPDLC-air-ASE shall:

a) invoke DSC-end service confirmation with:

1) the APDU contained in the D-END *User Data* parameter as the DSC-end service *CPDLC/IC Data* parameter value; and

2) the abstract value “rejected” as the CPDLC-end service *Result* parameter value; and

b) enter the *DIALOGUE* state.

3.5.3.7 CPDLC-start service request

Upon receipt of a CPDLC-start service request, if the CPDLC-air-ASE is in the *IDLE* state, the CPDLC-air-ASE shall:

a) create an AircraftPDUs APDU with a StartDownMessage APDU element containing:

1) the abstract value “cpdlc” as the mode; and

2) the *CPDLC/IC Data* parameter as the ICDownlinkMessage;

b) invoke D-START request with:

1) the CPDLC-start service *Called Peer Identifier* parameter value as the D-START *Called Peer ID* parameter value;

2) the CPDLC-start service *Calling Peer Identifier* parameter value as the D-START *Calling Peer ID* parameter value;

3) the CPDLC-start service *CPDLC Message Set Version Number* parameter value if specified by the CPDLC-user as the D-START *DS-User Version Number* parameter value;

4) the CPDLC-start service *Security Required* parameter value if specified by the CPDLC-user or otherwise the abstract value “no security” as the D-START *Security Requirements* parameter value;

5) the D-START *Quality of Service* parameters set as follows:

i) if provided, the CPDLC-start service *Class of Communication* parameter value as the D-START *QOS Routing Class* parameter value; and

ii) the abstract value of "high priority flight safety messages" as the D-START *QOS Priority* parameter value; and

iii) the abstract value of "low" as the D-START *QOS Residual Error Rate* parameter value; and

6) the APDU as the D-START *User Data* parameter value;

c) start timer t_{start}; and

d) enter the *START-REQ* state.

3.5.3.8 CPDLC-start service response

3.5.3.8.1 Upon receipt of a CPDLC-start service response, if the CPDLC-air-ASE is in the *START-IND* state, and the CPDLC-start service *Result* parameter has the abstract value "accepted", and DSC has the abstract value "false", the CPDLC-air-ASE shall:

a) create an AircraftPDUs APDU with an ICDownlinkMessage APDU element based on the *Response CPDLC/IC Data* parameter;

b) invoke D-START response with the following:

1) the APDU as the D-START *User Data* parameter value;

2) the abstract value received in the D-START indication *Security Requirements* parameter as the D-START *Security Requirements* parameter value; and

3) the abstract value "accepted" as the D-START *Result* parameter value; and

c) enter the *DIALOGUE* state.

3.5.3.8.2 Upon receipt of a CPDLC-start service response, if the CPDLC-air-ASE is in the *START-IND* state, and the CPDLC-start service *Result* parameter has the abstract value "rejected", and DSC has the abstract value "false", the CPDLC-air-ASE shall:

a) create an AircraftPDUs APDU with an ICDownlinkMessage APDU element based on the *Response CPDLC/IC Data* parameter;

b) invoke D-START response with:

1) the APDU as the D-START *User Data* parameter value;

2) the abstract value received in the D-START indication *Security Requirements* parameter as the D-START *Security Requirements* parameter value; and

3) the abstract value "rejected (permanent)" as the D-START *Result* parameter value; and

c) enter the *IDLE* state.

3.5.3.9 DSC-start service request

Upon receipt of a DSC-start service request, if the CPDLC-air-ASE is in the *IDLE* state, the CPDLC-air-ASE shall:

a) create an AircraftPDUs APDU with a StartDownMessage APDU element containing:

1) the abstract value “dsc” as the mode; and

2) the *CPDLC/IC Data* parameter as the ICDownlinkMessage;

b) invoke D-START request with:

1) the DSC-start service *Facility Designation* parameter value as the D-START *Called Peer ID* parameter value;

2) the DSC-start service *Aircraft Address* parameter value as the D-START *Calling Peer ID* parameter value;

3) the DSC-start service *CPDLC Message Set Version Number* parameter value if specified by the CPDLC-user as the D-START *DS-User Version Number* parameter value;

4) the DSC-start service *Security Required* parameter value if specified by the CPDLC-air-user or otherwise the abstract value “no security” as the D-START *Security Requirements* parameter value;

5) the D-START *Quality of Service* parameters set as follows:

i) if provided, the DSC-START service *Class of Communication* parameter value as the D-START *QOS Routing Class* parameter value;

ii) the abstract value of “high priority flight safety messages” as the D-START *QOS Priority* parameter value; and

iii) the abstract value of “low” as the D-START *QOS Residual Error Rate* parameter value; and

6) the APDU as the D-START *User Data* parameter value;

c) set DSC to the abstract value “true”;

d) start timer t_{start}; and

e) enter the *START-REQ* state.

3.5.3.10 CPDLC-message service request

3.5.3.10.1 Upon receipt of a CPDLC-message service request and the *CPDLC/IC Data* parameter contains a CPDLC message, if the CPDLC-air-ASE is in the *DIALOGUE* state, the CPDLC-air-ASE shall:

a) create an AircraftPDUs APDU with an ICDownlinkMessage APDU element based on the CPDLC-message service *CPDLC/IC Data* parameter;

b) invoke D-DATA request with the APDU as the D-DATA *User Data* parameter value; and

c) remain in the *DIALOGUE* state.

3.5.3.10.2 Upon receipt of a CPDLC-message service request and the *CPDLC/IC Data* parameter contains a CPDLC message, if the CPDLC-air-ASE is in the *END* state and DSC has the abstract value “false”, the CPDLC-air-ASE shall:

a) create an AircraftPDUs APDU with an ICDownlinkMessage APDU element based on the CPDLC-message service *CPDLC/IC Data* parameter;

b) invoke D-DATA request with the APDU as the D-DATA *User Data* parameter value; and

c) remain in the *END* state.

3.5.3.11 CPDLC-end service response

3.5.3.11.1 Upon receipt of a CPDLC-end service response, if the CPDLC-air-ASE is in the *END* state, and the CPDLC-end service *Result* parameter has the abstract value “accepted”, and DSC has the abstract value “false”, the CPDLC-air-ASE shall:

a) create an AircraftPDUs APDU with an ICDownlinkMessage APDU element based on the CPDLC-end service *CPDLC/IC Data* parameter;

b) invoke D-END response with:

1) the APDU as the D-END *User Data* parameter value; and

2) the abstract value “accepted” as the D-END *Result* parameter value; and

c) enter the *IDLE* state.

3.5.3.11.2 Upon receipt of a CPDLC-end service response, if the CPDLC-air-ASE is in the *END* state, and the CPDLC-end service *Result* parameter has the abstract value “rejected”, and DSC has the abstract value “false”, the CPDLC-air-ASE shall:

a) create an AircraftPDUs APDU with an ICDownlinkMessage APDU element based on the CPDLC-end service *CPDLC/IC Data* parameter;

b) invoke D-END response with:

1) the APDU as the D-END *User Data* parameter value; and

2) the abstract value “rejected” as the D-END *Result* parameter value; and

c) enter the *DIALOGUE* state.

3.5.3.12 DSC-end service request

Upon receipt of a DSC-end service request, if the CPDLC-air-ASE is in the *DIALOGUE* state and DSC has the abstract

value “true”, the CPDLC-air-ASE shall:

a) create an AircraftPDUs APDU with an ICDownlinkMessage APDU element based on the DSC-end service *CPDLC/IC Data* parameter;

b) invoke D-END request with the APDU as the D-END *User Data* parameter value; and

c) enter the *END* state.

3.5.3.13 CPDLC-user-abort service request

Upon receipt of a CPDLC-user-abort service request, if the CPDLC-air-ASE is not in the *IDLE* state, the CPDLC-air-ASE shall:

a) stop any timer;

b) if the CPDLC-user-abort service *Reason* parameter is provided, create an AircraftPDUs APDU with a CPDLCUserAbortReason APDU element based on the CPDLC-user-abort service *Reason* parameter;

c) if the CPDLC-user-abort service *Reason* parameter is not provided, create an AircraftPDUs APDU with a CPDLCUserAbortReason [undefined] APDU element;

d) invoke D-ABORT request with:

1) the D-ABORT *Originator* parameter set to the abstract value “user”; and

2) the APDU as the D-ABORT *User Data* parameter value;

e) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

f) enter the *IDLE* state.

3.5.3.14 D-ABORT indication

3.5.3.14.1 Upon receipt of a D-ABORT indication, if the CPDLC-air-ASE is not in the *IDLE* state, and the D-ABORT *Originator* parameter is “user”, and the D-ABORT *User Data* parameter contains a GroundPDUs [CPDLCUserAbortReason] APDU, the CPDLC-air-ASE shall:

a) stop any timer;

b) if the CPDLC-air-user is an active user, invoke CPDLC-user-abort service indication with the APDU contained in the D-ABORT *User Data* parameter as the CPDLC-user-abort service *Reason* parameter value;

c) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

d) enter the *IDLE* state.

3.5.3.14.2 Upon receipt of a D-ABORT indication, if the CPDLC-air-ASE is not in the *IDLE* state, and the D-ABORT *Originator* parameter is "provider", and the D-ABORT *User Data* parameter is provided, and the D-ABORT *User Data* parameter contains a GroundPDUs [CPDLCProviderAbortReason] APDU, the CPDLC-air-ASE shall:

a) stop any timer;

b) if the CPDLC-air-user is an active user, invoke CPDLC-provider-abort service indication with the D-ABORT *User Data* parameter as the CPDLC-provider-abort service *Reason* parameter value, if provided;

c) if DSC has the abstract value "true", set DSC to the abstract value "false"; and

d) enter the *IDLE* state.

3.5.3.15 D-P-ABORT indication

Upon receipt of a D-P-ABORT indication, if the CPDLC-air-ASE is not in the *IDLE* state, the CPDLC-air-ASE shall:

a) stop any timer;

b) if the CPDLC-air-user is an active user, invoke CPDLC-provider-abort service indication with the CPDLC-provider-abort service *Reason* parameter set to the abstract value "communication-service-failure";

c) if DSC has the abstract value "true", set DSC to the abstract value "false"; and

d) enter the *IDLE* state.

3.5.4 CPDLC-air-ASE exception handling

3.5.4.1 A timer expires

If a CPDLC-air-ASE detects that a timer has expired, that CPDLC-air-ASE shall:

a) interrupt any current activity;

b) create an AircraftPDUs APDU with a CPDLCProviderAbortReason [timer-expired] APDU message element;

c) invoke D-ABORT request with:

1) the abstract value "provider" as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

d) if the CPDLC-air-user is an active user, invoke CPDLC-provider-abort service indication with the abstract value "timer-expired" as the CPDLC-provider-abort service *Reason* parameter value;

e) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

f) enter the *IDLE* state.

3.5.4.2 Unrecoverable system error

If a CPDLC-air-ASE has an unrecoverable system error, the CPDLC-air-ASE should:

a) stop any timer;

b) create an AircraftPDUs APDU with a CPDLCProviderAbortReason [undefined-error] APDU message element;

c) invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

d) if the CPDLC-air-user is an active user, invoke CPDLC-provider-abort service indication with the abstract value “undefined-error” as the CPDLC-provider-abort service Reason parameter value;

e) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

f) enter the *IDLE* state.

3.5.4.3 Invalid PDU

3.5.4.3.1 If the *User Data* parameter of a D-END confirmation with the *Result* parameter set to the abstract value “rejected”, or the *User Data* parameter of a D-START confirmation with the *Result* parameter set to the abstract value "accepted", or the *User Data* parameter of a D-START indication, a D-DATA indication or a D-END indication does not contain a valid PDU, the CPDLC-air-ASE shall:

a) stop any timer;

b) create an AircraftPDUs APDU with a CPDLCProviderAbortReason [invalid-PDU] APDU message element;

c) invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

d) if the CPDLC-air-user is an active user, invoke CPDLC-provider-abort service indication with the abstract value “invalid-PDU” as the CPDLC-provider-abort service *Reason* parameter value;

e) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

f) enter the *IDLE* state.

3.5.4.3.2 If the *User Data* parameter of a D-START confirmation with the *Result* parameter set to the abstract value "rejected (permanent)", or a D-END confirmation with the *Result* parameter set to the abstract value "accepted" is not a valid PDU, then the CPDLC-air-ASE shall:

a) stop any timer;

b) if the CPDLC-air-user is an active user, invoke CPDLC-provider-abort service indication with the CPDLC-provider-abort service *Reason* parameter set to the abstract value "invalid-PDU";

c) if DSC has the abstract value "true", set DSC to the abstract value "false"; and

d) enter the *IDLE* state.

3.5.4.4 Protocol error

3.5.4.4.1 If the *User Data* parameter of a D-START indication, D-START confirmation, D-DATA indication or D-END indication is a valid PDU; however, it is not a PDU for which action is described within a given state in 3.5.3, the CPDLC-air-ASE shall:

a) stop any timer;

b) create an AircraftPDUs APDU with a CPDLCProviderAbortReason [protocol-error] APDU message element;

c) invoke D-ABORT request with:

1) the abstract value "provider" as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

d) if the CPDLC-air-user is an active user, invoke CPDLC-provider-abort service indication with the abstract value "protocol-error" as the CPDLC-provider-abort service *Reason* parameter value;

e) if DSC has the abstract value "true", set DSC to the abstract value "false"; and

f) enter the *IDLE* state.

3.5.4.4.2 If the *User Data* parameter of a D-END confirmation is a valid PDU; however, it is not a permitted PDU as defined in 3.5.3, the CPDLC-air-ASE shall:

a) stop any timer;

b) if the D-END *Result* parameter is set to the abstract value "rejected", then:

1) create an AircraftPDUs APDU with a CPDLCProviderAbortReason [protocol-error] APDU message element; and

2) invoke D-ABORT request with:

i) the abstract value "provider" as the D-ABORT *Originator* parameter value; and

ii) the APDU as the D-ABORT *User Data* parameter value;

c) if the CPDLC-air-user is an active user, invoke CPDLC-provider-abort service indication with the abstract value “protocol-error” as the CPDLC-provider-abort service *Reason* parameter value;
d) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

e) enter the *IDLE* state.

3.5.4.4.3 Upon receipt of a DS primitive for which there are no instructions in 3.5.3 (i.e. the primitive was not expected or was expected under other conditions or with other parameter values), the CPDLC-air-ASE shall:

a) stop any timer;

b) create an AircraftPDUs APDU with a CPDLCProviderAbortReason [protocol-error] APDU message element;

c) if a dialogue exists, invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

d) if the CPDLC-air-user is an active user, invoke CPDLC-provider-abort service indication with the abstract value “protocol-error” as the CPDLC-provider-abort service *Reason* parameter value;

e) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

f) enter the *IDLE* state.

3.5.4.5 D-START confirmation Result parameter or Reject Source parameter not as expected

If a D-START confirmation *Result* parameter has the abstract value of “rejected (transient)” or if the *Reject Source* parameter has the abstract value of “DS-provider”, the CPDLC-air-ASE shall:

a) stop any timer;

b) if the CPDLC-air-user is an active user, invoke CPDLC-provider-abort service indication with the CPDLC-provider-abort service *Reason* parameter set to the abstract value “communication-service-error”;

c) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

d) enter the *IDLE* state.

3.5.4.6 D-START indication Quality of Service parameter not as expected

If a D-START indication *QOS Priority* parameter does not have the abstract value of “high priority flight safety messages”, or if the *QOS Residual Error Rate* parameter does not have the abstract value of “low”, or if the *QOS Routing Class* parameter does not have one of the abstract values specified in Table 3-13 (see 3.6.2.2), the CPDLC-air-

ASE shall:

a) stop any timer;

b) create an AircraftPDUs APDU with a CPDLCProviderAbortReason [invalid-QOS-parameter] APDU message element;

c) invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

d) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

e) enter the *IDLE* state.

3.5.4.7 Expected PDU missing

If the *User Data* parameter of a D-START indication or confirmation, a D-DATA indication, or a D-END indication or confirmation does not contain a PDU, the CPDLC-air-ASE shall:

a) stop any timer;

b) create an AircraftPDUs APDU with a CPDLCProviderAbortReason [expected-PDU-missing] APDU message element;

c) invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

d) if the CPDLC-air-user is an active user, invoke CPDLC-provider-abort service indication with the abstract value “not-permitted-PDU” as the CPDLC-provider-abort service *Reason* parameter value;

e) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

f) enter the *IDLE* state.

3.5.4.8 D-START Security Requirements parameter not as expected

Note.— This section applies to the case when the D-START Security Requirements *parameter does not meet the local security policy.*

Upon receipt of a D-START indication with the *Security Requirements* parameter not consistent with the local security policy, or upon receipt of a D-START confirmation with the *Security Requirements* parameter not equal to the value that was set in the D-START request, the CPDLC-air-ASE shall:

a) stop all timers;

b) if the dialogue with the peer is still open:

1) create an AircraftPDUs APDU with a CPDLCProviderAbortReason [communication-service-failure] APDU message element; and

2) invoke D-ABORT request with:

i) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

ii) the APDU as the D-ABORT *User Data* parameter value;

c) if the CPDLC-air-user is active, invoke CPDLC-provider-abort service indication with the abstract value “communication-service-failure” as the CDPLC-provider-abort service *Reason* parameter value;

d) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

e) enter the *IDLE* state.

3.5.5 CPDLC-ground-ASE protocol description

3.5.5.1 Introduction

3.5.5.1.1 If no actions are described for a CPDLC service primitive when a CPDLC-ground-ASE is in specific state, then the invocation of that service primitive shall be prohibited while the CPDLC-ground-ASE is in that state.

3.5.5.1.2 Upon receipt of a PDU, if no actions are described for the arrival of that PDU when a CPDLC-ground-ASE is in a specific state, then that PDU is considered not permitted and exception handling procedures as described in 3.5.6.4 shall apply.

3.5.5.1.3 If a PDU is received that cannot be decoded, then exception handling procedures as described in 3.5.6.3 for invalid PDU shall apply.

3.5.5.1.4 If a PDU is not received when one is required, then exception handling procedures as described in 3.5.6.3 shall apply.

3.5.5.1.5 The states defined for the CPDLC-ground-ASE are the following:

a) IDLE;

b) START-REQ;

c) START-IND;

d) DIALOGUE;

e) END; and

f) FORWARD.

3.5.5.1.6 The CPDLC-ground-user is an active user from the time:

a) the CPDLC-ground-user invokes the CPDLC-start service request until it:

1) receives a CPDLC-start service confirmation with the Result parameter equal to the abstract value “rejected”; or
2) receives a CPDLC-end service confirmation with the Result parameter equal to the abstract value “accepted”; or

3) invokes a CPDLC-user-abort service request; or

4) receives a CPDLC-user-abort service indication; or

5) receives a CPDLC-provider-abort service indication; or

b) the CPDLC-ground-user receives the CPDLC-start service indication until it:

1) invokes a CPDLC-start service response with the Result parameter set to the abstract value “rejected”; or

2) receives a CPDLC-end service confirmation with the Result parameter equal to the abstract value “accepted”; or

3) invokes a CPDLC-user-abort service request; or

4) receives a CPDLC-user-abort service indication; or

5) receives a CPDLC-provider-abort service indication; or

c) the CPDLC-ground-user receives the DSC-start service indication until it:

1) invokes a DSC-start service response with the Result parameter equal to the abstract value “rejected”; or

2) invokes a CPDLC-user-abort service request; or

3) receives a CPDLC-user-abort service indication; or

4) receives a CPDLC-provider-abort service indication; or

d) the CPDLC-ground-user invokes the CPDLC-forward service request until it:

1) receives a CPDLC-forward service confirmation; or

2) invokes a CPDLC-user-abort service request; or

3) receives a CPDLC-user-abort service indication; or

4) receives a CPDLC-provider-abort service indication.

3.5.5.1.7 On initiation, the CPDLC-ground-ASE shall be in the *IDLE* state.

3.5.5.1.8 The CPDLC-ground-ASE contains a Boolean called DSC. DSC has the abstract value “true” when the dialogue is a DSC dialogue and has the abstract value “false” otherwise.

3.5.5.1.9 On the initiation of a CPDLC-ground-ASE, DSC shall be set to the abstract value “false”.

3.5.5.2 D-START indication

3.5.5.2.1 Upon receipt of a D-START indication, if the CPDLC-ground-ASE is in the *IDLE* state, and the abstract value of the D-START *Calling Peer ID* parameter is the ICAO 24-bit aircraft address, and the D-START *User Data* parameter contains an AircraftPDUs [StartDownMessage] APDU with the APDU element mode “cpdlc”, and the D-START *QOS Priority* parameter has the abstract value “high priority flight safety messages” and the D-START *QOS Residual Error Rate* parameter has the abstract value “low”, and the D-START *QOS Routing Class* parameter has one of the abstract values specified in Table 3-13 (see 3.6.2.2), and the D-START *Security Requirements* parameter is consistent with the local security policy, the CPDLC-ground-ASE shall:

a) invoke CPDLC-start service indication with:

1) the D-START *Calling Peer ID* parameter value as the CPDLC-start service *Calling Peer Identifier* parameter value;

2) the D-START *QOS Routing Class* parameter value as the CPDLC-start service *Class of Communication* parameter value;

3) if the D-START *DS-User Version Number* parameter is present, the D-START *DS-User Version Number* parameter is present as the *CPDLC Message Set Version Number* parameter value; and

4) the AircraftPDUs APDU element as the CPDLC-start service *CPDLC/IC Data* parameter value; and

b) enter the *START-IND* state.

3.5.5.2.2 Upon receipt of a D-START indication, if the CPDLC-ground-ASE is in the *IDLE* state, and the abstract value of the D-START *Calling Peer ID* parameter is the ICAO 24-bit aircraft address, and the D-START *User Data* parameter contains an AircraftPDUs [StartDownMessage] APDU with the APDU element mode “dsc”, and the D-START *QOS Priority* parameter has the abstract value “high priority flight safety messages”, and the D-START *QOS Residual Error Rate* parameter has the abstract value “low”, and the D-START *QOS Routing Class* parameter has one of the abstract values specified in Table 3-13 (see 3.6.2.2), and the D-START *Security Requirements* parameter is consistent with the local security policy, the CPDLC-ground-ASE shall:

a) invoke DSC-start service indication with:

1) the D-START *Calling Peer ID* parameter value as the DSC-start service *Aircraft Address* parameter value;

2) the D-START *QOS Routing Class* parameter value as the CPDLC-start service *Class of Communication* parameter value;

3) if the D-START *DS-User Version Number* parameter is present, the D-START *DS-User Version Number* parameter is present as the *CPDLC Message Set Version Number* parameter value; and

4) the AircraftPDUs APDU as the DSC-start service *CPDLC/IC Data* parameter value;

b) set DSC to “true”; and

c) enter the *START-IND* state.

3.5.5.2.3 Upon receipt of a D-START indication, if the CPDLC-ground-ASE is in the *IDLE* state, and the abstract value of the D-START *Calling Peer ID* parameter is a Facility Designation, and the D-START *User Data* parameter contains a GroundPDUs [ATCForwardMessage] APDU, and the CPDLC-ground-ASE supports the CPDLC-forward service, and the D-START *QOS Priority* parameter has the abstract value “high priority flight safety messages”, and the D-START *QOS Residual Error Rate* parameter has the abstract value “low”, the CPDLC-ground-ASE shall:

a) if the D-START *DS-User Version Number* parameter value is equal to the CPDLC-ground-ASE version number and the D-START *Security Requirements* parameter is consistent with the local security policy:

1) invoke CPDLC-forward service indication with:

i) the D-START *Calling Peer ID* parameter value as the CPDLC-forward service *Calling Facility Designation* parameter value; and

ii) the D-START GroundPDUs APDU element as the CPDLC-forward service *CPDLC Message* parameter value;

2) create a GroundPDUs APDU with an ATCForwardResponse [success] APDU element;

3) invoke D-START response with:

i) the APDU as the D-START *User Data* parameter value; and

ii) the abstract value “rejected (permanent)” as the D-START *Result* parameter value; and

4) remain in the *IDLE* state; and

b) if the D-START *DS-User Version Number* parameter value is not equal to the CPDLC-ground-ASE version number:

1) create a GroundPDUs APDU with an ATCForwardResponse [version-not-equal] APDU element;

2) invoke D-START response with:

i) the CPDLC-ground-ASE version number as the D-START *DS-User Version Number* parameter value;

ii) the APDU as the D-START *User Data* parameter value; and

iii) the abstract value “rejected (permanent)” as the D-START *Result* parameter value; and

3) remain in the *IDLE* state.

3.5.5.2.4 Upon receipt of a D-START indication, if the CPDLC-ground-ASE is in the *IDLE* state, and the abstract value of the D-START *Calling Peer ID* parameter is a Facility Designation, and the D-START *User Data* parameter contains a GroundPDUs APDU, and the APDU element is an ATCForwardMessage, and the CPDLC-ground-ASE does not support the CPDLC-forward service, and the D-START *Security Requirements* parameter is consistent with the local

security policy, the CPDLC-ground-ASE shall:

a) create a GroundPDUs APDU with an ATCForwardResponse [service-not-supported] APDU element;

b) invoke D-START response with:

1) the APDU as the D-START *User Data* parameter value; and

2) the abstract value “rejected (permanent)” as the D-START *Result* parameter value; and

c) remain in the *IDLE* state.

3.5.5.3 D-START confirmation

3.5.5.3.1 Upon receipt of a D-START confirmation, if the CPDLC-ground-ASE is in the *START-REQ* state, and the D-START Result parameter has the abstract value “accepted”, and DSC has the abstract value of “false”, and D-START *User Data* parameter contains an AircraftPDUs [ICDownlinkMessage] APDU, and the D-START *Security Requirements* parameter value is the same as the one set for the D-START request, the CPDLC-ground-ASE shall:

a) stop timer t_{start};

b) invoke CPDLC-start service confirmation with:

1) the APDU contained in the D-START *User Data* parameter as the CPDLC-start service *Response CPDLC/IC Data* parameter value; and

2) the abstract value “accepted” as the CPDLC-start service *Result* parameter value; and

c) enter the *DIALOGUE* state.

3.5.5.3.2 Upon receipt of a D-START confirmation, if the CPDLC-ground-ASE is in the *START-REQ* state, and the D-START *Result* parameter has the abstract value “rejected (permanent)”, and the D-START *Reject Source* parameter has the abstract value “DS-user”, and DSC has the abstract value “false”, and the D-START *User Data* parameter contains an AircraftPDUs [ICDownlinkMessage] APDU, and the D-START *Security Requirements* parameter value is the same as the one set for the D-START request, the CPDLC-ground-ASE shall:

a) stop timer t_{start};

b) invoke CPDLC-start service confirmation with:

1) the APDU contained in the D-START *User Data* parameter as the CPDLC-start service *Response CPDLC/IC Data* parameter value; and

2) the abstract value “rejected” as the CPDLC-start service *Result* parameter value; and

c) enter the *IDLE* state.

3.5.5.3.3 Upon receipt of a D-START confirmation, if the CPDLC-ground-ASE is in the *FORWARD* state, and the D-START *Result* parameter has the abstract value “rejected (permanent)”, and the *Reject Source* parameter has the abstract value “DS-user”, and the D-START *User Data* parameter contains a GroundPDUs [ATCForwardResponse] APDU, and the D-START *Security Requirements* parameter value is the same as the one set for the D-START request, the CPDLC-ground-ASE shall:

a) stop timer t_{start};

b) invoke CPDLC-forward service confirmation with:

1) the D-START GroundPDUs APDU element as the CPDLC-forward service *Result* parameter value; and

2) if provided, the D-START *DS-User Version Number* parameter value as the CPDLC-forward service *ASE Version Number* parameter value; and

c) enter the *IDLE* state.

3.5.5.4 D-DATA indication

3.5.5.4.1 Upon receipt of a D-DATA indication, if the CPDLC-ground-ASE is in the *DIALOGUE* state, and the APDU contained in the D-DATA *User Data* parameter is an AircraftPDUs [ICDownlinkMessage] APDU, the CPDLC-ground-ASE shall:

a) invoke CPDLC-message service indication with the APDU contained in the D-DATA *User Data* parameter as the CPDLC-message service *CPDLC/IC Data* parameter value; and

b) remain in the *DIALOGUE* state.

3.5.5.4.2 Upon receipt of a D-DATA indication, if the CPDLC-ground-ASE is in the *END* state, and DSC has the abstract value “false”, and the APDU contained in the D-DATA *User Data* parameter is an AircraftPDUs [ICDownlinkMessage] APDU, the CPDLC-ground-ASE shall:

a) invoke CPDLC-message service indication with the APDU contained in the D-DATA *User Data* parameter as the CPDLC-message service *CPDLC/IC Data* parameter value; and

b) remain in the *END* state.

3.5.5.5 D-END indication

Upon receipt of a D-END indication, if the CPDLC-ground-ASE is in the *DIALOGUE* state, and DSC has the abstract value “true”, and the D-END *User Data* parameter contains an AircraftPDUs [ICDownlinkMessage] APDU, the CPDLC-ground-ASE shall:

a) invoke DSC-end service indication with the APDU contained in the D-END *User Data* parameter as the DSC-end service *CPDLC/IC Data* parameter value; and

b) enter the *END* state.

3.5.5.6 D-END confirmation

3.5.5.6.1 Upon receipt of a D-END confirmation, if the CPDLC-ground-ASE is in the *END* state, and the D-END *Result* parameter has the abstract value “accepted”, and DSC has the abstract value “false”, and the D-END *User Data* parameter contains an AircraftPDUs [ICDownlinkMessage] APDU, the CPDLC-ground-ASE shall:

a) invoke CPDLC-end service confirmation with:

1) the APDU contained in the D-END *User Data* parameter as the CPDLC-end service *CPDLC/IC Data* parameter value; and

2) the abstract value “accepted” as the CPDLC-end service *Result* parameter value; and

b) enter the *IDLE* state.

3.5.5.6.2 Upon receipt of a D-END confirmation, if the CPDLC-ground-ASE is in the *END* state, and the D-END *Result* parameter has the abstract value “rejected”, and DSC has the abstract value “false”, and the D-END *User Data* parameter contains an AircraftPDUs [ICDownlinkMessage] APDU, the CPDLC-ground-ASE shall:

a) invoke CPDLC-end service confirmation with:

1) the APDU contained in the D-END *User Data* parameter as the CPDLC-end service *CPDLC/IC Data* parameter value; and

2) the abstract value “rejected” as the CPDLC-end service *Result* parameter value; and

b) enter the *DIALOGUE* state.

3.5.5.7 CPDLC-start service request

Upon receipt of a CPDLC-start service request, if the CPDLC-ground-ASE is in the *IDLE* state, the CPDLC-ground-ASE shall:

a) create a GroundPDUs APDU with an UplinkMessage(startup) APDU element containing the *CPDLC/IC Data* parameter as the UplinkMessage;

b) invoke D-START request with:

1) the CPDLC-start service *Called Peer Identifier* parameter value as the D-START *Called Peer ID* parameter value;

2) the CPDLC-start service *Calling Peer Identifier* parameter value as the D-START *Calling Peer ID* parameter value;

3) the CPDLC-start service *CPDLC Message Set Version Number* parameter value if specified by the CPDLC-user as the D-START *DS-User Version Number* parameter value;

4) the CPDLC-start service *Security Required* parameter value if specified by the CPDLC-user or otherwise the abstract value “no security” as the D-START *Security Requirements* parameter value;

5) the D-START *Quality of Service* parameters set as follows:

i) if provided, the CPDLC-start service *Class of Communication* parameter value as the D-START *QOS Routing Class* parameter value; and

ii) the abstract value of “high priority flight safety messages” as the D-START *QOS Priority* parameter value; and

iii) the abstract value of “low” as the D-START *QOS Residual Error Rate* parameter value; and

6) the APDU as the D-START *User Data* parameter value;

c) start timer t_{start}; and

d) enter the *START-REQ* state.

3.5.5.8 CPDLC-start service response

3.5.5.8.1 Upon receipt of a CPDLC-start service response, if the CPDLC-ground-ASE is in the *START-IND* state, and the CPDLC-start service *Result* parameter has the abstract value “accepted”, and DSC has the abstract value “false”, the CPDLC-ground-ASE shall:

a) create a GroundPDUs APDU with an ICUplinkMessage(send) APDU element based on the *Response CPDLC/IC Data* parameter;

b) invoke D-START response with:

1) the APDU as the D-START *User Data* parameter value;

2) the abstract value received in the D-START indication *Security Requirements* parameter as the D-START *Security Requirements* parameter value; and

3) the abstract value “accepted” as the D-START *Result* parameter value; and

c) enter the *DIALOGUE* state.

3.5.5.8.2 Upon receipt of a CPDLC-start service response, if the CPDLC-ground-ASE is in the *START-IND* state, and the CPDLC-start service *Result* parameter has the abstract value “rejected”, and DSC has the abstract value “false”, the CPDLC-ground-ASE shall:

a) create a GroundPDUs APDU with an ICUplinkMessage(send) APDU element based on the *Response CPDLC/IC Data* parameter;

b) invoke D-START response with:

1) the APDU element as the D-START *User Data* parameter value;

2) the abstract value received in the D-START indication *Security Requirements* parameter as the D-START *Security Requirements* parameter value; and

3) the abstract value “rejected (permanent)” as the D-START *Result* parameter value; and

c) enter the *IDLE* state.

