1. NTRODUCTION

1.1 Purpose

1.1.1 In line with normal ICAO practice, this document was developed as a companion document to the ATN Controller Pilot Data Link Communication (CPDLC) Standards And Recommended Practices (SARPs). It may be read alongside the CPDLC SARPs, in order to provide a greater understanding of the specification itself, or it may be read instead of the CPDLC SARPs by readers that simply want to understand the purpose of the CPDLC Application rather than the detail of the specification.

1.2 Scope

1.2.1 This document provides guidance material for those implementing the CPDLC Application.

1.2.2 This document does not define any mandatory or optional requirements for CPDLC, neither does it define any recommended practices. This document is not intended to instruct user on how to use CPDC in a particular operational environment.

1.2.3 The CPDLC application SARPs deal exclusively with Air Traffic Services (ATS). Aeronautical Operational Control (AOC) may choose to use the CPDLC SARPs as a model for their own applications.

1.3 History

1.3.1 The CPDLC application allows a pilot and controller to exchange messages. In a fully operational ATS data link environment, CPDLC is expected to be used as the main means of pilot/controller dialogue with voice available for backup.

1.3.2 The ADS Panel has developed the operational requirements for the CPDLC application over a period of several years. These requirements are specified in the Draft ICAO Manual of ATS Data Link Applications, submitted to the 2nd Meeting of the ADS Panel in September 1996 (see 1.8.5). In addition to the operational requirements specified for CPDLC, the ADSP has specified that the CPDLC application should conform to the ATN protocols for its data link operations.

1.3.3 In the initial implementation of CPDLC, CPDLC messages, their format and intent, were based on the relevant ICAO documentation, in particular Annexes 3 and 11 and Doc 4444, Procedures for Air Navigation and Rules of the Air (PANS/RAC). The format and content of the messages was developed from the voice message set documented in Doc 4444.

1.3.4 At its first meeting, the ATN Panel established a working group (WG3) to develop SARPs for the applications and upper layers. The working group, established a three sub-groups, one of which (SG2), was tasked to develop validated SARPs and Guidance material for four applications: Context Management (CM), Automatic Dependent Surveillance (ADS), Controller Pilot Data Link Communication (CPDLC), and Flight Information Services (FIS).

1.3.5 The initial development of the CPDLC SARPs was relatively straight-forward: the exchange of data-link messages between a pilot and controller. However it was recognized that the use of data link is not as flexible as

voice, and a set of rules has had to be developed indicating, for example, how a dialogue is opened and closed, how CPDLC dialogues are transitioned between data authorities, how communication with downstream data authorities could take place, and how CPDLC messages could be exchanged ground-ground, to complement the basic air-ground nature of CPDLC.

1.3.6 Some of the CPDLC capabilities such as the use of downstream clearances and ground-ground CPDLC message forwarding were seen as having limited applicability. States would not be willing to incur costs of implementing the complete CPDLC if they only intended to use certain elements of the application. The SARPs therefore took account of the need to separate out the functionalities to enable partial implementation, while still retaining the interoperability required by the ICAO Standards. This led to the development of subsetting rules, and the identification of conformant configurations.

1.3.7 The ATNP worked very closely with the ADSP to ensure that the development of both the operational concepts and the technical means of achieving them are consistent. However, the ADSP generally looked at a longer timescale than the current ATNP initial implementation programme, and this mean that some elements of their work have not been incorporated into the present SARPs.

1.3.8 The ATNP identified a set of packages to accommodate the continuous specification of operational requirements by the ADSP. This document provides guidance material for Version 1 of the CPDLC application. This has been previously known as CNS/ATM-1.

1.3.9 ICAO approved the SARPs and established a configuration control board (CCB) to manage any changes required to the SARPs.

1.4 Structure

1.4.1 Chapter 1 - NTRODUCTION - This chapter contains the reason for providing guidance material as well as the scope. In addition, it provides a brief overview of the CPDLC functionality, CPDLC's relationship with other SARPs, and identifies applicable reference documents.

1.4.2 Chapter 2 - OVERALL GENERAL FUNCTIONALITY - This chapter describes generic concepts that are used throughout the CPDLC SARPs and guidance material. This chapter also covers some implementation issues that are not addressed in the SARPs.

1.4.3 Chapter 3 - CPDLC SERVICE DESCRIPTION - This chapter gives a functional breakdown of the various services that CPDLC provides. It describes a peer to peer interaction, including reasons for why particular information is used or not used, and what operations on the information are expected.

1.4.4 Chapter 4 - CPDLC SECTION DESCRIPTION- This chapter clarifies any functionality that was not addressed in Chapter 3 on a CPDLC SARPs section by section basis.

1.4.5 Chapter 5 - DIMENSIONS - This chapter gives some sample encoding sizes for guidance on what capacities need to be allowed for in order to meet normal operational expectations.

1.4.6 Chapter 6 - INDEXES / TABLES - This chapter provides a CPDLC variables glossary, and hierarchical relationships between the CPDLC message elements and variables.

1.4.7 Chapter 7 - EXAMPLE SCENARIOS - This chapter gives some examples as to what typical scenarios are expected in course of normal CPDLC operation.

1.4.8 Chapter 8 - EXAMPLE ENCODING - This chapter outlines some actual sample PER encoding of typical CPDLC messages.

1.5 CPDLC Application Overview

1.5.1 This chapter gives a high level overview of the CPDLC application, as an application whose primary function is to allow the data link exchange of message betwee a controller an a pilot. It contains an outline description of the functions which CPDLC application provides, namely:

a) Controller-Pilot Message Exchange Function - allows pilots and controller to communicate via data link,

b) Transfer of Data Authority Function - allows a transfer in data authority without loss of an available CPDLC data link connction,

c) Down Stream Clearance Function - allows receipt of clearances affecting an aircraft's future flight plan from a data authority not currently having data authority, and

d) Ground Forward Function - allows a ground system to forward CPDLC messages to another ground system.

1.5.2 Controller-Pilot Message Exchange Function

1.5.2.1 The Controller-Pilot Message Exchange Function defines a method for a controller and pilot to exchange messages via data link. This function provides messages for the following :

a) general information exchange;

b) clearance

    1) delivery,

    2) request, and

    3) response;

c) altitude/identity surveillance;

d) monitoring of current/planned position;

e) advisories

    1) request and

    2) delivery;

f) system management functions; and

g) emergency situations.

1.5.3 Transfer of Data Authority Function

1.5.3.1 The Transfer of Data Authority Function provides the capability for a smooth transition of the CPDLC function when an aircraft transfers from one data authority to the subsequent one. The data authority having the

capability to have an active (exchange of CPDLC messages affecting the aircraft's current flight path) CPDLC dialogue with an aircraft is known as the Current Data Authority (CDA). A given aircraft may only have one CDA at a time. Only a CDA to allowed to designate another ground system as the Next Data Authority (NDA). An inactive (no exchange of CPDLC messages)CPDLC dialogue can be opened with or by the NDA at a time before it becomes the CDA. This capability is intended to prevent a loss of communication that would occur if a NDA were prevented from actually establisheing a dialogue with an aircraft until it became the CDA. The designation of a NDA is accomplished using a CPDLC message. A CDA is not required to designate a NDA, in which case no NDA connection is established. The CDA may also change the NDA, in which case any connection in place with the now previous NDA is terminated, and a connection is established with the newly designated NDA. The CDA may also cancel a NDA without designating another NDA, in which case any connection in place with the cancelled NDA is terminated. An aircrfat may have only one NDA connection at a time.

1.5.4 Down Stream Clearance Function

1.5.4.1 The Down Stream Clearance Function provides the capability for an aircraft to contact an air traffic service unit which is not the current data authority, but is expected to be in the future, for the purpose of receiving a DownStream Clearance (DSC). This data authority is designated the Downstream Data Authority (DDA). Downstream clearance information information is exchanged using CPDLC message(s). Care must be taken that downstream clerance exhanges do not affect an aircraft's current flight trajectory. To satisfy this requirement, a set of CPDLC messages elements permitted to be exchanged with a DDA is an operationaly restricted subset of the full CPDLC message set. A list of the permitted message elements is found in the Draft ICAO Manual of Air Traffic Services (ATS) Data Link Applications. Note that this function could also be accomplished using ground-ground connectivity whith a CDA obtaining the required information from a DDA and providing it to an aircraft using the CDA CPDLC connection. The DSC functionality has been included since it has been recognized that ground-ground connectivity is not global.

1.5.5 Ground Forward Function

1.5.6 The Ground Forward Function provides the capability for a ground system to forward a CPDLC message to another ground system. The ground forward function can be used by the controlling data authority to forward an aircraft request to the next data authority, so that an 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 CPDLC message(s). It is a one-way forwarding of information with an indication of success, failure or non-support from the receiving ground system. A ground foward CPDLC connections can be used to forward one message and is then terminated.

1.5.7 Mapping of CPDLC Functionailty To SARPs CPDLC Services

1.5.7.1 Unlike the other appliction SARPs (CM, ADS, and FIS), there is not a one-to-one mapping (except abort services) between the CPDLC functions (operational) described above and the CPDLC services (technical) defined in the SARPs. This is due to the fact that all of the above functions exhange a common set of CPDLC messages and have common technical capabilities. Thus the SARPs defines CPDLC services to allow the message exchange between air/ground or ground/ground. A mapping showing the CPDLC services used by a given CPDLC function is provided in the following table.

CPDLC Function	**CPDLC SARPs Service**
Controller Plot Message Exchange	CPDLC-Start Service CPDLC-Message Service CPDLC-End Service CPDLC-User-Abort Service CPDLC-Provider-Abort Service
Transfer of Data Authority	CPDLC-Start Service CPDLC-Message Service CPDLC-User-Abort Service CPDLC-Provider-Abort Service
DownStream Clearance Delivery	DSC-Start Service CPDLC-Message Service DSC-End Service CPDLC-User-Abort Service CPDLC-Provider-Abort Service
Ground Forward	CPDLC-Forward Service CPDLC-User-Abort Service CPDLC-Provider-Abort Service

Table 1.1 — Mapping of CPDLC Services to CPDLC Functions

1.6 Inter-relationships with Other SARPs

1.6.1 The pre-requisite for establishment of a CPDLC or DSC dialogue is the knowledge by the initiating peer of the name, address, and version number of the contacted peer. The Context Management (CM) application is the most natural way to obtain this information. There is no direct interaction between the CM application and the CPDLC application, however the CM user must make this information available for use by the CPDLC application.

1.6.2 The CPDLC SARPs also make use of the Upper Layer Application SARPs (sub-volume 4) to perform dialogue service functions required by the CPDLC application.

1.7 Structure of the SARPs

1.7.1 All the air-ground SARPs are produced to a standard format. This has greatly helped the maintenance of document stability, commonality and presentation. The CPDLC SARPs are no different in basic layout from all other air-ground applications SARPs.

1.7.2 The CPDLC SARPs constitute the third part of sub-volume 2.

1.7.3 Section 2.3.1 - INTRODUCTION - gives a very brief, high level description of CPDLC, as an application enabling CPDLC services to be provided to a pilot and controller via the exchange of messages between aircraft avionics and ground CPDLC systems. Since this overview contains no information directly related to the stipulation of specific standards, it is almost entirely written as series of informative notes.

1.7.4 Section 2.3.2 - GENERAL REQUIREMENTS - contains information and high level requirements for error processing and CPDLC version number requirements.

1.7.5 Section 2.3.3 - ABSTRACT SERVICE - defines the abstract service interface for the CPDLC Application. The CPDLC Application Service Element (CPDLC-ASE) abstract service is described from the viewpoint of the CPDLC-air-user, the CPDLC-ground-user, and the CPDLC-service-provider.

1.7.6 Section 2.3.4 - FORMAL DEFINITION OF MESSAGES - describes the contents of all permissible CPDLC messages through definition of the CPDLC ASN.1 abstract syntax. All possible combinations of message parameters and their range of values are detailed.

1.7.7 Section 2.3.5 - PROTOCOL DEFINITION - splits up the specification of the CPDLC protocol into three parts: sequence diagrams for the services covered by the abstract service, protocol descriptions and error handling for the CPDLC-Air-ASE and CPDLC-Ground-ASE, and State Tables.

1.7.8 Section 2.3.6 - COMMUNICATION REQUIREMENTS - specifies the use of Packed Encoding Rules (PER) to encode/decode the ASN.1 message structure and stipulates the Dialogue Service requirements, including Quality of Service (QOS).

1.7.9 Section 2.3.7 - CPDLC USER REQUIREMENTS - describes the requirements imposed on the CPDLC-users concerning CPDLC messages and interfacing with the CPDLC-ASEs.

1.7.10 Section 2.3.8 - SUBSETTING RULES - specifies conformance requirements which all implementations of the CPDLC protocol obey. The protocol options are tabulated, and indication is given as to whether mandatory, optional or conditional support is required to ensure conformance to the SARPs. These subsetting rules permit applications to be tailored to suit individual implementations, commensurate with the underlying task, while still maintaining an acceptable level of interoperability.

1.8 References

1.8.1 Controller Pilot Data Link Communication Application, Annex 10, Volume III, Part I, Chapter 3 (ATN), Appendix A, Sub-volume III - Air-Ground Applications, section 2.3.

1.8.2 Context Management Application, Annex 10, Volume III, Part I, Chapter 3 (ATN), Appendix A, Sub-volume III - Air-Ground Applications, section 2.1.

1.8.3 Upper Layers Communications Service, Annex 10, Volume III, Part I, Chapter 3 (ATN), Appendix A, Sub-volume IV.

1.8.4 Draft ICAO Manual of Air Traffic Services (ATS) Data Link Applications, ICAO ADS Panel.

2. OVERALL GENERAL FUNCTIONALITY

2.1 General

2.1.1 CPDLC is defined as a single application handling: CPDLC air-ground exchanges, DSC air-ground exchanges, and CPDLC ground-ground exchanges.

