1. INTRODUCTION

1.1 Purpose

1.1.1 In line with normal ICAO practice, this document was developed as a companion document to the ATN Flight Information Services (Automatic Terminal Information Service) - FIS(ATIS) - SARPs. It may be read alongside the FIS(ATIS) SARPs, in order to provide a greater understanding of the specification itself, or it may be read instead of the ATN FIS(ATIS) SARPs by readers that simply want to understand the purpose of the FIS(ATIS) Application rather than the detail of the specification.

1.1.2 This document also provides some historical information on the development of the FIS(ATIS) Application and explanations as to why the FIS(ATIS) Application is specified the way it is, including corresponding notes and recommendations, in the SARPs.

1.2 Scope

1.2.1 This document provides guidance material for those implementing of the Automatic Terminal Information Service as part of the ATN Flight Information Service Application.

1.2.2 This document does not define any mandatory or optional requirements for the FIS(ATIS) Application, neither does it define any recommended practices. This document does not instruct users on how to use the FIS(ATIS) Application in a particular operational environment.

1.2.3 The FIS(ATIS) SARPs are dedicated to Air Traffic Services.

1.3 History

1.3.1 The FIS Application allows a pilot to request and receive FIS services from ground FIS systems. In a fully operational ATS data link environment, the FIS Application is expected to be used as the main means of passing flight information (e.g. automatic terminal information (ATIS), notices to airmen (NOTAMs), meteorological aerodrome reports (METARs) and extracts from Aeronautical information circulars (AICs)) to aircraft, whether in flight or on the ground.

1.3.2 In the initial implementation of FIS, only ATIS information will be passed. The FIS(ATIS) Application identifies the instance of the FIS Application dealing with ATIS information. ATIS messages, their format and intent, are 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 will be identical to the current voice based systems.

1.3.3 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, and how a particular sequence of messages within a dialogue is ended. However, the intention is that this should be as automatic as possible, with an apparently seamless line of communication between end users. The extent of automation will ultimately be the responsibility of the system designers, both from the engineering and operational aspects.

1.3.4 In addition to the ICAO documentation noted in paragraph 1.3.2 above, the main document from which the FIS(ATIS) SARPs have been developed are the Draft ICAO Manual of ATS Data Link Applications, submitted to the 2nd Meeting of the ADS Panel in September 1996. This specifies operating concepts in some detail. ICAO has specified that the FIS(ATIS) Application should conform to the ATN protocols for its data link operations.

1.3.5 The initial development of the SARPs centred around the requirement to replace a broadcast service with a service based on individual contact between the user (the aircraft) and the provider (the ground system). An ATIS broadcast system allows the pilot to obtain current information almost exactly when he/she wants it, and if the information becomes obtrusive, it can be switched off. If at any stage in the flight a controller detects that the aircraft is not in possession of the current information, it can easily be updated by voice.

1.3.6 This functionality is replicated to the extent possible by having two basic modes of operation in a data link FIS, namely a single request capability, and a 'contract' with the ground system, which provides updates to the aircraft as and when the information is updated by the ground. In stable conditions update rates may be virtually nil, whereas during the passage of an active front the ATIS may be updated several times per hour.

1.3.7 States might not be willing to incur the costs of implementing a complete system if they only ever intended to use certain elements of the application, e.g. although they would be obliged to make data link FIS information available on request, they may not wish to implement the provision of the update functionality. The SARPs therefore took account of the need to enable partial implementation whilst still retaining the interoperability required by the ICAO standards. This led to the development of subsetting rules, and the identification of conformant configurations.

1.3.8 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 keep in step with each other. However, the ADSP looked generally at a longer timescale than the current ATNP initial implementation programme, and this inevitably meant that some elements of their work has not been incorporated into the present SARPs.

1.3.9 The ATNP identified a set of packages to accommodate the continued specification of operational requirements by the ADSP. This document provides guidance material for version 1 of the FIS(ATIS) Application, previously known as CNS/ATM-1 Package.

1.3.10 ICAO approved the SARPs and established a Configuration Control Board (CCB) to manage any change required to that SARPs.