3.5.5.9 DSC-start service response

3.5.5.9.1 Upon receipt of a DSC-start service response, if the CPDLC-ground-ASE is in the *START-IND* state, and the DSC-start service *Result* parameter has the abstract value “accepted”, and DSC has the abstract value “true”, the CPDLC-ground-ASE shall:

a) create a GroundPDUs APDU with an ICUplinkMessage(send) APDU element based on the *Response CPDLC/IC Data* parameter;

b) invoke D-START response with:

1) the APDU element as the D-START *User Data* parameter value;

2) the abstract value received in the D-START indication *Security Requirements* parameter as the D-START *Security Requirements* parameter value; and

3) the abstract value “accepted” as the D-START *Result* parameter value; and

c) enter the *DIALOGUE* state.

3.5.5.9.2 Upon receipt of a DSC-start service response, if the CPDLC-ground-ASE is in the *START-IND* state, and the DSC-start service *Result* parameter has the abstract value “rejected”, and DSC has the abstract value “true”, the CPDLC-ground-ASE shall:

a) create a GroundPDUs APDU with an ICUplinkMessage(send) APDU element based on the *Response CPDLC/IC Data* parameter;

b) invoke D-START response with:

1) the APDU element as the D-START *User Data* parameter value;

2) the abstract value received in the D-START indication *Security Requirements* parameter as the D-START *Security Requirements* parameter value; and

3) the abstract value “rejected (permanent)” as the D-START *Result* parameter value;

c) set DSC to the abstract value “false”; and

d) enter the *IDLE* state.

3.5.5.10 CPDLC-message service request

3.5.5.10.1 Upon receipt of a CPDLC-message service request and the *CPDLC/IC Data* parameter contains a CPDLC message, if the CPDLC-ground-ASE is in the *DIALOGUE* state, the CPDLC-ground-ASE shall:

a) create a GroundPDUs APDU with an ICUplinkMessage(send) APDU element based on the CPDLC-message service *CPDLC/IC Data* parameter;

b) invoke D-DATA request with the APDU as the D-DATA *User Data* parameter value; and

c) remain in the *DIALOGUE* state.

3.5.5.10.2 Upon receipt of a CPDLC-message service request and the *CPDLC/IC Data* parameter contains a CPDLC message, if the CPDLC-ground-ASE is in the *END* state and DSC has the abstract value “true”, the CPDLC-ground-ASE

shall:

a) create a GroundPDUs APDU with an ICUplinkMessage(send) APDU element based on the CPDLC-message service *CPDLC/IC Data* parameter;

b) invoke D-DATA request with the APDU as the D-DATA *User Data* parameter value; and

c) remain in the *END* state.

3.5.5.11 CPDLC-end service request

Upon receipt of a CPDLC-end service request, if the CPDLC-ground-ASE is in the *DIALOGUE* state, and DSC has the abstract value “false”, the CPDLC-ground-ASE shall:

a) create a GroundPDUs APDU with an ICUplinkMessage(send) APDU element based on the CPDLC-end service *CPDLC/IC Data* parameter;
b) invoke D-END request with the APDU as the D-END *User Data* parameter value; and

c) enter the *END* state.

3.5.5.12 DSC-end service response

3.5.5.12.1 Upon receipt of a DSC-end service response, if the CPDLC-ground-ASE is in the *END* state, and DSC has the abstract value “true”, and the DSC-end service *Result* parameter has the abstract value “accepted”, the CPDLC-ground-ASE shall:

a) create a GroundPDUs APDU with an ICUplinkMessage(send) APDU element based on the DSC-end service *CPDLC/IC Data* parameter;

b) invoke D-END response with:

1) the APDU as the D-END *User Data* parameter value; and

2) the abstract value “accepted” as the D-END *Result* parameter value;

c) set DSC to the abstract value “false”; and

d) enter the *IDLE* state.

3.5.5.12.2 Upon receipt of a DSC-end service response, if the CPDLC-ground-ASE is in the *END* state, and DSC has the abstract value “true”, and the DSC-end service *Result* parameter has the abstract value “rejected”, the CPDLC-ground-ASE shall:

a) create a GroundPDUs APDU with an ICUplinkMessage(send) APDU element based on the DSC-end service *CPDLC/IC Data* parameter;

b) invoke D-END response with:

1) the APDU as the D-END *User Data* parameter value; and

2) the abstract value “rejected” as the D-END *Result* parameter value; and

c) enter the *DIALOGUE* state.

3.5.5.13 CPDLC-forward service request

Upon receipt of a CPDLC-forward service request, if the CPDLC-ground-ASE is in the *IDLE* state, the CPDLC-ground-ASE shall:

a) create a GroundPDUs APDU with an ATCForwardMessage APDU element based on the CPDLC-forward service *CPDLC Message* parameter;

b) invoke D-START request with:

1) the CPDLC-forward service *Called Facility Designation* parameter value as the D-START *Called Peer ID* parameter value;
2) the CPDLC-start service *Calling Facility Designation* parameter value as the D-START *Calling Peer ID* parameter value;

3) the CPDLC-forward service *Security Required* parameter value if specified by the CPDLC-user or otherwise the abstract value “no security” as the D-START *Security Requirements* parameter value;

4) the D-START *Quality of Service* parameters set as follows:

i) if provided, the CPDLC-forward service *Class of Communication* parameter value as the D-START *QOS Routing Class* parameter value;

ii) the abstract value of “high priority flight safety messages” as the D-START *QOS Priority* parameter value; and

iii) the abstract value of “low” as the D-START *QOS Residual Error Rate* parameter value; and

5) the APDU as the D-START *User Data* parameter value;

c) start timer t_{start}; and

d) enter the *FORWARD* state.

3.5.5.14 CPDLC-user-abort service request

Upon receipt of a CPDLC-user-abort service request, if the CPDLC-ground-ASE is not in the *IDLE* state, the CPDLC-ground-ASE shall:

a) stop any timer;

b) if the CPDLC-user-abort service *Reason* parameter is provided, create a GroundPDUs APDU with a CPDLCUserAbortReason APDU element based on the CPDLC-user-abort service *Reason* parameter;

c) if the CPDLC-user-abort service *Reason* parameter is not provided, create a GroundPDUs APDU with

a CPDLCUserAbortReason [undefined] APDU element;

d) invoke D-ABORT request with:

1) the D-ABORT *Originator* parameter set to the abstract value “user”; and

2) the APDU as the D-ABORT *User Data* parameter value;

e) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

f) enter the *IDLE* state.

3.5.5.15 D-ABORT indication

3.5.5.15.1 Upon receipt of a D-ABORT indication, if the CPDLC-ground-ASE is not in the *IDLE* state, and the D-ABORT *Originator* parameter is “user”, and the D-ABORT *User Data* parameter contains an AircraftPDUs [CPDLCUserAbortReason] APDU, the CPDLC-ground-ASE shall:

a) stop any timer;

b) if the CPDLC-ground-user is an active user, invoke CPDLC-user-abort service indication with the APDU contained in the D-ABORT *User Data* parameter as the CPDLC-user-abort service *Reason* parameter value;

c) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

d) enter the *IDLE* state.

3.5.5.15.2 Upon receipt of a D-ABORT indication, if the CPDLC-ground-ASE is not in the *IDLE* state, and the D-ABORT Originator parameter is “provider”, and the D-ABORT *User Data* parameter is provided, and the D-ABORT *User Data* parameter contains either an AircraftPDUs [CPDLCProviderAbortReason] APDU or a GroundPDUs [CPDLCProviderAbortReason] APDU, the CPDLC-ground-ASE shall:

a) stop any timer;

b) if the CPDLC-ground-user is an active user, invoke CPDLC-provider-abort service indication with the D-ABORT *User Data* parameter as the CPDLC-provider-abort service *Reason* parameter value, if provided;

c) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

d) enter the *IDLE* state.

3.5.5.16 D-P-ABORT indication

Upon receipt of a D-P-ABORT indication, if the CPDLC-ground-ASE is not in the *IDLE* state, the CPDLC-ground-ASE shall:

a) stop any timer;

b) if the CPDLC-ground-user is an active user, invoke CPDLC-provider-abort service indication with the

CPDLC-provider-abort service *Reason* parameter set to the abstract value “communication-service-failure”;

c) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

d) enter the *IDLE* state.

3.5.6 CPDLC-ground-ASE exception handling

3.5.6.1 A timer expires

If a CPDLC-ground-ASE detects that a timer has expired, that CPDLC-ground-ASE shall:

a) interrupt any current activity;

b) create a GroundPDUs APDU with a CPDLCProviderAbortReason [timer-expired] APDU message element;

c) invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

d) if the CPDLC-ground-user is an active user, invoke CPDLC-provider-abort service indication with the abstract value “timer-expired” as the CPDLC-provider-abort service *Reason* parameter value”;

e) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

f) enter the *IDLE* state.

3.5.6.2 Unrecoverable system error

If a CPDLC-ground-ASE has an unrecoverable system error, the CPDLC-ground-ASE should:

a) stop any timer;

b) create a GroundPDUs APDU with a CPDLCProviderAbortReason [undefined-error] APDU message element;

c) invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT Originator parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

d) if the CPDLC-ground-user is an active user, invoke CPDLC-provider-abort service indication with the abstract value “undefined-error” as the CPDLC-provider-abort service Reason parameter value”;

e) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

f) enter the *IDLE* state.

3.5.6.3 Invalid PDU

3.5.6.3.1 If the *User Data* parameter of a D-END confirmation with the *Result* parameter set to the abstract value “rejected”, or the *User* Data parameter of a D-START confirmation with the *Result* parameter set to the abstract value "accepted", or the *User Data* parameter of a D-START indication, a D-DATA indication or a D-END indication does not contain a valid PDU, the CPDLC-ground-ASE shall:

a) stop any timer;

b) create a GroundPDUs APDU with a CPDLCProviderAbortReason [invalid-PDU] APDU message element;

c) invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

d) if the CPDLC-ground-user is an active user, invoke CPDLC-provider-abort service indication with the abstract value “invalid-PDU” as the CPDLC-provider-abort service *Reason* parameter value;

e) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

f) enter the *IDLE* state.

3.5.6.3.2 If the *User Data* parameter of a D-START confirmation with the *Result* parameter set to the abstract value “rejected (permanent)”, or a D-END confirmation with the *Result* parameter set to the abstract value “accepted” is not a valid PDU, the CPDLC-ground-ASE shall:

a) stop any timer;

b) if the CPDLC-ground-user is an active user, invoke CPDLC-provider-abort service indication with the CPDLC-provider-abort service *Reason* parameter set to the abstract value “invalid-PDU”;

c) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

d) enter the *IDLE* state.

3.5.6.4 Protocol error

3.5.6.4.1 If the *User Data* parameter of a D-START indication, a D-START confirmation, a D-DATA indication or a D-END indication is a valid PDU; however, it is not a PDU for which action is described within a given state as defined in 3.5.5, the CPDLC-ground-ASE shall:

a) stop any timer;

b) create a GroundPDUs APDU with a CPDLCProviderAbortReason [protocol-error] APDU message element;

c) invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

d) if the CPDLC-ground-user is an active user, invoke CPDLC-provider-abort service indication with the abstract value “protocol-error” as the CPDLC-provider-abort service *Reason* parameter value;

e) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

f) enter the *IDLE* state.

3.5.6.4.2 If the *User Data* parameter of a D-END confirmation is a valid PDU; however, it is not a permitted PDU for which action is described within a given state as defined in 3.5.5, the CPDLC-ground-ASE shall:

a) stop any timer;

b) if the D-END *Result* parameter is set to the abstract value “rejected”, then:

1) create a GroundPDUs APDU with a CPDLCProviderAbortReason [protocol-error] APDU message element; and

2) invoke D-ABORT request with:

i) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

ii) the APDU as the D-ABORT *User Data* parameter value;

c) if the CPDLC-ground-user is an active user, invoke CPDLC-provider-abort service indication with the abstract value “protocol-error” as the CPDLC-provider-abort service *Reason* parameter value;

d) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

e) enter the *IDLE* state.

3.5.6.4.3 Upon receipt of a DS primitive for which there are no instructions in 3.5.3 (i.e. the primitive was not expected or was expected under other conditions or with other parameter values), the CPDLC-ground-ASE shall:

a) stop any timer;

b) create a GroundPDUs APDU with a CPDLCProviderAbortReason [protocol-error] APDU message element;

c) if a dialogue exists, invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

d) if the CPDLC-ground-user is an active user, invoke CPDLC-provider-abort service indication with the abstract value “protocol-error” as the CPDLC-provider-abort service *Reason* parameter value;

e) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

f) enter the *IDLE* state.

3.5.6.5 D-START confirmation Result parameter or Reject Source parameter not as expected

If a D-START confirmation *Result* parameter has the abstract value of “rejected (transient)”, or if the *Reject Source* parameter has the abstract value of “DS-provider”, the CPDLC-ground-ASE shall:

a) stop any timer;

b) if the CPDLC-ground-user is an active user, invoke CPDLC-provider-abort service indication with the CPDLC-provider-abort service *Reason* parameter set to the abstract value “communication-service-error”;

c) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

d) enter the *IDLE* state.

3.5.6.6 D-START indication Quality of Service parameter not as expected

If a D-START indication *QOS Priority* parameter does not have the abstract value of “high priority flight safety messages”, or if the *QOS Residual Error Rate* parameter does not have the abstract value of “low”, or if the *QOS Routing Class* parameter does not have one of the abstract values specified in Table 3-13 (see 3.6.2.2), the CPDLC-ground-ASE shall:

a) stop any timer;

b) create a GroundPDUs APDU with a CPDLCProviderAbortReason [invalid-QOS-parameter] APDU message element;

c) invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

d) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

e) enter the *IDLE* state.

3.5.6.7 Expected PDU missing

If the *User Data* parameter of a D-START indication or confirmation, a D-DATA indication, or a D-END indication or confirmation does not contain a PDU, the CPDLC-ground-ASE shall:

a) stop any timer;

b) create a GroundPDUs APDU with a CPDLCProviderAbortReason [expected-PDU-missing] APDU message element;

c) invoke D-ABORT request with:

1) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

2) the APDU as the D-ABORT *User Data* parameter value;

d) if the CPDLC-ground-user is an active user, invoke CPDLC-provider-abort service indication with the abstract value “not-permitted-PDU” as the CPDLC-provider-abort service *Reason* parameter value;

e) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

f) enter the *IDLE* state.

3.5.6.8 D-START Security Requirements parameter not as expected

Note.— This section applies to the case when the D-START Security Requirements *parameter does not meet the local security policy.*

Upon receipt of a D-START indication with the *Security Requirements* parameter not consistent with the local security policy, or upon receipt of a D-START confirmation with the *Security Requirements* parameter not equal to the value that was set in the D-START request, the CPDLC-ground-ASE shall:

a) stop all timers;

b) if the dialogue with the peer is still open:

1) create a GroundPDUs APDU with a CPDLCProviderAbortReason [communication-service-failure] APDU message element; and

2) invoke D-ABORT request with:

i) the abstract value “provider” as the D-ABORT *Originator* parameter value; and

ii) the APDU as the D-ABORT *User Data* parameter value;

c) if the CPDLC-ground-user is active, invoke CPDLC-provider-abort service indication with the abstract value “communication-service-failure” as the CDPLC-provider-abort service *Reason* parameter value;

d) if DSC has the abstract value “true”, set DSC to the abstract value “false”; and

e) enter the *IDLE* state.

3.5.7 CPDLC-ASE state tables

3.5.7.1 Priority

3.5.7.1.1 If the state tables for the CPDLC-air-ASE and the CPDLC-ground-ASE at Tables 3-11 and 3-12, respectively, conflict with textual statements made elsewhere in this manual, the textual statements shall take precedence.

3.5.7.1.2 In the state tables at Tables 3-11 and 3-12, the statement “cannot occur” means that if the implementation conforms to the SARPs, it is impossible for this event to occur. If the event does occur, this implies that there is an error in the implementation. If such a situation is detected, it is suggested that the ASE abort with the error “unrecoverable system error”.

3.5.7.1.3 In the state tables at Tables 3-11 and 3-12, the statement “not permitted” means that the implementation must prevent this event from occurring through some local means. If the event does occur, this implies that there is an error in the implementation. If such a situation is detected, it is suggested that the ASE perform a local rejection of the request rather than aborting the dialogue.

Table 3-11. CPDLC-air-ASE state table

STATE → *EVENT ↓*	*IDLE*	*START-REQ*	*START-IND*	*DIALOGUE*	*END*
Dialogue Service Events					
D-START Indication APDU = ICUplinkMessage (startup)	● CPDLC-start indication →*START-IND*	cannot occur	cannot occur	cannot occur	cannot occur
D-START Confirmation *Result* "accepted", DSC = "false" APDU = ICUplinkMessage (send)	cannot occur	●Stop timer t_{start} ● CPDLC-start confirmation →*DIALOGUE*	cannot occur	cannot occur	cannot occur
D-START Confirmation *Result* "rejected (permanent)" and Reject Source "DS-user", DSC = "false" APDU = ICUplinkMessage (send)	cannot occur	●Stop timer t_{start} ● CPDLC-start confirmation →*IDLE*	cannot occur	cannot occur	cannot occur
D-START Confirmation *Result* "accepted", DSC = "true" APDU = ICUplinkMessage (send)	cannot occur	●Stop timer t_{start} ● DSC-start confirmation, →*DIALOGUE*	cannot occur	cannot occur	cannot occur
D-START Confirmation *Result* "rejected (permanent)" and Reject Source "DS-user", DSC = "true" APDU = ICUplinkMessage (send)	cannot occur	●Stop timer t_{start} ● DSC-start confirmation ●Set DSC = "false" →*IDLE*	cannot occur	cannot occur	cannot occur
D-DATA Indication APDU = ICUplinkMessage (send)	cannot occur	cannot occur	cannot occur	●*CPDLC*-message indication →*DIALOGUE*	●if DSC = "true" ● CPDLC-message indication →*END* ●else cannot occur
D-END Indication DSC = "false" APDU = ICUplinkMessage (send)	cannot occur	cannot occur	cannot occur	● CPDLC-end indication →*END*	cannot occur
D-END Confirmation DSC = "true" *Result* "accepted" APDU = ICUplinkMessage (send)	cannot occur	cannot occur	cannot occur	cannot occur	● DSC-end confirmation ●Set DSC "false" →*IDLE*
D-END Confirmation DSC = "true" *Result* "rejected" APDU = ICUplinkMessage (send)	cannot occur	cannot occur	cannot occur	cannot occur	● DSC-end confirmation →*DIALOGUE*

STATE → *EVENT ↓*	*IDLE*	*START-REQ*	*START-IND*	*DIALOGUE*	*END*
CPDLC-User Events					
CPDLC-start Request	●D-START request ●Start timer t_{start} →*START-REQ*	not permitted	not permitted	not permitted	not permitted
CPDLC-start Response DSC = "false" *Result* "accepted"	not permitted	not permitted	●D-START response →*DIALOGUE*	not permitted	not permitted
CPDLC-start Response DSC = "false" *Result* "rejected"	not permitted	not permitted	●D-START response →*IDLE*	not permitted	not permitted
DSC-start Request	●D-START request ●set DSC = "true" ●Start timer t_{start} →*START-REQ*	not permitted	not permitted	not permitted	not permitted
CPDLC-message Request	not permitted	not permitted	not permitted	●D-DATA request →*DIALOGUE*	●if DSC = "false" ●D-DATA request →*END* ●else not permitted
CPDLC-end Service Response DSC = "false" *Result* "accepted"	cannot occur	cannot occur	cannot occur	not permitted	●D-END response →*IDLE*
CPDLC-end Service Response DSC = "false" *Result* "rejected"	cannot occur	cannot occur	cannot occur	not permitted	●D-END response →*DIALOGUE*
DSC-end Request DSC = "true"	not permitted	not permitted	not permitted	●D-END request →*END*	not permitted
ABORT Events					
CPDLC-user-abort Request	not permitted	●Stop timer t_{start}, if set ●D-ABORT request ●If DSC = "true", set DSC = "false" →*IDLE*	●Stop timer t_{start}, if set ●D-ABORT request ●If DSC = "true", set DSC = "false" →*IDLE*	●D-ABORT request ●If DSC = "true", set DSC = "false" →*IDLE*	●D-ABORT request ●If DSC = "true", set DSC = "false" →*IDLE*
D-ABORT Indication *Originator* "user"	cannot occur	●Stop timer T_{start}, if set ●If active user: CPDLC-user-abort indication ●If DSC = "true", set DSC = "false" →*IDLE*	●Stop timer T_{start}, if set ●If active user: CPDLC-user-abort indication ●If DSC = "true", set DSC = "false" →*IDLE*	●If active user: CPDLC-user-abort indication ●If DSC = "true", set DSC = "false" →*IDLE*	●If active user: CPDLC-user-abort indication ●If DSC = "true", set DSC = "false" →*IDLE*

STATE → *EVENT ↓*	*IDLE*	*START-REQ*	*START-IND*	*DIALOGUE*	*END*
D-ABORT Indication *Originator* "provider"	cannot occur	●Stop timer T_{start}, if set ●If active user: CPDLC-provider-abort indication ●If DSC = "true", set DSC "false" →*IDLE*	●Stop timer T_{start}, if set ●If active user: CPDLC-provider-abort indication ●If DSC = "true", set DSC "false" →*IDLE*	●If active user: CPDLC-provider-abort indication ●If DSC = "true", set DSC "false" →*IDLE*	●If active user: CPDLC-provider-abort indication ●If DSC = "true", set DSC "false" →*IDLE*
D-P-ABORT Indication	cannot occur	●Stop timer t_{start}, if set ●If active user: CPDLC-provider-abort indication ●If DSC = "true", set DSC = "false" →*IDLE*	●Stop timer t_{start}, if set ●If active user: CPDLC-provider-abort indication ●If DSC = "true", set DSC = "false" →*IDLE*	●If active user: CPDLC-provider-abort indication ●If DSC = "true", set DSC = "false" →*IDLE*	●If active user: CPDLC-provider-abort indication ●If DSC = "true", set DSC = "false" →*IDLE*
T_{start} Expires	cannot occur	●D-ABORT request ● CPDLC-provider-abort indication ●If DSC = "true", set DSC = "false" →*IDLE*	cannot occur	cannot occur	cannot occur

Table 3-12. CPDLC-ground-ASE state table

STATE → *EVENT ↓*	*IDLE*	*START-REQ*	*START-IND*	*DIALOGUE*	*END*	*FORWARD*
Dialogue Service Events						
D-START Indication APDU Aircraft mode "cpdlc"	● CPDLC-start indication →*START-IND*	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur
D-START Indication APDU Aircraft mode "dsc"	● DSC-start indication, ●Set DSC = "true" →*START-IND*	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur
D-START Indication APDU Forward version equal and function supported	● CPDLC-forward indication ●D-START response →*IDLE*	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur
D-START Indication APDU Forward version not equal or function not supported	●D-START response →*IDLE*	cannot occur	cannot occur	cannot occur	cannot occur	cannot occur

STATE → *EVENT ↓*	*IDLE*	*START-REQ*	*START-IND*	*DIALOGUE*	*END*	*FORWARD*
D-START Confirmation *Result* "accepted" DSC = "false"	cannot occur	●Stop timer t_{start} ● CPDLC-start confirmation →*DIALOGUE*	cannot occur	cannot occur	cannot occur	cannot occur
D-START Confirmation *Result* "rejected (permanent)" and *Reject Source* "DS-user" DSC = "false"	cannot occur	●Stop timer t_{start} ● CPDLC-start confirmation →*IDLE*	cannot occur	cannot occur	cannot occur	●Stop timer t_{start} ●CPDLC-forward confirmation →*IDLE*
D-DATA Indication	cannot occur	cannot occur	cannot occur	●CPDLC-message indication →*DIALOGUE*	●if DSC = "false ●CPDLC-message indication →*END* ●else not permitted	cannot occur
D-END Indication DSC = "true"	cannot occur	cannot occur	cannot occur	●DSC-end indication →*END*	cannot occur	cannot occur
D-END Confirmation DSC = "false" *Result* "accepted"	cannot occur	cannot occur	cannot occur	cannot occur	●CPDLC-end confirmation →*IDLE*	cannot occur
D-END Confirmation DSC = "false" *Result* "rejected"	cannot occur	cannot occur	cannot occur	cannot occur	●CPDLC-end confirmation →*DIALOGUE*	cannot occur
CPDLC-User Events						
CPDLC-start Request	●D-START request ●Start timer t_{start} →*START-REQ*	not permitted	not permitted	not permitted	not permitted	not permitted
CPDLC-start Response DSC = "false" *Result* "accepted"	not permitted	not permitted	●D-START response →*DIALOGUE*	not permitted	not permitted	not permitted
CPDLC-start Response DSC = "false" *Result* "rejected"	not permitted	not permitted	●D-START response →*IDLE*	not permitted	not permitted	not permitted
DSC-start Response DSC = "true" *Result* "accepted"	not permitted	not permitted	●D-START response →*DIALOGUE*	not permitted	not permitted	not permitted
DSC-start Response DSC = "true" *Result* "rejected"	not permitted	not permitted	●D-START response ●Set DSC = "false" →*IDLE*	not permitted	not permitted	not permitted

STATE → *EVENT ↓*	*IDLE*	*START-REQ*	*START-IND*	*DIALOGUE*	*END*	*FORWARD*
CPDLC-message Request	not permitted	not permitted	not permitted	●D-DATA request →*DIALOGUE*	●If DSC = "true" ●D-DATA request →*END* ●else not permitted	not permitted
CPDLC-end Request DSC = "false"	not permitted	not permitted	not permitted	●D-END request →*END*	not permitted	not permitted
DSC-end Service Response DSC = "true" *Result* "accepted"	cannot occur	cannot occur	cannot occur	not permitted	●D-END response ●Set DSC = "false" →*IDLE*	not permitted
DSC-end Service Response DSC = "true" *Result* "rejected"	cannot occur	cannot occur	cannot occur	not permitted	●D-END response →*DIALOGUE*	not permitted
CPDLC-forward Request	●D-START request ●Start timer t_{start} →*FORWARD*	not permitted	not permitted	not permitted	not permitted	not permitted
ABORT Events						
CPDLC-user-abort Request	not permitted	●Stop timer t_{start}, if set ●D-ABORT request ●If DSC = "true", set DSC= "false" →*IDLE*	●Stop timer t_{start}, if set ●D-ABORT request ●If DSC = "true", set DSC= "false" →*IDLE*	●D-ABORT request ●If DSC = "true", set DSC= "false" →*IDLE*	●D-ABORT request ●If DSC = "true", set DSC= "false" →*IDLE*	●Stop timer t_{start}, if set ●D-ABORT request →*IDLE*
D-ABORT Indication *Originator* "provider"	cannot occur	●Stop timer T_{start}, if set ●If active user: CPDLC-provider-abort indication ●If DSC = "true", set DSC= "false" →*IDLE*	●Stop timer T_{start}, if set ●If active user: CPDLC-provider-abort indication ●If DSC = "true", set DSC= "false" →*IDLE*	●If active user: CPDLC-provider-abort indication ●If DSC = "true", set DSC= "false" →*IDLE*	●If active user: CPDLC-provider-abort indication ●If DSC = "true", set DSC= "false" →*IDLE*	●Stop timer T_{start}, if set ●If active user: CPDLC-provider-abort indication →*IDLE*
D-ABORT Indication *Originator* "user"	cannot occur	●Stop timer T_{start}, if set ●If active user: CPDLC-user-abort indication ●If DSC = "true", set DSC= "false" →*IDLE*	●Stop timer T_{start}, if set ●If active user: CPDLC-user-abort indication ●If DSC = "true", set DSC= "false" →*IDLE*	●If active user: CPDLC-user-abort indication ●If DSC = "true", set DSC= "false" →*IDLE*	●If active user: CPDLC-user-abort indication ●If DSC = "true", set DSC= "false" →*IDLE*	●Stop timer T_{start}, if set ●If active user: CPDLC-user-abort indication →*IDLE*

STATE → *EVENT ↓*	*IDLE*	*START-REQ*	*START-IND*	*DIALOGUE*	*END*	*FORWARD*
D-P-ABORT Indication	cannot occur	●Stop timer t_{start}, if set ●If active user: CPDLC-provider-abort indication ●If DSC = "true", set DSC= "false" →*IDLE*	●Stop timer t_{start}, if set ●If active user: CPDLC-provider-abort indication ●If DSC = "true", set DSC= "false" →*IDLE*	●If active user: CPDLC-provider-abort indication ●If DSC = "true", set DSC= "false" →*IDLE*	●If active user: CPDLC-provider-abort indication ●If DSC = "true", set DSC= "false" →*IDLE*	●Stop timer t_{start}, if set ●If active user: CPDLC-provider-abort indication →*IDLE*
T_{start} Expires	cannot occur	●D-ABORT request ● CPDLC-provider-abort indication ●If DSC = "true", set DSC= "false" →*IDLE*	cannot occur	cannot occur	cannot occur	●D-ABORT request ● CPDLC-provider-abort indication →*IDLE*

3.6 COMMUNICATION REQUIREMENTS

3.6.1 Encoding rules

3.6.1.1 The CPDLC application shall apply PER encoding as defined in ISO/IEC 8825-2, using the basic unaligned variant to encode/decode the ASN.1 CPDLC message structure and content specified in 3.4.

3.6.1.2 When CPDLC APDUs are treated as bit-oriented values that are not padded to an integral number of octets, the length determinant includes only the significant bits of the encoding corresponding to the ASN.1 type.

3.6.2 Dialogue service requirements

3.6.2.1 Primitive requirements

Where dialogue service primitives, i.e. D-START, D-DATA, D-END, D-ABORT and D-P-ABORT, are described as being invoked in 3.5, the CPDLC-ground-ASE and the CPDLC-air-ASE shall exhibit external behaviour consistent with the dialogue service, as described in Part III of this manual, having been implemented and its primitives invoked.

3.6.2.2 Quality of Service requirements

3.6.2.2.1 The application service priority for CPDLC shall have the abstract value of "high priority flight safety messages".

3.6.2.2.2 The RER Quality of Service parameter of the D-START for CPDLC shall be set to the abstract value of "low".

3.6.2.2.3 The CPDLC-ASE shall map the CPDLC-start service or DSC-start service Class of Communication parameter abstract value to the ATSC routing class abstract value part of the D-START Quality of Service parameter as presented in Table 3-13.

Table 3-13. Mapping between class of communication and routing class abstract values

Class of communication abstract value	*Routing class abstract value*
No preference	No traffic type policy preference
A	Traffic follows Class A ATSC route(s)
B	Traffic follows Class B ATSC route(s)
C	Traffic follows Class C ATSC route(s)
D	Traffic follows Class D ATSC route(s)
E	Traffic follows Class E ATSC route(s)
F	Traffic follows Class F ATSC route(s)
G	Traffic follows Class G ATSC route(s)
H	Traffic follows Class H ATSC route(s)

Note.— ATSC class values are defined in Part IV of this manual.

3.6.2.3 ATN Security Requirements parameter

The *Security Requirements* parameter of the D-START shall be set to either “secured exchange” or “no security” for the CPDLC-start, DSC-start and CPDLC-forward services.

Note. – Indication of security is conveyed through the use of the Security Required parameter. The specific security mechanisms that may provide the security functionality are outside the scope of this document.

3.7 CPDLC-USER REQUIREMENTS

3.7.1 General

Note 1.— Requirements imposed on CPDLC-users concerning CPDLC messages and interfacing with the CPDLC-ASEs are presented in this section.

Note 2.— Where reference is made to the “CPDLC-user”, this implies both the CPDLC-air-user and the CPDLC-ground-user.

Note 3.— “CPDLC/IC Data” is:

a) what the CPDLC-user provides to the CPDLC-service as the CPDLC/IC Data or Response CPDLC/IC Data parameter when invoking a CPDLC-service request or response primitive; or

b) what the CPDLC-user receives in the same parameters from the CPDLC-service indication or confirmation primitives.

Note 4.— A "CPDLC message" is an uplink or downlink message exchanged between CPDLC-users as part of the CPDLC/IC Data.

Note 5.— The terms "CPDLC message", "message", "uplink message" and "downlink message" are used interchangeably and equate to a CPDLC message. When the terms "send" and "transmit" are used, it means that the CPDLC-user has invoked a CPDLC-service request or response primitive. When the term "receive" is used, it means that a CPDLC indication or confirmation primitive parameter containing a CPDLC/IC Data value has been provided by the CPDLC-service.

Note 6.— A CPDLC-air-user uses the contents of an uplink CPDLC/IC Data item to verify that it has been delivered to the correct destination, as identified by the combination of flight identification and ICAO 24-bit aircraft address, and is from the appropriate data authority, and that it was encoded and decoded by the CPDLC-users using the same CPDLC abstract syntax version.