2.2 Topology

2.2.1 Air-Ground Links

2.2.1.1 The primary connection that is made in CPDLC is between an aircraft and the ground system responsible for control of the aircraft. This ground system is designated the Current Data Authority (CDA). This connection can be initiated by either the air or the ground. An aircraft can have only one CDA connection at any one time. The CDA connection is the only CPDLC connection permitting the exchange of control information that can affect the aircraft's current flight path. With the exception of abort conditions, the ground system is always responsible for initiating connection closure. While the CDA connection is active either the pilot (air-borne system) or controller (ground system) can initiate CPDLC message exchanges. This is core functionality of CPDLC and must be support by all CPDLC ground and airborne systems.

2.2.1.2 In addition to the CDA connection, a given aircraft can have a CPDLC connection with another ground system if instructed to do so by the CDA. This secondary connection is between the aircraft specified in the preceding paragraph and the Next Data Authority (NDA) as designated by the CDA. This connection can be initiated by either the air or the ground. An aircraft can have only one NDA connection at any one time, and this can only be established under direction by the CDA. The NDA connection CANNOT be used to exchange any operational messages. The NDA connection is used to facilitate the transfer of data authority between ground system without loss of CPDLC connectivity on the airborne side. With the exception of abort conditions, a NDA connection is not terminated by either the airborne or ground side, but rather it is converted to a CDA connection upon termination of the CDA connection. (Note that instructions from the CDA canceling a NDA or changing a NDA will result in an airborne abort of a NDA connection). This is core functionality of CPDLC and must be support by all CPDLC ground and airborne systems.

2.2.1.3 The CPDLC topology also permits another connection between an aircraft and a ground system This connection is to facilitate the delivery of a DownStream Clearance (DSC) from a Downstream Data Authority (DDA). The DSC connection can only be established and terminated (except aborts) by the airborne system. An aircraft can have only one DSC connection at any one time. The DSC connection is independent of any CDA or NDA connection. The DDA and the NDA can be the same ground system, and a NDA and DSC connection to the same ground system are considered operationally valid. If an airborne system has a DSC connection with a ground system, and a CDA connection is established with that ground system, the airborne system must terminate the DSC connection. A CDA connection and DSC connection to the same ground system is not considered operationally valid. Operationally the DSC connection is expected by be of short duration (request and receipt of a downstream clearance). While the DSC connection is active, either the pilot (air-borne system) or controller (ground system) can initiate CPDLC message exchanges. The operationally permissible message elements in DSC exchanges are defined in the ADSP Draft Manual. An airborne system does not need to support the DSC capability and to ba a conformant CPDLC configuration. A ground system must, at a minimum, recognize the request to establish a DSC (and can subsequently reject such a request) to have a conformant CPDLC configuration.

2.2.2 Ground-Ground Link

2.2.2.1 A given CPDLC ground system can establish a CPDLC link with another ground system for the purpose of forwarding a CPDLC message. The connection is essentially "one-shot" and is closed after receipt of confirmation

from the receiving ground system This is a one way exchange of operational information. A ground system does not need to implement the capability to establish a ground-ground CPDLC link, but must at a minimum recognize the request to establish ground-ground CPDLC connection (and can subsequently reject such a request) to have a conformant CPDLC configuration.

2.2.3 Illustration of Typical Toplogies

2.2.3.1 The following figure illustrates the CPDLC topology when an aircraft has a CDA and NDA connection. Ground connectivity is available for forwarding of CPDLC messages for transfer of data authority, as well as for delivery of downstream clearance information.

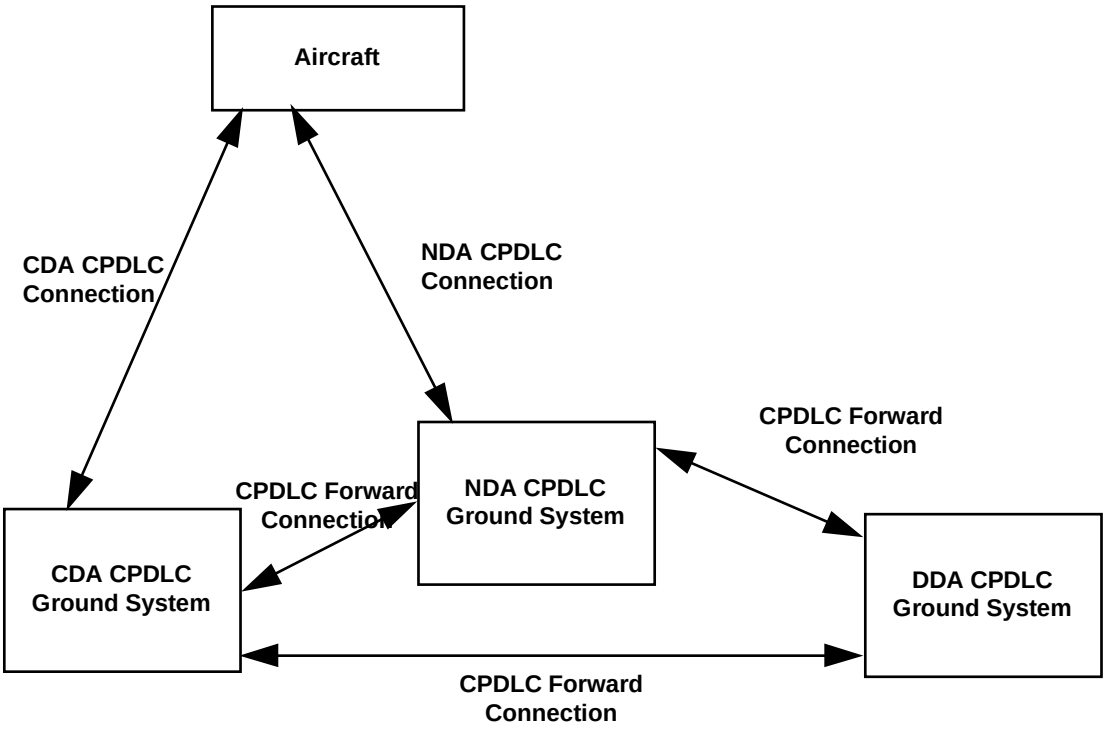


Figure 2.1:Aircraft With CDA and NDA Connection, Ground Ground CPDLC Connection

2.2.3.2 The following figure illutrate the CPDLC topology when an aircraft has a CDA, NDA, and DSC connection. Ground connectivity is available for forwarding of CPDLC messages for transfer of data authority, but not for delivery of downstream clearance information.

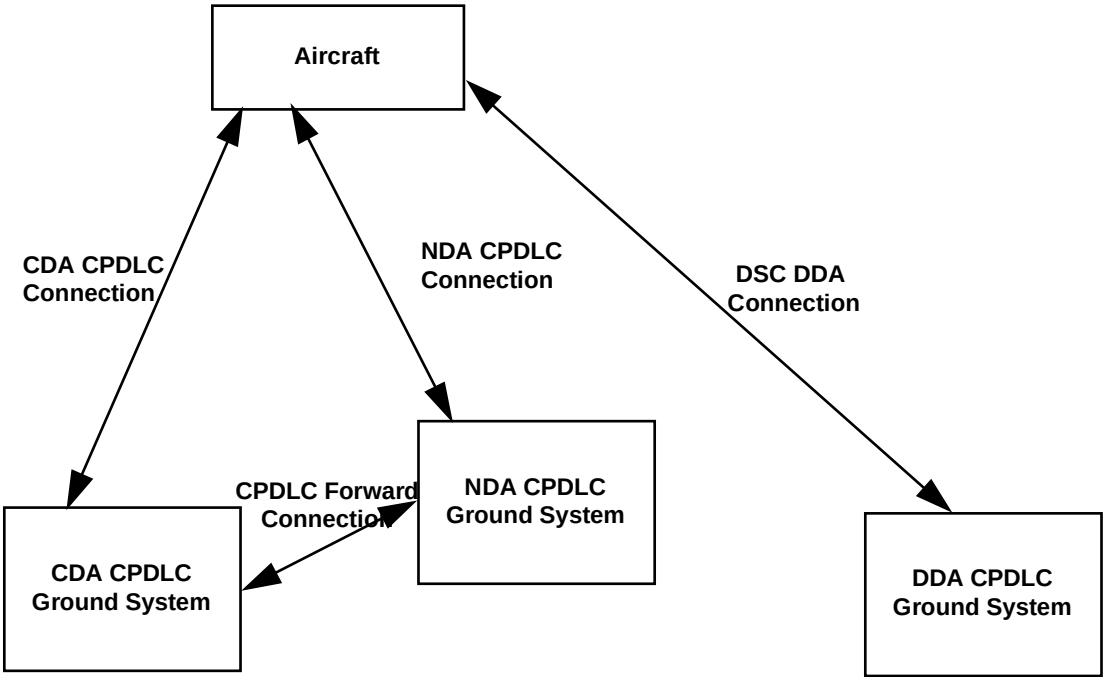


Figure 2.2:Aircraft With CDA, NDA and DDA Connection, No CDA-DDA Ground-Ground Connectivity

2.2.3.3 The following figure illutrate the CPDLC topology when an aircraft has a CDA, NDA, and DSC connection. Ground connectivity is not available, and the NDA and the DDAare the same ground system.

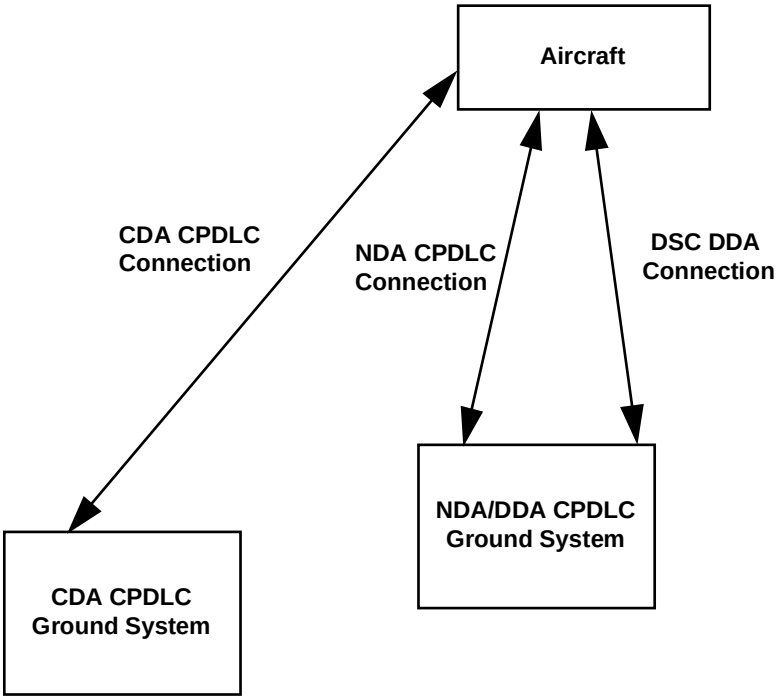


Figure 2.3:Aircraft With CDA, NDA, and DDA Connection, No Ground-Ground Connectivity

2.3 Abstract Internal Architecture

2.3.1.1 The architectural model for the CPDLC Application conforms to the ULA SARPs (Ref [2]). The architectural model is depicted in Figure 2.4. One Application Entity (AE) is defined per CPDLC Application. Each AE contains an ATN Application Service Element (ASE), which is the communication element responsible for the ATN Application. The CPDLC SARPs specify the CPDLC AE and CPDLC ASE.

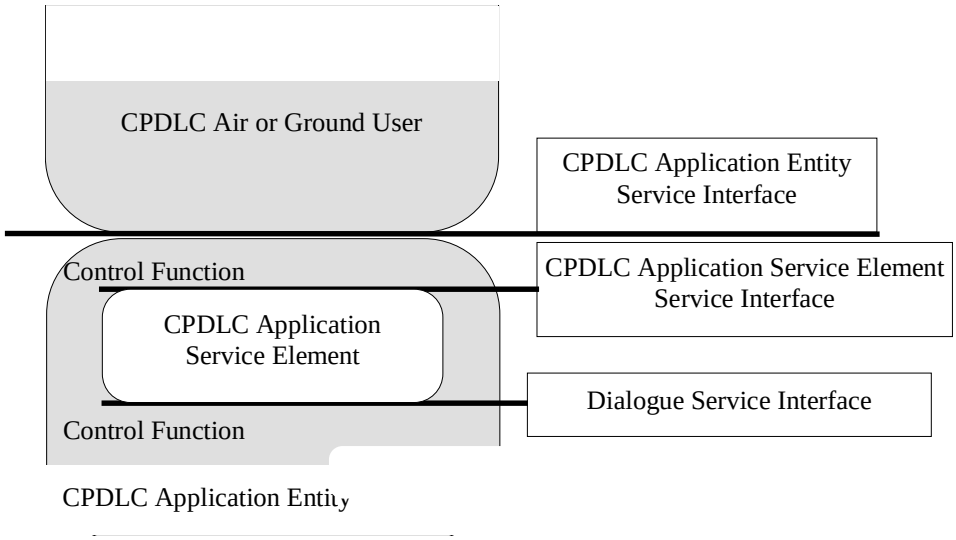


Figure 2.4. — CPDLC Architectural Model

2.3.2 Type of ASE in CPDLC

2.3.2.1 A single module handles the CPDLC protocol for the CPDLC application. If individual functions are not supported, as allowed by the subsetting rules (a conformant subset), the module will ensure that the protocol will the remaining subset of the full CPDLC functionality.