1.4

Structure

1.4.1 Chapter 1 - 1. Introduction - contains the reason for providing guidance material as well as the scope. In addition, it provides a brief overview of the FIS(ATIS) functionality, FIS' relationship with other SARPs, and identifies applicable reference document.

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

1.4.3 Chapter 3 - FIS SERVICE DESCRIPTION - gives a functional breakdown of the various services that CM 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 - FIS(ATIS) CHAPTER DESCRIPTION - clarifies any functionality that was not addressed in Chapter 3 on a chapter by chapter basis.

1.4.5 Chapter 5 - DIMENSIONS - 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 - ASN.1 INDEXES and TABLES - describes each ASN.1 field composing the FIS protocol data units as well as provides a cross-reference between fields.

1.4.7 Chapter 7 - EXAMPLE SCENARIOS - gives some examples as to what typical scenarios one can expect in course of normal FIS(ATIS) operation.

1.4.8 Chapter 8 - EXAMPLE ENCODING - outlines some actual sample PER encoding of typical FIS(ATIS) messages.

1.5 FIS(ATIS) Data Link Application Overview

1.5.1 Summary

1.5.1.1 The FIS(ATIS) Application described in the FIS(ATIS) SARPs allows a pilot to request and receive ATIS information from ground FIS systems via data link. It provides both air and ground users with the FIS Data Link Service limited to the ATIS information. The ATIS data link service supplements the existing availability of ATIS as a voice broadcast service, provided at aerodromes world-wide. All types of ATIS currently in use are encompassed (i.e. arrival, departure and combined).

1.5.1.2 The aircraft (pilot and/or avionics) requests the service by generating a request message for transmission to a FIS ground system. A FIS contract is then established by the FIS service provider which could take one of the two following forms:

a) the *FIS Demand Contract* where the ground FIS system provides the information immediately and once only, and

b) the *FIS Update Contract* where the ground FIS system provides the information and any subsequent update of this information.

1.5.1.3 These two types of FIS contract have been identified based on the analysis of the ATIS and METAR services as described in [3]. It is likely that additional type of contracts (e.g. FIS Periodic Contract) will be identified to support other data link FIS Services.

1.5.2 Establishment and Operation of a FIS Demand Contract for ATIS Information

1.5.2.1 This function allows the air system to establish a FIS demand contract with a FIS ground system, and then for the conditions of that contract to be realised. Realisation of the demand contract involves the sending of a single FIS report from the ground system to the aircraft, optionally after the sending of a positive acknowledgement.

1.5.2.2 Multiple FIS demand contracts may be established in parallel with the same ground FIS system.

1.5.2.3 The aircraft sends a FIS demand contract request to the FIS ground system. This contains an indication of which airport the requested ATIS information is related to. If the aircraft wants to retrieve the ATIS of several airport, it will establish one FIS demand contract per airport.

1.5.2.4 The ground FIS system then determines whether or not it is able to comply with the request.

1.5.2.4.1 If the ground FIS system detects that the requested ATIS information can be retrieved but is not yet available, the ground FIS system formats and sends to the airborne FIS system a FIS-positive-acknowledgement message first to indicate its acceptance of the contract, and the FIS-report message later.

1.5.2.4.2 If the ground FIS system can comply promptly with the FIS demand contract request it sends the FIS-report message as soon as possible.

1.5.2.4.3 If there are errors in the FIS-demand-contract message, or if the ground FIS system cannot comply with the request, it sends a FIS-contract-reject message to the airborne FIS system indicating the reason for its inability to accept the contract.

1.5.3 Establishment and Operation of a FIS Update Contract for ATIS Information

1.5.3.1 This function allows the airborne FIS system to establish an Update Contract with a ground FIS system. Realisation of the update contract involves the sending of FIS reports from the ground FIS system to the aircraft each time the requested FIS information is modified.

1.5.3.2 Multiple FIS update contracts may be established in parallel with the same ground FIS system.

1.5.3.3 The airborne FIS system sends a FIS update contract request to the ground FIS system. This contains which airport the requested ATIS information is related to.