Note 7.— A CPDLC-ground-user uses the contents of a downlink CPDLC/IC Data item to verify that it has been delivered to the correct destination and is from the expected aircraft, as identified by the combination of flight identification and ICAO 24-bit aircraft address, and that it was encoded and decoded by the CPDLC-users using the same CPDLC abstract syntax version.

Note 8.— The verification of correct delivery is performed by the CPDLC-user and not by the CPDLC-ASE. This avoids placing any reliance on the communications protocols or their implementation for a proof of correct delivery or a proof that data integrity has been maintained.

Note 9.— Use of an integrity check generation/validation algorithm other than the ATN message checksum specified in Chapter 5 of this manual is out of scope of this specification and requires prior agreement by both CPDLC-air-users and CPDLC-ground-users.

3.7.1.1 General CPDLC-service requirements

3.7.1.1.1 A CPDLC-ground-user shall invoke CPDLC-start service, DSC-start service, CPDLC-message service, CPDLC-end service and DSC-end service only when communicating with a CPDLC-air-user.

3.7.1.1.2 A CPDLC-ground-user shall invoke CPDLC-forward service only when communicating with another CPDLC-ground-user.

3.7.1.1.3 When a CPDLC-user invokes the CPDLC-start service, the DSC-start service or the CPDLC-forward service and requires a particular class of communication service, the CPDLC-user sets the *Class of Communication Service* parameter.

3.7.1.1.4 When a CPDLC-user does not require a particular class of communication, the user does not set the *Class of Communication Service* parameter.

3.7.1.1.5 When the CPDLC-user requires secure or unsecure services, the user sets the *Security Required* parameter as appropriate.

3.7.2 CPDLC-air-user requirements

3.7.2.1 The CPDLC-start service

3.7.2.1.1 Invoking the CPDLC-start request

3.7.2.1.1.1 If there is no CPDLC-service, the only CPDLC-service primitives the CPDLC-air-user shall be permitted to invoke are the CPDLC-start request or the DSC-start request.

3.7.2.1.1.2 If a CPDLC-air-user has invoked a CPDLC-start request, the CPDLC-air-user shall be prohibited from invoking any CPDLC-service primitive on the CPDLC dialogue initiated by the CPDLC-start service request, except the CPDLC-user-abort request, until after it has received a CPDLC-start confirmation.

3.7.2.1.2 Receipt of a CPDLC-start indication and invoking CPDLC-start response

If a CPDLC-start indication has been received, the CPDLC-air-user shall be prohibited from invoking any CPDLC-service primitive with this ground system, except the CPDLC-user-abort request, until after it has invoked the CPDLC-start response.

3.7.2.2 The DSC-start service

3.7.2.2.1 Invoking the DSC-start request

3.7.2.2.1.1 Only a CPDLC-air-user shall be permitted to invoke the DSC-start service request primitive.

3.7.2.2.1.2 A CPDLC-air-user shall only be permitted to invoke the DSC-start-service request primitive if the CPDLC-air-user has no existing DSC dialogue.

3.7.2.2.1.3 If a CPDLC-air-user has invoked a DSC-start request, the CPDLC-air-user shall be prohibited from invoking any CPDLC-service primitive on the DSC dialogue initiated by the DSC-start service request, except the CPDLC-user-abort request, until after it has received a DSC-start confirmation.

3.7.2.2.2 Receipt of a DSC-start confirmation

If a DSC-start confirmation has been received with a *Result* parameter containing the abstract value “accepted”, the CPDLC-air-user shall:

a) establish an association between a CPDLC-ASE invocation and the ground system facility designation contained in the DSC-start request *Facility Designation* parameter; and

b) associate the CPDLC-ASE invocation with a downstream data authority.

3.7.2.3 The CPDLC-end service

3.7.2.3.1 The CPDLC-end service request

The CPDLC-air-user shall be prohibited from invoking the CPDLC-end request.

3.7.2.4 The DSC-end service

3.7.2.4.1 The DSC-end request

3.7.2.4.1.1 Only the CPDLC-air-user shall be permitted to invoke the DSC-end request.

3.7.2.4.1.2 If a CPDLC-air-user has invoked a DSC-end service request primitive, the CPDLC-air-user shall be prohibited from invoking any CPDLC-service primitive with this ground system (except the CPDLC-user-abort request primitive) until it receives a DSC-end service confirmation primitive.

3.7.3 CPDLC-ground-user requirements

3.7.3.1 The CPDLC-start service

3.7.3.1.1 Invoking the CPDLC-start request

3.7.3.1.1.1 If there is no CPDLC-service, the only CPDLC-service primitives the CPDLC-ground-user shall be permitted to invoke are the CPDLC-start-request or the CPDLC-forward request.

3.7.3.1.1.2 If a CPDLC-ground-user has invoked a CPDLC-start request, the CPDLC-ground-user shall be prohibited from invoking any CPDLC-service primitive on the CPDLC dialogue initiated by the CPDLC-start service request, except the CPDLC-user-abort request, until after it has received a CPDLC-start confirmation.

3.7.3.1.2 Receipt of a CPDLC-start indication and invoking CPDLC-start response

3.7.3.1.2.1 If a CPDLC-start indication is received from an aircraft with which the ground system currently has a CPDLC dialogue, the CPDLC-ground-user shall:

a) invoke the CPDLC-start response with the *Result* parameter set to the abstract value “accepted”; and

b) invoke the CPDLC-user-abort request for the first CPDLC dialogue with that aircraft.

3.7.3.1.2.2 The CPDLC-ground-user shall be prohibited from invoking the CPDLC-start response unless and until it has received a CPDLC-start indication.

3.7.3.1.2.3 If a CPDLC-start indication has been received, the CPDLC-ground-user shall be prohibited from invoking any CPDLC-service primitive, except the CPDLC-user-abort request with that aircraft, until after it has invoked the CPDLC-start response.

3.7.3.2 The DSC-start service

3.7.3.2.1 Receipt of a DSC-start indication and invoking DSC-start response

3.7.3.2.1.1 The CPDLC-ground-user shall be prohibited from invoking the DSC-start response unless and until it has received a DSC-start indication.

3.7.3.2.1.2 If a DSC-start indication is received from an aircraft with which the ground system currently has a DSC dialogue, the CPDLC-ground-user shall:

a) invoke the DSC-start response with the *Result* parameter set to the abstract value “accepted”; and

b) invoke the CPDLC-user-abort request for the first DSC dialogue with that aircraft.

3.7.3.2.1.3 If a DSC-start indication has been received, the CPDLC-ground-user shall be prohibited from invoking any CPDLC-service primitive, except the CPDLC-user-abort request, until after it has invoked the DSC-start response.

3.7.3.3 The CPDLC-end service

3.7.3.3.1 The CPDLC-end request

3.7.3.3.1.1 Only the CPDLC-ground-user shall be permitted to invoke the CPDLC-end request.

3.7.3.3.1.2 If a CPDLC-ground-user has invoked a CPDLC-end service request primitive, the CPDLC-ground-user shall be prohibited from invoking any CPDLC-service primitive with this aircraft, except the CPDLC-user-abort request primitive, until after it has received a CPDLC-end service confirmation primitive.

3.7.3.4 The DSC-end service

3.7.3.4.1 Receipt of a DSC-end indication and invoking DSC-end response

If a DSC-end service response is invoked with an “accepted” Result, this will result in ending the dialogue regardless of any CPDLC messages contained in the *CPDLC/IC Data* parameter.

3.7.3.5 The CPDLC-forward service

3.7.3.5.1 Invoking the CPDLC-forward request

3.7.3.5.1.1 Only the CPDLC-ground-user shall be permitted to invoke the CPDLC-forward request.

3.7.3.5.1.2 The CPDLC-ground-user shall include in the CPDLC-forward request *CPDLC Message* parameter a MessageForward APDU element containing an ATCDownlinkMessageData or an ATCUplinkMessageData encoded using the unaligned basic PER.

3.7.3.6 The CPDLC-user-abort service

3.7.3.6.1 Issuing a CPDLC-user-abort request

The CPDLC-ground-user shall have the capability to invoke a CPDLC-user-abort request with the *Reason* parameter set to CPDLCUserAbortReason value [commanded-termination].

3.8 SUBSETTING RULES

3.8.1 General

Note.— This section specifies conformance requirements that all implementations of the CPDLC protocol obey.

3.8.1.1 An implementation of either the CPDLC ground-based service or the CPDLC air-based service claiming conformance to this section shall support the CPDLC protocol features as shown in Tables 3-18 to 3-21.

3.8.1.2 The "status" column indicates the level of support required for conformance to the CPDLC-ASE protocol. The values are as follows:

"M" mandatory support is required;

"O" optional support is permitted for conformance to the CPDLC protocol;

"N/A" the item is not applicable; and

"C.n" the item is conditional where n is the number that identifies the condition that is applicable. The definitions for the conditional statements used in this section are written under the tables in which they first appear.

Table 3-18. Protocol versions implemented

Version	*Status*	*Associated predicate*
Version 1	M	None

Table 3-19. CPDLC protocol options

	Status	*Associated predicate*
CPDLC-air-ASE	C.1	CPDLC/air
CPDLC-ground-ASE	C.1	CPDLC/ground
DSC function supported	If (CPDLC/air), then O, else M	DSC-FU
DSC function supported by CPDLC-ground-user	If (CPDLC/ground), then O, else N/A	DSC-USER
Forward function supported by initiating user	If (CPDLC/ground), then O, else N/A	FWD-INIT
Forward function supported by receiving user	If (CPDLC/ground), then O, else N/A	FWD-USER

C.1: A conformant implementation will support one, and only one, of these two options.

Table 3-20. CPDLC-air-ASE conformant configurations

	List of predicates	*Functionality description*
I.	CPDLC/air	A CPDLC-air-ASE supporting just the core (see Note) CPDLC functionality (no downstream clearance capability)
II.	CPDLC/air + DSC-FU	A CPDLC-air-ASE supporting the core CPDLC functionality and the downstream clearance capability (complete CPDLC-air-ASE functionality)
Note.— The core CPDLC functionality is defined as support for the CPDLC-start, message, and end services plus the abort services.		

Table 3-21. CPDLC-ground-ASE conformant configurations

	List of predicates	*Functionality description*
I.	CPDLC/ground	A CPDLC-ground-ASE supporting the core CPDLC functionality plus: – functionality for receiving and CPDLC-ground-user rejecting a request for a DSC dialogue – functionality for receiving and indicating that the forward function is not supported
II.	CPDLC/ground + DCS-FU + DSC-USER	A CPDLC-ground-ASE supporting the core CPDLC functionality and downstream clearance function plus: – functionality for receiving and indicating that the ground forward function is not supported
III.	CPDLC/ground + DSC-FU + FWD-INIT	A CPDLC-ground-ASE supporting the core CPDLC functionality plus: – functionality for receiving and CPDLC-ground-user rejecting a request for a DSC dialogue – functionality for receiving and indicating that the ground forward function is not supported – functionality for supporting the capability to initiate the ground forwarding of CPDLC messages
IV.	CPDLC/ground + DCS-FU + DSC-USER + FWD-INIT	A CPDLC-ground-ASE supporting the core CPDLC functionality and downstream clearance function plus: – functionality for receiving and indicating that the ground forward function is not supported – functionality for supporting the capability to initiate the ground forwarding of CPDLC messages
V.	CPDLC/ground + DSC-FU FWD-USER	A CPDLC-ground-ASE supporting the core CPDLC functionality plus: – functionality for receiving and CPDLC-ground-user rejecting a request for DSC dialogue

	List of predicates	*Functionality description*
VI.	CPDLC/ground + DCS-FU + DSC-USER + FWD-USER	A CPDLC-ground-ASE supporting the core CPDLC functionality and downstream clearance function plus: – functionality for CPDLC-ground-user support for the receipt of CPDLC ground forward messages
VII.	CPDLC/ground + DSC-FU + FWD-INIT + FWD-USER	A CPDLC-ground-ASE supporting the core CPDLC functionality plus: – functionality for receiving and CPDLC-ground-user rejecting a request for a DSC dialogue – full CPDLC ground forwarding functionality (initiating and user receiving)
VIII.	CPDLC/ground + DSC-FU + DSC-USER + FWD-INIT + FWD-USER	A CPDLC-ground-ASE supporting the core CPDLC functionality and downstream clearance function plus: – full CPDLC ground forwarding functionality (initiating and user receiving) (complete CPDLC-ground-ASE functionality)

Chapter 4

FLIGHT INFORMATION SERVICES (FIS)

(Placeholder awaiting development of new applications, such as D-ATIS
Take another look at text “to be developed”))

Chapter 5

AUTOMATIC DEPENDENT SURVEILLANCE — CONTRACT (ADS-C)

Note.— Structure of 5.1 defines the air-ground communication aspects of ADS-C. Section 4.2 defines the ground-ground (i.e. ADS-C report forwarding) aspects of ADS-C.

5.1 AUTOMATIC DEPENDENT SURVEILLANCE (CONTRACT) APPLICATION

5.1.1 Introduction

5.1.1.1 The ADS-C air ground application will allow users to obtain positional and other information from suitably equipped aircraft in a timely manner in accordance with their requirements. The ADS-C application is designed to give automatic reports about aircraft to a user. The ADS-C reports give positional as well as other information likely to be of use to the air traffic management function, including air traffic control. The aircraft provides the information to the user under the following circumstances:

a) under a contract (known as a demand contract) agreed with the ground system, the aircraft provides the information as soon as possible and once only;

b) under a contract (known as a periodic contract) agreed with the ground system, the aircraft provides information on a regular basis;

c) under a contract (known as an event contract) agreed with the ground system, the aircraft provides information when certain events are detected by the avionics;

Note 1.— Structure of 5.1: this chapter defines the air-ground communication aspects of the ADS-C application only.

a) 5.1.1: INTRODUCTION contains 5.1's purpose, structure, and a summary of the functions of the ADS-C application.

b) 5.1.2: GENERAL REQUIREMENTS contains backwards compatibility and error processing requirements.

c) 5.1.30: THE ABSTRACT SERVICE contains the description of the abstract service provided by the application service elements (ASE) defined for the ADS-C application.

d) 5.1.4: FORMAL DEFINITION OF MESSAGES contains the formal definition of Application Protocol Data Unit exchanged by ADS-C ASEs using Abstract Syntax Notation Number One (ASN.1).

e) 5.1.5: PROTOCOL DEFINITION describes the exchanges of messages allowed by the ADS-C protocol, as well as time constraints and the exception handling procedures associated with these exchanges. 0 describes also the ADS-C protocol in terms of state tables.

f) 5.1.6: COMMUNICATION REQUIREMENTS contains the requirements that the ADS-C ASE application imposes on the underlying communication system.

g) 5.1.7: ADS-USER REQUIREMENTS outlines the requirements that a user of an ADS-C ASE must meet.

h) 5.1.8: SUBSETTING RULES provides rules for subsetting the ADS-C application.

Note 2.— General Functionality

a) It will be necessary for an implementation to provide information which is both accurate and timely in the ADS-C reports; however, quantification of the age and accuracy of the information is beyond the scope of 5.1.

b) The ground system can request that authentication and data integrity services be performed to ensure the identity of participating entities is as claimed and to ensure that ADS-C data is not altered or destroyed in an unauthorized manner.

Note 3.— Establishment and Operation of a Demand Contract

a) Functional Description

1) This function allows the ground system to establish a demand contract with an aircraft. Realisation of the contract involves the sending of a single report from an aircraft to the ground system.

2) Any number of demand contracts may be sequentially established with an aircraft.

3) The ground system sends a demand contract request message to the avionics. The avionics then determines whether or not there are errors in the request, and if there are no errors, whether or not it is able to comply with the request. If the avionics can comply with the demand contract request it sends the ADS-C report in a contract accept message as soon as possible. If there are errors in the contract request, or if the avionics cannot comply with the request, it sends a contract reject message to the ground system indicating the reason for its inability to accept the contract. If the avionics can partially comply with the request, it sends a non-compliance notification message indicating those parts of the contract with which it cannot comply, and then it sends an ADS-C report message.

4) *If the avionics cannot send the report within the contract response time, it sends a positive acknowledgement message first to indicate its acceptance of the contract.*

b) Message Descriptions

1) The demand contract request message stipulates which information fields are to be included in the ADS-C report.

2) The ADS-C report message contains information blocks.

3) *A contract accept message indicates acceptance of the contract and contains an ADS-C report.*

4) *A positive acknowledgement message indicates acceptance of the contract and contains no further information.*

5) *A contract reject message contains an indication of the reason why the contract has not been accepted.*

6) *A non-compliance notification message contains an indication of which information fields cannot be sent.*

Note 4.— Establishment and Operation of an Event Contract

a) Functional Description

1) *This function allows the ground system to establish an event contract with the aircraft. Realisation of the contract involves the sending of reports from the aircraft to the ground system when certain agreed events occur. The definition of the events is outside the scope of 5.1*

2) *Only one event contract may exist between the ground system and avionics at any one time, but this may contain multiple event types.*

3) *Acceptance of an event contract request implicitly cancels an existing event contract, if one exists.*

4) *The ground system sends an event contract request to the avionics. The avionics then determines whether or not there are errors in the request, and if not, whether or not it is able to comply with the request. If the avionics can comply with the event contract request it sends any required baseline ADS-C report in a contract accept message. If the contracted event occurs, an ADS-C report is sent.*

5) *If there are errors in the event contract request, or if the avionics cannot comply with the request, it sends a contract reject message to the ground system indicating the reason for its inability to accept the contract within contract response time.*

6) *If the avionics can partially comply with the request, it sends a non-compliance notification indicating those parts of the contract with which it cannot comply. If a contracted event occurs with which it can comply, an ADS-C report is sent.*

7) *Depending of the type of event, if the event occurs, either a report is sent periodically while the limit(s) specified in the contract are exceeded or a single report is sent every time the event occurs.*

b) Message Descriptions

1) The event contract request message contains an indication of the events to be reported on, together with clarifying information.

2) The ADS-C report message has the same structure as in the operation of a demand contract. The inclusion of data blocks can be driven by the type of the triggering event(s). The definition of the reports is outside the scope of 4.1

3) A contract accept message indicates acceptance of the contract and contains an ADS-C report.

4) A positive acknowledgement message indicates acceptance of the contract and contains no further information.

5) A contract reject message contains an indication of the reason why the contract has not been accepted.

6) A non-compliance notification message contains an indication of the events which the avionics cannot detect.

Note 5.— Establishment and Operation of a Periodic Contract

a) Functional Description

1) This function allows the ground system to establish a periodic contract with the aircraft. Realisation of the contract involves the sending of reports from the aircraft to the ground system at regular intervals (the reporting rate).

2) Only one periodic contract may exist between a ground system and the avionics at any one time.

3) The ground system sends a periodic contract request message to the avionics. This message contains the reporting rate and an indication of which information blocks are required. The avionics then determines whether or not there are errors in the request, and if not, whether or not it is able to comply with the request. If the avionics can comply with the periodic contract request it sends its first report in a contract accept message, and then sends other ADS-C reports at the intervals requested. If it cannot send the first report within the contract response time, it sends a positive acknowledgement message first to indicate its acceptance of the contract.

4) Acceptance of a periodic contract request implicitly cancels any existing periodic contract.

5) If there are errors in the periodic contract request message, or if the avionics cannot accept the contract, it sends a contract reject message to the ground system indicating the reason for its inability to accept the contract within the contract response time.

6) If the avionics can partially comply with the request, it sends a non-compliance notification message indicating those parts of the contract with which it cannot comply. Non-compliance can be caused by either inability to meet the requested reporting rate, and/or inability to supply the requested information.

b) Message Descriptions

1) The periodic contract request message may optionally contain the reporting interval and the modulus for the optional information blocks

Moduli indicate the multiple of the reporting rate that the information block is sent at (e.g. meteorological information modulus of 5 means that the meteorological information block is sent with every 5th report).

2) The ADS-C report message has the same structure as in the operation of a demand contract.

3) A contract accept message indicates acceptance of the contract and contains an ADS-C report.

4) A positive acknowledgement message indicates acceptance of the contract and contains no further information.

5) A contract reject message contains an indication of the reason why the contract has not been accepted.

6) A non-compliance notification message contains an indication of which optional information fields cannot be sent, and/or indicates that the requested periodic report cannot be met.

Note 6.— Cancellation of Contracts

a) Functional Description

1) This function allows the ground system explicitly to cancel a contract that is in operation. The ground system sends a cancel contract message to the avionics. The avionics cancels the contract and acknowledges the cancellation.

2) Implicit cancellation occurs when a periodic contract is in place, and then the ground system establishes a new periodic contract - the first one is implicitly cancelled on the establishment of the second; similarly with event contracts. Demand contracts are implicitly cancelled when the report is sent. There are no additional information flows associated with implicit cancellation.

3) The ground system may also cancel all contracts in a single cancel all contracts message. The avionics cancels all contracts and acknowledges the cancellation.

b) Message Descriptions

1) The cancel contract message contains an indication of the contract to be cancelled.

2) The cancel all contracts message contains no additional information.

3) A positive acknowledgement message contains no additional information.

4) A negative acknowledgement message contains an indication that no such contract existed (i.e. if a periodic contract was attempted to be cancelled when no periodic contract was active).

5.1.2 General Requirements

5.1.2.1 ADS-C ASE Version Number

The ADS-air-ASE and the ADS-ground-ASE version numbers shall both be set to one.

5.1.2.2 Error Processing Requirements

5.1.2.2.1 In the event of information input by the ADS-user being incompatible with that able to be processed by the system, the user shall be notified.

5.1.2.2.2 In the event of an ADS-user invoking an ADS-C service primitive, when the ADS-C ASE is not in a state specified in 5.1.5, the user shall be notified.

5.1.3 The Abstract Service

5.1.3.1 Service Description

5.1.3.1.1 An implementation of either the ADS-C ground based service or the ADS-C air based service shall exhibit behaviour consistent with having implemented an ADS-ground-ASE, or ADS-air-ASE respectively.

5.1.3.1.2 This section defines the abstract service interface for the ADS-C service. The ADS-C ASE abstract service is described in this section from the viewpoint of the ADS-air-user, the ADS-ground-user and the ADS-C service-provider.

5.1.3.1.3 This section defines the static behaviour (i.e. the format) of the ADS-C service. Its dynamic behaviour (i.e. how it is used) is described in 5.7.

5.1.3.1.4 Figure 5-1 shows the functional model of the ADS-C Application. The functional modules identified in this model are the following:

a) the ADS-user,

b) the ADS-C Application Entity (ADS-AE) service interface,

c) the ADS-AE,

d) the ADS-C Control Function (ADS-CF),

e) the ADS-C application service element (ADS-C ASE) service interface,

f) the ADS-C ASE, and

g) the DS interface.

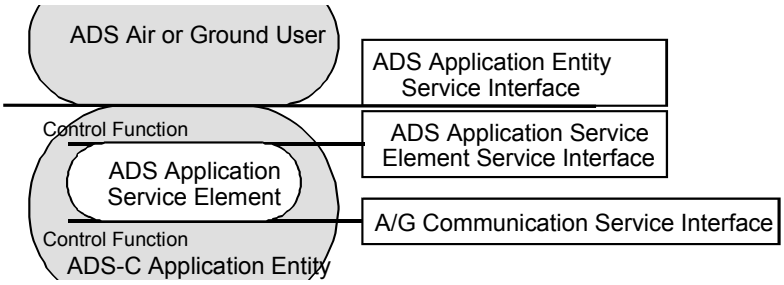


Figure 5-1: Functional Model of the ADS-C Application

5.1.3.1.5 The ADS-user represents the operational part of the ADS-C system. This user does not perform the communication functions but relies on a communication service provided to it via the ADS-AE through the ADS-AE service interface. The individual actions at this interface are called ADS-AE service primitives. Similarly, individual actions at other interfaces in the communication system are called service primitives at these interfaces.

5.1.3.1.6 The ADS-AE consists of several elements, including the ADS-C ASE and the ADS-CF. The air/ground communication service interface is made available by the ADS-CF to the ADS-C ASE for communication with the peer ADS-C ASE.

5.1.3.1.7 The ADS-C ASE is the element in the communication system which executes the ADS-C specific protocol. In other words, it takes care of the ADS-C specific service primitive sequencing actions, message creation, timer management, error and exception handling.

5.1.3.1.8 The ADS-C ASE interfaces only with the ADS-CF. This ADS-CF is responsible for mapping service primitives received from one element (such as the ADS-C ASE and the ADS-user) to other elements which interface with it. The part of the ADS-CF which is relevant from the point of view of these specifications, i.e. the part between the ADS-user and the ADS-C ASE, will map ADS-AE service primitives to ADS-C ASE service primitives transparently.

5.1.3.1.9 The DS interface is the interface between the ADS-C ASE and part of ADS-CF underneath, the ADS-C ASE and provides the ATN dialogue service. The ADS-C ASE is designed to move on top of another communication service with no impact on the application protocol.

5.1.3.2 The ADS-C ASE Abstract Service

5.1.3.2.1 There is no requirement to implement the service in an ADS-C product; however, it is necessary to implement the ground based and air based system in such a way that it will be impossible to detect (from the peer system) whether or not an interface has been built.

5.1.3.2.2 The ADS-C ASE abstract service shall consist of a set of the following services as allowed by the subsetting rules defined in 5.8:

a) ADS-demand-contract service as defined in 5.1.3.3;

b) ADS-event-contract service as defined in 5.1.3.4;

c) ADS-periodic-contract service as defined in 5.1.3.5;

d) ADS-report service as defined In 5.1.3.6;

e) ADS-cancel service as defined in 5.1.3.7;

f) ADS-cancel-all-contracts service as defined in 5.1.3.8

g) ADS-user-abort service as defined in 5.1.3.9;

h) ADS-provider-abort service as defined in 5.1.3.10.

5.1.3.2.3 ADS-demand-contract, ADS-event-contract, ADS-periodic-contract, ADS-cancel and ADS-cancel-all-contracts are services only initiated by the ADS-ground-user.

5.1.3.2.4 ADS-report is a service only initiated by the ADS-air-user.

5.1.3.2.5 ADS-user-abort is a service initiated by an ADS-air-user or an ADS-ground-user.

5.1.3.2.6 ADS-provider-abort is a service only initiated by the ADS-C service-provider.

5.1.3.2.7 An abstract syntax is a syntactical description of a parameter which does not imply a specific implementation. Only when the ADS-C ASE maps a parameter onto an APDU field, or vice-versa, is the abstract syntax of the parameter described by using the ASN.1 of 0 for this field.

5.1.3.2.8 For a given primitive, the presence of each parameter is described by one of the following values in the parameter tables in 5.1.3

blank	not present;
C	conditional upon some predicate explained in the text;
C(=)	conditional upon the value of the parameter to the immediate left being present, and equal to that value;
M	mandatory;
M(=)	mandatory, and equal to the value of the parameter to the immediate left;
U	user option.

5.1.3.2.9 The following abbreviations are used in this document:

Req	request; data is input by an ADS-user initiating the service to its respective ADS-C ASE;
Ind	indication; data is indicated by the receiving ADS-CASE to its respective ADS-user;
Rsp	response; data is input by receiving ADS-user to its respective ADS-C ASE;
Cnf	confirmation; data is confirmed by the initiating ADS-CASE to its respective ADS-user.

5.1.3.2.10 An unconfirmed service allows just one message to be transmitted, in one direction.

5.1.3.2.11 A confirmed service provides end-to-end confirmation that a message sent by one user was received by its peer user.

5.1.3.3 ADS-demand-contract Service

5.1.3.3.1 General

5.1.3.3.1.1 The ADS-demand-contract service allows the ADS-ground-user to request a demand contract with the aircraft. It is a confirmed service, initiated by the ADS-ground-user.

5.1.3.3.1.2 The ADS-demand-contract service shall contain primitives and parameters as presented in Table 5-1.

Table 5-1: ADS-demand-contract service parameters

Parameter Name	*Req*	*Ind*	*Rsp*	*Cnf*
Aircraft Identity	M			
Class of Communication Service	U			
ICAO Facility Designation	C	C(=)		
ADS-C Message Set Version Number	U	C(=)		
Security Required	C			
ADS/IC Contract Data	M	M(=)		
Result			M	M(=)
ADS/IC Positive Acknowledgement			C	C(=)
ADS/IC Report Data			C	C(=)
ADS/IC Non Compliance Data			C	C(=)
ADS/IC Reject Data			C	C(=)

5.1.3.3.2 Aircraft Identity parameter

5.1.3.3.2.1 This parameter contains the 24-bit ICAO aircraft address of the aircraft or the aircraft identification with which the contract is being made.

5.1.3.3.2.2 When an aircraft address is used to identify the aircraft, the *Aircraft Identity* parameter value shall conform to an abstract value corresponding to a 24-bit ICAO aircraft address.

5.1.3.3.2.3 When an aircraft identification is used to identify the aircraft, the *Aircraft Identity* parameter value shall conform to an abstract value corresponding to a 2 to 7 character aircraft identification.

5.1.3.3.3 Class of Communication Service parameter

5.1.3.3.3.1 This parameter contains the value of the required class of communication service, if specified by the

ADS-ground-user.

5.1.3.3.3.2 Where specified by the ADS-ground-user, the *Class Of Communication Service* parameter shall have one of the following abstract values: “A”, “B”, “C”, “D”, “E”, “F”, “G” or “H”.

5.1.3.3.3.3 If contracts are currently in place, the *Class Of Communication Service* parameter is not used by the ADS-C service provider.

5.1.3.3.3.4 Where not specified by the ADS-ground-user, when there are no contracts already in force, this indicates that there will be no routing preference.

5.1.3.3.4 ICAO Facility Designation parameter

5.1.3.3.4.1 This parameter contains the 4 to 8 character ICAO Facility Designation of the ICAO facility which is initiating the contract. If contracts are currently in place, this parameter is not used by the ADS-service-provider.

5.1.3.3.4.2 The *ICAO Facility Designation* parameter value shall conform to an abstract value corresponding to a 4 to 8 character ICAO Facility Designation.

5.1.3.3.4.3 The *ICAO Facility Designation* parameter value shall be provided when the ADS-ground-user has no other contracts in place with the aircraft.

5.1.3.3.5 ADS-C Message Set Version Number parameter

5.1.3.3.5.1 This parameter contains the version number of the ADS-ASE and identifies the user data abstract syntax.

5.1.3.3.5.2 When provided by the ADS-user, the *ADS-C Message Set Version Number* parameter shall conform to an abstract integer value from 1 to 255.

5.1.3.3.6 Security Required parameter

5.1.3.3.6.1 This parameter contains the value of the required level of security, if specified by the ADS-ground-user.

5.1.3.3.6.2 If the received *Security Required* parameter is not as expected per the local security policy, the receiving ADS-air-ASE will abort.

5.1.3.3.6.3 Where specified by the ADS-ground-user, the *Security Required* parameter shall have one of the following abstract values: “no security” or “secured exchange”.

5.1.3.3.6.4 If contracts are currently in place, the *Security Required* parameter is not used by the ADS-C service provider.

5.1.3.3.7 ADS/IC Contract Data parameter

5.1.3.3.7.1 This parameter contains the details of the demand contract as requested by the ADS-ground-user.

5.1.3.3.7.2 This parameter contains an Application Message Integrity Check.

5.1.3.3.7.3 The *ADS/IC Contract Details* parameter value shall conform to the ASN.1 abstract syntax *ICContractRequest*.

5.1.3.3.8 Result parameter

5.1.3.3.8.1 This parameter indicates the extent to which the demand contract request can be complied with:

a) if it has the value “positive acknowledgement”, it indicates that the demand contract has been accepted but the ADS-C report will be sent later, and the *ADS/IC Positive Acknowledgement* parameter indicates the contract type,

b) if it has the value “accepted”, it indicates that the demand contract has been accepted and the *ADS/IC Report Data* parameter contains the report,

c) if it has the value “rejected”, it indicates that the demand contract has been rejected and the *ADS/IC Reject Data* parameter gives reasons,

d) if it has the value “non compliance notification”, it indicates that only some parts of the demand contract can be complied with and the *ADS/IC Non Compliance Data* parameter indicates which ones have been rejected.

5.1.3.3.8.2 If it has the value “positive acknowledgement”, this indicates that the contract has been accepted but cannot be satisfied immediately because the information is not available within the contract response time.

5.1.3.3.8.3 The *Result* parameter value shall conform to one of the following abstract values: “accepted”, “rejected”, “positive acknowledgement”, or “non compliance notification”.

5.1.3.3.9 ADS/IC Positive Acknowledgement parameter

5.1.3.3.9.1 This parameter contains the details of the ADS-C positive acknowledgement.

5.1.3.3.9.2 This parameter contains an Application Message Integrity Check.

5.1.3.3.9.3 The *ADS/IC positive acknowledgement* parameter value shall conform to the ASN.1 abstract syntax *ICPositiveAck*.

5.1.3.3.9.4 The *ADS/IC positive acknowledgement* parameter value shall be present if and only if the Result parameter contains the abstract value “positive acknowledgement”.