2.3.2.2 The CPDLC application defines only one type of ASE, the CPDLC-ASE. The CPDLC-ASE has two variations: the CPDLC-air-ASE and the CPDLC-ground-ASE. The CPDLC-air-ASE in turn has two functional modes: CPDLC or DSC. Both of these modes are used exclusively in air/ground links. The CPDLC-ground-ASE also has two functional modes: CPDLC or DSC. The DSC mode is used exclusively in air/ground links. The ground CPDLC funtional mode can in turn be either in a air/ground or ground/ground mode. A ground/ground CPDLC-ASE can be either an initiating or receiving ASE. The ground/ground CPDLC-ASE is used exclusively in the CPDLC-forward function. A CPDLC-ASE can act in only one mode at a time. For example a CPDLC-air ASE cannot be both in DSC mode and CPDLC mode at the same time. The different modes of operation are depicted in Figure 2.5.

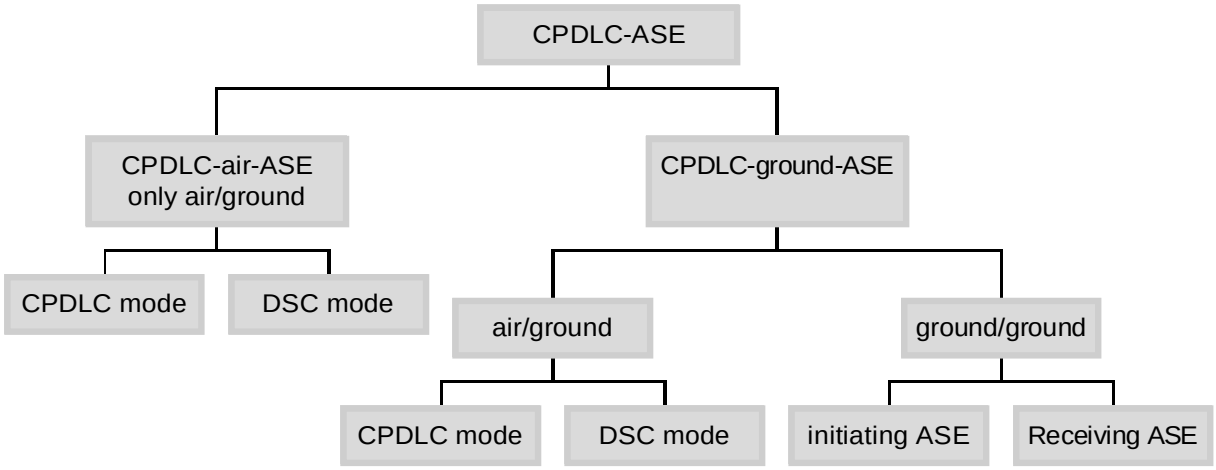


Figure 2.5: CPDLC-ASE Modes

2.3.3 CPDLC-ASE Functionality

2.3.3.1 The CPDLC-ASEs are responsible for all aspects air/ground and ground/ground communication related to CPDLC. They encode and decode the PDUs. They maintain a state machine that only allows PDUs to be sent at an appropriate time, and detect if the peer application send PDUs at an inappropriate time.

2.3.3.2 There is no architectural capability for multiple instances of the CPDLC-ASE within the same CPDLC AE. This implies that the CPDLC-ASE will generate and manage only one dialogue over the lifetime of the ASE

2.3.4 CPDLC-User Functionality

2.3.4.1 In the model of the CPDLC in Figure 2.4, there is a module called the CPDLC-user. The functionality of the CPDLC-users is described in Section 2.3.7 of the CPDLC SARPs. Either the CPDLC-air-user or the CPDLC-ground-user is permitted to initiate the air/ground CPDLC service. Only the CPDLC-air-user is permitted to initiate the DSC service. Once an air/ground CPDLC or DSC dialogue is established either user can initiate CPDLC message service. Only the CPDLC-ground-user can normally (non-abort conditions) initiate termination of a CPDLC dialogue. Only the CPDLC-air-user can normally (non-abort conditions) initiate termination of a DSC dialogue. Although there are requirements placed on the air/ground CPDLC-users, these tend to be the minimum required for interoperabilty. There is still a great deal of freedom in the method of implementation of this component.

2.3.4.2 The CPDLC-forward service is initiated by a CPDLC-ground-user. It is a ground-ground service. A CPDLC-forward dialogue is maintained just long enough for the initiator to receive confirmation on the forwarded CPDLC message. There is no concept of a two-way exchange of CPDLC messages in the forward service. The service (although technically confirmed) is operationally one-shot, one message operation. Although there are requirements placed on the ground/ground CPDLC-users, these tend to be the minimum required for interoperabilty. There is still a great deal of freedom in the method of implementation of this component.

2.3.5 Product Architecture

2.3.5.1 The CPDLC SARPs have defined an abstract model as a method to convey the CPDLC requirements. That is, the SARPs have split the functionality between the ASE and the user and has defined an abstract interface between the two. It should be noted that there is no requirement for an implementer to build such an interface.

2.3.5.2 Also note that the internal structure of the ATN ASE is not standardized across applications; for example, CPDLC and CM are defined as a single module while ADS and FIS are defined as multiple modules. The CPDLC module handles the protocol for the CPDLC application. If individual functions are not supported as allowed by the subsetting rules, the module will ensure that the protocol will handle the remaining subset of the full CPDLC functionality.

2.3.5.3 The dialogue service interface is the ATN ASE's view of the ATN Upper Layer Architecture. The CPDLC module uses the Dialogue Service for communication with the peer ASE through the ATN. The control function is used to for communication between different elements, such as the CPDLC-ASE and CPDLC-user.

2.4 Implementation Dependent Functionality

2.4.1 The CPDLC SARPs specify some of the requirements for the user, but leave a lot up to the implementor. There are no requirements that state how the user interface appears, how CPDLC interacts with the systems generating the CPDLC information, how CPDLC interacts with higher level functionality or with other applications such as CM. All this is implementation dependent.

2.4.2 The definition of the message elements composing a CPDLC message in the SARPs is quite open. It has not been possible to specify with the ASN.1 notation all the constraints and relationships between message elements. The rules for building a consistent CPDLC message are not checked by the CPDLC ASEs and have to be implemented by the CPDLC-users. There are a few constraints placed on message composition outlined in SARPs Section 2.3.7. These constraints are not checked by the sending or receiving ASE, but are requirements placed solely on the CPDLC-user.

2.5 Rationale for ASE / Users Split

2.5.1 The rationale for the split in functionality between the CPDLC ASEs and the CPDLC-users is as follows:

2.5.1.1 The ASE contains all functionality that is necessary to ensure the interoperability at the syntactic level. That is, two valid implementations of ASEs will be able to interact, passing data to each other in the correct order. They will be able to check the format of the data, ensure that it has been sent with the appropriate point in the dialogue, and also ensure that the peer ASE is behaving according to the requirements in the SARPs. Thus the ASE ensures interoperability.

2.5.1.2 The SARPs define some requirements for the users. These are the minimum user requirements necessary to ensure the semantic interoperability of the two peers. The user requirements explain how the data that is transported by the ASE should be interpreted. The user requirements explain how the following types of CPDLC/DSC dialogues operate:

a) a CPDLC dialogue between a CPDLC-air user and the Current Data Authority (CDA) CPDLC-ground-user,

b) a CPDLC dialogue between a CPDLC-air user and the Next Data Authority (NDA) CPDLC-ground-user,

c) a DSC dialogue between a CPDLC-air user and the Downstream Data Authority (DDA) CPDLC-ground-user, and

d) a CPDLC dialogue between two CPDLC-ground-users.

2.5.1.3 Some care has been taken to ensure that the requirements are not over-specified. That is, they do not specify rules which are not absolutely essential to the syntactic and semantic interoperability of the CPDLC function. This implementation dependent part can be built by different manufacturers in different ways, without

affecting the interoperability between different implementations. This implementation dependent part has not been specified in the SARPs, but should be specified by individual product manufacturers or regional standards.

2.6 Inter-relationship with other ATN Applications

2.6.1 There is no interaction required between the CPDLC Application and the other Version 1 Data Link applications. The CPDLC application is a stand alone application which can be developed, certified, installed and operated completely independently from the other ATN Applications.

2.6.2 However, in order to be able to initiate a CPDLC or DSC dialogue, some naming and addressing information have to be known from the dialogue initiator:

a) Establishment of air/ground dialogues:

   1) Ground Initiation (CPDLC):

      i) the ATN address of the airborne CPDLC system must be present somewhere in the ground data link communication system. Without this information, a CPDLC dialogue with a airborne CPDLC system can not be established.

      ii) the version number of the CPDLC protocol run on the airborne system must be available to the CPDLC-ground-user. The protocol version negotiation is not performed by the CPDLC Application Entities and the CPDLC-ground-user is responsible for checking the version compatibility before establishing a CPDLC dialogue.

   2) Air Initiation (CPDLC and DSC):

      i) the ATN address of the ground CPDLC system must be present somewhere in the airborne data link communication system. Without this information, a CPDLC or DSC dialogue with a ground CPDLC system can not be established.

      ii) the version number of the CPDLC protocol run on the ground system must be available to the CPDLC-air-user. The protocol version negotiation is not performed by the CPDLC Application Entities and the CPDLC-air-user is responsible for checking the version compatibility before establishing a CPDLC or DSC dialogue.

b) Establishment of ground/ground dialogues:

   1) the ATN address of the ground CPDLC system must be present somewhere in the ground data link communication system. Without this information, a CPDLC dialogue with another CPDLC ground system can not be established.

   2) CM does not perform any address or version number exchange function for ground/ground dialogues, this information will have to acquired and made available to the CPDLC ground application by some other mechanism

   3) version number compatibility between two CPDLC ground systems is checked at the time a dialogue is initiated (see Sections 2.3.3 and 2.3.5 of the CPDLC SARPs)

2.6.3 The Context Management (CM) Application is the mean defined in Version 1 to exchange this information for air/ground dialogues. Other solutions could be chosen, like the hard-coding of the addressing data bases or the loading of this information at the gate before the take-off.

2.7 Ground CPDLC Exchanges

2.7.1 No operational requirement related to the ground/ground forwarding of CPDLC-addressing information have been specified in the ADSP Manual [ref. xx].

2.8 Dialogue Management

2.8.1 Optimization of the use of dialogues

2.8.1.1 Establishment and termination of an air/ground (CPDLC or DSC) dialogue is always done explicitly by a CPDLC-user. There is no implicit dialogue establishment and termination as described in the ADS and FIS applications. Since a CPDLC dialogue can technically be initiated by either a CPDLC-air or a CPDLC-ground user there are explicit instructions in the CPDLC SARPs for the situation when two CPDLC dialogues are established between an air/ground peer pair.

2.8.1.2 Establishment of a ground/ground CPDLC dialogue is always done explictly by a CPDLC-user. There is no implicit dialogue establishment described in the ADS and FIS applications. Dialogue termination is always.done upon receipt of a confirmation of the ground forward request (completion of the CPDLC-forward service). Dialogue termination occurs whether or not the function was operationally successful. The CPDLC-forward service is a “one-shot” service, with no concept of maintaining a dialogue.

2.8.2 Dialogue Establishment

2.8.2.1 A CPDLC or DSC dialogue can only be established by an explicit request by a CPDLC-user. There is no implicit CPDLC/DSC dialogue establishment. A CPDLC dialogue can be established only in any of the following three ways:

	Initiation Requirement	Type of Dialogue	Valid Source
1.	CPDLC-start request	air/ground CPDLC dialogue	CPDLC-air-user CPDLC-ground-user
2.	DSC-start request	air/ground DSC dialogue	CPDLC-air-user
3.	CPDLC-forward request	ground/ground CPDLC dialogue	CPDLC-ground-user

Table 2.1 — CPDLC/DSC Dialogue Initiation/Establishment

2.8.2.2 The CPDLC SARPs outlines specific requirements for when a CPDLC/DSC dialogue can be established.. During the time of the dialogue establishment (a start or forward request has been issued and the confirmation has not yet been received), no new request, other than an abort, can be issued by the CPDLC-user for a given peer connection.

Note: since a CPDLC air/ground dialogue can be established by either the CPDLC-air-user or the CPDLC-ground-user there is a possibility for “simultaneous” starts. This situation is covered explictly in 2.3.7 of the CPDLC SARPs. In this case the CPDLC-ground-user aborts gorund initated dialogue and accepts the air initiated dialogue.

2.8.2.3 Summary of CPDLC-Air-User Dialogue Establishment Requirements

2.8.2.3.1 The SARPs permit a CPDLC-air-user to issue a CPDPC-start request with any ground system if the CPDLC-air-user currently has no CPDLC dialogues. Ground acceptance of any CPDLC-start request will be determined by operational or procedural requirements. For example, a given ground system may not permit establishment of air initiated CPDLC dialogues.

2.8.2.3.2 Once any CPDLC dialogue is established, the SARPs constrains the CPDLC-air-user to only being permitted to issue a CPDLC-start request with an NDA. As in the preceding paragraph, ground acceptance of any air initiated CPDLC-start request will be determined by operational or procedural requirements.

2.8.2.3.3 The SARPs permit a CPDLC-air-user to issue a DSC-start request with any ground system if the CPDLC-air-user currently has no DSC dialogues. Ground acceptance of any DSC-start request will be determined by operational or procedural requirements. For example, a given ground system may not support the DSC function. (It must support a stub of the function to enable rejection of the request for a DSC dialogue).

2.8.2.4 An aircraft can have only one DSC dialogue at a time. That is, the SARPs do not permit a CPDLC-air-user to establish a DSC dialogue with two ground systems concurrently; nor do they permit a CPDLC-air-user to have more than one DSC dialogue with the same ground system.

2.8.2.5 Although the SARPs do not prohibit the CPDLC-air-user from opening a DSC dialogue with its CDA (provided another DSC connection is not already established, since there can only be one DSC connection for a given CPDLC-air-user), the SARPs do require a CPDLC-air-user to terminate a DSC dialogue with a ground system that is its CDA. So, if such a DSC dialogue were opened with a CDA, the CPDLC-air-user would have to promptly abort such a connection. Operationally this configuration of a ground system simultaneously being a CDA and a DSC is contradictory.