1.5.3.4 The ground FIS system then determines whether or not it is able to comply with the request.

1.5.3.4.1 If the ground FIS system can comply with the FIS-update-contract request, it sends the first FIS report as soon as possible, and whenever the requested FIS information is modified, it sends a new FIS report to the aircraft.

1.5.3.4.2 If the ground FIS system detects that the requested FIS information can be retrieved but is not yet available, then it sends to the airborne FIS system a FIS positive acknowledgement first to indicate its acceptance of the contract, and then sends the FIS reports.

1.5.3.4.3 If there are errors in the FIS update contract request, or if the ground FIS system cannot comply with the request, it sends a FIS-contract-reject message to the airborne FIS system indicating the reason for its inability to accept the contract.

1.5.3.4.4 If the ground FIS system does not support the update contract function, it sends a FIS-contract-reject message containing, if available, the requested FIS information.

1.5.4 Cancellation of FIS Contracts

1.5.4.1 This function allows both air and ground FIS systems to cancel a particular FIS update contract that is in operation.

1.5.4.2 A FIS-update-contract-cancel request is sent by the system initiating the termination. Any FIS report previously sent is delivered to the aircraft before the contract is effectively ended. Other pending FIS contracts are not disrupted by the termination of a particular FIS update contract.

1.5.4.3 The cancellation of the FIS update contract is confirmed to the FIS-user by the system receiving the FIS update contract cancel message.

1.5.4.4 The airborne FIS system may also cancel all FIS contracts (demand and update contracts) of the same type by sending a FIS-cancel-contracts request. The ground FIS system cancels these contracts and acknowledges the cancellation by sending a FIS-cancel-contracts-accept request. The cancellation is made on the basis of the type(s) of contract supplied by the airborne FIS system.

1.5.5 Aborts

1.5.5.1 This function allows the airborne system, the ground system or the communications system to abort a connection in cases where a serious problem has occurred.

1.5.5.2 If the communications part of the airborne or ground systems, or the network itself, detects an error, either in itself or in the protocol arriving from its peer, it will initiate a FIS provider abort.

1.5.5.3 Id the user part of the airborne or the ground system detects an error, it has the option of initiating a FIS user abort.

1.5.5.4 In either of these cases, the result is that the connection is closed down immediately. Some of the messages already transmitted, but not yet confirmed, may be lost in transit.

1.6 Inter-relationships with Other SARPs

1.6.1 There is no interaction between FIS(ATIS) SARPs and the other CNS/ATM-1 Applications SARPs.

1.6.2 The FIS(ATIS) SARPs make use of the Upper Layer Application SARPs [2] to perform dialogue service functions required by the FIS(ATIS) Application.

1.7 Structure of the FIS(ATIS) 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. FIS(ATIS) SARPs are no different in basic layout from all other air-ground applications SARPs.

1.7.2 The FIS(ATIS) SARPs constitute the fourth part of sub-volume 2.

1.7.3 SARPs Section 2.4.1 - INTRODUCTION - gives a very brief, high level description of FIS, as an application enabling FIS services to be provided to a pilot via the exchange of messages between aircraft avionics and ground FIS 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 SARPs Section 2.4.2 - GENERAL REQUIREMENTS - contains information and high level requirements for the maintenance of backward compatibility and error processing.

1.7.5 SARPs Section 2.4.3 - ABSTRACT SERVICE - defines the abstract service interface for the FIS Application. The FIS Application Service Element (FIS-ASE) abstract service is described from the viewpoint of the FIS-air-user, the FIS-ground-user and the FIS-service-provider.

1.7.6 SARPs Section 2.4.4 - FORMAL DEFINITION OF MESSAGES - describes the contents of all permissible FIS messages through definition of the FIS ASN.1 abstract syntax. All possible combinations of message parameters and their range of values are detailed.

1.7.7 SARPs Section 2.4.5 - PROTOCOL DEFINITION - splits up the specification of the FIS(ATIS) protocol into three parts: sequence diagrams for the services covered by the abstract service, protocol descriptions and error handling for the FIS-Air- and Ground-ASEs, and State Tables.