5.1.3.3.10 ADS/IC Report Data parameter

5.1.3.3.10.1 This parameter contains the details of the ADS-C report.

5.1.3.3.10.2 This parameter contains an Application Message Integrity Check.

5.1.3.3.10.3 The *ADS/IC Report Data* parameter value shall conform to the ASN.1 abstract syntax *ICReport*.

5.1.3.3.10.4 The *ADS/IC Report Data* parameter value shall be present if and only if the *Result* parameter contains the abstract value “accepted”.

5.1.3.3.11 ADS/IC Reject Data parameter

5.1.3.3.11.1 This parameter contains the reason of the contract rejection.

5.1.3.3.11.2 This parameter contains an Application Message Integrity Check.

5.1.3.3.11.3 The *ADS/IC Reject Data* parameter value shall conform to the ASN.1 abstract syntax *ICReject*.

5.1.3.3.11.4 The *ADS/IC Reject Data* parameter value shall be present if and only if the *Result* parameter contains the abstract value “rejected”.

5.1.3.3.12 ADS/IC Non Compliance Data parameter

5.1.3.3.12.1 This parameter contains an indication of which optional information fields cannot be sent.

5.1.3.3.12.2 This parameter contains an Application Message Integrity Check.

5.1.3.3.12.3 The *ADS/IC Non Compliance Data* parameter value shall conform to the ASN.1 abstract syntax *ICNonCompliance*.

5.1.3.3.12.4 The *ADS/IC Non Compliance Data* parameter value shall be present if and only if the *Result* parameter contains the abstract value “non compliance notification”.

5.1.3.4 ADS-event-contract Service

5.1.3.4.1 General

5.1.3.4.1.1 The ADS-event-contract service allows the ADS-ground-user to request an event contract with the aircraft. It is a confirmed service, initiated by the ADS-ground-user.

5.1.3.4.1.2 The ADS-event-contract service shall contain primitives and parameters as presented in Table 5-2.

Table 5-2: ADS-event-contract service parameters

Parameter Name	*Req*	*Ind*	*Rsp*	*Cnf*
Aircraft Identity	M			
Class of Communication Service	U			
ICAO Facility Designation	C	C(=)		
ADS-C Message Set Version Number	U	C(=)		
Security Required	C			
ADS/IC Contract Data	M	M(=)		
Result			M	M(=)
ADS/IC Positive Acknowledgement			C	C(=)
ADS/IC Report Data			C	C(=)
ADS/IC Reject Data			C	C(=)
ADS/IC Non Compliance Data			C	C(=)

5.1.3.4.2 Aircraft Identity

5.1.3.4.2.1 This parameter contains the 24 bit ICAO address of the aircraft or the aircraft identification with which the contract is being made.

5.1.3.4.2.2 When an aircraft address is used to identify the aircraft, the *Aircraft Identity* parameter value shall conform to an abstract value corresponding to a 24-bit ICAO aircraft address.

5.1.3.4.2.3 When an aircraft identification is used to identify the aircraft, the *Aircraft Identity* parameter value shall conform to an abstract value corresponding to a 2 to 7 character aircraft identification.

5.1.3.4.3 Class of Communication Service

5.1.3.4.3.1 This parameter contains the value of the required class of communication service, if specified by the ADS-ground-user.

5.1.3.4.3.2 Where specified by the ADS-ground-user, the *Class Of Communication Service* parameter shall have one of the following abstract values: “A”, “B”, “C”, “D”, “E”, “F”, “G” or “H”.

5.1.3.4.3.3 If contracts are currently in place, the *Class Of Communication Service* parameter is not used by the ADS-C service provider.

5.1.3.4.3.4 Where not specified by the ADS-ground-user, when there are no contracts already in force, this indicates that there will be no routing preference.

5.1.3.4.4 ICAO Facility Designation

5.1.3.4.4.1 This parameter contains the 4 to 8 character ICAO Facility Designation of the ICAO facility which is initiating the contract. If contracts are currently in place, this parameter is not used by the ADS-service-provider.

5.1.3.4.4.2 The *ICAO Facility Designation* parameter value shall conform to an abstract value corresponding to a 4 to 8 character ICAO Facility Designation.

5.1.3.4.4.3 The *ICAO Facility Designation* parameter value shall be provided when the ADS-ground-user has no other contracts in place with the aircraft.

5.1.3.4.5 ADS-C Message Set Version Number parameter

5.1.3.4.5.1 This parameter contains the version number of the ADS-ASE and identifies the user data abstract syntax.

5.1.3.4.5.2 When provided by the ADS-user, the *ADS-C Message Set Version Number* parameter shall conform to an abstract integer value from 1 to 255.

5.1.3.4.6 Security Required

5.1.3.4.6.1 This parameter contains the value of the required level of security, if specified by the ADS-ground-user.

5.1.3.4.6.2 If the received *Security Required* parameter is not as expected per the local security policy, the receiving ADS-air-ASE will abort.

5.1.3.4.6.3 Where specified by the ADS-ground-user, the *Security Required* parameter shall have one of the following abstract values: “no security” or “secured exchange”.

5.1.3.4.6.4 If contracts are currently in place, the *Security Required* parameter is not used by the ADS-C service provider.

5.1.3.4.7 ADS/IC Contract Data

5.1.3.4.7.1 This parameter contains the details of the event contract as requested by the ADS-ground-user.

5.1.3.4.7.2 This parameter contains an Application Message Integrity Check.

5.1.3.4.7.3 The *ADS/IC contract data* parameter value shall conform to the ASN.1 abstract syntax *ICContractRequest*.

5.1.3.4.8 Result

5.1.3.4.8.1 This parameter indicates the extent to which the event contract request can be complied with:

a) If it has the value “rejected”, it indicates that the event contract has been rejected and the *ADS/IC Reject Data* parameter gives reasons,

b) If it has the value “accepted”, it indicates that the event contract has been accepted and the *ADS/IC*

Report Data parameter contains the report,

c) If it has the value “non compliance notification”, it indicates that only some parts of the event contract can be complied with, and the *ADS/IC Non Compliance Data* parameter indicates which ones have been rejected,

d) If it has the value “positive acknowledgement”, it indicates full compliance with the event contract has been accepted but the ADS-C report will be sent later, and the *ADS/IC Positive Acknowledgement* parameter indicates the contract type.

5.1.3.4.8.2 The *Result* parameter value shall conform to one of the following abstract values: “accepted”, “rejected”, “positive acknowledgement”, or “non compliance notification”.

5.1.3.4.9 ADS/IC Positive Acknowledgement

5.1.3.4.9.1 This parameter contains the details of the ADS-C positive acknowledgement.

5.1.3.4.9.2 This parameter contains an Application Message Integrity Check.

5.1.3.4.9.3 The *ADS/IC positive acknowledgement* parameter value shall conform to the ASN.1 abstract syntax *ICPositiveAck*.

5.1.3.4.9.4 The *ADS/IC positive acknowledgement* parameter value shall be present if and only if the *Result parameter* contains the abstract value “positive acknowledgement”.

5.1.3.4.10 ADS/IC Report Data

5.1.3.4.10.1 This parameter contains the details of the ADS-C report.

5.1.3.4.10.2 This parameter contains an Application Message Integrity Check.

5.1.3.4.10.3 The *ADS/IC report data* parameter value shall conform to the ASN.1 abstract syntax *ICReport*.

5.1.3.4.10.4 The *ADS/IC report data* parameter value shall be present if and only if the *Result* parameter contains the abstract value “accepted”.

5.1.3.4.11 ADS/IC Reject Data

5.1.3.4.11.1 This parameter contains the reason of the contract rejection.

5.1.3.4.11.2 This parameter contains an Application Message Integrity Check.

5.1.3.4.11.3 The *ADS/IC reject data* parameter value shall conform to the ASN.1 abstract syntax *ICReject*.

5.1.3.4.11.4 The *ADS/IC reject data* parameter value shall be present if and only if the *Result parameter* contains the abstract value “rejected”.

5.1.3.4.12 ADS/IC Non Compliance Data

5.1.3.4.12.1 This parameter contains an indication of which optional information fields cannot be sent.

5.1.3.4.12.2 This parameter contains an Application Message Integrity Check.

5.1.3.4.12.3 The *ADS/IC non compliance data* parameter value shall conform to the ASN.1 abstract syntax *ICNonCompliance*.

5.1.3.4.12.4 The *ADS/IC non compliance data* parameter value shall be present if and only if the *Result parameter* contains the abstract value “non compliance notification”.

5.1.3.5 ADS-periodic-contract Service

5.1.3.5.1 General

5.1.3.5.1.1 The ADS-periodic-contract service allows the ADS-ground-user to request a periodic contract with the aircraft. It is a confirmed service, initiated by the ADS-ground-user.

5.1.3.5.1.2 The ADS-periodic-contract service shall contain primitives and parameters as presented in Table 5-3.

Table 5-3: ADS-periodic-contract service parameters

Parameter Name	*Req*	*Ind*	*Rsp*	*Cnf*
Aircraft Identity	M			
Class of Communication Service	U			
ICAO Facility Designation	C	C(=)		
ADS-C Message Set Version Number	U	C(=)		
Security Required	C			
ADS/IC Contract Data	M	M(=)		
Result			M	M(=)
ADS/IC Positive Acknowledgement			C	C(=)
ADS/IC Report Data			C	C(=)
ADS/IC Reject Data			C	C(=)
ADS/IC Non Compliance Data			C	C(=)

5.1.3.5.2 Aircraft Identity parameter

5.1.3.5.2.1 This parameter contains the 24 bit ICAO address of the aircraft or the aircraft identification with which the contract is being made.

5.1.3.5.2.2 When an aircraft address is used to identify the aircraft, the *Aircraft Identity* parameter value shall

conform to an abstract value corresponding to a 24-bit ICAO aircraft address.

5.1.3.5.2.3 When an aircraft identification is used to identify the aircraft, the *Aircraft Identity* parameter value shall conform to an abstract value corresponding to a 2 to 7 character aircraft identification.

5.1.3.5.3 Class of Communication Service parameter

5.1.3.5.3.1 This parameter contains the value of the required class of communication service, if specified by the ADS-ground-user.

5.1.3.5.3.2 Where spcified by the ADS-ground-user, the *Class Of Communication Service* parameter shall have one of the following abstract values: “A”, “B”, “C”, “D”, “E”, “F”, “G” or “H”.

5.1.3.5.3.3 If contracts are currently in place, the *Class Of Communication Service* parameter is not used by the ADS-C service provider.

5.1.3.5.3.4 Where not specified by the ADS-ground-user, when there are no contracts already in force, this indicates that there will be no routing preference.

5.1.3.5.4 ICAO Facility Designation parameter

5.1.3.5.4.1 This parameter contains the 4 to 8 character ICAO Facility Designation of the ICAO facility which is initiating the contract. If contracts are currently in place, this parameter is not used by the ADS-service-provider.

5.1.3.5.4.2 The *ICAO Facility Designation* parameter value shall conform to an abstract value corresponding to a 4 to 8 character ICAO Facility Designation.

5.1.3.5.4.3 The *ICAO Facility Designation* parameter value shall be provided when the ADS-ground-user has no other contracts in place with the aircraft.

5.1.3.5.5 ADS-C Message Set Version Number parameter

5.1.3.5.5.1 This parameter contains the version number of the ADS-ASE and identifies the user data abstract syntax.

5.1.3.5.5.2 When provided by the ADS-user, the *ADS-C Message Set Version Number* parameter shall conform to an abstract integer value from 1 to 255.

5.1.3.5.6 Security Required parameter

5.1.3.5.6.1 This parameter contains the value of the required level of security, if specified by the ADS-ground-user.

5.1.3.5.6.2 If the received *Security Required* parameter is not as expected per the local security policy, the receiving ADS-air-ASE will abort.

5.1.3.5.6.3 Where specified by the ADS-ground-user, the *Security Required* parameter shall have one of the following abstract values: “no security” or “secured exchange”.

5.1.3.5.6.4 If contracts are currently in place, the *Security Required* parameter is not used by the ADS-C service provider.

5.1.3.5.7 ADS/IC Contract Data parameter

5.1.3.5.7.1 This parameter contains the details of the periodic contract as requested by the ADS-ground-user.

5.1.3.5.7.2 This parameter contains the Application Message Integrity Check.

5.1.3.5.7.3 The *ADS/IC Contract Data* parameter value shall conform to the ASN.1 abstract syntax ICContractRequest.

5.1.3.5.8 Result parameter

5.1.3.5.8.1 This parameter indicates the extent to which the periodic contract request can be complied with:

a) If it has the value “rejected” it indicates that the periodic contract has been rejected and the *ADS/IC Reject Data* parameter gives reasons,

b) If it has the value “accepted”, it indicates that the periodic contract has been accepted and the *ADS/IC Report Data* parameter contains the report,

c) If it has the value “non compliance notification”, it indicates that only some parts of the contract can be complied with, and the *ADS/IC Non Compliance Data* parameter indicates which ones have been rejected,

d) If it has the value “positive acknowledgement”, it indicates that the contract has been accepted but cannot be satisfied immediately because the information is not available within the contract response time, and the *ADS/IC Positive Acknowledgement* parameter indicates the contract type.

5.1.3.5.8.2 The *Result* parameter value shall conform to one of the following abstract values: “accepted”, “rejected”, “positive acknowledgement”, or “non compliance notification”.

5.1.3.5.9 ADS/IC Positive Acknowledgement parameter

5.1.3.5.9.1 This parameter contains the details of the ADS-C positive acknowledgement.

5.1.3.5.9.2 This parameter contains an Application Message Integrity Check.

5.1.3.5.9.3 The *ADS/IC positive acknowledgement* parameter value shall conform to the ASN.1 abstract syntax *ICPositiveAck*.

5.1.3.5.9.4 The *ADS/IC positive acknowledgement* parameter value shall be present if and only if the *Result parameter* contains the abstract value “positive acknowledgement”.

5.1.3.5.10 ADS/IC Report Data parameter

5.1.3.5.10.1 This parameter contains the details of the ADS-C report.

5.1.3.5.10.2 This parameter contains an Application Message Integrity Check.

5.1.3.5.10.3 The *ADS/IC report data* parameter value shall conform to the ASN.1 abstract syntax *ICReport*.

5.1.3.5.10.4 The *ADS/IC report data* parameter value shall be present if and only if the *Result parameter* contains the abstract value “accepted”.

5.1.3.5.11 ADS/IC Reject Data parameter

5.1.3.5.11.1 This parameter contains the reason of the contract rejection.

5.1.3.5.11.2 This parameter contains an Application Message Integrity Check.

5.1.3.5.11.3 The *ADS/IC reject data* parameter value shall conform to the ASN.1 abstract syntax ICReject.

5.1.3.5.11.4 The *ADS/IC reject data* parameter value shall be present if and only if the *Result parameter* contains the abstract value “rejected”.

5.1.3.5.12 ADS/IC Non Compliance Data parameter

5.1.3.5.12.1 This parameter contains an indication of which optional information fields cannot be sent.

5.1.3.5.12.2 This parameter contains an Application Message Integrity Check.

5.1.3.5.12.3 The *ADS/IC non compliance data* parameter value shall conform to the ASN.1 abstract syntax ICNonCompliance.

5.1.3.5.12.4 The *ADS/IC non compliance data* parameter value shall be present if and only if the *Result parameter* contains the abstract value “non compliance notification”.

5.1.3.6 ADS-report Service

5.1.3.6.1 General

5.1.3.6.1.1 The ADS-report service allows the ADS-air-user to send an ADS-C report to the ADS-ground-user. This is an unconfirmed service, initiated by the ADS-air-user.

5.1.3.6.1.2 The ADS-report service shall contain primitives and parameters as contained in Table 5-4.

Table 5-4: ADS-report service parameters

Parameter Name	*Req*	*Ind*
Contract type	M	M(=)
ADS/IC Report Data	M	M(=)

5.1.3.6.2 Contract Type parameter

5.1.3.6.2.1 This parameter identifies the type of contract that this report is in response to.

5.1.3.6.2.2 The *contract type* parameter value shall contain a value conforming to the abstract syntax *ContractType*.

5.1.3.6.3 ADS/IC Report Data parameter

5.1.3.6.3.1 This parameter contains the details of the ADS-C report.

5.1.3.6.3.2 This parameter contains the Application Message Integrity Check.

5.1.3.6.3.3 The *ADS/IC Report Data* parameter value shall conform to the ASN.1 abstract syntax ICReport.

5.1.3.7 ADS-cancel Service

5.1.3.7.1 General

5.1.3.7.1.1 The ADS-cancel service allows the ADS-ground-user to cancel an existing ADS-C contract. It is a confirmed service, initiated by the ADS-ground-user.

5.1.3.7.1.2 The ADS-cancel service shall contain primitives and parameters as contained in Table 5-5.

Table 5-5: ADS-cancel service parameters

Parameter Name	*Req*	*Ind*	*Cnf*
Contract type	M	M(=)	M(=)

5.1.3.7.2 Contract Type parameter

5.1.3.7.2.1 This parameter identifies the type of ADS-C contract that is to be cancelled.

5.1.3.7.2.2 The *Contract type* parameter value shall conform to the ASN.1 abstract syntax *CancelContract*.

5.1.3.8 ADS-cancel-all-contracts Service

5.1.3.8.1 General

5.1.3.8.1.1 The ADS-cancel-all-contracts service allows the ADS-ground-user to cancel all ADS-C contracts with a particular aircraft. It is a confirmed service, initiated by the ADS-ground-user.

5.1.3.8.1.2 The ADS-cancel-all-contracts service shall contain primitives as contained in Table 5-6.

Table 5-6: ADS-cancel-all-contracts service parameters

Parameter Name	*Req*	*Ind*	*Cnf*
None			

5.1.3.9 ADS-user-abort Service

5.1.3.9.1 General

5.1.3.9.1.1 The ADS-user-abort service allows the ADS-air-user to abort all ADS-C contracts with a particular ground system or ADS-ground-user to abort all ADS-C contracts with a particular aircraft. It is an unconfirmed service, initiated by an ADS-ground-user or the ADS-air-user. Messages in transit may be lost during this operation. It can be invoked at any time that the ADS-user is aware that any ADS-C service is in operation.

5.1.3.9.1.2 If the service is invoked prior to complete establishment of the dialogue, the ADS-user-abort indication may not be provided. An ADS-provider-abort indication may result instead.

5.1.3.9.1.3 The ADS-user-abort service shall contain primitives as contained in Table 5-7.

Table 5-7: ADS-user-abort service parameters

Parameter Name	*Req*	*Ind*
Reason	U	M

5.1.3.9.2 Reason parameter

5.1.3.9.2.1 This parameter is used to indicate a reason for aborting the ADS-C contracts.

5.1.3.9.2.2 If provided by the ADS-user, the parameter indicated to the peer ADS-user is that provided by the ADS-user, else it is what the ASE supplies.

5.1.3.9.2.3 The *reason* parameter value shall conform to the ASN.1 abstract syntax *UserAbortReason*.

5.1.3.9.2.4 When the *reason* parameter is provided by the ADS-user, the same value shall be indicated to the peer ADS-user.

5.1.3.10 ADS-provider-abort Service

5.1.3.10.1 General

5.1.3.10.1.1 The ADS-provider-abort service allows the ADS-service-provider to inform the ADS-ground-user and the ADS-air-user that it can no longer provide the ADS-C service for a particular ADS-ground-user - ADS-air-user pairing. It is initiated by the ADS-service-provider. Messages in transit may be lost during this operation.

5.1.3.10.1.2 The ADS-provider-abort service shall contain primitives and parameters as contained in Table 5-8.

Table 5-8: ADS-provider-abort service parameters

Parameter Name	*Ind*
Reason	M

5.1.3.10.2 Reason parameter

5.1.3.10.2.1 This parameter identifies the reason for the provider abort.

5.1.3.10.2.2 The *reason* parameter shall conform to the ASN.1 abstract syntax *ProviderAbortReason*.

5.1.4 Formal Definitions of Messages

5.1.4.1 Encoding/Decoding Rules

5.1.4.1.1 An ADS-air-ASE shall be capable of encoding [ADSAircraftPDUs] APDUs and decoding [ADSGroundPDUs] APDUs.

5.1.4.1.2 An ADS-ground-ASE shall be capable of encoding [ADSGroundPDUs] APDUs and decoding [ADSAircraftPDUs] APDUs.

5.1.4.2 ADS-C ASN.1 Abstract Syntax

5.1.4.2.1 The abstract syntax of the air-ground ADS-C protocol data units shall comply with the description contained in the ASN.1 module *PMADSCAPDUVersion1* (conforming to ISO/IEC 8824), as defined below.

```
PMADSCAPDUVersion1 DEFINITIONS ::=

BEGIN

-- ------------------------------------------------------------------------------------------------------------
-- Aircraft-generated and Ground-generated Message Choice
-- ------------------------------------------------------------------------------------------------------------
ADSAircraftPDUs ::= SEQUENCE
{
        timestamp                                       [0]     DateTimeGroup,
        adsAircraftPdu                                  [1]     ADSAircraftPDU
}
```

```
ADSAircraftPDU ::= CHOICE
{
        aDS-report-PDU                              [0]     Report,
        aDS-accepted-PDU                            [1]     Report,
        aDS-rejected-PDU                            [2]     Reject,
        aDS-ncn-PDU                                 [3]     NonCompliance,
        aDS-positive-acknowledgement-PDU            [4]     PositiveAcknowledgement,
        aDS-cancel-positive-acknowledgement-PDU     [5]     RequestType,
        aDS-cancel-negative-acknowledgement-PDU     [6]     CancelRejectReason,
        aDS-provider-abort-PDU                      [7]     ProviderAbortReason,
        aDS-user-abort-PDU                          [8]     UserAbortReason,
        ...
}

ADSGroundPDUs ::= SEQUENCE
{
        timestamp                                   [0]     DateTimeGroup,
        adsGroundPdu                                [1]     ADSGroundPDU
}


ADSGroundPDU ::= CHOICE
{
        aDS-cancel-all-contracts-PDU                [0]     NULL,
        aDS-cancel-contract-PDU                     [1]     CancelContract,
        aDS-contract-PDU                            [2]     ContractRequest,
        aDS-provider-abort-PDU                      [3]     ProviderAbortReason,
        aDS-user-abort-PDU                          [4]     UserAbortReason,
        ...
}

-- ------------------------------------------------------------------------------------------------------------------
-- Ground-generated and Aircraft-generated message components - Protocol Data Units
-- ------------------------------------------------------------------------------------------------------------------
ContractRequest ::= SEQUENCE
{
        contract-type                               [0] ContractType,
        ic-contract-request                         [1] ICContractRequest
}

ICContractRequest ::= SEQUENCE
{
        algorithmIdentifier                         [0] AlgorithmIdentifier OPTIONAL,
        aDSMessage                                  [1] ADSMessage,
                                                    -- PER encoded User Data ADSRequestContract
        integrityCheck                              [2] BIT STRING,
        ...
}

ICPositiveAck ::= SEQUENCE
{
```

```
        algorithmIdentifier                 [0] AlgorithmIdentifier OPTIONAL,
        aDSPositiveAck                      [1] ADSMessage,
                                            -- PER encoded User Data ADSPositiveAcknowledgement
        integrityCheck                      [2] BIT STRING,
        …
}

PositiveAcknowledgement ::= SEQUENCE
{
        contract-type                       [0]     ContractType,
        ic-positive-ack                     [1]     ICPositiveAck
}


Report ::= SEQUENCE
{
        contract-type                       [0] ContractType,
        ic-report                           [1] ICReport
}

ICReport ::= SEQUENCE
{
        algorithmIdentifier                 [0] AlgorithmIdentifier OPTIONAL,
        aDSMessage                          [1] ADSMessage,
                                            -- PER encoded User Data ADSReport
        integrityCheck                      [2] BIT STRING,
        ...
}

Reject ::= SEQUENCE
{
        contract-type                       [0] ContractType,
        ic-reject                           [1] ICReject
}

ICReject ::= SEQUENCE
{
        algorithmIdentifier                 [0] AlgorithmIdentifier OPTIONAL,
        aDSMessage                          [1] ADSMessage,
                                            -- PER encoded User Data ADSReject
        integrityCheck                      [2] BIT STRING,
        ...
}

NonCompliance ::= SEQUENCE
{
        contract-type                       [0] ContractType,
        ic-ncn                              [1] ICNonCompliance
}

ICNonCompliance ::= SEQUENCE
```

```
{
        algorithmIdentifier                 [0] AlgorithmIdentifier OPTIONAL,
        aDSMessage                          [1] ADSMessage,
                                            -- PER encoded User Data ADSNonCompliance
        integrityCheck                      [2] BIT STRING,
        ...
}

CancelContract ::= ENUMERATED
{
        event-contract                      (0),
        periodic-contract                   (1),
        ...
}


AlgorithmIdentifier ::= RELATIVE-OID -- root is {icao-arc atn-algorithms(9)}

ADSMessage ::= BIT STRING

RequestType ::= ENUMERATED
{
        cancel-event-contract               (0),
        cancel-periodic-contract            (1),
        cancel-all-contracts                (2),
        ...
}

CancelRejectReason ::= SEQUENCE
{
        requestType                         [0]     RequestType,
        rejectReason                        [1]     RejectReason,
        …
}

RejectReason ::=                ENUMERATED
{
        no-such-contract                    (0),
        …
}

ContractType ::= ENUMERATED
{
        event-contract                      (0),
        periodic-contract                   (1),
        demand-contract                     (2)
}

ProviderAbortReason ::= ENUMERATED
{
        communications-service-failure      (0),
        unrecoverable-system-error          (1),
```

```
        invalid-PDU                     (2),
        sequence-error                  (3),
        timer-expiry                    (4),
        cannot-establish-contact        (5),
        undefined-error                 (6),
        dialogue-end-not-accepted       (7),
        unexpected-PDU                  (8),
        decoding-error                  (9),
        invalid-qos-parameter           (10),
        ...
}

UserAbortReason ::= ENUMERATED
        {
        undefined                       (0),
        unknown-integrity-check         (1),
        validation-failure              (2),
        unable-to-decode-message        (3),
        ...
        }
DateTimeGroup ::= SEQUENCE
{
        date                            Date,
        time                            Time
}

Date ::= SEQUENCE
{
        year                            DateYear,
        month                           DateMonth,
        day                             DateDay
}

DateYear ::= INTEGER (1996..2095)
        -- unit = year
        -- Range = 1996 to 2095

DateMonth ::= INTEGER (1..12)
        -- unit = month
        -- Range = January to December

DateDay ::= INTEGER (1..31)
        -- unit = day
        -- Range = 1 to 31

Time ::= SEQUENCE
{
        timeHours                       [0] TimeHours,
        timeMinutes                     [1] TimeMinutes,
        timeSeconds                     [2] TimeSeconds OPTIONAL
}
```

TimeHours ::= INTEGER (0..23)
-- Unit = hours
-- Range = midnight to 23.00 (11 PM)

TimeMinutes ::= INTEGER (0..59)
-- Unit = minutes
-- Range = 0 minutes to 59 minutes

TimeSeconds ::= INTEGER (0..59)
-- Unit = seconds
-- Range = 0 seconds to 59 seconds

END

5.1.5 Protocol Definition

5.1.5.1 Sequence Rules

5.1.5.1.1 Only the sequence of primitives illustrated in Figure 6-2 to Figure 5-24 shall be permitted.

5.1.5.1.2 The following figures define the valid sequences of primitives that are possible to be invoked during the operation of the ADS-C application over the ATN Dialogue. They show the relationship in time between the service request and the resulting indication, and if applicable, the subsequent response and the resulting confirmation.

5.1.5.1.3 Abort primitive may interrupt and terminate any of the normal message sequences outlined below.

5.1.5.1.4 Primitives are processed in the order in which they are received.

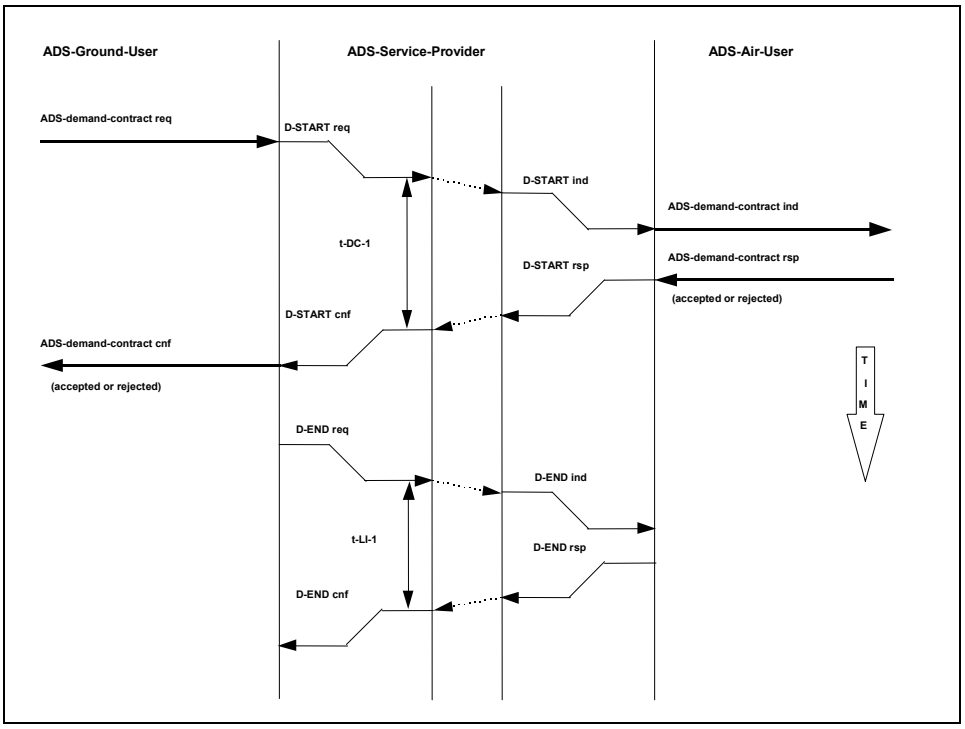


Figure 5-2: Use of demand contract (accepted or rejected) with no dialogue existing