2.8.2.6 The SARPs do not constrain in any way a CPDLC-air-user from opening a DSC dialogue with a ground system that is its NDA (provided another DSC connection is not already established, since there can only be one DSC connection for a given CPDLC-air-user). Operationally, the configuration is both possible and logical. The NDA connection cannot be used for operational message exchange, so communication with an NDA that is also the DSC must be done through the DSC connection.

2.8.2.7 Summary of Ground Dialogue Establishment Requirements

2.8.2.7.1 The SARPs do not constrain a CPDLC-ground-user from issuing a CPDPC-start request with any aircraft at any time other than the time of dialogue establishment with a given aircraft peer. There are specific requirements in the SARPs that totally determine when a CPDLC-air-user must accept or reject a CPDLC-start request. Since any ground system could technically establish a CPDLC air/ground connection, operational and procedural rules may constrain CPDLC-ground initiation. The first CPDLC connection that is established determines the CDA, and thus effectively locks out any other ground system (legitimate or not) from establishing any CPDLC connection other than a NDA CPDLC connection with a given aircraft.

2.8.2.7.2 The mapping of CDA, NDA, and DDA designations to ground systems is an CPDC-air-user function. A given ground system does not technically need to be aware of what (CDA, NDA, or DDA) if any of these it is.

2.8.2.7.3 The SARPs do not permit a CPDLC-ground-user to issue a DSC-start request.

2.8.2.7.4 The SARPs do not require that a ground system have the capability to issue a CPDLC-forward request.

2.8.2.7.5 The SARPs permit a CPDLC-user to issue a CPDLC-forward request, if CPDLC-forward request capable, with any other CPDLC ground system at any time other than the time of dialogue establishment with a given ground system peer. Requirements dictating rejection of ground/ground CPDLC dialogue establishment are specified in the SARPs. Any requirements determining acceptance of a ground/ground CPDLC dialogue will be determined by operational and procedural rules outside of the SARPs. All CPDLC ground systems must have the technical capability of receiving a CPDLC-forward request, and issuing a response.

2.8.3 Dialogue Termination

2.8.3.1 An air/ground CPDLC/DSC dialogue is not terminated unless specifically commanded to do so by either an orderly release, or abruptly by an abort condition. A CPDLC/DSC air/ground dialogue is never implicitly released for any reason (e.g. due to lack of CPDLC message traffic). A CPDLC ground/ground dialogue is never

maintained beyond the receipt of a CPDLC-forward confirmation, and can only be terminated prior to this by an abort condition. A CPDLC/DSC dialogue can be terminated only in any of the following ways:

	Termination Requirement	Type of Dialogue	Valid Source
1.	CPDLC-start response/confirmation *Result* "rejected"	air/ground CPDLC dialogue	CPDLC-air-user CPDLC-ground-user
2.	DSC-start response/confirmation *Result* "rejected"	air/ground DSC dialogue	CPDLC-ground-user
3.	CPDLC-forward response/confirmation	ground/ground CPDLC dialogue	CPDLC-ground-user
4	D-START response/confirmation *Result* "rejected"	ground/ground CPDLC dialogue	CPDLC-ground-ASE
5	CPDLC-user-abort	any CPDLC or DSC dialogue	CPDLC-air-user CPDLC-ground-user
6	CPDLC-provider-abort	any CPDLC or DSC dialogue	CPDLC-air-ASE CPDLC-ground-ASE
7	D-U-ABORT	any CPDLC or DSC dialogue	CPDLC-air-user CPDLC-ground-user CPDLC-air-ASE CPDLC-ground-ASE
8	D-ABORT	any CPDLC or DSC dialogue	CPDLC communication service provider
9	CPDLC-end response/conformation *Result* "accepted"	air/ground CPDLC dialogue	CPDLC-air-user
10	DSC-end response/confirmation	air/ground DSC dialogue	CPDLC-ground-user

Table 2.2 — CPDLC/DSC Dialogue Termination

2.8.3.2 The CPDLC SARPs outlines specific requirements for dialogue termination.

2.8.3.3 Summary of CPDLC-Air-User Dialogue Termination Requirements

2.8.3.3.1 The SARPs prohibit a CPDLC-air-user from issuing a CPDLC-end request.

2.8.3.3.2 The SARPs very specifically (but not totally) constrain the CPDLC-air-user CPDLC-end response requirements.

2.8.3.3.3 The SARPs permit only the CPDLC-air-user to invoke the DSC-end request. The SARPs require the CPDLC-air-user to terminate a DSC dialogue in two situations

a) when the DDA become a CDA (the CDA CPDLC dialogue is established), and

b) prior to issuing any other DSC-start request. (A given aircraft can only have one DSC dialogue at a time.)

2.8.3.3.4 Although, no further constraints are placed by the SARPs on the CPDLC-air-user DSC-end request invocation, the DSC is operationally envisaged as having "short term" duration. That is, a request for a downstream clearance is issued to the DDA and the DSC dialogue is terminated upon the receipt of the DSC clearance.

2.8.3.4 Summary of Ground Dialogue Termination Requirements

2.8.3.4.1

2.9 Protocol Monitoring

2.9.1 General

2.9.1.1 The generation and transmission of expected responses are monitored by the peer CPDLC-ASE. If there is an unexpected event, the ASE will abort and give a reason (if possible). Likewise, if an application timer expires, the ASE will also abort. These cases are described in the following sections.

2.9.2 Service Timers

2.9.2.1 If the CPDLC-start, DSC-start, or CPDLC-forward confirmation is not received by the requesting CPDLC-ASE within a locally specified period of time (the SARPs section 2.3.5 recommends 6 minutes), the dialogue being established is aborted. Both CPDLC-users are informed of this situation by a CPDLC-provider-abort indication with a reason of "timer-expiry". This timer is a service timer and is not directly connected to any operational timers set, except that it should be sufficiently larger than any operational timer to prevent "nuisance" timer-expiry aborts.

2.9.3 Operational Timers

2.9.3.1 Other than the CPDLC-start and DSC-start response time requirements, there are no operational timer requirements for when a response must be issued upon receipt of a CPDLC indication. The timer requirements (and a prohibition of message inclusion upon acceptance of a CPDLC or DSC start request) for a CPDLC and DSC start response are due to the technical constraint of not allowing the dialogue requester to invoke any other (except aborts) CPDLC or DSC primitives until a confirmation is received. Operationally this time must be minimized as much a possible.

2.9.3.2 No other operational response timers are established by the SARPs. All other operational response time requirements will be determined by operational and procedural requirements. No implicit aborts (i.e., a SARPs required CPDLC-user-abort, a CPDLC-provider-abort, a D-U-ABORT, or a D-ABORT) will occur due to the failure of any operational timer. (A local parameter could be set up to allow a given system to issue a CPDLC-user-abort due to operational timer failure.)

2.9.4 Errors

2.9.4.1 Unrecoverable System Error

2.9.4.1.1 The unrecoverable system error is intended to cover cases where a fault causes a system lockup or the system to become unstable. Upon determining that it has become unstable, the CPDLC application will attempt to abort with the reason "undefined-error".

2.9.4.1.2 The unrecoverable system error processing is written as a recommendation in the SARPs instead of a requirement, as it is recognized that depending on the nature of the error in the system, it may not be possible to regain control in order to either perform an abort, or inform the user of the abort situation.

2.9.4.2 Invalid PDU

2.9.4.2.1 An invalid PDU is defined as a PDU that cannot be decoded or that is not received when it is expected. For example, a PDU that somehow becomes garbled would be an invalid PDU. There are two cases for handling invalid PDUs, the first requiring both CPDLC-users be informed, if active, because a dialogue is either in place or being established (thus D-ABORT is invoked), and the second requiring only the local user be informed, if active, because a dialogue is in the process of being closed (D-ABORT is not invoked.) An abort due to an invalid PDU is always invoked by a receiving CPDLC-ASE.

2.9.4.2.2 Dialogue In Place or Being Established

2.9.4.2.2.1 For the case of a dialogue in place or being established (receipt of any dialogue service indication or a D-END confirmation with a rejection of the end request) there is inherently a connection in place and if there is an invalid PDU then D-ABORT must be invoked. In this case, a CPDLCProviderAbortReason APDU is created with the reason [invalid-PDU]. D-ABORT is invoked with the abstract value of "provider" for the Originator parameter value and the created APDU as the User Data parameter value The abstract value "provider" is used since this is a CPDLC-ASE not a CPDLC-user invoked abort. Finally, if the CPDLC-air-user or CPDLC-ground-user is an active user (as defined in 2.3.5.1.4 Note 2 and 2.3.5.5.1.4 Note 2), then the user is informed of an abort due to receipt of an invalid PDU using the CPDLC-provider-abort service.

Note: In the case of the receipt of a D-START confirmation with the dialogue accepted, no User Data is ever permitted, so any included PDU would not be examined by the receiving ASE for validity, but rather rejected as a "not-permitted PDU" situation.

2.9.4.2.3 Dialogue In Process of Being Closed

2.9.4.2.3.1 For the case of the dialogue in the process of being closed (D-START confirmation with the start request rejected or D-END confirmation with the end request accepted) the ASE invoking the dialogue service response is no longer part of any connection. In this case, only the if the CPDLC-air-user or CPDLC-ground-user is an active user (as defined in 2.3.5.1.4 *Note 2* and 2.3.5.5.1.4 *Note 2*), then the receiving ASE informs its user of an abort due to receipt of an invalid PDU using the CPDLC-provider-abort service. D-ABORT is not invoked.

2.9.4.3 Not Permitted PDU

2.9.4.3.1 A not permitted PDU is defined as a PDU that arrives when the CPDLC-ASE is in a state that does not allow that PDU. For example, receiving a *User Data* parameter in a D-START confirmation when the start has been accepted. This does not mean that the PDU cannot be decoded; and in fact it may well be a valid PDU. There are two cases for handling not permitted PDUs, the first requiring both CPDLC-users be informed, if active, because a dialogue is either in place or being established (thus D-ABORT is invoked), and the second requiring only the local user be informed, if active, because a dialogue is in the process of being closed (D-ABORT is not invoked.) An abort due to a not permitted PDU is always invoked by a receiving CPDLC-ASE.

2.9.4.3.2 Dialogue In Place or Being Established

2.9.4.3.2.1 For the case of a dialogue in place or being established (receipt of any dialogue service indication a D-START confirmation with acceptance of the start requests, or a D-END confirmation with a rejection of the end request) there is inherently a connection in place and if there is an not permitted PDU then D-ABORT must be invoked. In this case, a CPDLCProviderAbortReason APDU is created with the reason [not-permitted-PDU]. D-ABORT is invoked with the abstract value of "provider" for the *Originator* parameter value and the created APDU as the *User Data* parameter value The abstract value "provider" is used since this is a CPDLC-ASE not a CPDLC-user invoked abort. Finally, if the CPDLC-air-user or CPDLC-ground-user is an active user (as defined in 2.3.5.1.4

Note 2 and 2.3.5.5.1.4 *Note 2*), then the user is informed of an abort due to receipt of an invalid PDU using the CPDLC-provider-abort service.

Note: In the case of the receipt of a D-START confirmation with the dialogue accepted, no User Data is ever permitted, so any included PDU would be a "not-permitted PDU" situation.

2.9.4.3.3 Dialogue In Process of Being Closed

2.9.4.3.3.1 For the case of the dialogue in the process of being closed (D-START confirmation with the start request rejected or D-END confirmation with the end request accepted) the ASE invoking the dialogue service response is no longer part of any connection. In this case, only the if the CPDLC-air-user or CPDLC-ground-user is an active user (as defined in 2.3.5.1.4 *Note 2* and 2.3.5.5.1.4 *Note 2*), then the receiving ASE informs its user of an abort due to receipt of an not permitted PDU using the CPDLC-provider-abort service. D-ABORT is not invoked.

2.9.4.4 D-START Confirmation Result or Reject Source Parameter Values Not as Expected

2.9.4.4.1 CPDLC-ASEs should never receive a D-START confirmation with the D-START *Result* parameter having the abstract value "rejected (transient)" nor D-START *Reject Source* parameter "DS provider". The D-START *Result* parameter is set by the CPDLC-ASEs, and is if SARPs compliant is either "rejected (permanent)" or "accepted". The D-START *Reject Source* parameter is set by the dialogue service provider, and is normally "DS User". The D-START *Result* abstract value "rejected (transient)" or the D-START *Reject Source* abstract value "DS provider" is indicative of a communication failure. Therefore, an indication that an abort due to a communication service error is given to the CPDLC-air-user or CPDLC-ground-user, if active. A D-ABORT is not invoked since there is inherently no connection due to the communication error condition.

2.9.4.5 D-START Indication Quality of Service Parameter Not as Expected

2.9.4.5.1 The ATN Sub-volume 1 (Ref [x]) dictates the values used for application service priority and RER quality of service parameters for all ATN applications. These values must be adhered to so proper levels of flight safety and performance are maintained. For CPDLC, the values used are "high priority flight safety message" for application service priority and "low" for RER quality of service. If these values are not present (as checked in sections 2.3.5.4.6 and 2.3.5.6.6 of the CPDLC SARPs), then the CPDLC-ASE will abort with reason "invalid-QOS-parameter" and the abstract value of "provider" for the *Originator* parameter value. The abstract value "provider" is used since this is not a user-invoked abort.

2.10 Version Number Negotiation (Air/Ground)

2.10.1 The CPDLC SARPs specify the operation of version 1 of the CPDLC application. The version number is a value inherent to the CPDLC-ASE and is not provided by the CPDLC-users.

2.10.2 The version numbers supported by the air and ground systems for the CPDLC application are exchanged during the CM procedures. The CPDLC-user initiating a air/ground CPDLC or DSC dialogue must check whether the ASE implemented is compatible with one of the versions implemented on the peer system. A dialogue must not be initiated if the peer ASE version numbers are not compatible.

2.10.3 The version number negotiation is implicitly performed by the upper layers when establishing the application-association. The application context name proposed by the initiator must be supported by the receptor.

2.11 Version Number Negotiation (Ground/Ground)

2.11.1 The CPDLC SARPs specify the operation of Version 1 of the CPDLC application. The version number is a value inherent to the CPDLC-ASE and is not provided by the CPDLC-users.

2.11.2 The CM application does not support any ground/ground version negotiation. The version number of the ASE is indicated to the receiving ASE using CPDLC-forward service when establishing aground/ground CPDLC dialogue. The CPDLC-ground-ASE responding to a request for a ground/ground CPDLD dialogue must provide its version number to the initiating CPDLC-ground-user whenever it does not match that of the initiating ASE.

2.11.3 The version number negotiation is implicitly performed by the upper layers when establishing the application-association. The application context name proposed by the initiator must be supported by the receptor.

3. FUNCTIONALITY OF SERVICES

3.1 Concepts

3.1.1 Users of the CPDLC service are termed *CPDLC-ground-user* and *CPDLC-air-user* or just *CPDLC-user* when it applies to both air and ground. The CPDLC-user represents the operational part of the CPDLC system. It is either the final end-user (e.g. a crew member or controller) or an automated system. The CPDLC-user that initiates a CPDLC air-ground or ground-ground service is termed the *calling* CPDLC-user or *initiator*. The CPDLC-user that the initiator is trying to contact is termed the *called* CPDLC-user or *responder*.

3.1.2 This section describes first the information required by the ASEs from the CPDLC-users. Then, it considers CPDLC services in turn and provides an overview of the data flow within the ASE which handles the service primitives. The primitives are grouped according to the services they provide: dialogue initiation (CPDLC-start, DSC start, or CPDLC-forward), message exchange (specifically CPDLC-message, although all CPDLC services can contain a CPDLC message or abort reason), ending a dialogue (CPDLC-end, DSC-end, or CPDLC-forward), or aborting a dialogue (CPDLC-user-abort or CPDLC-provider-abort).

3.1.3 This section considers each of primitives composing the CPDLC Service. The primitives are grouped according to the services they provide:

a) starting a CPDLC/DSC dialogue
    1) CPDPC-start service
    2) DSC-start service
    3) CPDLC-forward service
b) CPDLC message exchange
    1) CPDLC-message service
c) ending a CPDLC/DSC dialogue
    1) CPDLC-end service
    2) DSC-end service
d) aborting a CPDLC/DSC dialogue
    1) CPDLC-user-abort service
    2) CPDLC-provider-abort service

3.2 CPDLC User to ASE Service Parameters

3.2.1 Called Peer Identifier

3.2.1.1 During CPDLC air/ground dialogue establishment (CPDLC-start request), the CPDLC-user must supply the identification of the called peer. The *Called Peer Identifier* is required for dialogue addressing purposes. When the dialogue is air initiated, the *Called Peer Identifier* is a *Facility Designation*. When the dialogue is ground initiated, the *Called Peer Identifier* is an *Aircraft Address*.

3.2.1.2 The *Called Peer Identifier* must be retrievable from the addressing data base of the aircraft when air initiated. The *Called Peer Identifier* must be retrievable from the addressing data base of the ground system when ground initiated. The CM application can be used to obtain and make available the required information.

3.2.1.3 See *Facility Designation* and *Aircraft Address* for the format of the *Called Peer Identifier.*

3.2.2 Calling Peer Identifier

3.2.2.1 During CPDLC air/ground dialogue establishment (CPDLC-start request), the CPDLC-user must supply its identification. The *Called Peer Identifier* is required for for indication to the peer-user. When the dialogue is air initiated, the *Calling Peer Identifier* is an *Aircraft Address.* When the dialogue is ground initiated, the *Calling Peer Identifier* is a *Facility Designation.*

3.2.2.2 See *Facility Designation* and *Aircraft Address* for the format of the *Calling Peer Identifier.*

3.2.3 Facility Designation

3.2.3.1 During DSC dialogue establishment (DSC-start request) the CPDLC-air-user must supply the *Facility Designation* of the called peer. The *Facility Designation* is required for dialogue addressing purposes. The *Facility Designation* must be retrievable from the addressing data base of the aircraft. The CM application can be used to obtain and make available the required information.

3.2.3.2 The *Facility Designation* parameter can be four to eight characters. Four characters are used to identify a particular facility, such as "ZYVR" for Vancouver. Up to eight characters may be used to identify a particular system within a facility; i.e. "ZYVRA123" may be the address for Vancouver en-route.

3.2.4 Aircraft Address

3.2.4.1 During DSC dialogue establishment (DSC-start request) the CPDLC-air-user must supply its *Aircraft Address.* The *Aircraft Address* is required for for indication to the peer-ground-user

3.2.4.2 The *Aircraft Address* is 24 bit aircraft address.

3.2.5 Called Facility Designation

3.2.5.1 During CPDLC ground/ground dialogue establishment (CPDLC-forward request), the initiating CPDLC-ground-user must supply the identification of the called ground peer. The *Called Facility Designation* is required for dialogue addressing purposes.

3.2.5.2 The *Called Facility Designation* must be retrievable from the addressing data base of the initiating ground system.. The CM application is not used for ground/ground addressing information. The SARPs do not address this area.

3.2.5.3 See *Facility Designation* for the format of the *Called Facility Designation.*

3.2.6 Calling Facility Designation

3.2.6.1 During CPDLC dialogue establishment (CPDLC-forward request), the initiating CPDLC-ground-user must supply its identification. The *Calling Facility Designation* is required for for indication to the receiving ground-peer-user.

3.2.6.2 See *Facility Designation* for the format of the *Calling Facility Designation.*

3.2.7 Class Of Communication Service

3.2.7.1 The *Class of Communication* may be provided by the CPDLC-user initaing a CPDLCor DSC dialogue (CPDLC-start request, DSC-start request, or CPDLC-forward request). Technically supplying the *Class of Communication* is up to the user, although operational procedures beyoond the SARPs requirements may dictate when and what value *Class of Communication* is supplied.

3.2.7.2 The *Class of Communication* is a means for the user to indicate the required performance in terms of end-to-end transit delay. The *Class of Communication* provided by the CPDLC-user is supplied to the Transport Service Provider (TSP) when the connection is established. Values for these transit delays are given in Sub-Volume 1 of the ATN SARPs. The *Class of Communication* is not guaranteed nor a degradation of the provided class is indicated to the users. It is the responsibility of the application to determine the actual transit delay achieved by local means such as time stamping.

3.2.7.3 There is no negotiation of the *Class of Communication* between air and ground CPDLC-users. However, the ground-user may have operational requirements dictating the minimum acceptable *Class of Communication*. Therefore the *Class of Communication* is always indicated to the receiving user and can be used as a basis for a CPDLC-ground-user to reject the request to start a CPDLC or DSC dialogue.

3.2.7.4 If the CPDLC-user does not require a particular *Class of Communication*, the *Class of Communication* parameter may be left blank, indicating no routing preference. This means Class of Communications is chosen by the dialogue service provider. The value supplied by dialogue service provider is then indicated to the receiving user and can be used as a basis for a CPDLC-user to reject the request to start a CPDLC or DSC dialogue.

3.2.8 CPDLC Message The *CPDLC Message* parameter contains a CPDLC message provided by the user of the CPDLC service. When invoking the CPDLC-start, DSC-start, CPDLC-end, or DSC-end request primitive, the user has the choice of whether or not to supply this parameter. When invoking the CPDLC-forward and CPDLC-message request primitive, the user must supply a CPDLC message. When invoking CPDLC-end or DSC-end service response primitive, the user has the choice of whether or not to supply this parameter. Whenever this parameter is supplied by a user it is always indicated/confirmed to the receiving user.

3.2.8.2 User requirements in Section 7 of the SARPs mandate the provision of a CPDLC message in the CPDLC-end and DSC-end *CPDLC Message* parameter under specified conditions.

3.2.8.3 Furthermore, operational and procedural requirements may further constrain when a CPDLC message must/must not be supplied by the user of a service, and what particular message is supplied under given conditions. These requirements are outside of the SARPs.

3.2.8.4 Whenever the provision of the *CPDLC Message* parameter is optional according to the protocol in Section 3 (i.e., a “U”), the receiving ASE does not check if a message is present. Any requirements to supply a message based on Section 7 of the SARPs or based on operational and procedural requirements are not checked by the receiving ASE. (The receiving ASE does check any supplied PDU as per Section 5 protocol and will abort if an error is detected.)

3.2.8.5 The *CPDLC Message* parameter passes transparently through the CPDLC Service provider. The CPDLC service provider will attach a small header (not to be confused with the CPDLC message header provided by the used as part of the CPDLC message) to the user data used for the coordination of the two CPDLC ASEs. This header is termed the Protocol Control Information (PCI).

3.2.9 Reject Reason

3.2.9.1 The *Reject Reason* parameter contains a CPDLC message to be provided by the user of the CPDLC service only to indicate an operational reason for the rejection of the requested service. The parameter is optionally

supplied by the CPDLC-user receiving CPDLC-start, DSC-start if and only if the receiving user is rejecting the requested start.

3.2.9.2 Operational and procedural requirements may further constrain when a CPDLC message must/must not be supplied as a *Reject Reason* parameter by the user of a service, and what particular message is supplied under given conditions. These requirements are outside of the SARPs.

3.2.9.3 The ASE receiving the start confirmation primitive will issue an abort if a CPDLC message is supplied when the user accepts the requested start.

3.2.9.4 When the requested start is rejected, the receiving ASE does not check if a message is present. The ASE does not take into account any required message presence based on operational and procedural requirements. (The receiving ASE does check any supplied PDU as per Section 5 protocol and will abort as required if an error is detected.)

3.2.9.5 The *Reject Reason* parameter passes transparently through the CPDLC Service provider. The CPDLC service provider will attach a small header (not to be confused with the CPDLC message header provided by the user as part of the CPDLC message) to the user data used for the coordination of the two CPDLC ASEs. This header is termed the Protocol Control Information (PCI).

3.3 CPDLC ASE to CPDLC-User Primitives

3.4 The CPDLC-start Service

3.4.1 The *CPDLC-start* service is used to set up a CPDLC dialogue between a CPDLC-air-user and a CPDLC-ground-user. It is a confirmed service, that can be initiated by either the CPDLC-air-user or CPDLC-ground-user.

- the CPDLC-start request is passed to its CPDLC-ASE.
- the ASE creates an APDU
    - if the ASE is an air-ASE is set the mode as “cpdlc”
    - includes the user supplied message if any, else sets the field as “NULL”
- invokes the D-START service to pass the APDU and
    - the called and calling peer identifiers supplied by the user,
    - sets the QOS parameters, making use of the *Class of Communication* parameter, if provided by the user
- sets the start timer
- the receiving ASE
    - confirms the values in the *Calling Peer ID* parameter
    - determines if the APDU is valid
    - if the ASE is an ground-ASE is checks that the mode is “cpdlc”
    - checks that the QOS parameters are set correctly
- if any of the above check fails the ASE aborts the request to start a dialogue
- if all of the above checks pass, the ASE passes the following to its user

- the calling peer identifier
- the CPDLC message provided by the requesting user, if any
- the class of communication value

3.4.2 The SARPs completely determine the whether the or not a CPDLC-air-user accepts or rejects a CPDLC-ground-users request to establish a dialogue.

3.4.3 The SARPs place no requirements on whether or not a CPDLC-ground-user accepts or rejects a CPDLC-air-users request to establish a dialogue.

3.4.4 The CPDLC-start response is passed to its CPDLC-ASE

- the ASE checks the value of the *Result* parameter
- if the value is "accepted" and no *Reject Reason* parameter is supplied by the user and DSC is "false"
    - the ASE invokes the D-START service with the "accepted" value
    - the receiving ASE
        - checks that DSC is "false"
        - checks that the *Result* parameter has the value "accepted"
        - and checks that a *User Data* parameter is not present
    - if the any of the above checks fail the ASE aborts
    - if all of the above checks pass, the ASE stops the start timer and passes the "accepted" value to its user
- if the value is "rejected" and DSC is "false"
    - if the *Reject Reason* is provided by the user, the ASE creates an APDU based on the parameter
    - the ASE invokes the D-START service with the "rejected (permanent)" value and the APDU, if created
    - the receiving ASE
        - checks that DSC is "false"
        - checks that the *Result* parameter has the value "rejected (permanent)"
        - checks that the *Reject Source* parameter is "DS user"
        - checks that a *User Data* parameter, if present, contains a valid APDU
    - if the any of the above checks fail the ASE aborts
    - if all of the above checks pass, the ASE
        - stops the start timer
        - passes the "rejected" value and if present, the APDU to its user

3.4.5 A CPDLC message can be included in the start request. If the request to open a dialogue is rejected the message is ignored.

3.4.6 If a CPDLC message is included in the start request, and the request is accepted, any CPDLC response message required is sent using the CPDLC-message or CPDLC-end services.

3.5 The DSC-start service

3.5.1 The *DSC-start* service is used to set up a DSC dialogue between a CPDLC-air-user and a CPDLC-ground-user. It is a confirmed service, that can only be initiated by the CPDLC-air-user.