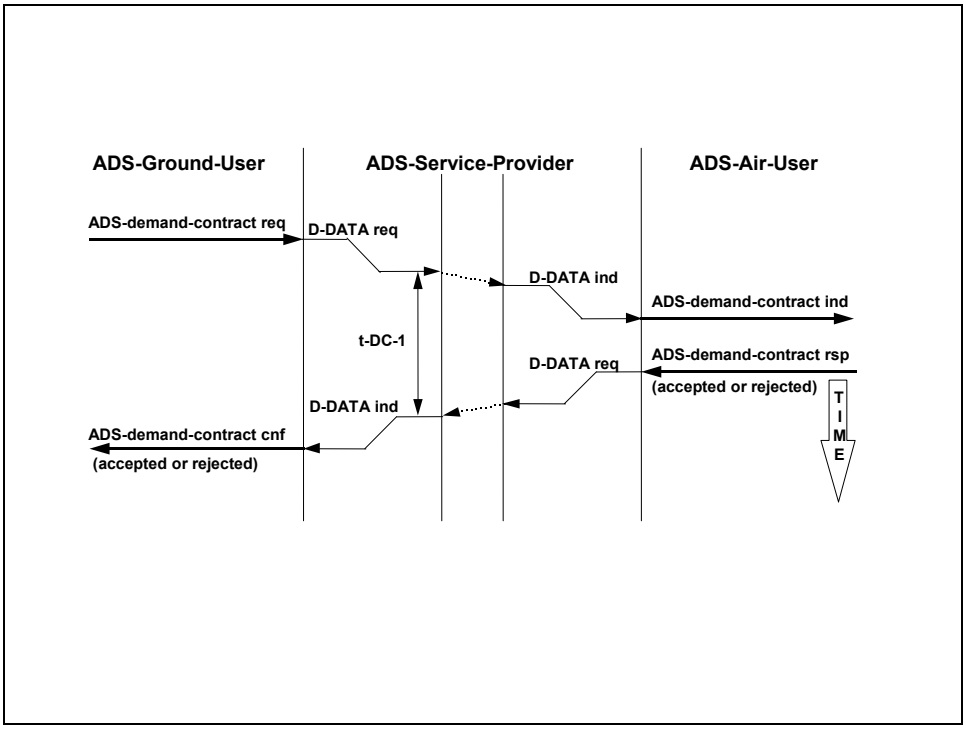


Figure 5-3: Use of demand contract (accepted or rejected) with dialogue existing

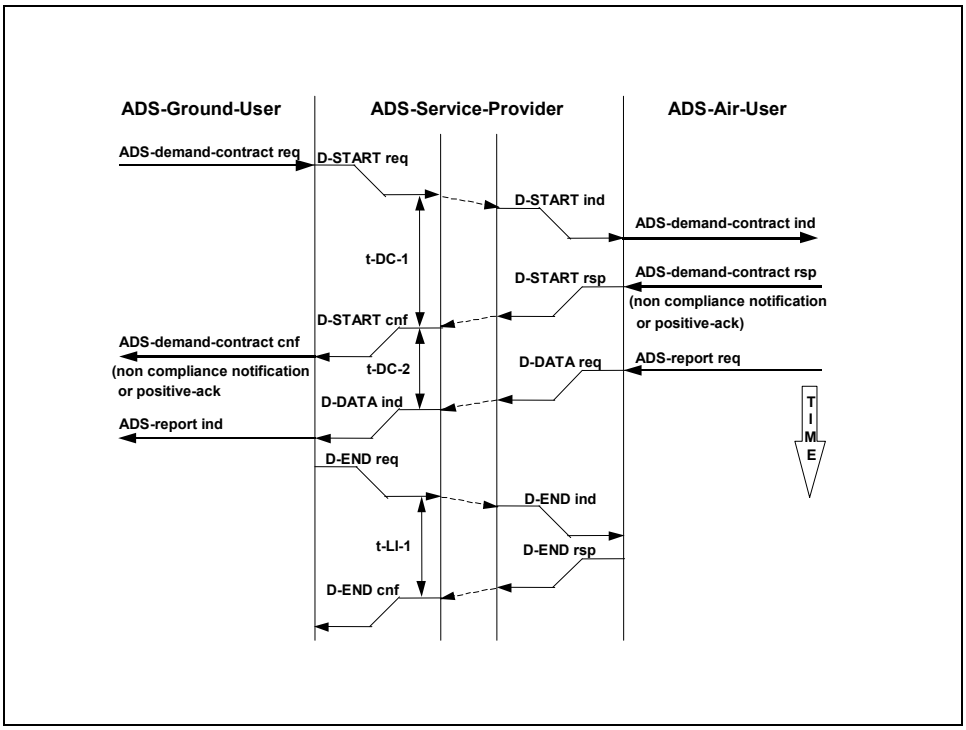


Figure 5-4: Use of demand contract (non compliance notification or

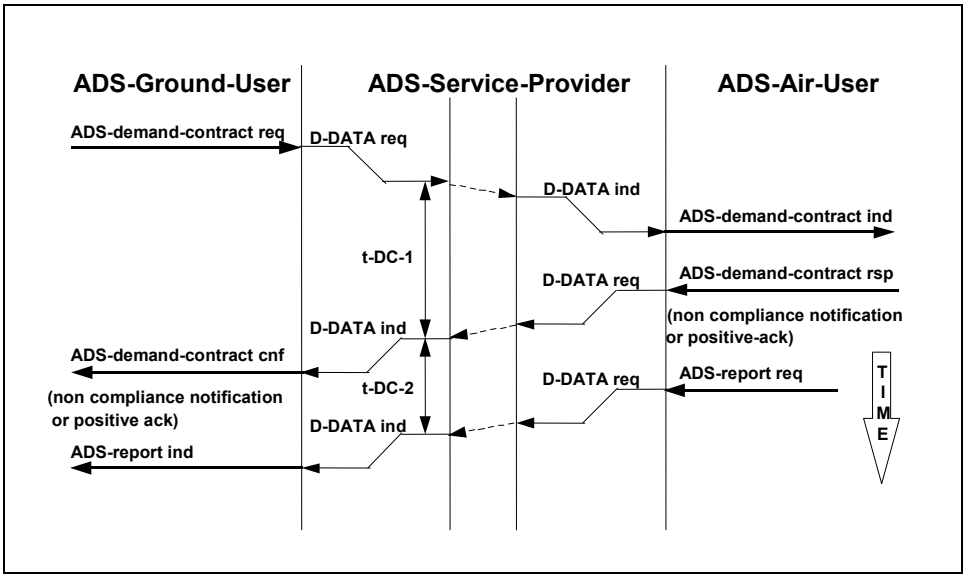


Figure 5-5: Use of demand contract (non compliance notification or positive acknowledgement) with dialogue existing

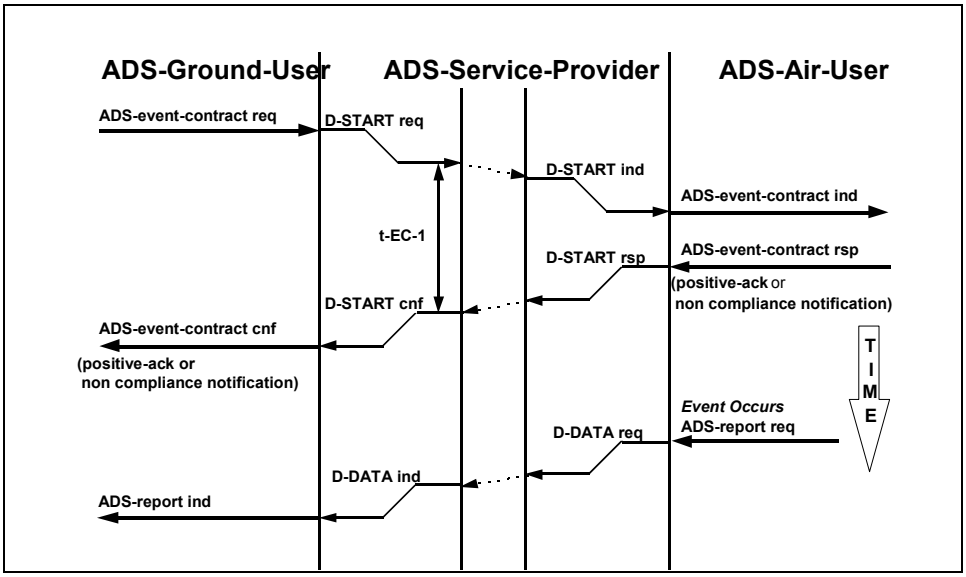


Figure 5-6: Use of event contract (positive acknowledgement or non compliance notification)

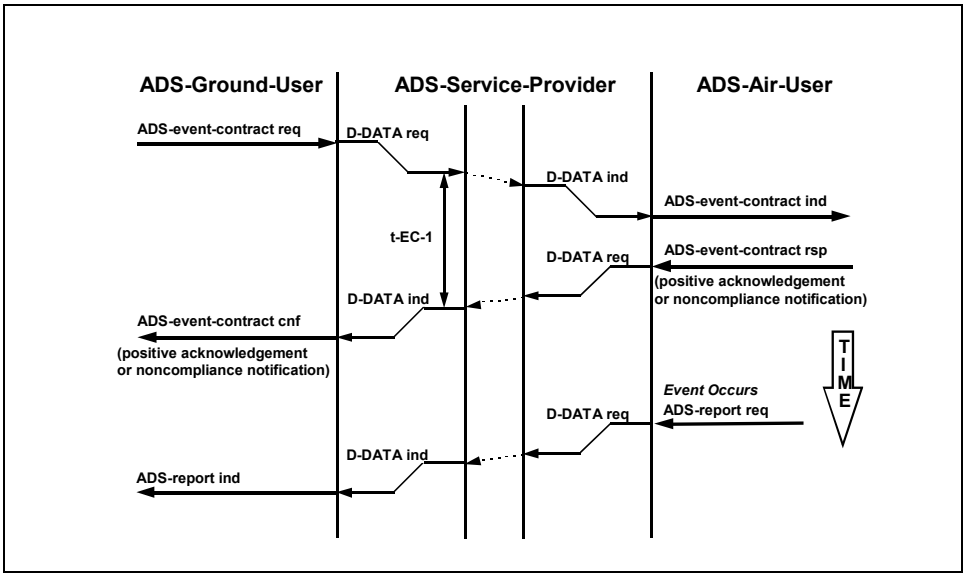
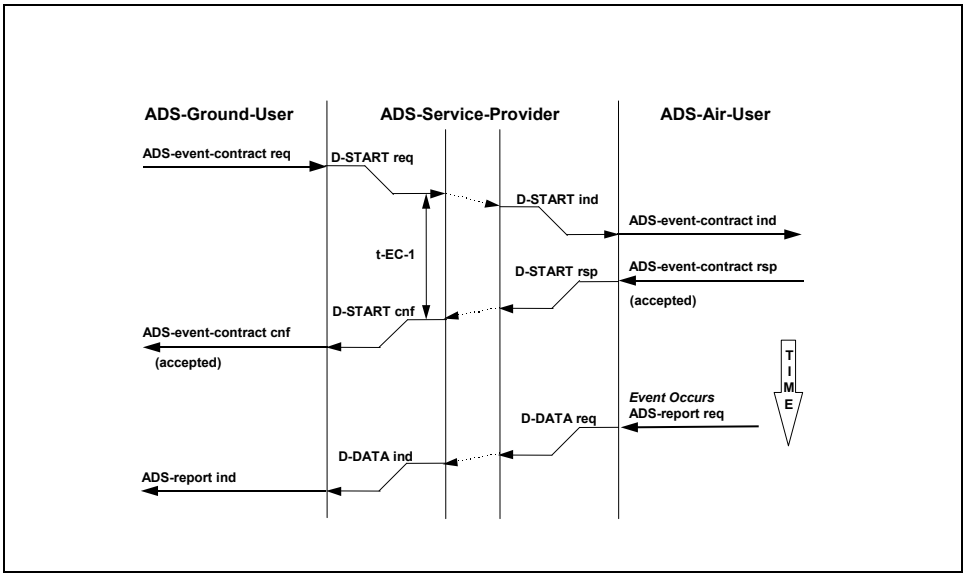


Figure 5-7: Use of event contract (positive acknowledgement or non compliance notification) with dialogue existing



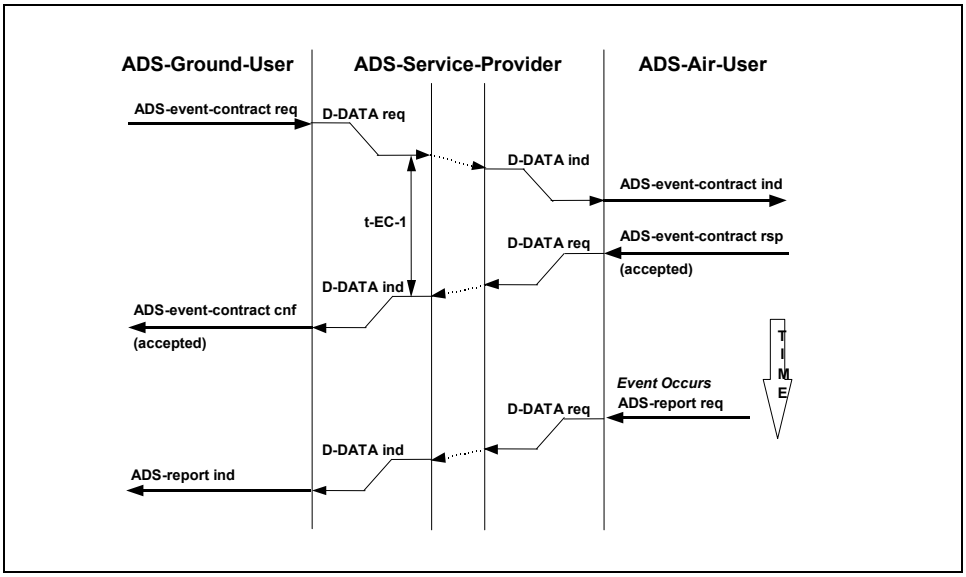


Figure 5-9: Use of event contract (accepted) with dialogue existing

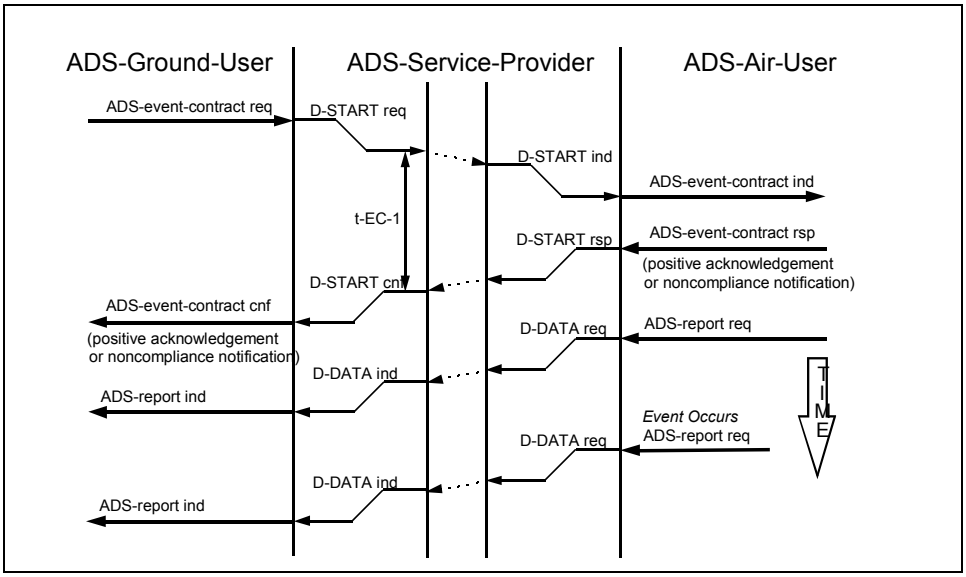


Figure 5-10: Use of event contract (positive acknowledgement or

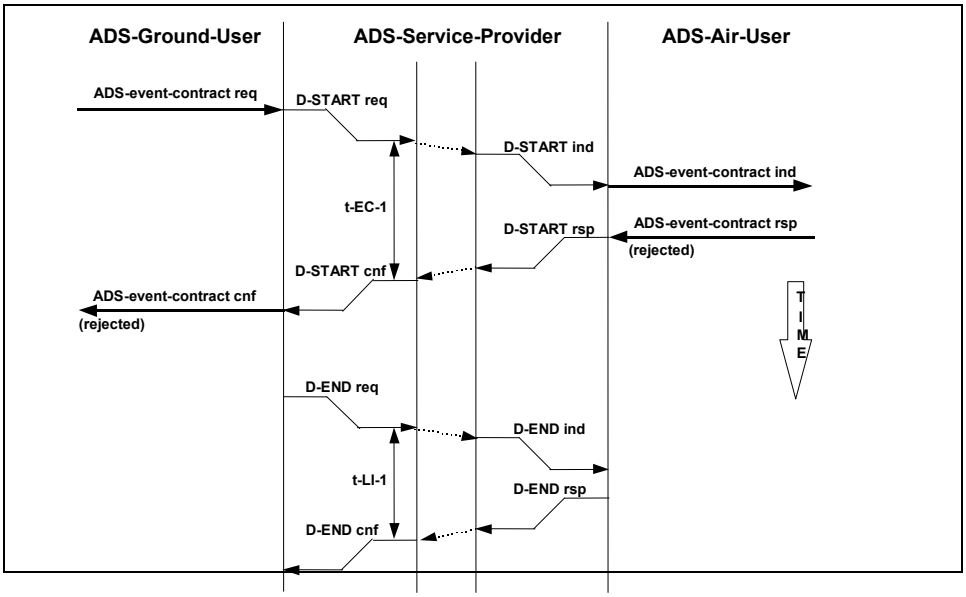
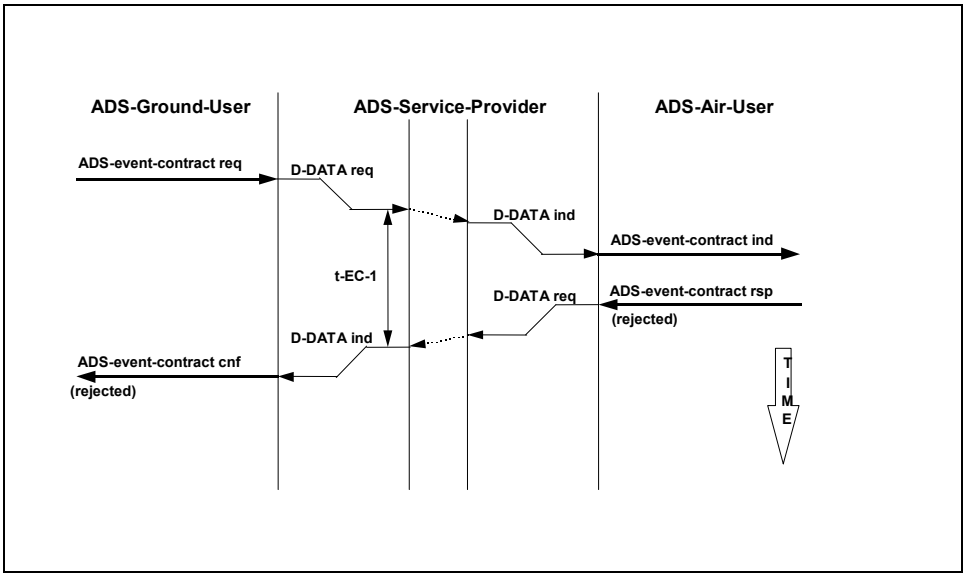


Figure 5-11: Use of event contract (rejected)
with no dialogue existing



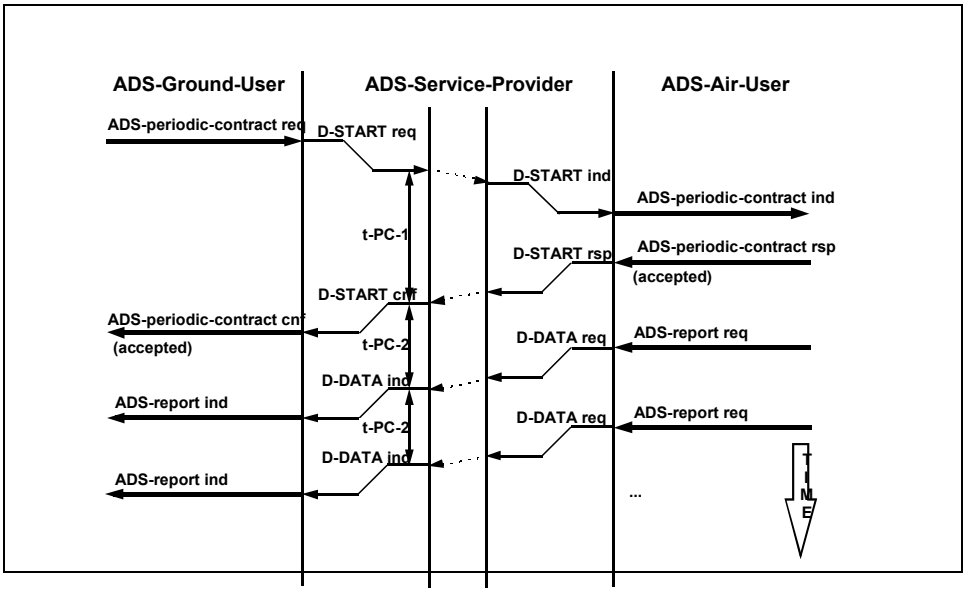
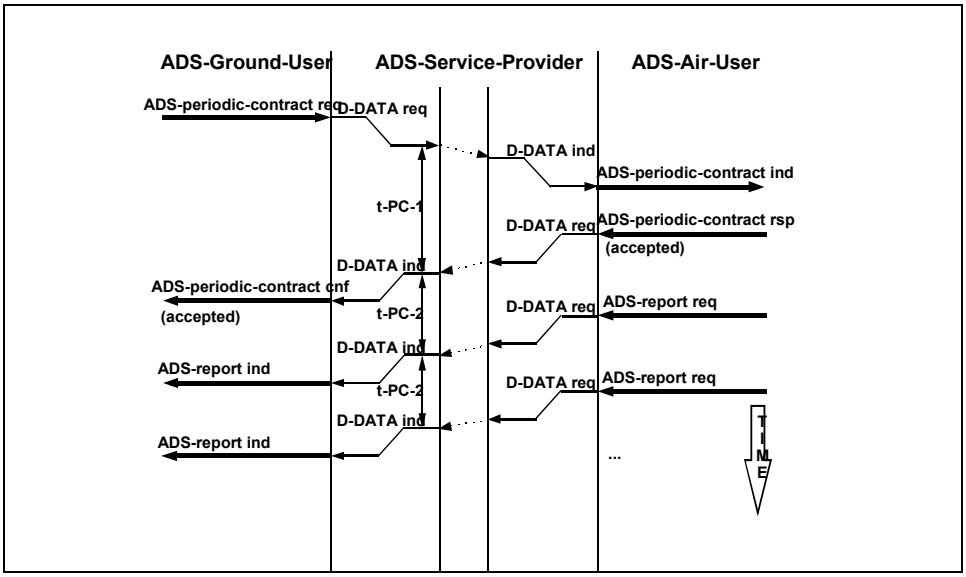


Figure 5-13: Use of periodic contract (accepted) with no dialogue existing



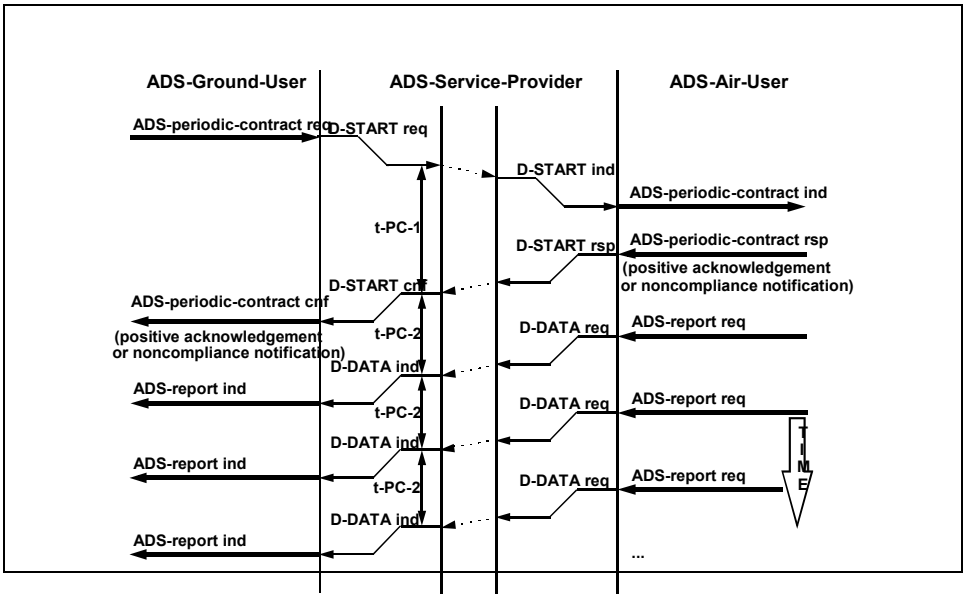


Figure 5-15: Use of periodic contract (positive acknowledgement or non compliance notification) with no dialogue existing

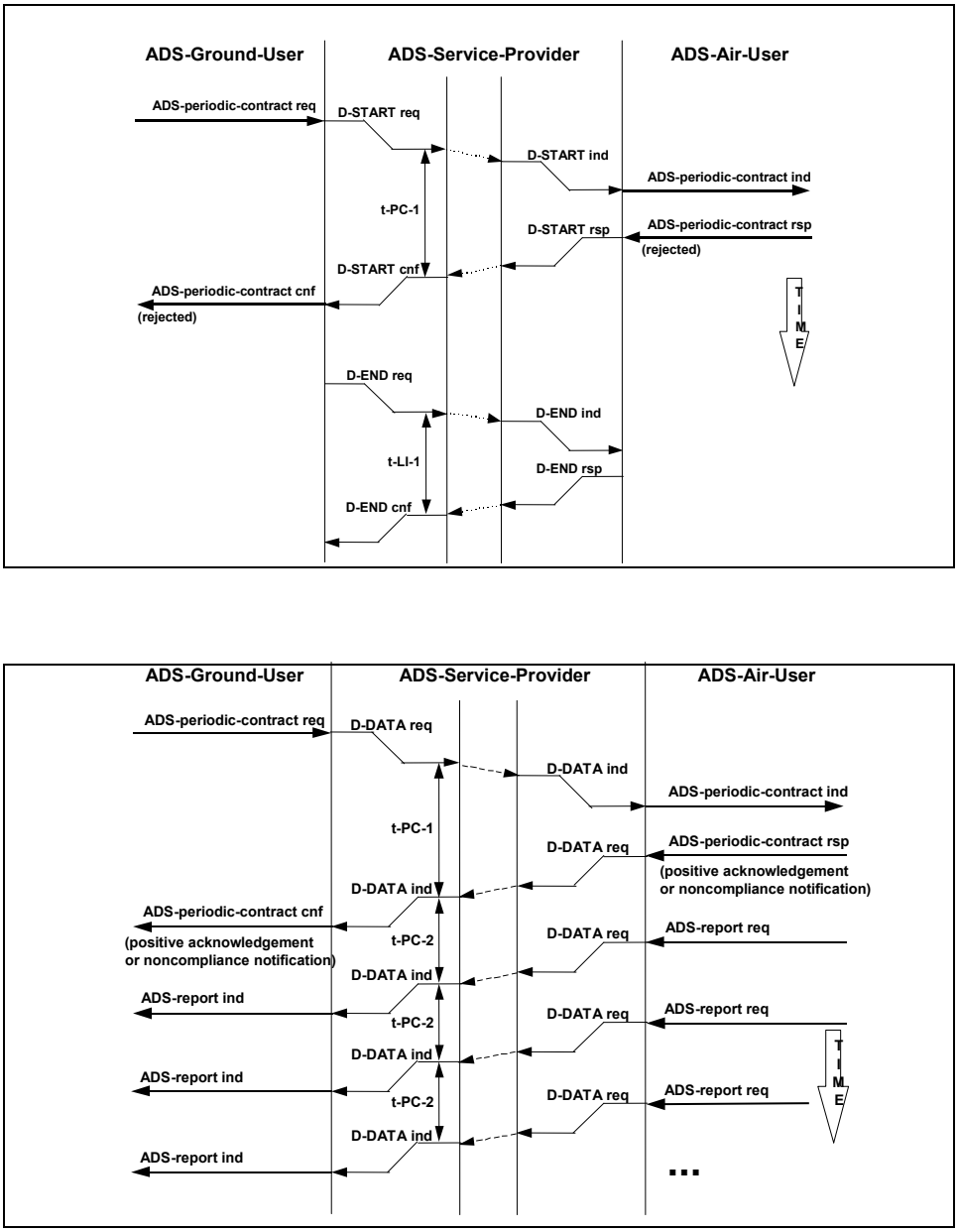
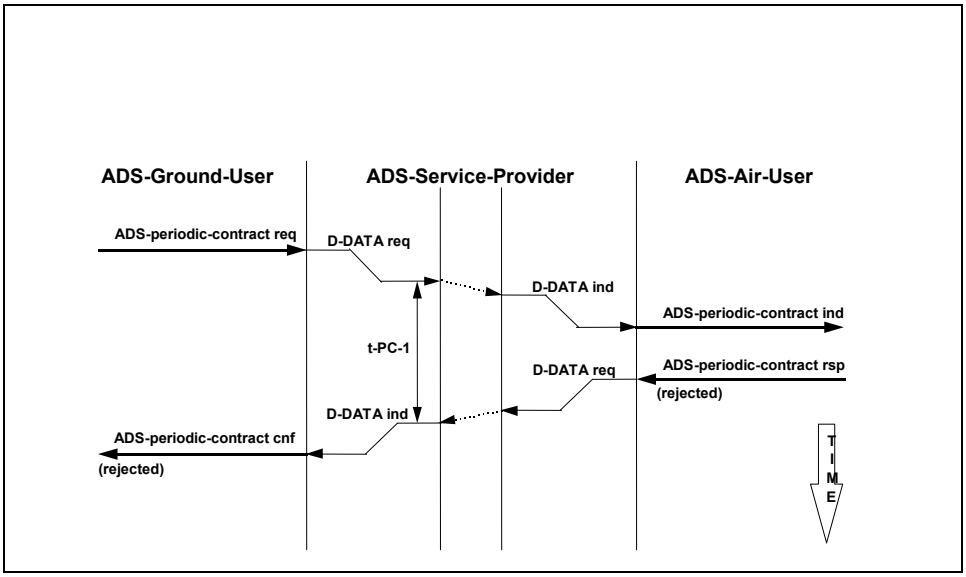


Figure 5-17: Use of periodic contract (positive acknowledgement or non compliance notification) with dialogue existing



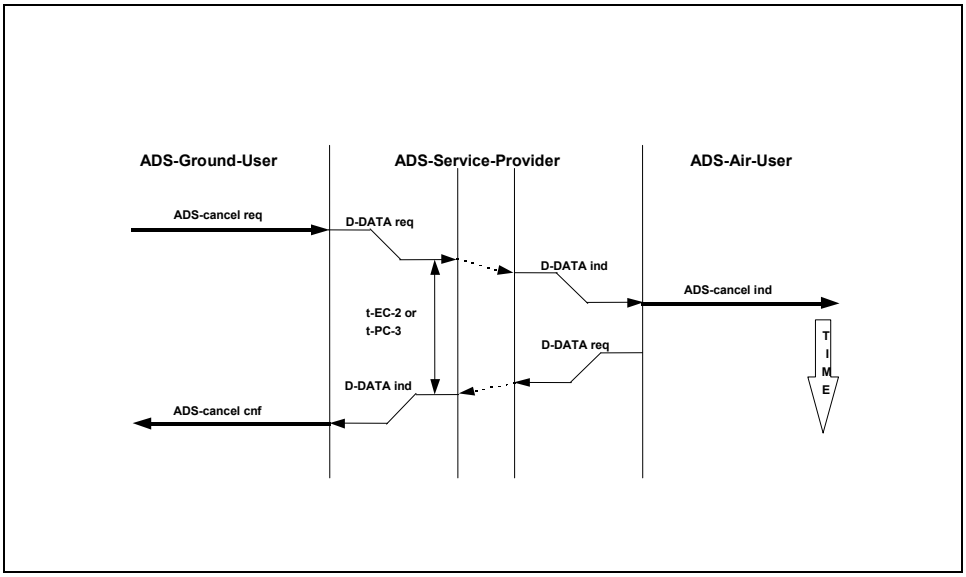


Figure 5-19: Use of ADS-cancel contract service

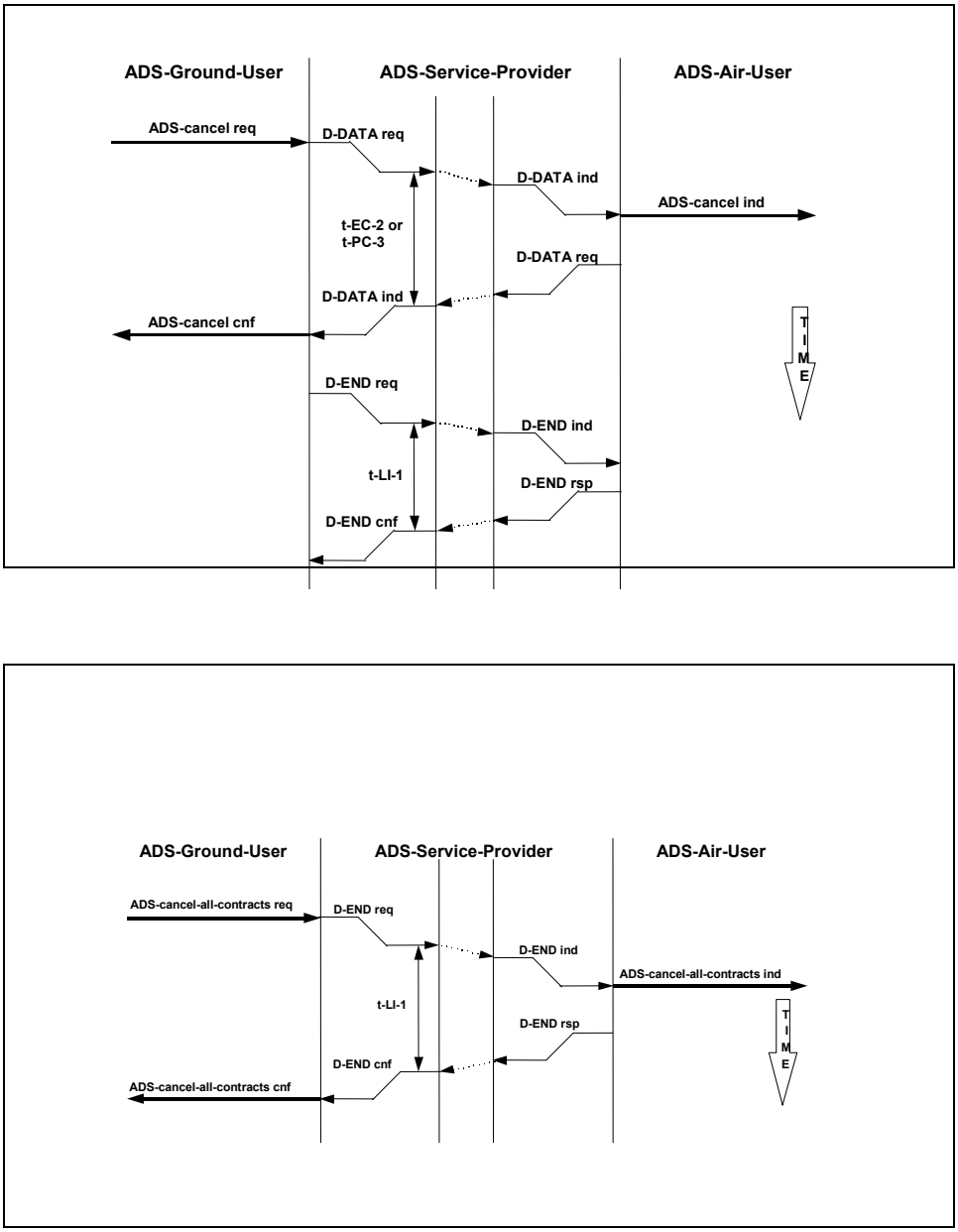
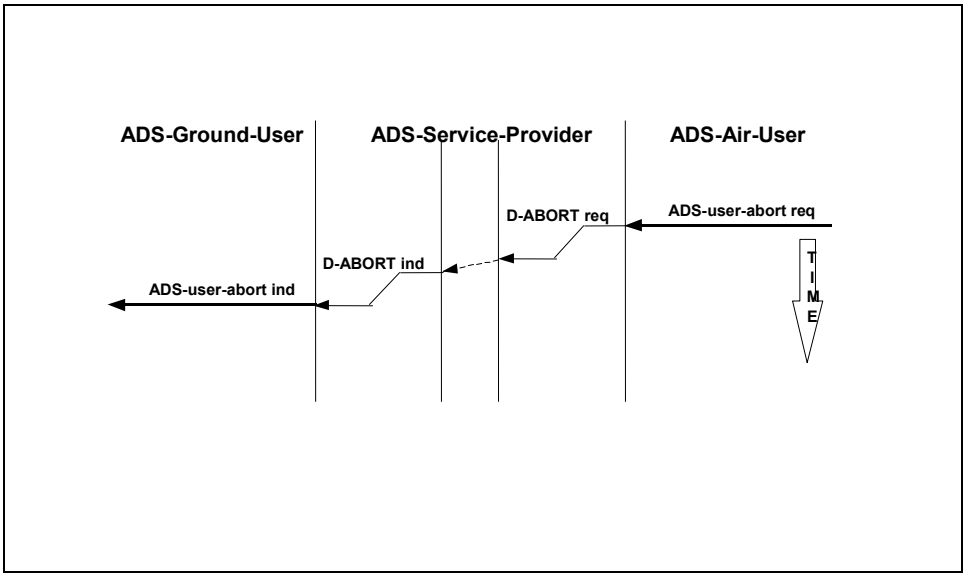