- the DSC-start request is passed to its CPDLC-air-ASE.
- the ASE creates an APDU
    - the ASE sets the mode as “cpdlc”
    - includes the user supplied message if any, else sets the field as “NULL”
- invokes the D-START service to pass the APDU and
    - the Facility Designation and Aircraft Address supplied by the user,
    - sets the QOS parameters, making use of the *Class of Communication* parameter, if provided by the user
- sets the start timer
- the receiving ground-ASE
    - confirms the values in the *Calling Peer ID* parameter
    - determines if the APDU is valid
    - if the ASE is an ground-ASE is checks that the mode is “dsc”
    - checks that the QOS parameters are set correctly
- if any of the above check fails the ground-ASE aborts the request to start a dialogue
- if all of the above checks pass, the ground-ASE sets to DSC to “true” and passes the following to the ground-user:
    - the calling peer identifier
    - the CPDLC message provided by the requesting user, if any
    - the class of communication value

3.5.2 The SARPs place no requirements on whether or not a CPDLC-ground-user accepts or rejects a CPDLC-air-users request to establish a DSC dialogue.

3.5.3 The DSC-start response is passed to the CPDLC-ground-ASE

- the ASE checks the value of the *Result* parameter

- if the value is “accepted” and no *Reject Reason* parameter is supplied by the user and DSC is “true”
    - the ASE invokes the D-START service with the “accepted” value
    - the receiving ASE
        - checks that DSC is “true”
        - checks that the *Result* parameter has the value “accepted”
        - and checks that a *User Data* parameter is not present
    - if the any of the above checks fail the ASE aborts
    - if all of the above checks pass, the air-ASE stops the start timer and passes the “accepted” value to the air-user
- if the value is “rejected” and DSC is “true”
    - if the *Reject Reason* is provided by the user, the ASE creates an APDU based on the parameter
    - the ASE invokes the D-START service with the “rejected (permanent)” value and the APDU, if created
    - the receiving ASE
        - checks that DSC is “true”
        - checks that the *Result* parameter has the value “rejected (permanent)”
        - checks that the *Reject Source* parameter is “DS user”
        - checks that a *User Data* parameter, if present, contains a valid APDU
    - if the any of the above checks fail the ASE aborts
    - if all of the above checks pass, the ASE
        - stops the start timer
        - passes the “rejected” value and if present, the APDU to the air-user

3.5.4 A CPDLC message can be included in the start request. If the request to open a dialogue is rejected the message is ignored.

3.5.5 If a CPDLC message is included in the start request, and the request is accepted, any CPDLC response message required is sent using the CPDLC-message service.

3.6 The CPDLC-message Service

3.6.1 The *CPDLC-message* service allows the exchange of air/ground CPDLC messages on CPDLC and DSC dialogues. A CPDLC-message service request can be invoked by either the air or ground user. This service is unconfirmed.

- the CPDLC-message request is passed to its CPDLC-ASE.
- the ASE creates an APDU based on the user supplied message

- the ASE invokes the D-DATA service to pass the APDU
- the receiving ASE checks that the APDU is valid
    - if not it aborts
    - if so it passes the APDU to its user

3.7 The CPDLC-end Service

3.7.1 The *CPDLC-end* service allows the CPDLC-ground-user to initiate a smooth closure of a CPDLC dialogue. This service is confirmed.

3.7.2 more here

3.8 The DSC-end Service

3.8.1 The *DSC-end* service allows the CPDLC-air-user to initiate a smooth closure of a CPDLC dialogue. This service is confirmed.

3.8.2 more here

3.9 The CPDLC-forward Service

3.9.1 The *CPDLC-forward* service allows the CPDLC-ground-user to send a CPDLC message to another CDLC-ground-user This service is confirmed.

3.9.2 more here

3.10 The CPDLC-user-abort Service

3.10.1 User-Initiated Abort

3.10.1.1 The CPDLC-user can abort a CPDLC or DSC dialogue at any time, subject to operational or procedural reasons. There are several operational reasons when the CPDLC-user must invoke a CPDLC or DSC dialogue. There are considered operationally catastrophic. When a CPDLC-user invokes a CPDLC-user-abort the user can supply a reason.. Messages in transit may be lost during this operation. This is an unconfirmed service.

3.10.1.2 The CPDLC-user-abort service can be invoked at any time that the CPDLC-user is aware that any CPDLC or DSC service is in operation. After this primitive is invoked, no further primitives may be invoked for the current dialogue.

- the CPDLC-user invokes the CPDLC-user-abort request with a Reason, if desired to its ASE
- the requesting ASE:
    - stops any timer that is set

- if a CPDLC message is provided in the *Reason* parameter, creates an APDU based on the supplied CPDLC message
- else creates an APDU with the element “undefined”
- invoke D-ABORT
  - with the APDU
  - the *Originator* parameter set to “user”
- set DSC to “false”

- the receiving ASE
  - checks that:
    - *Originator* is “user”
    - the APDU is valid
  - if so:
    - stops any timer
    - if its user is active, passes the APDU to its user
    - if DSC if “true” sets DSC to “false”

3.10.2 Provider-Initiated Abort

3.10.2.1 When the CPDLC-ASE detects an error from which it cannot recover, the *CPDLC-provider-abort* service is invoked. This is a CPDLC provider-initiated service with a single, mandatory parameter telling why the dialogue is being aborted. Messages in transit may be lost during this operation.

3.10.2.2 The CPDLC-provider-abort service is part of a core functionality of the CPDLC systems. It shall be supported by any configuration of airborne and ground CPDLC systems.

- The detecting ASE module passes the CPDLC-provider-abort indication to the its CPDLC-user if it is still active,
- The detecting ASE requests D-ABORT to abort the communication with a reason value request if the dialogue is still open. The abort originator parameter is set to "provider". The APDU is passed to the remote system by way of the upper and lower layers and emerges in the receiving ASE
- The receiving ASE examines the APDU
- The receiving ASE module recognizes a provider abort situation (the abort originator of the D-ABORT indication is set to "provider"). It decodes the APDU and passes a CPDLC-provider-abort indication to its user, if it is still active.

3.10.2.3 Errors detected by the ASE are the following:

- *timer expiration*: a timer set previously by the ASE expires.
- *not permitted PDU*: the PDU is a valid PDU, no action is described for the PDU

- *invalid PDU*: the ASE is not able to decode the data packet received in the user data parameter of a dialogue service primitive.
- *communication service error*: a system function call has failed, for instance a memory allocation/release function, an I/O operation, etc...
- *communication service failure*: a system function call has failed, for instance a memory allocation/release function, an I/O operation, etc...
- *invalid QOS parameter*: a QOS parameter has not been set correctly for CPDLC
- *undefined error*: unrecoverable system error.

4. CPDLC SECTION DESCRIPTION

4.1 Section 2.3.2: GENERAL REQUIREMENTS

4.1.1 Version Number

4.1.1.1 This section is included to allow the Context Management (CM) application to exchange version numbers of the CPDLC application. It is necessary to allow for future versions of the protocol to be negotiated by CM. It has no effect on the CPDLC functionality.

4.1.2 Error Processing Requirements

4.1.2.1 In the abstract service definition, each service has a set of parameters and the abstract syntax of those parameters specified. Thus information which is not a valid syntax is not allowed to be input.

4.1.2.2 In the protocol definition, there is a requirement that no service is permitted to be called when in an ASE is in an incompatible state. Thus making use of the abstract services is not permitted at these times.

4.1.2.3 An implementation should not allow the user to take invalid actions; however, there is no requirement to prevent an implementation from allowing this. The error processing requirements section thus says that *if* the implementation allows the user to enter invalid information, the system must inform the user that an entry error has occurred. In that case, the error is locally detected and the dialogue does not need to be aborted.

4.2 Section 2.3.3: ABSTRACT SERVICE DEFINITION

4.2.1 Concept of an Abstract Service

4.2.3 Section 2.3.3 concerns the CPDLC abstract service. The following provide an explanation of the term “abstract service”.

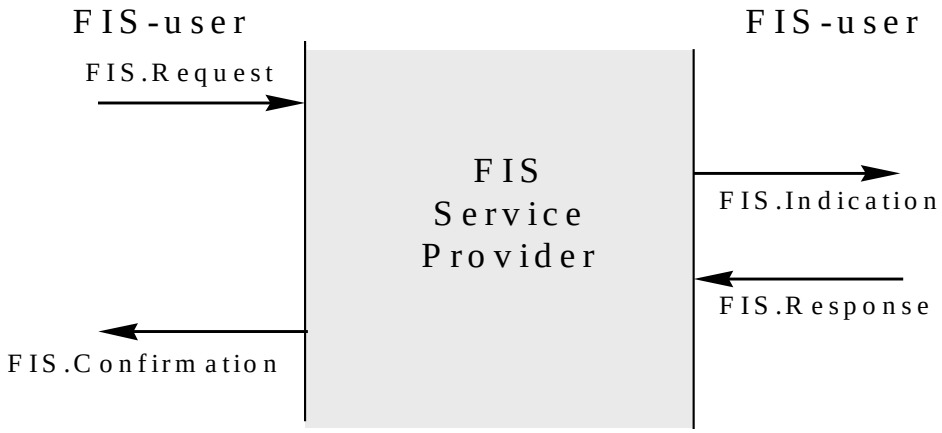
4.2.3.1 The CPDLC Abstract Service represents a set of functions offered to the CPDLC-user by the CPDLC-service provider.

4.2.4 Conventions

4.2.4.1 Service Primitives

4.2.4.1.1 A CPDLC service may consist of one to four primitives. Each primitive consists of the name of the CPDLC service (CPDLC-start, DSC-start, CPDLC-message, CPDLC-end, DSC-end, CPDLC-forward, CPDLC-user-abort and CPDLC-provider-abort) and a suffix that indicates at what point in the service the primitive occurs (request, indication, response, confirmation).

4.2.4.1.2 A *confirmed CPDLC service* is one that involves a handshake between the user that requests the service and the user that is informed that the service has been requested:



4.2.4.1.3 An *unconfirmed CPDLC service* involves no handshake.

4.2.4.1.4 A *provider-initiated service* is generated by the service provider in response to some internal condition. They consist of one indication primitive which is given to both users by the CPDLC service provider.

4.2.4.2 Service Parameters

4.2.4.2.1 Some primitive parameters have the same contents than APDU fields in the ASN.1 description. In most cases, the CPDLCASE is tasked to copy the parameter value within the APDU field. In order to avoid confusion by defining identical data structures twice, the type of those primitive parameters is specified by simply referring to the corresponding ASN.1 type in the APDU. The ASN.1 is used in the service definition as a syntax notation only and does not implicitly imply any local encoding of these parameters. The local implementation of these parameters remains a local implementation issue.

4.2.4.2.2 For the other parameters, the syntax is described by enumerating the authorized abstract values.

4.2.4.2.3 These service conventions have no impact on implementations. For instance a service primitive may be implemented through a function call or through an exchange via a mailbox (IPC mechanism).

4.2.4.2.4 There is no requirement for a SARPs compliant CPDLC system to be conformed to the abstract service specified in the SARPs.

4.3 Section 2.3.4: FORMAL DEFINITION OF MESSAGES

4.3.1 Purpose of Section 4

<Editor's note: tbd>

4.3.2 ASN.1

4.3.2.1 Data types exchanged by CPDLC ASEs are described in the CPDLC SARPs by using a machine-independent and language-independent syntax. There is no constraint put on the implementors concerning the machine nor the development language to be selected for implementing the protocol.

4.3.2.2 The ASN.1 module *MessageSetVersion1* contains the data types of the protocol data units handled by the CPDLC ASEs. Unlike common OSI ASEs (e.g. ACSE), no object identifier has been attached to the CPDLC) ASN.1 specification. Indeed, the ULCS architecture releases the applications from negotiating during the dialogue establishment the applicable abstract syntax. Object identifiers related to CPDLC application (application context name and version number) are defined in the ULCS SARPs.

4.3.2.3 ASN.1 Tags

4.3.2.3.1 Tags are used in ASN.1 to allow to distinguish data types when confusion is possible. For instance, when a data type contains two optional elements of the same type, if only one is encoded there is no means for the decoder to know which element the decoded value is attached to.

4.3.2.3.2 Even if tag values are not used by the Packed Encoding Rules, the ASN.1 grammar mandate the use of tags in some cases. When specifying the CPDLC data types, the following rules have been used:

- tags are always used within CHOICE data type, starting at 0 and then incremented by 1 for each entry.
- tags are not used at all in SEQUENCE data type when no confusion is possible. When any optional element is defined, all elements in the sequence are tagged.

4.3.2.4 Extensibility Markers

4.3.2.4.1 In order to allow the upgrade of the ASN.1 specification when new CPDLC capabilities or message elements are made available, the extensibility ASN.1 feature (ellipse) has been used in some data types in the CPLC service. Add examples

4.3.2.5 Data type specification when unit, range and resolution are defined

4.3.2.5.1 Some operational data are real. REAL data types exist in ASN.1 but the associated encoding procedures are not optimized in term of data size (PER do not specify specific rules for real but import the BER ones which requires the encoding of a mantissa, a base and a exponent taking several octets). Real values are therefore specified using the INTEGER data type.

4.3.2.5.2 ASN.1 constraints are attached to the INTEGER data type to take advantage of both range and resolution. Lower bound is equal to the operational minimal value devised by the resolution and the upper bound is equal to the operational maximal value devised by the resolution.

4.3.2.5.3 For instance:

```
PressureMeasure ::= CHOICE
{
hPa             [0] INTEGER (7500..12500)
-- units = hPa, range (750.0..1250.0), resolution 0.1
inches [1] INTEGER (2200..3200)
-- units = inches of Mercury, range (22.00..32..00), resolution = 0.01
}
```

4.3.2.6 Entry Points

If the implementor is using an ASN.1 compiler this may require an entry point. For here

1. add to tims 4311A system is not required to be able to encode or decode all messages specified in the ASN.1 description. Typically, an aircraft system is not required to be able to encode a GroundPDUs APDU nor to decode an AircraftPDUs APDU.

4.3.2.7 Data types common to several Package 1 Applications

Data names same between applications are dictated by ADSP operation reqs as stated by ADSP.