Figure 5-21: Use of ADS-cancel-all-contracts service



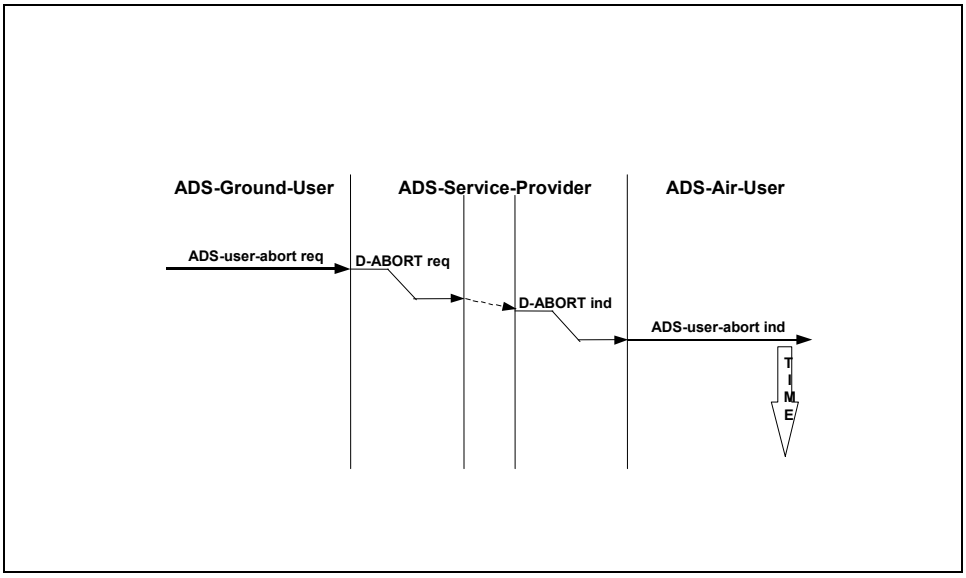
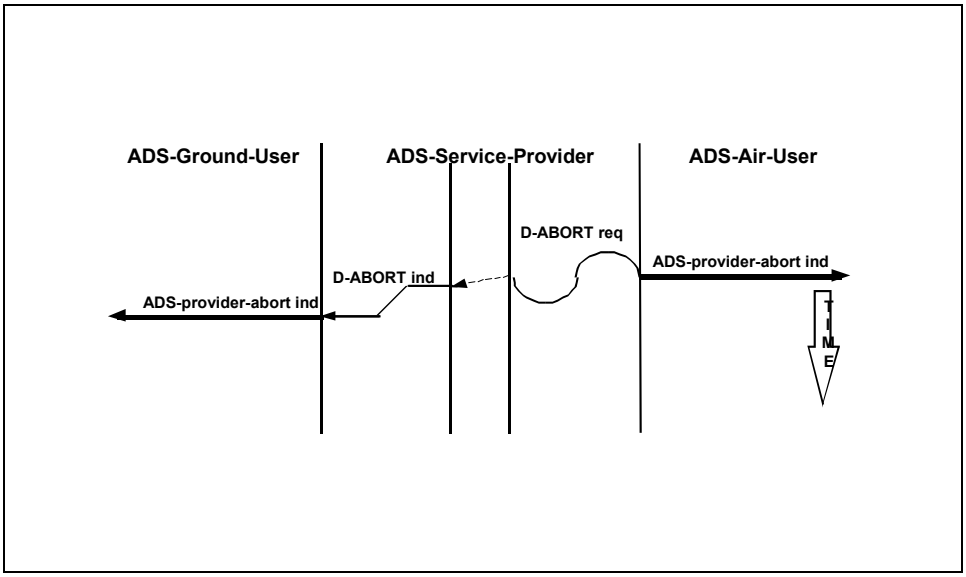


Figure 5-23: Ground user abort service



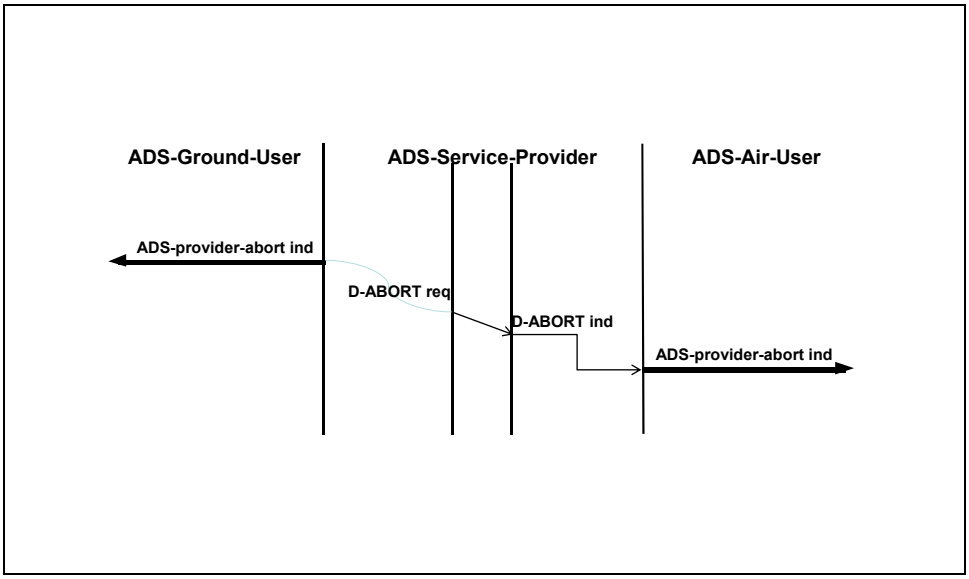


Figure 5-25: Ground ASE abort

5.1.5.2 ADS-C Service Provider Timers

5.1.5.2.1 The ADS-C ASE shall be capable of detecting when a timer expires.

5.1.5.2.2 Table 5-9 lists the time constraints related to the ADS-C application. Each time constraint requires a timer to be set in the ADS-C protocol machine.

5.1.5.2.3 If the timer expires before the final event has occurred, the ADS-C ASE takes the appropriate action as defined in 0.

5.1.5.2.4 The timer values should be as indicated in Table 5-9.

Table 5-9: ADS-C Service Provider Timers

ADS-C service	*Timer*	*Timer Value*	*Timer Start Event*	*Timer Stop Event*
ADS-demand-contract	t-DC-1	6 minutes	ADS-demand-contract request	ADS-demand-contract confirmation
	t-DC-2	3 minutes 30 seconds	ADS-demand-contract confirmation (pos-ack, ncn)	ADS-report indication
ADS-event-contract	t-EC-1	6 minutes	ADS-event-contract request	ADS-event-contract confirmation
	t-EC-2	6 minutes	ADS-cancel request	ADS-cancel-contract confirmation
ADS-periodic-contract	t-PC-1	6 minutes	ADS-periodic-contract request	ADS-periodic-contract confirmation
	t-PC-2	123 minutes (longest reporting rate + 3 minutes)	ADS-report indication or ADS-periodic-contract confirmation	ADS-report indication
	t-PC-3	6 minutes	ADS-cancel request	ADS-cancel-contract confirmation
General	t-LI-1	6 minutes	D-END request	D-END confirmation

Note.— The receipt of ADS-user-abort request, D-ABORT indication or D-P-ABORT indication are also timer stop events.

5.1.5.3 ADS-C ASE Protocol Description

5.1.5.3.1 Description

5.1.5.3.1.1 ADS-C ASE Functional Model

5.1.5.3.1.1.1 The ADS-air-ASE is functionally made of 6 modules as shown in Figure 5-26 and the ADS-ground-ASE is functionally made of a similar 6 modules as shown in Figure 5-27:

a) the ADS-C High Interface Module (ADS-C HI module) interfaces with the ASE-user through the abstract service interface as defined in 5.1.3.

b) the ADS-C Demand Contract Module (ADS-C DC module) manages all demand contracts with a single ground system.

c) the ADS-C Event Contract Module (ADS-C EC module) manages event contracts with a single ground system.

d) the ADS-C Periodic Contract Module (ADS-C PC module) manages periodic contracts with a single ground system.

e) the ADS-C Abort Module (ADS-C AB module) handles aborts in case of unrecoverable error.

f) the Low Interface Modules (LI modules) described in this document interfaces the ATN Dialogue Service Provider on behalf of the DC, EC and PC and AB modules.

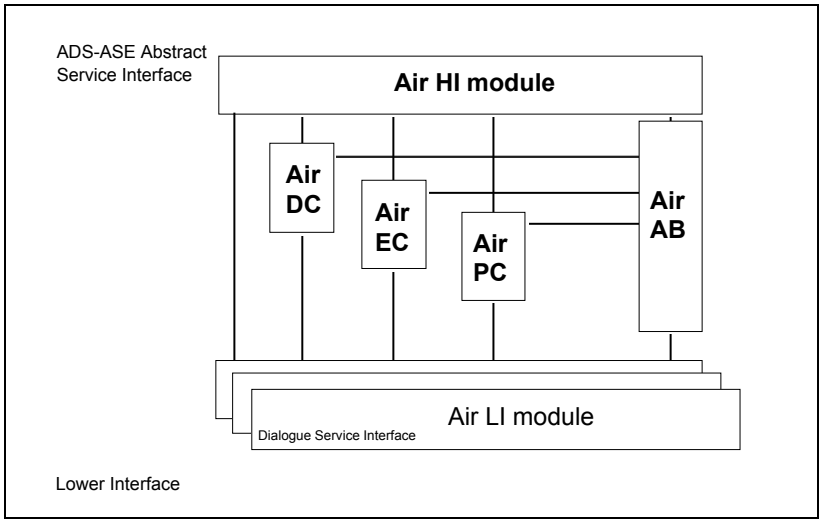


Figure 5-26: Functional model of the ADS-air-ASE

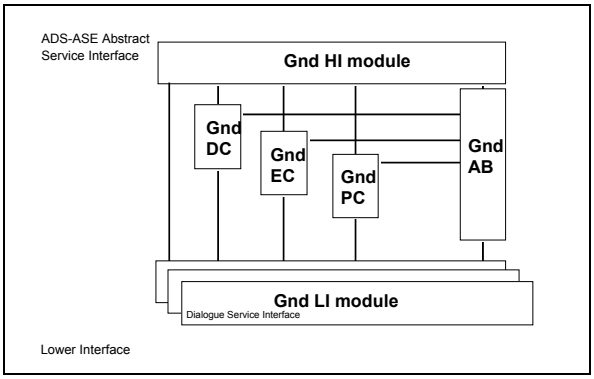


Figure 5-27: Functional model of the ADS-ground-ASE

5.1.5.3.1.1.2 There is no difference between the ADS-ground-ASE and the ADS-air-ASE functional models.

5.1.5.3.1.1.3 Section 5.1.5.3 describes the actions of the individual modules in both the air and ground systems. Section 5.1.5.5 contains state tables for the individual modules.

5.1.5.3.1.1.3 The ADS-ground-user is considered an active user from the time at which it invokes the first ADS-demand-contract request, an ADS-event-contract request or an ADS-periodic-contract request until such time that:

a) the ADS-ground-user receives an ADS-cancel-all-contracts confirmation,

b) the ADS-ground-user receives an ADS-cancel confirmation, and there are no other contracts in place,

c) the ADS-ground-user receives an ADS-demand-contract confirmation, an ADS-event-contract confirmation or an ADS-periodic-contract confirmation, with the *Result* parameter value set to “rejected”, and there are no other contracts in place,

d) the ADS-ground-user receives an ADS-demand-contract confirmation with the *Result* parameter value set to “accepted” an ADS-report indication, with the Contract type parameter value set to “demand contract”, and there are no other contracts in place,

e) the ADS-ground-user receives an ADS-user-abort indication,

f) the ADS-ground-user receives an ADS-provider-abort indication, or

g) the ADS-ground-user invokes an ADS-user-abort request.

5.1.5.3.1.1.4 The ADS-air-user is considered an active user from the time at which it receives the first ADS-demand-contract indication, an ADS-event-contract indication or an ADS-periodic-contract indication until such time that:

a) the ADS-air-user receives an ADS-cancel-all-contracts indication,

b) the ADS-air-user receives an ADS-cancel indication, and there are no other contracts in place,

c) the ADS-air-user invokes an ADS-demand-contract response, an ADS-event-contract response or an ADS-periodic-contract response, with the *Result* parameter value set to “rejected”, and there are no other contracts in place,

d) the ADS-air-user invokes an ADS-demand-contract response with the *Result* parameter value set to “accepted” an ADS-report request, with the *Contract type* parameter value set to “demand contract”, and there are no other contracts in place,

e) the ADS-air-user receives an ADS-user-abort indication,

f) the ADS-air-user receives an ADS-provider-abort indication, or

g) the ADS-air-user invokes an ADS-user-abort request.

5.1.5.3.2 In section 5.1.5.3, if no actions are described for an ADS-C service primitive in a particular state, then the invocation of that service primitive shall be prohibited in that state.

5.1.5.3.3 Possible errors arising upon Receipt of an APDU or a Dialogue Service Primitive.

5.1.5.3.3.1 If an APDU is not received when one is required, or one is received in an inappropriate dialogue service primitive, then exception handling procedures as described in 5.1.5.4.3 shall apply.

5.1.5.3.3.2 Upon receipt of an APDU or dialogue service primitive, if no actions are described for their arrival when in a particular state, then exception handling procedures as described in 5.1.5.4.4 shall apply.

5.1.5.3.3.3 Upon receipt of an APDU that cannot be decoded, then exception handling procedures as described in 5.1.5.4.7 shall apply.

5.1.5.3.4 Ground ADS-C HI Module

5.1.5.3.4.1 Upon receipt of a service primitive, the ADS-C HI module shall pass it to the module as shown in Table 5-10.

Table 5-10: Request and response primitive to ground module mapping

Service Primitive	*ADS-ground-ASE Module*
ADS-demand-contract request	ADS-C DC
ADS-event-contract request	ADS-C EC
ADS-periodic-contract request	ADS-C PC
ADS-cancel request with contract type “event-contract”	ADS-C EC
ADS-cancel request with contract type “periodic-contract”	ADS-C PC
ADS-cancel-all-contracts request	ADS-C LI
ADS-user-abort request	ADS-C AB

5.1.5.3.4.2 Upon receipt of a request to invoke a service primitive from one of the ground modules in the ADS-ground-ASE as shown in Table 5-11, the ADS-C ground HI module shall do so.

Table 5-11: Indication and confirmation primitive to ground module mapping

ADS-ground-ASE Module	*Service Primitive*
ADS-C DC	ADS-demand-contract confirmation
ADS-C EC	ADS-event-contract confirmation
ADS-C PC	ADS-periodic-contract confirmation
ADS-C DC	ADS-report indication with contract type “demand-contract”
ADS-C EC	ADS-report indication with contract type “event-contract”
ADS-C PC	ADS-report indication with contract type “periodic-contract”
ADS-C EC	ADS-cancel confirmation with contract type “event-contract”
ADS-C PC	ADS-cancel confirmation with contract type “periodic-contract”
ADS-C LI	ADS-cancel-all-contracts confirmation

5.1.5.3.4.3 On receipt of a request to invoke ADS-provider-abort indication from the ADS-C ground AB module, the ADS-C ground HI module shall:

a) if the ADS-ground-user is not an active user, take no further action;

b) if the ADS-ground-user is an active user, invoke ADS-provider-abort indication.

5.1.5.3.4.4 On receipt of a request to invoke ADS-user-abort indication from the ADS-C ground AB module, the ADS-C ground HI module shall:

a) if the ADS-ground-user is not an active user, take no further action;

b) if the ADS-ground-user is an active user, invoke ADS-user-abort indication.

5.1.5.3.4.5 The ADS-C ground HI module shall reject requests and responses, apart from ADS-user-abort requests, when the ground LI module is in the LI-G-START state or the LI-G-END state.

5.1.5.3.5 Air ADS-C HI Module

5.1.5.3.5.1 Upon receipt of a service primitive, the air HI module shall pass it to the air module as shown in Table 5-12

Table 5 -12: Request and response primitive to air module mapping

Service Primitive	*ADS-air-ASE Module*
ADS-demand-contract response	ADS-C DC
ADS-event-contract response	ADS-C EC
ADS-periodic-contract response	ADS-C PC
ADS-report request with contract type “event-contract”	ADS-C EC
ADS-report request with contract type “periodic-contract”	ADS-C PC
ADS-user-abort request	ADS-C AB

5.1.5.3.5.2 Upon receipt of a request to invoke a service primitive from one of the air modules in the ADS-air-ASE as shown in Table 5-13, the air HI module shall do so.

Table 5-13: Indication and confirmation primitive to air module mapping

ADS-air-ASE Module	*Service Primitive*
ADS-C DC	ADS-demand-contract indication
ADS-C EC	ADS-event-contract indication
ADS-C PC	ADS-periodic-contract indication
ADS-C EC	ADS-cancel indication with contract type “event-contract”
ADS-C PC	ADS-cancel indication with contract type “periodic-contract”
ADS-C LI	ADS-cancel-all-contracts indication

5.1.5.3.5.3 On receipt of a request to invoke ADS-provider-abort indication from the ADS-C air AB module, the air HI module shall:

a) if the ADS-air-user is not an active user, take no further action;

b) if the ADS-air-user is an active user, invoke ADS-provider-abort indication.

5.1.5.3.5.4 On receipt of a request to invoke ADS-user-abort indication from the ADS-C air AB module, the air HI module shall:

a) if the ADS-air-user is not an active user, take no further action;

b) if the ADS-air-user is an active user, invoke ADS-user-abort indication.

5.1.5.3.6 Ground ADS-C DC Module

5.1.5.3.6.1 The states defined for the ground ADS-C DC module are the following:

a) DC-G-IDLE

b) DC-G-PENDING

c) DC-G-ACTIVE

5.1.5.3.6.2 On initiation, the ADS-C ground DC module shall be in the DC-G-IDLE state.

5.1.5.3.6.3 Upon receipt of an ADS-demand-contract request:

5.1.5.3.6.3.1 If in the DC-G-IDLE state, the ADS-C ground DC module shall:

a) eate an ADS-contract-PDU with “demand contract” in the contract-type PDU elements and the *ADS/IC contract data* parameter in the ic-contract-request PDU element,

b) pass it, together with the *Aircraft Address* parameter value, *ICAO Facility Designation* parameter value, the *Class Of Communication Service* parameter value, and the *Security Required* parameter to the ground LI module,

c) start timer t-DC-1, and

d) enter the DC-G-PENDING state.

5.1.5.3.6.4 Upon receipt of an ADS-accepted-PDU containing the *ContractType* element set to the abstract value “demand-contract”:

5.1.5.3.6.4.1 If in the DC-G-PENDING state, the ADS-C ground DC module shall:

a) stop the t-DC-1 timer,

b) request the ADS-C ground HI module to invoke ADS-demand-contract confirmation with “accepted” in the *Result* parameter and the ic-report element of the ADS-accepted-report-PDU in the *ADS/IC Report Data* parameter,

c) enter the DC-G-IDLE state.

5.1.5.3.6.5 Upon receipt of an ADS-report-PDU containing the *ContractType* element set to the abstract value “demand-contract”:

5.1.5.3.6.5.1 If in the DC-G-ACTIVE state, the ADS-C ground DC module shall:

a) stop the t-DC-2 timer,

b) request the ADS-C ground HI module to invoke ADS-report indication with “demand contract” in the *Contract Type* parameter and the ic-report element of the ADS-report-PDU in the *ADS/IC Report Data* parameter, and

c) enter the DC-G-IDLE state.

5.1.5.3.6.6 Upon receipt of an ADS-positive-acknowledgement-PDU containing the *ContractType* element set to the abstract value “demand-contract”:

5.1.5.3.6.6.1 If in the DC-G-PENDING state, the ADS-C ground DC module shall:

a) stop the t-DC-1 timer,

b) request the ADS-C ground HI module to invoke ADS-demand-contract confirmation, with “positive-acknowledgement” in the *Result* parameter and the ic-positive-ack element of the ADS-positive-acknowledgement-PDU in the *ADS/IC Positive Acknowledgement* parameter,

c) start the t-DC-2 timer,

d) enter the DC-G-ACTIVE state.

5.1.5.3.6.7 Upon receipt of an ADS-rejected-PDU containing the *ContractType* element set to the abstract value “demand-contract”:

5.1.5.3.6.7.1 If in the DC-G-PENDING state, the ADS-C ground DC module shall:

a) stop the t-DC-1 timer,

b) request the ADS-C ground HI module to invoke ADS-demand-contract confirmation with “rejected” in the *Result* parameter and the ic-reject element of the ADS-rejected-PDU in the *ADS/IC Reject Data* parameter, and

c) enter the DC-G-IDLE state.

5.1.5.3.6.8 Upon receipt of an ADS-ncn-PDU containing the *ContractType* element set to the abstract value “demand-contract”:

5.1.5.3.6.8.1 If in the DC-G-PENDING state, the ADS-C ground DC module shall:

a) stop the t-DC-1 timer,

b) request the ADS-C ground HI module to invoke an ADS-demand-contract confirmation with “non compliance notification” in the *Result* parameter and the ic-ncn element of the ADS-ncn-PDU in the *ADS/IC Non Compliance Data* parameter,

c) start the t-DC-2 timer, and

d) enter the DC-G-ACTIVE state.

5.1.5.3.6.9 Upon expiry of the t-DC-1 timer or the t-DC-2 timer, the ADS-C ground DC module shall:

a) request the ADS-C ground AB module to abort with reason timer-expiry, and

b) enter the DC-G-IDLE state

5.1.5.3.6.10 Upon receipt of a request from the ground AB or ground LI module to stop operation, the ADS-C ground DC module shall:

a) stop any timers, and

b) enter the DC-G-IDLE state.

5.1.5.3.7 Air ADS-C DC Module

5.1.5.3.7.1 The states defined for the air ADS-C DC module are the following:

a) DC-A-IDLE

b) DC-A-PENDING

c) DC-A-ACTIVE

5.1.5.3.7.2 On initiation, the ADS-C air DC module shall be in the DC-A-IDLE state.

5.1.5.3.7.3 Upon receipt of an ADS-demand-contract response with the *Result* parameter value set to “rejected”:

5.1.5.3.7.3.1 If in the DC-A-PENDING state, the ADS-C air DC module shall:

a) create an ADS-rejected-PDU with “demand-contract” in the contract-type element and the *ADS/IC Reject Data* parameter value in the ic-reject element,

b) pass it to the air LI module, and

c) enter the DC-A-IDLE state.

5.1.5.3.7.4 Upon receipt of an ADS-demand-contract response with the *Result* parameter value set to “positive-acknowledgement”:

5.1.5.3.7.4.1 If in the DC-A-PENDING state, the ADS-C air DC module shall:

a) create an ADS-positive-acknowledgement-PDU with “demand-contract” in the contrat-type element and the ic-positive-ack element of the ADS-positive-acknowledgement-PDU in the *ADS/IC Positive Acknowledgement* parameter,

b) pass it to the air LI module, and

c) enter the DC-A-ACTIVE state.

5.1.5.3.7.5 Upon receipt of an ADS-demand-contract response with the *Result* parameter value set to “non compliance notification”:

5.1.5.3.7.5.1 If in the DC-A-PENDING state, the ADS-C air DC module shall:

a) create an ADS-ncn-PDU with “demand-contract” in the contract-type element and the *ADS/IC Non Compliance Data* parameter value in the ic-ncn element,

b) pass it to the air LI module, and

c) enter the DC-A-ACTIVE state.

5.1.5.3.7.6 Upon receipt of an ADS-demand-contract response with the *Result* parameter value set to set to “accepted”:

5.1.5.3.7.6.1 If in the DC-A-PENDING state, the ADS-C air DC module shall:

a) create an ADS-accepted-PDU with “demand-contract” in the contract-type element and the *ADS/IC Report Data* parameter value in the ic-report element,

b) pass it to the air LI module, and

c) enter the DC-A-IDLE state.

5.1.5.3.7.7 Upon receipt of an ADS-report request containing the *ContractType* element set to the abstract value “demand-contract”:

5.1.5.3.7.7.1 If in the DC-A-ACTIVE state, the ADS-C air DC module shall:

a) create ADS-report-PDU with “demand-contract” in the contract-type element and the *ADS/IC Report Data* parameter value in the ic-report element,

b) pass it to the air LI module, and

c) enter the DC-A-IDLE state.

5.1.5.3.7.8 Upon receipt of an ADS-contract-PDU containing the *ContractType* element set to the abstract value “demand-contract”:

5.1.5.3.7.8.1 If in the DC-A-IDLE state, the ADS-C air DC module shall:

a) request the air HI module to invoke ADS-demand-contract indication with:

1) the ic-contract-request PDU element in the *ADS/IC Contract Data* parameter,

2) the Calling peer id PDU element, if provided by the air LI module, in the *ICAO Facility Designation* parameter,

3) the DS-User Version Number, if provided by the air LI module, in the *ADS-C Message Set Version Number* parameter, and

b) enter the DC-A-PENDING state.

5.1.5.3.7.9 Upon receipt of a request from the air AB or air LI module to stop operation, the ADS-C air DC module shall enter the DC-A-IDLE state.

5.1.5.3.8 Ground ADS-C EC Module

5.1.5.3.8.1 The states defined for the ground ADS-C EC module are the following:

a) EC-G-IDLE

b) EC-G-START-PENDING

c) EC-G-ACTIVE

d) EC-G-PENDING

e) EC-G-CANCEL

5.1.5.3.8.2 On initiation, the ADS-C ground EC module shall be in the EC-G-IDLE state.

5.1.5.3.8.3 Upon receipt of an ADS-event-contract request:

5.1.5.3.8.3.1 If in the EC-G-IDLE state, the ADS-C ground EC module shall:

a) create an ADS-contract-PDU with elements as defined in Table 5-18,

b) pass it, together with the *Aircraft Address* parameter value, *ICAO Facility Designation* parameter value, the *Class Of Communication Service* parameter value, the *ADS-C Message Set Version Number* parameter value and the *Security Required* parameter value, to the ground LI module,

c) start timer t-EC-1, and

d) enter the EC-G-START-PENDING state.

5.1.5.3.8.3.2 If in the EC-G-ACTIVE state, the ADS-C ground EC module shall:

a) create an ADS-contract-PDU with “event-contract” in the contract-type element and the *ADS/IC Contract Data* parameter in the ic-contract-request element,

b) pass it to the ground LI module,

c) start timer t-EC-1, and

d) enter the EC-G-PENDING state.

5.1.5.3.8.4 Upon receipt of an ADS-cancel request:

5.1.5.3.8.4.1 If in the EC-G-ACTIVE state, the ADS-C ground EC module shall:

a) create an ADS-cancel-contract-PDU with “event-contract” in the ADS-cancel-contract-PDU element,

b) pass it to the ground LI module,

c) start timer t-EC-2, and

d) enter the EC-G-CANCEL state.

5.1.5.3.8.5 Upon receipt of an ADS-accepted-PDU containing the *ContractType* element set to the abstract value “event-contract”:

5.1.5.3.8.5.1 If in the EC-G-PENDING or the EC-G-START-PENDING state, the ADS-C ground EC module shall:

a) stop the t-EC-1 timer,

b) request the ADS-C ground HI module to invoke ADS-demand-contract indication, with “accepted” in

the *Result* parameter and the ic-report element of the ADS-accepted-PDU in the *ADS/IC Report Data* parameter, and

c) enter the EC-G-ACTIVE state.

5.1.5.3.8.6 Upon receipt of an ADS-report-PDU containing the *ContractType* element set to the abstract value "event-contract":

5.1.5.3.8.6.1 If in the EC-G-ACTIVE, EC-G-PENDING or EC-G-CANCEL state, the ADS-C ground EC module shall

a) request the ADS-C ground HI module to invoke ADS-report indication, with "event contract" in the *Contract Type* parameter and the ic-report PDU element of the ADS-report-PDU in the *ADS/IC Report Data* parameter, and

b) remain in the same state.

5.1.5.3.8.7 Upon receipt of an ADS-positive-acknowledgement-PDU containing a *ContractType* element set to the abstract value "event-contract":

5.1.5.3.8.7.1 If in the EC-G-START-PENDING state or the EC-G-PENDING state, the ADS-C ground EC module shall:

a) stop the t-EC-1 timer,

b) request the ADS-C ground HI module to invoke ADS-event-contract confirmation, with "positive-acknowledgement" in the *Result* parameter and the ic-positive-ack element of the ADS-positive-acknowledgement-PDU in the *ADS/IC Positive Acknowledgement* parameter, and

c) enter the EC-G-ACTIVE state.

5.1.5.3.8.8 Upon receipt of an ADS-cancel-positive-acknowledgement-PDU containing a *RequestType* element set to the abstract value "cancel-event-contract":

5.1.5.3.8.8.1 If in the EC-G-CANCEL state, the ADS-C ground EC module shall:

a) stop the t-EC-2 timer,

b) request the ADS-C ground HI module to invoke ADS-cancel-contract confirmation, with "event-contract" in the *Contract Type* parameter, and

c) enter the EC-G-IDLE state.

5.1.5.3.8.9 Upon receipt of an ADS-rejected-PDU containing the *ContractType* element set to the abstract value "event-contract":

5.1.5.3.8.9.1 If in the EC-G-START-PENDING state, the ADS-C ground EC module shall:

a) stop the t-EC-1 timer,

b) request the ADS-C ground HI module to invoke ADS-event-contract confirmation, with "rejected" in the *Result* parameter and ic-reject element of the ADS-rejected-PDU in the *ADS/IC Reject Data* parameter, and

c) enter the EC-G-IDLE state.

5.1.5.3.8.9.2 If in the EC-G-PENDING state, the ADS-C ground EC module shall:

a) stop the t-EC-1 timer,

b) request the ADS-C ground HI module to invoke ADS-event-contract confirmation, with “rejected” in the *Result* parameter and ic-reject element of the ADS-rejected-PDU in the *ADS/IC Reject Data* parameter, and

c) enter the EC-G-ACTIVE state.

5.1.5.3.8.10 Upon receipt of an ADS-ncn-PDU:

5.1.5.3.8.10.1 If in the EC-G-START-PENDING state or the EC-G-PENDING, the ADS-C ground EC module shall:

a) stop the t-EC-1 timer,

b) request the ADS-C ground HI module to invoke ADS-event-contract confirmation, with “non compliance notification” in the *Result* parameter and the ic-ncn element of the ADS-ncn-PDU in the *ADS/IC Non Compliance Data* parameter, and

c) enter the EC-G-ACTIVE state.

5.1.5.3.8.11 Upon expiry of the t-EC-1 timer or the t-EC-2 timer, the ADS-C ground EC module shall:

a) request the ADS-C ground AB module to abort with reason timer-expiry, and

b) enter the EC-G-IDLE state

5.1.5.3.8.12 Upon receipt of a request from the ADS-C ground AB module or LI module to stop operation, the EC module shall:

a) stop any timers, and

b) enter the EC-G-IDLE state.