4.3.2.8 Time Representation

4.3.2.8.1 Data types have been specified for containing time indication (Date, DateTimeGroup, Year, Month, Hours, Minutes, Seconds). This way to represent time has been preferred over the pre-defined ASN.1 representations (GeneralizedTime and UTCTime) for optimization of the PER encoding.

4.3.3 ASN.1 Glossary

4.3.3.1 .

4.3.3.2 CPDLC Protocol data

4.3.3.2.1 These data are used to coordinate the processing of remote CPDLC ASEs. They are Protocol Control Information which do not carry operational data but are used for protocol processing purpose. They are used: does CPDLC have this?

4.3.3.2.2 The following data are used as the CPDLC message variables, or component of the variables, and are shown here in the alphabetic order:

Aircraft Address:

Aircraft Flight Identification*:* Field 7 of the ICAO flight plan.

Airport: Four characters that specifies the ICAO four-letter identifier for the airport.

Airway Identifier: Specifies the particular airway to be used within the route of the aircraft. (IA5 string of 2-5 characters.)

Altimeter: Indicates the aircraft altimeter setting in metric or English measurement units.

ATIS Code: Specifies the alphanumeric value for the current version of the automatic terminal information service (ATIS) in effect at a given location.

ATW Along Track Waypoint: Sequence of information used to compute additional way-points to an aircraft's route of flight. The following data composes the *ATW Along Track Waypoint:*

a) *Position*,

b) *ATW Distance*,

c) *Speed* (optional), and

d) *ATW Altitude Sequence* (optional).

ATW Level: Contains *ATW Level Tolerance* and *Level*

ATW Level Sequence: Sequence of 1 or 2 *ATW Levels.*

ATW Level Tolerance: Indicates the vertical tolerance factor for level clearances. Used in level clearances to indicate the acceptable vertical clearance of an aircraft relative to a particular level. Indicates:

a) at,

b) at or above, or

c) at or below.

ATW Distance: Used to specify the distance along a route of flight at which point to add the fix. Composed of *ATW Distance Tolerance* and *Distance*.

ATW Distance Tolerance: Indicates whether a distance can be plus or minus.

Clearance Type: Specifies a particular type of clearance. Where specified, the following clearance types are permitted:

a) approach,

b) departure,

c) further,

d) start-up,

e) pushback

f) taxi,

g) take-off,

h) landing,

i) oceanic,

j) en-route, or

k) downstream.

Code: Specifies the Mode A value (beacon code) for the aircraft. A sequence of 4 *CodeOctalDigit.*

Code Octal Digit: An octal digit used in *Code*.

Controlled Time:

Date: Gives the date in YYMMDD format using *Year*, *Month*, and *Day* data.

Date Time Track Generated: Date and time of the creation of a track.

Date Time Group: Provides date and time as YYMMDD and HHMMSS.

Day: Day of the month.

Degree Increment: Specifies the number of degrees (of latitude or longitude) separating reporting points.

Degrees: Indicates the degree value in degrees magnetic or degrees true.

Departure Clearance: Sequence of data structures necessary to provide a departure clearance. The sequence of data structures that compose a departure clearance data structure are:

a) Aircraft Identification,

b) Clearance Limit,

c) Flight Information,

d) *Further Instructions (optional).*

Departure Minimum Interval: Specifies the minimum interval of time to depart behind the preceding aircraft.

Direction: Indicates the horizontal direction specified in terms of the current direction relative to the aircraft or in terms of the cardinal points of the compass. Values are as indicated:

a) left,

b) right

c) either side,

d) north,

e) south,

f) east,

g) west,

h) north east,

i) north west,

j) south east, or

k) south west.

Direction Degrees: A sequence of Direction and Degrees

Distance: Provides the distance in metric or English units.

Distance Offset: Specifies the offset distance from the aircraft's route in metric or English units.

Distance Offset Direction: Sequence of *Distance Offset* and *Direction* data.

Distance Offset Direction Time: Sequence of *Distance Offset* and *Direction* and *Time* data.

Error Information: Indicates the error conditions as follows:

a) unrecognized message reference number,

b) logical acknowledgment not accepted,

c) insufficient resources

d) invalid message element combination, or

e) invalid message element.

Facility: Provides a choice of whether or not to indicate a *Facility*.

Facility Designation: Specifies the ICAO four to eight letter location indicator for the facility.

Facility Designation Altimeter: Sequence of *Facility Designation* and *Altimeter*.

Facility Designation ATIS Code: Sequence of *Facility Designation* and *ATISCode*.

Facility Function: Identifies type of facility:

a) center,

b) approach,

c) tower,

d) final,

e) ground control,

f) clearance delivery,

g) departure, or

h) control.

Facility Identification: Provides an ICAO facility identification as either an *ICAO Facility Designation* or a *Facility Name*.

Facility Name: Specifies the ICAO facility name.

Fix: Specifies the ICAO identifier for a given fix.

Fix Name: Sequence of Fix and optionally a Latitude and longitude for the fix. .

Flight Information*:* Information for a route of flight. Specified as:

a) *Route of Flight*, or

b) *Levels of Flight*, or

c) *Route of Flight* and *Levels of Flight.*

Free Text: Used to convey unstructured information.

Frequency: Specifies the frequency and an indicator of the RF spectrum used for the given frequency. The types of frequency that can be provided include:

a) *Frequency HF*,

b) *Frequency VHF*,

c) *Frequency UHF*,

d) *Frequency Sat Channel*, or

e) *Frequencyvhfchannel*

Frequency Sat Channel: Specifies the appropriate address for use with a satellite voice system.

Further Instructions*:* Provides additional information in a departure clearance as follows:

a) *Beacon Code* (optional),

b) *Departure Frequency* (optional),

c) *Clearance Expiry Time* (optional),

d) *Departure Airport* (optional),

e) *Destination Airport* (optional),

f) *Departure Time* (optional),

g) *Departure Runway* (optional)

h) *Revision Number* (optional), or

i) *ATIS Code* (optional).

Hold At Waypoint: Sequence of data structures used to define the holding procedure to be used at a particular point. The *Hold At Waypoint* consists of an sequence of the following:

a) *Position*,

b) *Hold At Waypoint Speed Low* (optional),

c) *ATW Altitude* (optional),

d) *Hold At Waypoint Speed High* (optional),

e) *Direction* (optional),

f) *Degrees* (optional),

g) *EFC Time* (optional), and

h) *Legtype* (optional).

Hold Clearance: Provides a holding clearance to the aircraft. The *Hold Clearance* is provided using:

a) *Position*,

b) *Altitude*,

c) *Degrees*,

d) *Direction*, and

e) *Legtype* (optional).

Intercept Course From: The *Intercept Course From* is used to specify a fix and a bearing from that fix needed to intercept a route using *Intercept Course From Selection* and *Degrees*.

Intercept Course From Selection. Used to specify the point from which the intercept course originates and an indication of which type of fix is specified. Provided as one of the following:

a) *Published Identifier*,

b) *Latitude Longitude*,

c) *Place Bearing Place Bearing*, or

d) *Place Bearing Distance*.

Icing:

Latitude Degrees: Degrees of latitude.

Latitude Degrees Minutes: Specifies Latitude in whole degrees and minutes of a degree.

Latitude Degrees Minutes Seconds: Specifies Latitude in whole degrees, whole minutes of a degree and seconds of a degree.

Latitude Direction: Indicates whether north or south latitude is specified.

Latitude Longitude: Sequence of *Latitude* and *Longitude*.

Latitude Reporting Points: Indicates the latitude on which to base incremental reporting points. Provided as of *Latitude Direction* and *Latitude Degrees*.

Latitude Type: Choice of specifying latitude in Degrees, degrees and minutes or degrees, minutes and seconds.

Latitude Whole Degrees: Specifies latitude in whole degrees.

Latlon Reporting Points: Provides either *Latitude Reporting Points* or *Longitude Reporting Points*.

Leg Distance: Indicates the aircraft leg in metric or English distance units.

Leg Time: Specifies aircraft leg in terms of minutes.

Leg Type: Provides either *Leg Distance* or *Leg Time*.

Level: Specifies the Level as a single or block altitude.

Level Position: A sequence of *Level* and *Position.*

Level Procedure Name: A sequence of *Level* and *Procedure Name.*

Levels of Flight*:* Specified as a choice of:

a) *Level*, or

b) *Procedure Name*, or

c) *Level* and *Procedure Name.*

Level Speed: A sequence of *Level* and *Speed.*

Level Time: A sequence of *Level* and *Time.*

Level Type: Specifies the Level in one of the following ways:

a) *Level* in feet or meters,

b) *Flight Level* in feet or meters.

Longitude: Provides longitude as*).*

Longitude Degrees: Degrees of longitude.

Longitude Degrees Minutes: Specifies Longitude in whole degrees and minutes of a degree.

Longitude Degrees Minutes Seconds: Specifies Longitude in whole degrees, whole minutes of a degree and seconds of a degree.

Longitude Direction: Indicates whether east or west longitude is specified.

Longitude Reporting Points: Indicates the longitude on which to base incremental reporting points. Provided as *Longitude Direction* and *Longitude Degrees.*

Longitude Type:

Longitude Whole Degrees:

Minutes Latlon:

Month: Month of the year.

Navaid: Specifies a particular navigation aid and optionally a lat lon of the navaid.

Navaid Name: Specifies a particular navaid.

Persons On Board: Specifies the number of persons on the aircraft.

Place Bearing: Sequence of *Published Identifier* and *Degrees.*

Place Bearing Distance: Used to indicate a location based on the degrees and distance from a known point. Provided using *Published Identifier, Degrees* and *Distance* data.

Place Bearing Place Bearing: Used to define a point as the intersection formed by two bearings from two known points. Provided as two *Place Bearing*.

Position: Information used to specify a location. Position can be specified as:

a) *Fix Name*,
b) *Navaid*,
c) *Airport*,
d) *Latitude Longitude*, or
e) *Place Bearing Distance*.

Position Degrees: Sequence of *Position* and *Degrees.*

Position Distance Offset Direction Sequence of *Position* and *Distance Offset Direction.*

Position Level*:* Sequence of *Position* and *Level.*

Position Level Speed: Sequence of *Position. Level* and *Speed.*

Position Procedure Name: Sequence of *Position* and *Procedure Name.*

Position Report: Uses the following data necessary to provide an aircraft position report as follows:

a) *Position Current*,
b) *Time At Position Current*,
c) *Altitude*,
d) *Fix Next* (optional),
e) *Time ETA At Fix Next* (optional),
f) *Fix Next Plus One* (optional),
g) *Time ETA destination* (optional),
h) *Remaining Fuel* (optional),
i) *Temperature* (optional),
j) *Winds* (optional),,
k) *Turbulence* (optional),
l) *Icing* (optional),
m) *Speed* (optional),
n) *Speed Ground* (optional),
o) *Vertical Change* (optional),
p) *Track Angle* (optional),

q) *True Heading* (optional),

r) *Distance* (optional),

s) *Supplementary Information* (optional),

t) *Reported Waypoint Position* (optional),

u) *Reported Waypoint Time* (optional), and

v) *Reported Waypoint Altitude* (optional).

Position Route Clearance: Sequence of *Position* and *Route Clearance.*

Position Speed: Sequence of *Position* and *Speed.*

PositionTime: Sequence of *Position* and *Time.*

Position Time Level: Sequence of *Position* and *Time* and *Level.*

Position Unit Name Frequency: Sequence of *Position, Unit Name* and *Frequency*.

Procedure: Specifies the name of the procedure.

Procedure Name: Used to uniquely identify the standard arrival, approach or departure procedure using the following:

a) *Procedure Type*,

b) *Procedure*, and

c) *Procedure Transition* (optional).

Procedure Type: Specifies the type of procedure as arrival, approach, or departure.

Procedure Transition: Specifies the name of the procedure transition.

Published Identifier: Used to indicate a *Fix Name* or a *Navaid.*

Remaining Fuel: Specifies the amount of fuel remaining on the aircraft using *Time* data.

Remaining Fuel Persons on Board: Sequence of *Remaining Fuel* and *Persons on Board*.

Reporting Points: Used to indicate reporting points along a route of flight based on a specific Latitude and/or Longitude increment expressed in degrees.

Revision Number: Specifies the revision number of the departure clearance. Used to differentiate different revisions of the departure clearance for a given aircraft flight.

Route And Levels: Sequence of *Route Information* and *Levels Of Flight*

Route Clearance: Data necessary to provide a route clearance. Provided using the following data:

a) *Aircraft Identification* (optional),

b) *Departure Airport* (optional),

c) *Destination* Airport (optional),

d) *Gate* (optional),

e) *Runway Departure* (optional),

f) *Procedure Departure* (optional),

g) *Runway Arrival* (optional),

h) *Procedure Approach* (optional),

i) *Procedure Arrival* (optional),

j) *Airway Intercept* (optional),

k) *Route Information Sequence* (optional), and

l) *Route Information Additional* (optional).

Route Information: Indicate the method used to define the aircraft route of flight. The actual aircraft route of flight will probably consist of multiple *Route Information* sequences as follows:

a) *Published Identifier* (optional)

b) *Latitude Longitude* (optional),

c) *Place Bearing Place Bearing* (optional),

d) *Place Bearing Distance* (optional),

e) *Airway Identifier* (optional), and

f) *Track Detail* (optional).

Route Information Additional: Additional data used to further specify a route clearance. Provided using the following:

a) *ATW Along Track Waypoint Sequence* (optional),

b) *Reporting Points* (optional),

c) *Intercept Course From Sequence* (optional),

d) *Hold At Waypoint Sequence* (optional),

e) *Waypoint Speed Altitude Sequence* (optional), and

f) *RTA Required Time Arrival Sequence* (optional).

Route Of Flight*:* Specifies route of flight using *Route Information*

RTA Required Time Arrival: Sequence used to associate an estimated time of arrival with a specific point along a route of flight. The *RTA Required Time Arrival* consists of:

a) *Position*,

b) *RTA Time*, and

c) *RTA Tolerance* (optional).

RTA Time: Used to specify the required time of arrival for an aircraft at a specific point.

RTA Tolerance: Specifies the possible tolerance expressed in minutes in the RTA time.

Runway: Specifies a runway using *Runway Direction* and *Runway Configuration*.

Runway Configuration: Used to specifically identify one runway in a group of parallel runways. Can be specified as left, right, or center.

Runway Direction: Specifies the direction of the runway.

Runway RVR: Sequence of *Runway* and *RVR*.

RVR: Runway Visual Range as distance in feet or meters.

Seconds Lat Lon: Seconds of latitude or longitude degrees.

Speed: Provides the aircraft speed as one of the following:

a) *Speed Indicated,*

b) *Speed True,*

c) *Speed Ground,* or

d) *Speed Mach.*

Speed Ground*:* Ground speed expressed in either English or metric units.

Speed Indicated*:* Indicated aircraft speed expressed in either English or metric units.

Speed Mach: Aircraft speed specified as a Mach value.

Speed Qualifier*:* Qualifies the given speed type. Values include minimum approach or cruise.

Speed Time: Sequence of *Speed* and *Time.*

Speed True*:* Aircraft true speed expressed in either English or metric units.

Speed Type: Indicates what type of speed is to be provided:

a) Indicated,

b) True,

c) Ground,

d) Mach, or

e) Not specified.

Temperature: Temperature specified in centigrade or Fahrenheit.

Time: Sequence of *Hours* and *Time Minutes.*

Time Departure: Time of departure given as an allocated time, Controlled departure Time, a departure clearance time, and a departure minimum interval**.**

Time Distance Offset Direction: Sequence of *Time* and *Distance Offset Direction*

Time Distance To From Position: Sequence of *Time, Distance, To From, and Position*

Time HHMMSS*:* Provides time as HHMMSS.

Time Hours: Time in Hours of a day.

Time Level: Sequence of *Time* and *Level.*

Time Minutes: Specifies time in minutes of an hour.

Time Position: Sequence of *Time* and *Position.*

Time Position Level: Sequence of *Time, Position,* and *Level.*

Time Position Level Speed: Sequence of *Time, Position, Level.,* and *Speed.*

Time Seconds*:* Specifies time in seconds of a minute.

Time Speed: Sequence of *Time* and *Speed.*

Time To From Position: Sequence of *Time, To From,* and *Position.*

Time Tolerance: Provides a time tolerance as: at, at or before, or at or after.

Time Unit Name Frequency: Sequence of *Time, UnitName,* and *Frequency*.

To From: Specifies to or from.

To From Position: Used to indicate a "to" or "from" relative to a specified position.

Track Detail: Associates a sequence of fixes with a particular track name. Specified using *Track Name* and *Latitude Longitude.*

Track Name: Specifies the name of an identified group of points which make up a section of a route.

Traffic Type*:* Indicates what type of traffic is present. Permitted types:

a) opposite direction,

b) same direction,

c) converging, or

d) crossing.

Turbulence: Specifies the severity of turbulence. Can be one of the following: "light", "moderate", or "severe".

Unit Name: Sequence of *ICAO Facility Identification* and *ICAO Facility Function*.

Unit Name Frequency: Sequence of *ICAO Unit Name* and *Frequency*.

Version Number: Specifies version number of CPDLC.

Vertical Change: Sequence of *Vertical Direction* and *Vertical Rate*.

Vertical Direction: Specifies whether the rate of vertical change is in the upward or downward direction.

Vertical Rate: Specifies the vertical rate of change in English or metric units.

Waypoint Speed Level Used to associate altitudes and speeds with particular points in a route clearance. It is composed using the following:

a) *Position*,

b) *Speed* (optional), and

c) *ATW Altitude Sequence* (optional).

Wind Direction: Specifies the direction of the wind using *Degree Value*.

Winds: Provides wind using *Wind Direction* and *Wind Speed*.

Wind Speed: Provides wind speed in metric or English measurement units.

Year: Provides year as last two digits of a year.

4.4 Section 2.3.5: PROTOCOL DEFINITION

4.4.1 Purpose of Section 5

<Editor's note: tbd>

4.4.2 Message Sequence Diagrams

4.4.2.1 Time sequence diagrams or message sequence diagrams are used to denote the relationship between the primitives that form a CPDLC service and the order in which they occur.

4.4.2.2 Implicitly, the concept of order is given through these Message Sequence Diagrams, e.g. the indication/confirmation primitives occurs some time after the request/response primitives.

4.4.2.3 Inherent to the service model is the notion of queuing. The CPDLC-service indications and confirmations are delivered to the CPDLC-users in the order that the corresponding CPDLC-service requests and responses were issued. The users can therefore initiate several services in parallel without having to wait between each invocation. One exception to the notion of queuing is the abortive services (CPDLC-user-abort and CPDLC-provider-abort services) which may overtake other primitives by empty the primitives in the queues.

4.4.2.4

4.4.2.5 add tims 4412 to 4418 with picture ot time seq

4.4.3 Technical Timers

<Editor's note: define what technical timers are>

<Editor's note: explain derivation of technical timers (as opposed to operational)>

4.4.3.1 The assignment of values for timers must be optimized based on operational testing of the application. In such testing, incompatible timer values and optimum combinations can be identified. Implementations of CPDLC protocol are required to support configurable values for all timers and protocol parameters, rather than having fixed values. This allows modification as operational experience is gained.

4.4.4 State Machines

4.4.4.1 The CPDLC service provider is described in the SARPs as finite state machines or protocol machines (PM). The protocol machine for a particular service starts in an initial state (IDLE). Events, which are service primitives received from the CPDLC-users above or the Dialogue Service provider below, as they occur, trigger activity on the part of the PM. As part of this activity, actions may be required (service primitives issued to the CPDLC-users and/or the underlying Dialogue Service provider).

4.4.4.2 The protocol is described under two forms: the textual and the tabular descriptions. The textual description takes precedence over the tabular description.

4.4.4.3 Functional Model

4.4.4.4 The protocol activity is composed Textual Description of the Protocol

4.4.4.4.1 Protocol is explained based on incoming events. An incoming event may be issued by the local CPDLC-service user or the Dialogue service-provider. For each state of the PM allowing the reception of the incoming event is listed the actions to be performed by the PM.

4.4.4.4.2 Actions are enumerated in the order they have to be carried out by the protocol machine.

4.4.4.4.3 Actions described must be implemented. What is not described is not allowed and must be covered by the Exception Handling procedures.

4.4.4.5 Tabular Description of the Protocol

<Editor's note: "Cannot occur" is used when...>

<Editor's note: "Not permitted" is used when...>

4.5 Section 2.3.6: COMMUNICATION REQUIREMENTS

4.5.1 Purpose of Section 6

<Editor's note: tbd>

4.5.2 Encoding Rules

<Editor's note: PER - choice and implementation>

4.5.3 Dialogue Service Requirements

<Editor's note: QOS priority: where it comes from (ADSP), how they relate to other application, use by the ATN (in case of congestion), benefits and affects>

<Editor's note: QOS RER High/low explanation>

4.6 Section 2.4.7: CPDLC USER REQUIREMENTS

4.6.1 Purpose of Section 7

<Editor's note: justify Section 7. Sections 3 and 5 guarantee interoperability but not operational acceptability (use of service primitives, use of optional ASN.1 data, required relationships between parameters in the request and in the response

<Range and Resolution - from Manual - Distance units.>

<Answers to some "whys". Stand alone Elements (justify rules beyond the ADSP instructions)>

<What is conveyed in a CPDLC message.>

<Messages inserted / ordered to preserve an element of compatibility with previous systems.>

4.7 Section 2.3.8: SUBSETTING RULES

4.7.1 Purpose of Section 8

<Editor's note: Any ground configuration works with any air configuration>

<a system will implement a configuration at a time>

<no correlation between version number and configuration>

<ASE implementation drives configuration>

<"do not use" options specify predicates>

4.7.1.1 An implementation of either the CPDLC ground based service or the CPDLC air based service claiming conformance to 2.4. must support the CPDLC protocol features as described below.

4.7.1.2 Only version 1 of the CPDLC protocol is defined. Any CNS/ATM-1 compliant CPDLC system must support this version.

4.7.1.3 Basic functionality supported by the air and ground CPDLC system is the same: the exchange of CPDLC messages. However, they are not identical. DSC-start and DSC end can only be initiated by the CPDLC-air-user, CPDLC-end and CPDLC-forward can only be initiated by a CPDLC-ground-user.

4.7.1.4

4.7.1.5 detail in full and partial stuff

5. DIMENSIONS

5.1 PDU Size

<Editor's note: tbd>

5.2 Rate of Message Transmission

<Editor's note: tbd>

5.3 Number of CPDLC/DSC Dialogues/Connections

5.3.1 Airborne CPDLC/DSC Dialogue Requirements

5.3.1.1 Total CPDLC/DSC Connections

5.3.1.1.1 The airborne CPDLC application must be capable of instantaneously supporting a total of up to 6 CPDLC or DSC dialogues (ASE instanteations) as follows:

a) 2 simultaneously with the CDA:

   1) in the situation that results from both an aircraft and a ground system in initiating a CPDLC dialogue simultaneously; the aircraft then subsequently terminates (aborts) one of the established dialogues, and the remaining dialogue is used for operational communication, and

   2) in the situation that an aircraft receives a second request from its CDA to establish a CPDLC dialogue; this request is accepted and the aircraft then terminates the previously established dialogue. This is to allow operational continuation of a CDA dialogue although a technical problem may have occurred with the first connection.

b) 2 simultaneously with the NDA

   1) in the situation that results from both an aircraft and a ground system in initiating a CPDLC dialogue simultaneously; the aircraft then subsequently terminates (aborts) one of the established dialogues, and the remaining dialogue remains open but is not used operationally until when and if the NDA becomes the CDA, and

   2) in the situation that an aircraft receives a second request from its NDA to establish a CPDLC dialogue, this request is accepted and the aircraft then terminates the previously established dialogue. This is to allow operational continuation of a NDA dialogue although a technical problem may have occurred with the first connection.

c) 1 with a DSC

1) only the aircraft can initiate a DSC dialogue, thus the above situations requiring 2 simultaneous connection with the CDA and NDA do not occur with the DSC.

d) 1 ground request

1) the aircraft has to be able to receive a CPDLC dialogue start request from any other ground system that is not the CDA or NDA and then reject such a request.

5.3.1.2 Connections With a Given Ground System Peer

5.3.1.2.1 The airborne CPDLC application must be capable of simultaneously supporting a maximum of 3 CPDLC or DSC dialogues with a given ground system peer as follows:

a) 2 simultaneously with the CDA or NDA (a given ground system is never simultaneously a CDA and NDA) as described above,

b) 1 with a DSC

1) an aircraft can have a DSC dialogue with a NDA ground system concurrently while it is has two NDA connection as described above. These are totally separate connections and have no operational or technical relationship. Thus when the given ground system peer is an NDA, or

2) an aircraft can have an existing NDA connection with a ground system when it becomes a CDA. The aircraft must then terminate the DSC connection with the CDA but for a period of time 2 simultaneous connections exist.

5.3.2 Ground CPDLC/DSC Dialogue Requirements

5.3.2.1 Total Connections

5.3.2.1.1 The total number of CPDLC/DSC connections (ASE instanteations) a given ground system must be able to simultaneously support cannot be determined here. A given ground system's connections are be determined by how many aircraft and other CPDLC ground systems it is communication with simultaneously. This will be a mix of how many aircraft it is connected to as a CDA, how many other aircraft it is connected to as a NDA, how many aircraft it is connected to as a DSC, and how many other CPDLC ground system it is connected to.

5.3.2.2 Connections With a Given Aircraft System Peer

5.3.2.2.1 (Same as above under Airborne = 3)

5.3.2.3 Connections With a Given Other Ground System Peer

A ground system must be capable of supporting a maximum of two CPDLC connections with any other ground system peer as follows: 1 connection with a ground system with which it initiates a CPDLC dialogue, and one connection with that same ground system when it receives a request to initiate a CPDLC dialogue.

6. ASN.1 INDEX

<Editor's note: tbd>

7. EXAMPLE OF OPERATIONAL SCENARIOS

7.1 Introduction

7.1.1 This section contains a set of example scenarios of use. The purpose of this section is to demonstrate different scenarios that are theoretically possible using CPDLC. It is not meant to indicate what is required from an operational point of view.

7.2 Scenario #1

7.2.1 The aircraft is equipped with CM, ADS, CPDLC and FIS(ATIS) applications.

7.2.2 Before take-off.

1. The pilot initiate the CM-Logon process with the ground CM covering the departure airport. The CM's response includes the address information for the ground CPDLC and FIS(ATIS) applications.

7.2.3 After take-off

1. The aircraft is now leaving the TMA environment and the pilot elects to

7.2.4 After landing

8. EXAMPLE ENCODING

8.1 Encoding/Decoding Rules

8.1.1 Samples of PER encoded CPDLC messages are provided in this section. For each APDU, the value tree containing the values assigned to the APDU components is provided. Then, the octet string resulting of the PER encoding is dumped in hexadecimal.

TBD.

add air initiation capability at all times from Whistler

8.1.2 Application Priority and Residual Error Rate

8.1.2.1 In addition to the Class Of Communication provided by the CPDLC-user requesting the CPDLC-start, DSC-start, or CDLC-forward service, other Quality of Service (QOS) parameters are attached by the ASE to the dialogue supporting the communication. Values for the Application Priority and the Residual Error Rate are constant for the CPDLC application and therefore are not required to be supplied by the CPDLC-user.

8.1.2.2 The Application Priority has the value "xx". Where does it comes from.

8.1.2.3 The Residual Error Rate has the value. What does it mean

8.1.2.4 movs somehwhere else