5.1.5.3.9 Air ADS-C EC Module

5.1.5.3.9.1 The states defined for the air ADS-C EC module are the following:

a) EC-A-IDLE

b) EC-A-PENDING

c) EC-A-ACTIVE

d) EC-A-ACTIVE-PENDING

5.1.5.3.9.2 On initiation, the ADS-C air EC module shall be in the EC-A-IDLE state.

5.1.5.3.9.3 Upon receipt of an ADS-event-contract response with the *Result* parameter value set to the abstract

value “positive acknowledgement”:

5.1.5.3.9.3.1 If in the EC-A-PENDING state or in the EC-A-ACTIVE-PENDING state, the ADS-C air EC module shall:

a) create an ADS-positive-acknowledgement-PDU with “event-contract“ in the contract-type PDU element and the ic-positive-ack element of the ADS-positive-acknowledgement-PDU in the *ADS/IC Positive Acknowledgement* parameter,

b) pass it to the air LI module, and

c) enter the EC-A-ACTIVE state.

5.1.5.3.9.4 Upon receipt of an ADS-event-contract response with the *Result* parameter value set to the abstract value “non compliance notification”:

5.1.5.3.9.4.1 If in the EC-A-PENDING state or in the EC-A-ACTIVE-PENDING state, the ADS-C air EC module shall:

a) create an ADS-ncn-PDU with “event-contract” in the contract-type PDU element and the *ADS/IC Non Compliance Data* parameter value in the ic-ncn PDU element,

b) pass it to the air LI module, and

c) enter the EC-A-ACTIVE state.

5.1.5.3.9.5 Upon receipt of an ADS-event-contract response with the *Result* parameter value set to the abstract value “rejected”:

5.1.5.3.9.5.1 If in the EC-A-PENDING state, the ADS-C air EC module shall:

a) create an ADS-rejected-PDU with “event-contract“ in the Contract-Type PDU element and the *ADS/IC Reject Data* parameter value in the ic-reject PDU element,

b) pass it to the air LI module, and

c) enter the EC-A-IDLE state.

5.1.5.3.9.5.2 If in the EC-A-ACTIVE-PENDING state, the ADS-C air EC module shall:

a) create an ADS-rejected-PDU with “event-contract“ in the Contract-Type PDU element and the *ADS/IC Reject Data* parameter value in the ic-reject PDU element,

b) pass it to the air LI module, and

c) enter the EC-A-ACTIVE state.

5.1.5.3.9.6 Upon receipt of an ADS-event-contract response with the *Result* parameter value set to the abstract value “accepted”:

5.1.5.3.9.6.1 If in the EC-A-PENDING state or in the EC-A-ACTIVE-PENDING state, the ADS-C air EC module shall:

a) create an ADS-accepted-PDU with “event-contract“ in the request-type PDU element and the

ADS/IC Report Data parameter value in the ic-report PDU element,

b) pass it to the air LI module, and

c) enter the EC-A-ACTIVE state.

5.1.5.3.9.7 Upon receipt of an ADS-report request containing the *ContractType* element set to the abstract value "event-contract":

5.1.5.3.9.7.1 If in the EC-A-ACTIVE state, the ADS-C air EC module shall:

a) create an ADS-report-PDU with "event-contract" in the contract-type PDU element and the *ADS/IC Report Data* parameter value in the ic-report PDU element,

b) pass it to the air LI module, and

c) remain in the EC-A-ACTIVE state.

5.1.5.3.9.8 Upon receipt of an ADS-contract-PDU containing the *ContractType* element set to the abstract value "event-contract":

5.1.5.3.9.8.1 If in the EC-A-IDLE state, the ADS-C air EC module shall:

a) request the air HI module to invoke ADS-event-contract indication with:

1) the ic-contract-request PDU element in the *ADS/IC Contract Data* parameter,

2) the Calling peer id, if provided by the air LI module, in the *ICAO Facility Designation* parameter,

3) the DS-User Version Number, if provided by the air LI module, in the *ADS-C Message Set Version Number* parameter, and

b) enter the EC-A-PENDING state.

5.1.5.3.9.8.2 If in the EC-A-ACTIVE state, the ADS-C air EC module shall:

a) request the air HI module to invoke ADS-event-contract indication with the ic-contract-request PDU element in the *ADS/IC Contract Data* parameter and the Calling peer id, if provided by the air LI module, in the *ICAO Facility Designation* parameter, and

b) enter the EC-A-ACTIVE-PENDING state.

5.1.5.3.9.9 Upon receipt of an ADS-cancel-contract-PDU:

5.1.5.3.9.9.1 If in the EC-A-ACTIVE state, the ADS-C air EC module shall:

a) request the ADS-C HI module to invoke ADS-cancel indication with "event-contract" in the *Contract Type* parameter,

b) create an ADS-cancel-positive-acknowledgement-PDU with “cancel-event-contract” in the Request type PDU element,

c) pass it to the air LI module, and

d) enter the EC-A-IDLE state.

5.1.5.3.9.9.2 If in the EC-A-IDLE state, the ADS-C air EC module shall:

a) create an ADS-cancel-negative-acknowledgement-PDU with “cancel-event-contract” for the RequestType element and “no-such-contract” for the RejectReason element in the CancelRejectReason PDU element,

b) pass it to the air LI module, and

c) remain in the EC-A-IDLE state.

5.1.5.3.9.10 Upon receipt of a request from the air AB or air LI module to stop operation, the ADS-C air EC module shall enter the EC-A-IDLE state.

5.1.5.3.10 Ground ADS-C PC Module

5.1.5.3.10.1 The states defined for the ground ADS-C PC module are the following:

a) PC-G-IDLE

b) PC-G-START-PENDING

c) PC-G-ACTIVE

d) PC-G-PENDING

e) PC-G-CANCEL

5.1.5.3.10.2 On initiation, the ADS-C ground PC module shall be in the PC-G-IDLE state.

5.1.5.3.10.3 Upon receipt of an ADS-periodic-contract request:

5.1.5.3.10.3.1 If in the PC-G-IDLE state, the ADS-C ground PC module shall:

a) create an ADS- contract-PDU with “periodic-contract” in the contract-type PDU element and the *ADS/IC contract data* parameter value in the ic-contract-request PDU element,

b) pass it, together with the *Aircraft Address* parameter value, *ICAO Facility Designation* parameter value, the *Class of Communication Service* parameter value, the *ADS-C Message Set Version Number* parameter value and the *Security Required* parameter value, to the ground LI module,

c) start timer t-PC-1, and

d) enter the PC-G-START-PENDING state.

5.1.5.3.10.3.2 If in the PC-G-ACTIVE state, the ADS-C ground PC module shall:

a) stop the t-PC-2 timer,

b) create an ADS- contract-PDU with “periodic-contract” in the contract-type PDU element and the *ADS/IC contract data* parameter value in the ic-contract-request PDU element,

c) pass it to the ground LI module,

d) start timer t-PC-1, and

e) enter the PC-G-PENDING state.

5.1.5.3.10.4 Upon receipt of an ADS-cancel request containing the *ContractType* element set to the abstract value “periodic-contract”:

5.1.5.3.10.4.1 If in the PC-G-ACTIVE state, the ADS-C ground PC module shall:

a) stop the t-PC-2 timer,

b) create an ADS-cancel-contract-PDU with “periodic-contract” in the ADS-cancel-contract-PDU element,

c) pass it to the ground LI module,

d) start timer t-PC-3, and

e) enter the PC-G-CANCEL state.

5.1.5.3.10.5 Upon receipt of an ADS-accepted-PDU containing the *ContractType* element set to the abstract value “periodic-contract”:

5.1.5.3.10.5.1 If in the PC-G-START-PENDING state, or the PC-G-PENDING state, the ADS-C ground PC module shall:

a) stop the t-PC-1 timer,

b) create the *ADS/IC report data* parameter of an ADS-periodic-indication,

c) request the ADS-C ground HI module to invoke ADS-periodic-contract indication with “accepted” in the *Result* parameter and the ic-report PDU element of the ADS-accepted-PDU in the *ADS/IC Report Data* parameter,

d) start the t-PC-2 timer,

e) enter the PC-G-ACTIVE state.

5.1.5.3.10.6 Upon receipt of an ADS-report-PDU:

5.1.5.3.10.6.1 If in the PC-G-PENDING state, the ADS-C ground PC module shall:

a) request the ADS-C ground HI module to invoke ADS-report indication with “periodic-contract“ in the

Contract Type parameter and the ic-report PDU element of the ADS-report-PDU in the *ADS/IC Report Data* parameter,

b) remain in the PC-G-PENDING state.

5.1.5.3.10.6.2 If in the PC-G-ACTIVE state, the ADS-C ground PC module shall:

a) stop the t-PC-2 timer,

b) request the ADS-C ground HI module to invoke ADS-report indication with “periodic-contract“ in the *Contract Type* parameter and the ic-report PDU element of the ADS-report-PDU in the *ADS/IC Report Data* parameter,

c) start the t-PC-2 timer,

d) remain in the PC-G-ACTIVE state.

5.1.5.3.10.6.3 If in the PC-G-CANCEL state, the ADS-C ground PC module shall:

a) request the ADS-C ground HI module to invoke ADS-report indication with “periodic-contract“ in the *Contract Type* parameter and the ic-report PDU element of the ADS-report-PDU in the *ADS/IC Report Data* parameter, and

b) remain in the PC-G-CANCEL state.

5.1.5.3.10.7 Upon receipt of an ADS-positive-acknowledgement-PDU containing a *ContractType* element set to the abstract value “periodic-contract”:

5.1.5.3.10.7.1 If in the PC-G-START-PENDING state, or the PC-G-PENDING state, the ADS-C ground PC module shall:

a) stop the t-PC-1 timer,

b) request the ADS-C ground HI module to invoke ADS-periodic-contract confirmation with “positive acknowledgement” in the *Result* parameter and the ic-positive-ack element of the ADS-positive-acknowledgement-PDU in the *ADS/IC Positive Acknowledgement* parameter,

c) start the t-PC-2 timer,

d) enter the PC-G-ACTIVE state.

5.1.5.3.10.8 Upon receipt of an ADS-cancel-positive-acknowledgement-PDU containing a *RequestType* element set to the abstract value “cancel-periodic-contract”:

5.1.5.3.10.8.1 If in the PC-G-CANCEL state, the ADS-C ground PC module shall:

a) stop the t-PC-3 timer,

b) request the ADS-C ground HI module to invoke ADS-cancel-contract confirmation with “periodic-contract“ in the *Contract Type* parameter, and

c) enter the PC-G-IDLE state.

5.1.5.3.10.9 Upon receipt of an ADS-rejected-PDU containing the *ContractType* element set to the abstract value "periodic-contract":

5.1.5.3.10.9.1 If in the PC-G-START-PENDING state, the ADS-C ground PC module shall:

a) stop the t-PC-1 timer,

b) request the ADS-C ground HI module to invoke ADS-periodic-contract confirmation with parameter values derived as in Table 2.1.5-40, and

c) enter the PC-G-IDLE state.

5.1.5.3.10.9.2 If in the PC-G-PENDING state, the ADS-C ground PC module shall:

a) stop the t-PC-1 timer,

b) request the ADS-C ground HI module to invoke ADS-periodic-contract confirmation with "rejected" in the *Result* parameter and the ic-reject element of the ADS-rejected-PDU in the *ADS/IR Reject Data* parameter,

c) start the t-PC-2 timer,

d) enter the PC-G-ACTIVE state.

5.1.5.3.10.10 Upon receipt of an ADS-ncn-PDU containing the *ContractType* element set to the abstract value "periodic-contract":

5.1.5.3.10.10.1 If in the PC-G-START-PENDING state, or the PC-G-PENDING state, the ADS-C ground PC module shall:

a) stop the t-PC-1 timer,

b) request the ADS-C ground HI module to invoke ADS-periodic-contract confirmation with "non compliance notification" in the *Result* parameter and the ic-ncn element of the ADS-ncn-PDU in the *ADS/IC Non Compliance Data* parameter,

c) start the t-PC-2 timer, and

d) enter the PC-G-ACTIVE state.

5.1.5.3.10.11 Upon receipt of a request from the ground AB or ground LI module to stop operation, the ADS-C ground PC module shall:

a) stop any timers,

b) enter the PC-G-IDLE state.

5.1.5.3.10.12 Upon expiry of the t-PC-1 timer, t-PC-2 timer, or t-PC-3 timer, the ADS-C ground PC module shall:

a) request the ADS-C ground AB module to abort with reason timer-expiry, and

b) enter the PC-G-IDLE state.

5.1.5.3.11 Air ADS-C PC Module

5.1.5.3.11.1 The states defined for the air ADS-C PC module are the following:

a) PC-A-IDLE

b) PC-A-PENDING

c) PC-A-ACTIVE

d) PC-A-ACTIVE-PENDING

5.1.5.3.11.2 On initiation, the ADS-C air PC module shall be in the PC-A-IDLE state.

5.1.5.3.11.3 Upon receipt of an ADS-periodic-contract response with the *Result* parameter value set to the abstract value “positive acknowledgement”, then:

5.1.5.3.11.3.1 If in the PC-A-PENDING state or PC-A-ACTIVE-PENDING state, the ADS-C air PC module shall:

a) If the *Result* parameter value contains the abstract value “positive acknowledgement”, create an ADS-positive-acknowledgement-PDU with “periodic-contract“ in the contract-type PDU element and the ic-positive-ack element of the ADS-positive-acknowledgement-PDU in the *ADS/IC Positive Acknowledgement* parameter,

b) pass it to the air LI module, and

c) enter the PC-A-ACTIVE state.

5.1.5.3.11.4 Upon receipt of an ADS-periodic-contract response with the *Result* parameter value set to the abstract value “non compliance notification”, then:

5.1.5.3.11.4.1 If in the PC-A-PENDING state or PC-A-ACTIVE-PENDING state, the ADS-C air PC module shall:

a) If the *Result* parameter value contains the abstract value “non compliance notification”, create an ADS-ncn-PDU with “periodic-contract” in the contract-type PDU element and the *ADS/IC Non Compliance Data* parameter value in the ic-ncn PDU element,

b) pass it to the air LI module, and

c) enter the PC-A-ACTIVE state.

5.1.5.3.11.5 Upon receipt of an ADS-periodic-contract response with the *Result* parameter value set to the abstract value “rejected”, then:

5.1.5.3.11.5.1 If in the PC-A-PENDING state, the ADS-C air PC module shall:

a) create an ADS-rejected-PDU with “periodic-contract “ in the contract-type PDU element and the

ADS/IC Reject Data parameter value in the ic-ncn PDU element,

b) pass it to the air LI module, and

c) enter the PC-A-IDLE state.

5.1.5.3.11.5.2 If in the PC-A-ACTIVE-PENDING state, the ADS-C air PC module shall:

a) create an ADS-rejected-PDU with “periodic-contract “ in the contract-type PDU element and the *ADS/IC Reject Data* parameter value in the ic-ncn PDU element,

b) pass it to the air LI module, and

c) enter the PC-A-ACTIVE state.

5.1.5.3.11.6 Upon receipt of an ADS-periodic-contract response with the *Result* parameter value set to the abstract value “accepted”, then:

5.1.5.3.11.6.1 If in the PC-A-PENDING state or PC-A-ACTIVE-PENDING, the ADS-C air PC module shall:

a) create ADS- -accepted-PDU with “periodic-contract “ in the contract-type PDU element and the *ADS/IC Report Data* parameter value in the ic-report PDU element,

b) pass it to the air LI module, and

c) enter the PC-A-ACTIVE state.

5.1.5.3.11.7 Upon receipt of an ADS-report request containing the *ContractType* element set to the abstract value “periodic-contract”:

5.1.5.3.11.7.1 If in the PC-A-ACTIVE state, the ADS-C air PC module shall:

a) create ADS-report-PDU with “periodic-contract” in the contract-type PDU element and the *ADS/IC Report Data* parameter value in the ic-report PDU element,

b) pass it to the air LI module, and

c) remain in the PC-A-ACTIVE state.

5.1.5.3.11.8 Upon receipt of an ADS-contract-PDU containing the *ContractType* element set to the abstract value “periodic-contract”:

5.1.5.3.11.8.1 If in the PC-A-IDLE state, the ADS-C air PC module shall:

a) request the air HI module to invoke ADS-periodic-contract indication with:

1) the ic-contract-request PDU element in the *ADS/IC Contract Data* parameter,

2) the Calling peer id, if provided by the air LI module, in the *ICAO Facility Designation* parameter,

3) the DS-User Version Number, if provided by the air LI module, in the *ADS-C Message Set*

Version Number parameter, and

b) enter the PC-A-PENDING state.

5.1.5.3.11.8.2 If in the PC-A-ACTIVE state, the ADS-C air PC module shall:

a) request the air HI module to invoke ADS-periodic-contract indication with the ic-contract-request PDU element in the *ADS/IC Contract Data* parameter and the Calling peer id, if provided by the air LI module, in the *ICAO Facility Designation* parameter, and

b) enter the PC-A-ACTIVE-PENDING state.

5.1.5.3.11.9 Upon receipt of an ADS-cancel-contract-PDU containing the *ContractType* element set to the abstract value “periodic-contract”, then:

5.1.5.3.11.9.1 If in the PC-A-ACTIVE state, the ADS-C air PC module shall:

a) request the air HI module to invoke ADS-cancel indication with “periodic-contract” in the *Contract Type* parameter,

b) create an ADS-cancel-positive-acknowledgement-PDU with “cancel-periodic-contract” in the RequestType PDU element,

c) pass it to the air LI module, and

d) enter the PC-A-IDLE state.

5.1.5.3.11.9.2 If in the PC-A-IDLE state, the ADS-C air PC module shall:

a) create an ADS-cancel-negative-acknowledgement-PDU with “cancel-periodic-contract” in the RequestType element and “no-such-contract” in the RejectReason element of the CancelRejectReason PDU element,

b) pass it to the air LI module, and

c) remain in the PC-A-IDLE state.

5.1.5.3.11.10 Upon receipt of a request from the air AB or air LI module to stop operation, the ADS-C air PC module shall enter the PC-A-IDLE state.

5.1.5.3.12 Ground and Air ADS-C AB Modules

5.1.5.3.12.1 All statements in 5.3.14 apply to both the ADS-C ground AB module and the ADS-C air AB module.

5.1.5.3.12.2 Upon receipt of an ADS-user-abort request, the ADS-C AB module shall:

a) request the DC, EC and PC modules to stop operation,

b) create an ADS-user-abort-PDU with the UserAbortReason PDU element derived from the *reason*

parameter, and

c) request the LI module to invoke D-ABORT request with “user” in the *originator* parameter and the the ADS-user-abort-PDU in the *User Data* parameter.

5.1.5.3.12.3 Upon receipt of a request to abort, the ADS-C AB module shall:

a) request the DC, EC and PC modules to stop operation,

b) create an ADS-provider-abort-PDU with the value provided by DC, EC or PC module in the AbortReason PDU element,

c) request the LI module to invoke D-ABORT request with “provider” in the *Originator* parameter and the ADS-provider-abort-PDU in the *User Data* parameter, and

d) request the ADS-C HI module to invoke ADS-provider-abort indication with the value provided by DC, EC or PC module in the *Reason* parameter.

5.1.5.3.12.4 Upon receipt of a D-P-ABORT indication, the ADS-C AB module shall:

a) request the DC, EC and PC modules to stop operation, and

b) request the ADS-C HI module to invoke ADS-provider-abort indication with “communications-service-failure” in the *Reason* parameter.

5.1.5.3.12.5 Upon receipt of a D-ABORT indication with the *Originator* parameter value set to the abstract value “user” and with an ADS-user-data-PDU in the *User Data* parameter, the ADS-C AB module shall:

a) request the DC, EC and PC modules to stop operation, and

b) request the ADS-C HI module to invoke ADS-user-abort indication with the value of the *User Data* parameter in the *Reason* parameter.

5.1.5.3.12.6 Upon receipt of a D-ABORT indication with the *Originator* parameter value set to the abstract value “provider” and with data in the *User Data* parameter, the ADS-C AB module shall:

a) request the DC, EC and PC modules to stop operation, and

b) request the ADS-C HI module to invoke ADS-provider-abort indication with the value of the *User Data* parameter in the *Reason* parameter.

5.1.5.3.13 Ground ADS-C LI Module

5.1.5.3.13.1 The states defined for the ground ADS-C ATN LI module are the following:

a) LI-G-IDLE

b) LI-G-START

c) LI-G-ACTIVE

d) LI-G-END

5.1.5.3.13.2 On initiation, the ground LI module shall be in the LI-G-IDLE state.

5.1.5.3.13.3 Upon receipt of an ADS-contract-PDU from the ground DC, EC or PC modules:

5.1.5.3.13.3.1 If in the LI-G-IDLE state, the ground LI module shall:

a) Invoke D-START request using parameter values as shown in Table 5-14, and

b) enter the LI-G-START state.

Table 5-14: D-START request parameter values

D-START parameter	Source
Called peer Id	*Aircraft address* parameter value from contract request
Calling peer Id	*ICAO Facility Designation* parameter value from contract request
DS-user version number	*ADS-C Message Set Version Number* parameter value from contract request
Security requirements	*Security Required* parameter value from contract request
Quality of service	Routing class: ATSC, with value from *Class of Communication Service* parameter value from contract request Priority: High priority flight safety messages RER: Low
User data	The contract PDU passed to LI

5.1.5.3.13.3.2 If in the LI-G-ACTIVE state, the ground LI module shall:

a) Invoke D-DATA request with the PDU as the *User Data* parameter value, and

b) remain in the LI-G-ACTIVE state.

5.1.5.3.13.4 Upon receipt of an ADS-cancel-all-contracts request:

5.1.5.3.13.4.1 If in the LI-G-ACTIVE state, the ground LI module shall:

a) Invoke D-END req with ADS-cancel-all-contracts-PDU in the user data,

b) start the t-LI-1 timer, and

c) enter the LI-G-END state.

5.1.5.3.13.5 Upon receipt of an ADS-cancel-contract-PDU from the ground EC or PC modules:

5.1.5.3.13.5.1 If in the LI-G-ACTIVE state, the ground LI module shall:

a) invoke D-DATA req with the PDU in the user data,

b) if the DC module is in the DC-G-IDLE state, and the EC module is in the EC-G-IDLE state, and the

PC module is in the PC-G-IDLE state, then:

1) invoke D-END req with no user data,

2) start the t-LI-1 timer, and

3) enter the LI-G-END state; or

c) otherwise, remain in the LI-G-ACTIVE state.

5.1.5.3.13.6 Upon receipt of request to invoke D-ABORT from the ADS-C ground AB module:

5.1.5.3.13.6.1 If in the LI-G-START state, the LI-G-ACTIVE state or the LI-G-END state, the ground LI module shall:

a) invoke D-ABORT request with the parameter values supplied, and

b) enter the LI-G-IDLE state.

5.1.5.3.13.6.2 If in the LI-G-IDLE state, the ground LI module shall ignore the request.

5.1.5.3.13.7 Upon receipt of a D-START confirmation with the *Result* parameter value containing the abstract value "accepted" and the *Security Requirements* parameter value containing an abstract value equal to the abstract value set in the D-START request *Security Requirements* parameter:

Note.— If a secure D-START request was issued (i.e. the Security Requirements parameter was set to "secured exchange") then the D-START confirmation Security Requirements parameter must be equal to that value. If an unsecure D-START request was issued (i.e. the Security Requirements parameter was set to "no security") then the D-START confirmation Security Requirements parameter will be equal to that value.

5.1.5.3.13.7.1 If in the LI-G-START state, the ground LI module shall:

a) pass the user data to the module as defined in Table 5-15,

b) if, after processing the PDU (i.e. the ADS-C ground HI module has issued the appropriate indication or confirmation), the ADS-C ground DC module is in the DC-G-IDLE state, and the ADS-C ground EC module is in the EC-G-IDLE state, and the ADS-C ground PC module is in the PC-G-IDLE state, then:

1) invoke D-END req with no user data, and

2) start the t-LI-1 timer, and

3) enter the LI-G-END state; or

c) if, after processing the PDU, the ADS-C ground DC module is not in the DC-G-IDLE state, or the ADS-C ground EC module is not in the EC-G-IDLE state, or the ADS-C ground PC module is not in the PC-G-IDLE state, then:

1) enter the LI-G-ACTIVE state.

5.1.5.3.13.8 Upon receipt of a D-DATA indication:

5.1.5.3.13.8.1 If in the LI-G-ACTIVE state, the ground LI module shall:

a) pass the user data to the module as defined in Table 5-15,

b) if, after processing the PDU (i.e. the ADS-C ground HI module has issued the appropriate indication or confirmation), the ADS-C ground DC module is in the DC-G-IDLE state, and the ADS-C ground EC module is in the EC-G-IDLE state, and the ADS-C ground PC module is in the PC-G-IDLE state, then:

1) invoke D-END req with no user data, and

2) start the T-LI-1 timer, and

3) enter the LI-G-END state; or

c) otherwise, remain in the LI-G-ACTIVE state.

5.1.5.3.13.8.2 If in the LI-G-END state, the ground LI module shall:

a) pass the user data to the module as defined in Table 5-15, and

b) remain in the LI-G-END state.

Table 5-15: PDU to ground module mapping

PDU	*Subfield*	*Ground Module*
ADS-report-PDU	*contract-type*	
	"demand-contract"	DC
	"event-contract"	EC
	"periodic-contract"	PC
ADS-accepted-PDU, ADS-rejected-PDU, ADS-ncn-PDU	*contract-type*	
	"demand-contract"	DC
	"event-contract"	EC
	"periodic-contract"	PC
ADS-cancel-positive-acknowledgement-PDU	*RequestType*	
	"cancel-event-contract"	EC
	"cancel-periodic-contract"	PC
ADS-positive-acknowledgement-PDU	*ContractType*	
	"demand-contract"	DC
	"periodic-contract"	PC
ADS-cancel-negative-acknowledgement-PDU		AB

5.1.5.3.13.9 Upon receipt of a D-END confirmation with the *Result parameter* value containing the abstract value "accepted":

5.1.5.3.13.9.1 If in the LI-G-END state, the ground LI module shall:

a) stop the t-LI-1 timer,

b) if the *User Data* parameter value contains an ADS-cancel-positive-acknowledgement-PDU (cancel-all-contracts), then request the ADS-C ground HI module to invoke ADS-cancel-all-contracts confirmation,

c) request the DC, EC and PC modules to stop operation, and

d) enter the LI-G-IDLE state.

5.1.5.3.13.10 Upon receipt of a D-ABORT indication, the ground LI module shall:

a) pass the D-ABORT indication to the ADS-C ground AB module, and

b) enter the LI-G-IDLE state.

5.1.5.3.13.11 Upon receipt of a D-P-ABORT indication, the ground LI module shall:

a) pass the D-P-ABORT indication to the ADS-C ground AB module, and

b) enter the LI-G-IDLE state.

5.1.5.3.13.12 Upon expiry of the t-LI-1 timer, the ground LI module shall:

a) request the ADS-C air AB module to abort with reason timer-expiry, and

b) remain in the same state.

5.1.5.3.14 Air ADS-C LI Module

5.1.5.3.14.1 The states defined for the air ADS-C ATN LI module are the following:

a) LI-A-IDLE

b) LI-A-START

c) LI-A-ACTIVE

5.1.5.3.14.2 On initiation, the air LI module shall be in the LI-A-IDLE state.

5.1.5.3.14.3 Upon receipt of an ADS-positive-acknowledgement-PDU, an ADS-accepted-PDU, ADS-rejected-PDU, or an ADS-ncn-PDU from the air DC, EC or PC modules:

5.1.5.3.14.3.1 If in the LI-A-START state, the air LI module shall:

a) Invoke D-START response using parameter values as shown in Table 5-16, and

b) enter the LI-A-ACTIVE state.

Table 5-16: D-START response parameter values

D-START parameter	Source
DS-user version number	Not used
Security requirements	the *Security Requirements* parameter received in the D-START indication
Quality of service	Not used
Result	"Accepted"
User data	The PDU passed to the air LI module

5.1.5.3.14.4 Upon receipt of an ADS-report-PDU, an ADS-positive-acknowledgement-PDU, an ADS-accepted-PDU, ADS-rejected-PDU, or an ADS-ncn-PDU from the air DC, EC or PC modules:

5.1.5.3.14.4.1 If in the LI-A-ACTIVE state, the air LI module shall:

a) Invoke D-DATA request using the PDU as the *User Data* parameter value, and

b) remain in the LI-A-ACTIVE state.

5.1.5.3.14.5 Upon receipt of a request to invoke D-ABORT from the ADS-C air AB module:

5.1.5.3.14.5.1 The air LI module shall:

a) If a dialogue exists, invoke D-ABORT request with the APDU as *User Data* parameter value and the value supplied by the ADS-C AB module as *Originator* parameter, and

b) enter the LI-A-IDLE state.

5.1.5.3.14.6 Upon receipt of a D-START indication:

5.1.5.3.14.6.1 If in the LI-A-IDLE state, and the *application service priority* parameter value is "high priority flight safety messages", the *RER quality of service* parameter is the abstract value "low", the *Routing Class* parameter identifies the traffic category "Air Traffic Service Communications (ATSC)", the *Calling Peer ID* parameter is a valid four to eight character facility designation and the *Security Requirements* parameter is consistent with the local security policy, the air LI module shall:

a) pass the user data, the calling peer id and the *DS-User Version Number* parameter values to the module as defined in Table 5-17, and

b) enter LI-A-START state.

5.1.5.3.14.7 Upon receipt of a D-DATA indication:

5.1.5.3.14.7.1 If in the LI-A-ACTIVE state, the air LI module shall:

a) pass the user data to the module as defined in Table 5-17, and

b) remain in the LI-A-ACTIVE state.

Table 5-17: PDU to air module mapping

PDU type	*Sub-element*	*Air Module*
ADS-contract-PDU	*contract-type*	
	"demand-contract"	DC
	"event-contract"	EC
	"periodic-contract"	PC
ADS-cancel-all-contracts-PDU		HI
ADS-cancel-contract-PDU	*CancelContract*	
	"event-contract"	EC
	"periodic-contract"	PC

5.1.5.3.14.8 Upon receipt of a D-END indication:

5.1.5.3.14.8.1 If in the LI-A-ACTIVE state:

5.1.5.3.14.8.2 If the *User Data* parameter value contains an ADS-cancel-all-contracts-PDU, the air LI module shall:

a) pass it to the air HI module,

b) invoke D-END response with the *Result* parameter value set to "accepted" and aDS-cancel-positive-acknowledgement-PDU ("cancel-all-contracts") in the user data,

c) request the DC, EC and PC modules to stop operation, and

d) enter LI-A-IDLE state.

5.1.5.3.14.8.3 If there is no user data, the air LI module shall:

a) invoke D-END response with the *Result* parameter value set to "accepted" and no user data, and

b) enter LI-A-IDLE state.

5.1.5.3.14.9 Upon receipt of a D-ABORT indication, the air LI module shall:

a) pass the D-ABORT indication to the ADS-C air AB module, and

b) enter the LI-A-IDLE state.

5.1.5.3.14.10 Upon receipt of a D-P-ABORT indication, the air LI module shall:

a) pass the D-P-ABORT indication to the ADS-C air AB module, and

b) enter the LI-A-IDLE state.

5.1.5.4 Exception Handling

5.1.5.4.1 Timer Expires

When any of the timers in any of the modules stated in 5.1.5.2 reaches its maximum time, the module shall request the air or ground AB module to abort with reason timer-expiry.

5.1.5.4.2 Unrecoverable System Error

When any module has an unrecoverable system error, the module should request the air or ground AB module to abort with reason unrecoverable-system-error.

5.1.5.4.3 Invalid PDU

5.1.5.4.3.1 When the *user data* parameter value of a D-START indication is a valid APDU and is not an ADS-contract-PDU, the air LI module shall request the ADS-C AB module to abort with reason invalid-PDU.

5.1.5.4.3.2 When the *user data* parameter value of a D-START confirmation is a valid APDU and is not an ADS-accepted-PDU, an ADS-positive-acknowledgement-PDU, an ADS-rejected-PDU or an ADS-ncn-PDU the ground LI module shall request the ADS-C AB module to abort with reason invalid-PDU.

5.1.5.4.3.3 When the *user data* parameter value of a D-DATA indication is a valid APDU and is not an ADS-contract-PDU or an ADS-cancel-contract-PDU, the air LI module shall request the ADS-C AB module to abort with reason invalid-PDU.

5.1.5.4.3.4 When the *user data* parameter value of a D-DATA indication is a valid APDU and is not an ADS-report-PDU, an ADS-accepted-PDU, an ADS-positive-acknowledgement-PDU, and ADS-rejected-PDU, or an ADS-ncn-PDU, the ground LI module shall request the ADS-C AB module to abort with reason invalid-PDU.

5.1.5.4.3.5 When the *user data* parameter value of a D-END indication is present, but does not contain an ADS-cancel-all-contracts-PDU, the air LI module shall request the ADS-C AB module to abort with reason invalid-PDU.

5.1.5.4.3.6 When the *user data* parameter value of a D-ABORT indication is a valid APDU and is not an ADS-provider-abort-PDU or an ADS-user-abort-PDU, the air LI module or the ground LI module shall request the ADS-C AB module to abort with reason invalid-PDU.

5.1.5.4.3.7 When the *user data* parameter value of a D-END confirmation is present, but does not contain an ADS-positive-acknowledgement-PDU (cancel-all-contracts), the ground LI module shall request the ADS-C AB module to abort with reason invalid-PDU.

5.1.5.4.4 Sequence Error

5.1.5.4.4.1 When a PDU is passed to a module for which there are no instructions in 5.1.5.3 (i.e. the PDU has arrived out of sequence), the air or ground AB module shall be requested to abort with reason sequence-error.

5.1.5.4.4.1 Upon receipt of a Dialogue service primitive for which there are no instruction in 5.1.5.3 (i.e. the primitive was not expected or was expected under other conditions or with other parameter values), the air or ground AB module shall be requested to abort with reason sequence-error.

5.1.5.4.5 D-START Rejection

5.1.5.4.5.1 Upon receipt of a D-START confirmation with a negative *Result parameter* value (i.e. containing the abstract value “rejected (transient)” or “rejected (permanent)”), and the *reject source* parameter value containing the abstract value “DS user”, the ground LI module shall:

a) request the ADS-C ground AB module to abort with reason sequence-error; and

b) enter the LI-G-IDLE state.

5.1.5.4.5.2 Upon receipt of a D-START confirmation with a negative *Result parameter* value (i.e. containing the abstract value “rejected (transient)” or “rejected (permanent)”), and the *reject source* parameter value containing the abstract value “DS provider”, the ground LI module shall:

a) request the ADS-C ground AB module to abort with reason “cannot-establish-contact”; and

b) enter the LI-G-IDLE state.

5.1.5.4.6 D-END Rejection

Upon receipt of a D-END confirmation with the *Result parameter* value containing the abstract value rejected, the ADS-C ground AB module shall be requested to abort with *reason* “dialogue-end-not-accepted”.

5.1.5.4.7 Decoding Error

5.1.5.4.7.1 When the air LI module or the ground LI module fails to decode an APDU, the LI module shall request the ADS-C AB module to abort with *reason* “decoding-error”.

5.1.5.4.7.2 Upon receipt of a D-START indication, a D-START confirmation or a D-DATA indication with no user data, the air or ground AB module shall be requested to abort with *reason* “decoding-error”.

5.1.5.4.8 Invalid QOS

Upon receipt of a D-START indication with the *application service priority* parameter set to a value other than the abstract value “high priority flight safety messages”, or the *RER quality of service* parameter set to a value other than the abstract value “low” or the *Routing Class quality of service* parameter set to a value other than one identifying the traffic category “Air Traffic Service Communications (ATSC)”, the air LI module shall request the ADS-C air AB module to abort with *reason* “invalid-qos-parameter”.

5.1.5.4.9 Invalid Security Parameter

5.1.5.4.9.1 Upon receipt of a D-START indication with the *Security Requirements* parameter not consistent with the local security policy, the air LI module shall:

a) request the ADS-C air AB module to abort with reason “communications-service-failure”; and

b) enter the LI-A-IDLE state.

5.1.5.4.9.2 Upon receipt of a positive D-START confirmation with the *Security Requirements* parameter value containing an abstract value no equal to the abstract value set in the D-START request *Security Requirements* parameter, the ground LI module shall:

a) request the ADS-C ground AB module to abort with reason “communications-service-failure”; and

b) enter the LI-G-IDLE state.

5.1.5.5 ADS-C ASE State Tables

5.1.5.5.1 Priority

5.1.5.5.1.1 If the state tables for the ADS-air-ASE and the ADS-ground-ASE shown below conflict with textual statements made elsewhere in this document, the textual statements shall take precedence.

5.1.5.5.1.2 In the following state tables, the statement “cannot occur” means that if the implementation conforms to this Specification, it is impossible for this event to occur. If the event does occur, this implies that there is an error in the implementation. If such a situation is detected, it is suggested that the ASE aborts with the error “unrecoverable error”.

5.1.5.5.1.3 In the following state tables, the statement “not permitted” means that the implementation must prevent this event from occurring through some local means. If the event does occur this implies that there is an error in the implementation. If such a situation is detected, it is suggested that the ASE performs a local rejection of the request rather than aborting the dialogue.

Table 5-18: ADS-C ground DC module state table

State ⇒ *Event* ⇓	*DC-G-IDLE* (Initial State)	*DC-G-PENDING*	*DC-G-ACTIVE*
Primitive Requests and Responses			
ADS-demand-contract req	Send ADS-contract-PDU (demand-contract) Start t-DC-1 *DC-G-PENDING*	Not permitted	Not permitted
ADS-C downlink PDUs			
ADS-accepted-PDU (demand-contract)	request AB to abort	Stop t-DC-1 ADS-demand-contract cnf (accepted) *DC-G-IDLE*	request AB to abort
ADS-report-PDU (demand-contract)	request AB to abort	request AB to abort	stop t-DC-2 ADS-report ind (demand-contract) *DC-G-IDLE*
ADS-rejected-PDU (demand-contract)	request AB to abort	stop t-DC-1 ADS-demand-contract cnf (rejected) *DC-G-IDLE*	request AB to abort
ADS-positive-acknowledgement-PDU (demand-contract)	request AB to abort	stop t-DC-1 ADS-demand-contract cnf (pos ack) start t-DC-2 *DC-G-ACTIVE*	request AB to abort
ADS-ncn-PDU (demand-contract)	request AB to abort	stop t-DC-1 ADS-demand-contract cnf (ncn) start t-DC-2 *DC-G-ACTIVE*	request AB to abort
Requests from other modules			

Request to stop operation	DC-G-IDLE	stop t-DC-1 *DC-G-IDLE*	stop t-DC-2 *DC-G-IDLE*
Timer expiry			
t-DC-1	cannot occur	request AB to abort	cannot occur
t-DC-2	cannot occur	cannot occur	request AB to abort

Table 5-19: ADS-C air DC module state table

State ⇒ *Event* ⇓	*DC-A-IDLE* (Initial State)	*DC-A-PENDING*	*DC-A-ACTIVE*
Primitive Requests and Responses			
ADS-demand-contract rsp (rejected)	Not permitted	Send ADS-rejected-PDU (demand-contract) *DC-A-IDLE*	Not permitted
ADS-demand-contract rsp (non compliance notification)	Not permitted	Send ADS-ncn-PDU (demand-contract) *DC-A-ACTIVE*	Not permitted
ADS-demand-contract rsp (accepted)	Not permitted	Send ADS-accepted-PDU (demand-contract) *DC-A-IDLE*	Not permitted
ADS-demand-contract rsp (positive acknowledgement)	Not permitted	Send ADS-positive-acknowledement-PDU (demand contract) *DC-A-ACTIVE*	Not permitted
ADS-report req (demand-contract)	Not permitted	Not permitted	Send ADS-report-PDU (demand-contract) DC-A-IDLE
Requests from other modules			
Requests to stop operation	*DC-A-IDLE*	*DC-A-IDLE*	*DC-A-IDLE*
ADS-C uplink PDUs			
ADS-contract-PDU (demand-contract)	ADS-demand-contract ind *DC-A-PENDING*	request AB to abort	request AB to abort

Table 5-20: ADS-C ground EC module state table

State ⇒ *Event* ⇓	*EC-G-IDLE* (Initial State)	*EC-G-START-PENDING*	*EC-G-ACTIVE*	*EC-G-PENDING*	*EC-G-CANCEL*
Primitive Requests and Responses					
ADS-event-contract req	Send ADS-contract-PDU (event) Start t-EC-1 *EC-G-START-PENDING*	Not permitted	Send ADS-contract–PDU Start t-EC-1 *EC-G-PENDING*	Not permitted	Not permitted
ADS-cancel req (event)	Not permitted	Not permitted	Send ADS-cancel-PDU Start t-EC-2 *EC-G-CANCEL*	Not permitted	Not permitted
ADS-C Aircraft PDUs					
ADS-positive-acknowledgementPDU (event)	request AB to abort	Stop t-EC-1 ADS-event-contract cnf *EC-G-ACTIVE*	request AB to abort	Stop t-EC-1 ADS-event-contract cnf *EC-G-ACTIVE*	request AB to abort
ADS-report-PDU (event)	request AB to abort	request AB to abort	ADS-report ind *EC-G-ACTIVE*	ADS-report ind *EC-G-PENDING*	ADS-report ind *EC-G-CANCEL*
ADS-accept-PDU (event)	request AB to abort	stop t-EC-1 ADS-event-contract cnf *EC-G-ACTIVE*	request AB to abort	stop t-EC-1 ADS-event-contract cnf *EC-G-ACTIVE*	request AB to abort
ADS-positive-acknowledgement-PDU (cancel-contract-event)	request AB to abort	request AB to abort	request AB to abort	request AB to abort	Stop t-EC-2 ADS-cancel-contract cnf *EC-G-IDLE*
ADS-cancel-negative-acknowledgement-PDU (no-such-contract)	request AB to abort	request AB to abort	request AB to abort	request AB to abort	Stop t-EC-2 ADS-cancel-contract cnf *EC-G-IDLE*
ADS-rejected-PDU (event)	request AB to abort	stop t-EC-1 ADS-event-contract cnf *EC-G-IDLE*	request AB to abort	stop t-EC-1 ADS-event-contract cnf *EC-G-ACTIVE*	request AB to abort
ADS-ncn-PDU (event)	request AB to abort	stop t-EC-1 ADS-event-contract cnf *EC-G-ACTIVE*	request AB to abort	stop t-EC-1 ADS-event-contract cnf *EC-G-ACTIVE*	request AB to abort

Requests from other modules					
Requests to stop operation	*EC-G-IDLE*	stop t-EC-1 *EC-G-IDLE*	*EC-G-IDLE*	stop t-EC-1 *EC-G-IDLE*	stop t-EC-2 *EC-G-IDLE*
Timer expiry					
t-EC-1	cannot occur	request AB to abort	cannot occur	request AB to abort	cannot occur
t-EC-2	cannot occur	cannot occur	cannot occur	cannot occur	request AB to abort

Table 5-21: ADS-C air EC module state table

State ⇒ *Event* ⇓	*EC-A-IDLE* (Initial State)	*EC-A-PENDING*	*EC-A-ACTIVE*	*EC-A-ACTIVE-PENDING*
Primitive Requests and Responses				
ADS-event-contract rsp (positive acknowledgement)	Not permitted	Send ADS-positive-acknowledgement-PDU *EC-A-ACTIVE*	Not permitted	Send ADS-positive-acknowledgementPDU *EC-A-ACTIVE*
ADS-event-contract rsp (non compliance notification)	Not permitted	Send ADS-ncn-PDU *EC-A-ACTIVE*	Not permitted	Send ADS-ncn-PDU *EC-A-ACTIVE*
ADS-event-contract rsp (rejected)	Not permitted	Send ADS-rejected-PDU *EC-A-IDLE*	Not permitted	Send ADS-rejected-PDU *EC-A-ACTIVE*
ADS-event-contract rsp (accepted)	Not permitted	Send ADS-accepted-PDU *EC-A-ACTIVE*	Not permitted	Send ADS-accepted-PDU EC-A-ACTIVE
ADS-report req (event)	Not permitted	Not permitted	Send ADS-report-PDU *EC-A-ACTIVE*	Not permitted
Requests from other modules				
Requests to stop operation	*EC-A-IDLE*	*EC-A-IDLE*	*EC-A-IDLE*	*EC-A-IDLE*
ADS-C Ground PDUs				
ADS-contract-PDU(event)	ADS-event-contract ind *EC-A-PENDING*	request AB to abort	ADS-event-contract ind *EC-A-ACTIVE-PENDING*	request AB to abort
ADS-cancel-PDU (event)	Send ADS-cancel-negative-acknowledgement *EC-A-IDLE*	request AB to abort	ADS-cancel ind Send ADS-cancel-positive-acknowledgement *EC-A-IDLE*	request AB to abort

Table 5-22: ADS-C ground periodic-contract state table

State ⇒ *Event ⇓*	*PC-G-IDLE* (Initial State)	*PC-G-START-PENDING*	*PC-G-ACTIVE*	*PC-G-PENDING*	*PC-G-CANCEL*
Primitive Requests and Responses					
ADS-periodic-contract req	Send ADS-contract-PDU Start t-PC-1 *PC-G-START-PENDING*	Not permitted	Stop t-PC-2 Send ADS-contract-PDU Start t-PC-1 *PC-G-PENDING*	Not permitted	Not permitted
ADS-cancel req (periodic)	Not permitted	Not permitted	Stop t-PC-2 Send ADS-cancel-PDU Start t-PC-3 *PC-G-CANCEL*	Not permitted	Not permitted
ADS-C Aircraft PDUs					
ADS-accepted-PDU (periodic)	request AB to abort	Stop t-PC-1 ADS-periodic-cnf Start t-PC-2 *PC-G-ACTIVE*	request AB to abort	Stop t-PC-1 ADS-periodic-cnf Start t-PC-2 *PC-G-ACTIVE*	request AB to abort
ADS-report-PDU (periodic)	request AB to abort	request AB to abort	Stop t-PC-2 ADS-report ind Start t-PC-2 *PC-G-ACTIVE*	ADS-report ind *PC-G-PENDING*	ADS-report ind *PC-G-CANCEL*
ADS-positive-acknowledgement-PDU (periodic-contract)	request AB to abort	stop t-PC-1 ADS-periodic-contract cnf Start t-PC-2 *PC-G-ACTIVE*	request AB to abort	stop t-PC-1 ADS-periodic-contract cnf Start t-PC-2 *PC-G-ACTIVE*	request AB to abort
ADS-positive-acknowledgement-PDU (cancel-periodic-contract)	request AB to abort	request AB to abort	request AB to abort	request AB to abort	Stop t-PC-3 ADS-cancel-contract cnf *PC-G-IDLE*
ADS-cancel-negative-acknowledgement-PDU (no-such-contract)	Request AB to abort	Request AB to abort	Request AB to abort	Request AB to abort	Stop t-PC-3 ADS-cancel-contract cnf *PC-G-IDLE*

ADS-rejected-PDU (periodic)	request AB to abort	stop t-PC-1 ADS-periodic-contract cnf *PC-G-IDLE*	request AB to abort	stop t-PC-1 ADS-periodic-contract cnf Start t-PC-2 *PC-G-ACTIVE*	request AB to abort
ADS-ncn-PDU (periodic)	request AB to abort	stop t-PC-1 ADS-periodic-contract cnf Start t-PC-2 *PC-G-ACTIVE*	request AB to abort	stop t-PC-1 ADS-periodic-contract cnf Start t-PC-2 *PC-G-ACTIVE*	request AB to abort
Requests from other modules					
Request to stop operation	*PC-G-IDLE*	stop t-PC-1 *PC-G-IDLE*	*Stop t-PC-2* *PC-G-IDLE*	stop t-PC-1 *PC-G-IDLE*	stop t-PC-3 *PC-G-IDLE*
Request to suspend periodic contract			stop t-PC-2		
Request to reinstate periodic contract			start t-PC-2		
Timer expiry					
t-PC-1	cannot occur	request AB to abort	cannot occur	request AB to abort	cannot occur
t-PC-2	cannot occur	cannot occur	request AB to abort	cannot occur	cannot occur
t-PC-3	cannot occur	cannot occur	cannot occur	cannot occur	request AB to abort

Table 5-23: ADS-C air PC module state table

State ⇒ *Event* ⇓	*PC-A-IDLE* (Initial State)	*PC-A-PENDING*	*PC-A-ACTIVE*	*PC-A-ACTIVE-PENDING*
Primitive Requests and Responses				
ADS-periodic-contract rsp (positive acknowledgement)	Not permitted	Send ADS-positive-acknowledgement-PDU *PC-A-ACTIVE*	Not permitted	Send ADS-positive-acknowledgement-PDU *PC-A-ACTIVE*
ADS-periodic-contract rsp (non compliance notification)	Not permitted	Send ADS-ncn-PDU *PC-A-ACTIVE*	Not permitted	Send ADS-ncPDU *PC-A-ACTIVE*
ADS-periodic-contract rsp (reject)	Not permitted	Send ADS-rejected-PDU *PC-A-IDLE*	Not permitted	Send ADS-rejected-PDU *PC-A-ACTIVE*
ADS-periodic-contract rsp (accepted)	Not permitted	Send ADS-accepted-PDU *PC-A-ACTIVE*	Not permitted	Send ADS-accepted-PDU *PC-A-ACTIVE*
ADS-report req (periodic)	Not permitted	Not permitted	Send ADS-report-PDU *PC-A-ACTIVE*	Not permitted
Requests from other modules				
Requests to stop operation	*PC-A-IDLE*	*PC-A-IDLE*	*PC-A-IDLE*	*PC-A-IDLE*
ADS-C uplink PDUs				
ADS-contract-PDU (periodic)	ADS-periodic-contract ind *PC-A-PENDING*	request AB to abort	ADS-periodic-contract ind *PC-A-ACTIVE-PENDING*	request AB to abort
ADS-cancel-contract-PDU (periodic)	Send ADS-cancel-negative-acknowledgement *PC-A-IDLE*	request AB to abort	ADS-cancel ind Send ADS-positive-acknowledgement *PC-A-IDLE*	request AB to abort

Table 5-24: Ground ADS-C LI module state table

State ⇒ *Event* ⇓	*LI-G-IDLE* (Initial State)	*LI-G-START*	*LI-G-ACTIVE*	*LI-G-END*
Data and requests passed from other modules				
ADS-contract-PDU	D-START req *LI-G-START*	Not permitted	D-DATA req *LI-G-ACTIVE*	Not permitted
ADS-cancel-all-contracts req	Not permitted	Not permitted	start t-LI-1 D-END req *LI-G-END*	Not permitted
ADS-cancel-contract-PDU	Not permitted	Not permitted	D-DATA req [1]	LI-G-END
ADS-provider-abort-PDU, ADS-user-abort-PDU	*LI-G-IDLE*	D-ABORT req *LI-G-IDLE*	D-ABORT req *LI-G-IDLE*	D-ABORT req *LI-G-IDLE*
ADS-forward-contract-response-PDU	Not permitted	Not permitted	Not permitted	Not permitted
Primitive Indications and Confirmations				
D-START ind	pass user data to appropriate module *LI-G-START-R*	cannot occur	cannot occur	cannot occur
D-START cnf	cannot occur	pass user data to appropriate module [1]	cannot occur	cannot occur
D-DATA ind	cannot occur	cannot occur	pass user data to appropriate module [1]	pass user data to appropriate module *LI-G-END*
D-END-ind	cannot occur	cannot occur	if ADS-end-forward-service-PDU, pass to HI module D-END rsp *LI-G-IDLE*	cannot occur
D-END cnf	cannot occur	cannot occur	cannot occur	stop t-LI-1 *ADS-cancel-all-contracts cnf* *LI-G-IDLE*
D-ABORT ind or D-P-ABORT ind	cannot occur	pass to AB module *LI-G-IDLE*	pass to AB module *LI-G-IDLE*	pass to AB module *LI-G-IDLE*
Timer expiry				

t-LI-1	cannot occur	cannot occur	cannot occur	request AB to abort
[1] If DC, EC and PC modules are all in their idle state then Invoke D-END req with no user data LI-G-END Else LI-G-ACTIVE				

Table 5-25: Air ADS-C LI module state table

State ⇒ *Event* ⇓	*LI-A-IDLE* (Initial State)	*LI-A-START*	*LI-A-ACTIVE*
Data and requests passed from other modules			
ADS-accepted-PDU, ADS-rejected-PDU, ADS-ncn-PDU, ADS-positive-acknowledgement-PDU	Not permitted	D-START rsp *LI-A-ACTIVE*	D-DATA req *LI-A-ACTIVE*
ADS-report-PDU, ADS-negative-acknowledgement-PDU	Not permitted	Not permitted	D-DATA req *LI-A-ACTIVE*
Primitive Indications and Confirmations			
D-START ind	pass to appropriate module *LI-A-START*	cannot occur	cannot occur
D-DATA ind	cannot occur	cannot occur	pass user data to appropriate module *LI-A-ACTIVE*
D-END ind	cannot occur	cannot occur	if ADS-cancel-all-contracts-PDU, pass to HI module D-END rsp *LI-A-IDLE*
D-ABORT ind or D-P-ABORT ind	cannot occur	pass to AB module *LI-A-IDLE*	pass to AB module *LI-A-IDLE*

5.1.6 Communication Requirements

5.1.6.1 Encoding Rules

5.1.6.1.1 The ADS-C application shall use PER as defined in ISO/IEC 8825-2, using the Basic Unaligned variant to encode/decode the ASN.1 message structure and content specified in 4.

5.1.6.1.2 When encoded ADS-C APDUs are treated as bit-oriented values that are not padded to an integral number of octets, the length determinant includes only the significant bits of the encoding, corresponding to the ASN.1 type.

5.1.6.2 Dialogue Service Requirements

5.1.6.2.1 Primitive Requirements

5.1.6.2.1.1 Where dialogue service primitives, that is D-START, D-END, D-ABORT, D-P-ABORT and D-DATA are described as being invoked in 5.1.5, the ADS-ground-ASE and the ADS-air-ASE shall exhibit external behaviour consistent with the dialogue service, as described in 5.14.2, having been implemented and its primitives invoked.

5.1.6.2.2 Quality of Service Requirements

5.1.6.2.2.1 The *application service priority* for ADS-C shall have the abstract value of “high priority flight safety messages”.

5.1.6.2.2.2 *The RER quality of service* parameter of the D-START request shall be set to the abstract value of “low”.

5.1.6.2.2.3 The ADS-C ASE shall map the class of communication service abstract values to the ATSC routing class abstract value part of the D-START *QOS* parameter as presented in Table 5-26.

Table 5-26: Mapping between class of communication and routing class abstract values

Class of Communication Abstract Value	*Routing Class Abstract Value*
A	Traffic follows Class A ATSC route(s)
B	Traffic follows Class B ATSC route(s)
C	Traffic follows Class C ATSC route(s)
D	Traffic follows Class D ATSC route(s)
E	Traffic follows Class E ATSC route(s)
F	Traffic follows Class F ATSC route(s)
G	Traffic follows Class G ATSC route(s)
H	Traffic follows Class H ATSC route(s)

5.1.6.2.2.4 ATSC values are defined in 1.3.

5.1.6.2.2.5 ATN Security Requirements

5.1.6.2.2.5.1 The *Security Requirements* parameter of the D-START request shall be set to the abstract value of either "secured exchange" or "no security" for the ADS-demand-contract, ADS-event-contract and ADS-periodic-contract services.

5.1.7 ADS-user Requirements

5.1.7.1 General

5.1.7.1.1 General Requirements

The ADS-ground-user shall only establish a demand contract, an event contract or a periodic contract with an ADS-air-user.

5.1.7.1.2 General Parameter Requirements

5.1.7.1.2.1 When an ADS-ground-user invokes ADS-demand-contract request, ADS-event-contract request, ADS-periodic-contract request or ADS-forward-contract request and requires a particular class of communication service, it provides the *Class of Communication Service* parameter.

5.1.7.1.2.2 When an ADS-ground-user invokes ADS-demand-contract request, ADS-event-contract request, ADS-periodic-contract request or ADS-forward-contract request, and does not provide the *Class of Communications Service* parameter, this indicates no routing preference.

5.1.7.1.2.3 When an ADS-ground-user specifies the *Class of Communications Service* parameter and there is an ADS-C contract in place, the parameter is ignored.

5.1.7.1.2.4 When a ADS-ground-user requires to establish a secure or unsecure air-ground ADS-C contracts, it sets the *Security Required* parameter as appropriate.

5.1.7.1.3 Timing Requirements

5.1.7.1.3.1 When a ADS-ground-user requires to establish a secure or unsecure air-ground ADS-C contracts, it sets the *Security Required* parameter as appropriate.

5.1.7.1.3.2 When an ADS-air-user or ADS-ground-user receives an indication that requires a response, it should invoke the response within the contract response time.

5.1.7.1.3.3 When a periodic contract is in place, the ADS-air-user should invoke ADS-report request (as described below) within the report generation time of the reporting interval as measured from the sending of the previous report.

5.1.7.1.4 Error Handling Requirements

5.1.7.1.4.1 If the ADS-air-user or ADS-ground-user has an unrecoverable system error, then it should invoke ADS-user-abort request for each affected peer system.

5.1.7.1.4.2 If the ADS-user receives an ADS-user-abort indication or an ADS-provider-abort indication, then it shall cease operation of all ADS-C contracts with the peer system to which the indication is related.

5.1.7.1.5 Miscellaneous Air User Requirements

With the permissible exception of ADS-user-abort and ADS-provider-abort, the ADS-air-user shall respond to indications and confirmations in the order in which they are received.

5.1.7.1.6 Miscellaneous Ground User Requirements

With the permissible exception of ADS-user-abort and ADS-provider-abort, the ADS-ground-user shall respond to indications and confirmations in the order in which they are received.

5.1.8 Subsetting Rules

5.1.8.1 This section specifies conformance requirements which all implementations of the ADS-C protocol obey.

5.1.8.2 An implementation of either the ADS-C ground based service or the ADS-C air based service claiming conformance to 5.1 shall support the ADS-C protocol features as shown in the tables below.

5.1.8.3 The 'status' column indicates the level of support required for conformance to the ADS-C ASE protocol described in the tables are as follows:

a) **'M'** mandatory support is required;

b) **'O'** optional support is permitted for conformance to the ADS-C protocol;

c) **'N/A'** the item is not applicable; and

d) **'C.n'** the item is conditional where n is the number which identifies the condition which is applicable.

Table 5-27: ADS-C Protocol Versions Implemented

	Status	*Associated Predicate*
Version 1	C	V1

C: a conformant implementation shall support one and only one of these two options

Table 5-28: ADS-C Protocol Functional Units

	Status	*Associated Predicate*
The ADS-C system acts as an airborne system	C.1	**ADS/air**
The ADS-C system acts as a ground system	C.1	**ADS/ground**
The ADS-C ground system can establish demand contract	If (ADS/ground) C.2 else N/A	**G-DC-FU**
The ADS-C ground system can establish event contracts	If (ADS/ground) C.2 else N/A	**G-EC-FU**
The ADS-C ground system can establish periodic contracts	If (ADS/ground) C.2 else N/A	**G-PC-FU**
The ADS-C air system can process demand contracts	If (ADS/air) C.3 else N/A	**A-DC-FU**
The ADS-C air system can process event contracts	If (ADS/air) C.3 else N/A	**A-EC-FU**
The ADS-C air system can process periodic contracts	If (ADS/air) C.3 else N/A	**A-PC-FU**

C.1: a conformant implementation shall support one and only one of these two options.
C.2: a conformant ground implementation shall support at least one of the three options.
C.3: a conformant air implementation shall support at least one of the three options.

Table 5-29: ADS-ground-ASE Conformant Configurations

	List of Predicates	*Functionality Description*
I	G-DC-FU + ADS/ground	ADS-ground-ASE supporting demand contract only Demand contract only can be established with the aircraft
II	G-EC-FU + ADS/ground	ADS-ground-ASE supporting event contracts Event contracts can be established with the aircraft
III	G-PC-FU + ADS/ground	ADS-ground-ASE supporting periodic contracts Periodic contracts can be established with the aircraft
IV	G-DC-FU + G-EC-FU + ADS/ground	ADS-ground-ASE supporting demand and event contracts Demand and Event contracts can be established with the aircraft
V	G-DC-FU + G-PC-FU + ADS/ground	ADS-ground-ASE supporting demand and periodic contracts Demand and Periodic contracts can be established with the aircraft
VI	G-EC-FU + G-PC-FU + ADS/ground	Event and Periodic contracts can be established with the aircraft
VII	G-DC-FU + G-EC-FU + G-PC-FU + ADS/ground	Demand, Event and Periodic contracts can be established with the aircraft

Table 5-30: ADS-air-ASE Conformant Configurations

	List of Predicates	*Functionality Description*
I	ADS/air + A-DC-FU	Demand contracts only can be established with the ground system. A Reject Contract with reason “ADS-service unavailable” is sent when Event or Periodic contracts are requested.
II	ADS/air + A-EC-FU	Event contracts only can be established with the ground system. A Reject Contract with reason “ADS-service unavailable” is sent when contracts are requested.
III	ADS/air + A-PC-FU	Periodic contracts only can be established with the ground system. A Reject Contract with reason “ADS-service unavailable” is sent when Demand or Event contracts are requested.
IV	ADS/air + A-DC-FU + A-EC-FU	Demand and Event contracts can be established with the ground system. A Reject Contract with reason “ADS-service unavailable” is sent when Periodic contracts are requested.
V	ADS/air + A-DC-FU + A-PC-FU	Demand and Periodic contracts can be established with the ground system. A Reject Contract with reason “ADS-service unavailable” is sent when Event contracts are requested.
VI	ADS/air + A-EC-FU + A-PC-FU	Event and Periodic contracts can be established with the ground system. A Reject Contract with reason “ADS-service unavailable” is sent when Demand contracts are requested.
VII	ADS/air + A-DC-FU + A-EC-FU +A-PC-FU	Demand, Event and Periodic contracts can be established with the ground system.

5.2 ADS REPORT FORWARDING APPLICATION

(To be developed.)

Chapter 6

ATN MESSAGE INTEGRITY CHECK ALGORITHM

6.1 USE OF THE INTEGRITY CHECK FUNCTION

6.1.1 The integrity check algorithm specified in this chapter may optionally be invoked by the ATN application-user specification to provide a proof of message integrity. If other information (e.g. sender and/or receiver identity) is also bound to the message when the integrity sequence is computed and verified, then it can also provide additional proof (e.g. a proof of delivery to an intended recipient).

6.1.2 By default, an application integrity check shall be computed using the default ATN message checksum algorithm specified in this chapter.

6.1.3 Other integrity check algorithms may be used and their use identified by including, in the application message, a relative OID that identifies the algorithm alongside the computed integrity check. However, the use of an alternative integrity check algorithm must be agreed in advance by both air and ground users using a procedure outside of the scope of this specification.

6.2 DEFAULT ATN MESSAGE CHECKSUM ALGORITHM

6.2.1 General

6.2.1.1 The default ATN message checksum algorithm generates a 32-bit (4-octet) checksum.

6.2.1.2 The default ATN message checksum algorithm is a 32-bit version of the Fletcher algorithm. This has been chosen in order to satisfy requirements for high data integrity with good performance characteristics.

6.2.1.3 The object identifier “atn-default-checksum”, as defined below, identifies the default ATN message checksum algorithm:

atn-default-checksum ::= OBJECT IDENTIFIER {iso(1) identified organization(3) icao(27) atn-algorithms(9) atc-chk32(0)}

6.2.2 Symbols

6.2.2.1 The symbols used in the ATN application-user specification are as follows:

C0, C1, C2, C3 are variables used by the algorithm.

i is the number (i.e. position) of an octet within the message to be protected.

L is the length of the message to be protected, in octets.

X_j is the value of the j'th octet of the checksum BIT STRING, where X_0 is the first octet in the BIT STRING; X_1 is the second octet; X_2 is the third octet; and X_3 is the fourth octet.

6.2.2.2 The data to be protected is a bit string provided by the user of this algorithm.

6.2.3 Arithmetic conventions

6.2.3.1 Addition shall be performed in one of the two following modes:

a) modulo 255; or

b) ones complement arithmetic.

6.2.3.2 In the ones complement mode, if any of the variables has the value minus zero (i.e. 0xFFFF), it shall be regarded as though it were plus zero (i.e. 0).

6.2.4 Algorithm for checksum generation

6.2.4.1 If the data to be protected is not an integral number of octets, it shall be right padded to the next octet boundary by appending bits with value “zero” (0).

6.2.4.2 The ATN message checksum shall be generated by the following algorithm:

a) Initialize C0, C1, C2 and C3 to zero.

b) Process each octet in the message sequentially from i = 1 to L by:

1) adding the value of the octet to C0; and

2) then adding the value of C0 to C1, C1 to C2, and C2 to C3.

c) Set the octets of the checksum as follows:

1) X0 = - (C0 + C1 + C2 + C3);

2) X1 = C1 + 2*C2 + 3*C3;

3) X2 = - (C2 + 3*C3); and

4) X_3 = C3.

6.2.5 Algorithm for checksum verification

When the integrity of a message is to be verified using this algorithm, the verification function shall:

a) append to the message the octets of the checksum field in the same order in which they appear in the checksum BIT STRING;

b) initialize C0, C1, C2 and C3 to zero;

c) process each octet in the message, including the appended checksum octets, sequentially from i = 1 to L by:

   1) adding the value of the octet to C0; and

   2) then adding the value of C0 to C1, C1 to C2, and C2 to C3;

d) discard the appended checksum octets; and

e) if, when all the octets have been processed, all of the variables C0, C1, C2 and C3 have the value zero, then a “success” indication is given to the algorithm user. Otherwise, a “verification failure” indication is given to the algorithm user.

— END —

ISBN 978-92-9258-140-4