1.7.8 SARPs Section 2.4.6 - 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 SARPs Section 2.4.7 - FIS USER REQUIREMENTS - describes the requirements imposed on the FIS-users concerning FIS messages and interfacing with the FIS-ASEs.

1.7.10 SARPs Section 2.4.8 - SUBSETTING RULES - specifies conformance requirements which all implementations of the FIS 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 will 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] Flight Information Services Application, Annex 10, Volume III, Part 1, Chapter 3 (ATN), Appendix A, Sub-Volume II - Air-Ground Applications, section 2.4.

[2] Upper Layer Communications Service, Annex 10, Volume III, Part 1, Chapter 3 (ATN), Appendix A, Sub-Volume IV.

[3] ICAO Manual of Air Traffic Service (ATS) Data Link Applications, ICAO ADS Panel, September 1996.

2.

OVERALL GENERAL FUNCTIONALITY

2.1 General

2.1.1 There is not just one FIS Data Link Application but several FIS(DFIS Service) Data Link Applications. There are as many FIS Applications as there are DFIS services. The FIS(ATIS) SARPs describe one of these applications: the FIS(ATIS) Data Link Application[1].

2.2 Topology

2.2.1 The FIS Application functions on a client/server architecture. FIS ground systems are servers providing FIS services to any clients, which are the data link equipped aircraft hosting the FIS application.

2.2.2 A FIS ground system maintains FIS information related to one or several DFIS services. For instance, a FIS ground system may be specialized for providing ATIS information only, or both ATIS and TAF/METAR information. There is a different FIS Application developed for each DFIS service. Each FIS Application is identified by an Application Name and an ATN address (both standardized and attributed by ICAO).

2.2.3 A FIS(ATIS) ground system maintains ATIS information related to one or several airports. For instance, a FIS(ATIS) system may be implemented in one airport to provide the ATIS for this airport or a centralized FIS(ATIS) system may provide the ATIS for several airports.

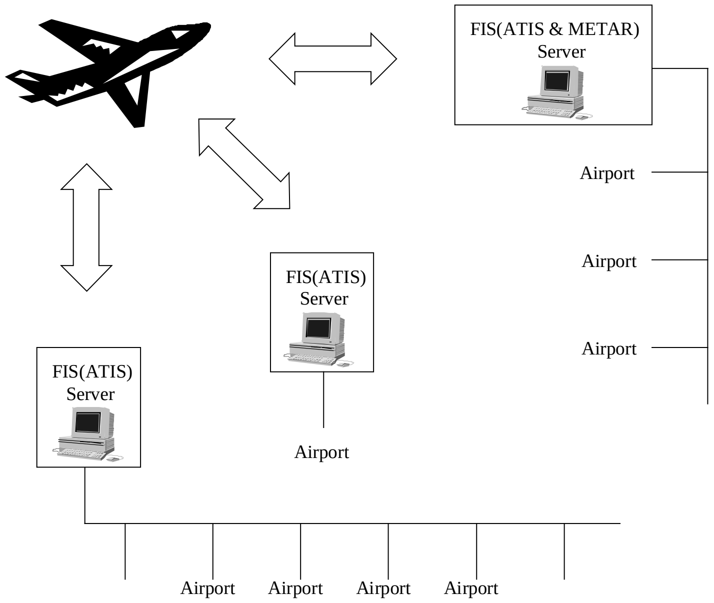
2.2.4 This flexible architecture permits multiple implementation choices concerning the ground topology definition. This choice is made by the authorities providing the FIS information based on their own criteria: for example accessibility of the FIS information, existence of communication infrastructure, distribution of the FIS addressing access points.

2.2.5 The selection of a particular ground topology does not impact the operational procedures for FIS. A FIS equipped aircraft can inter-operate with centralized and non-centralized FIS ground systems. However, as the airborne system must know the ATN address of the contacted FIS ground systems, the number of CM exchanges before establishing a FIS contract is dependent on the type of these systems: a single CM exchange is needed for a FIS ground system centralized at the country scale whereas several CM exchanges may be required when several FIS ground systems cover the country.

2.2.6 The following topologies may be envisaged:

1. the FIS(ATIS) ground system is tied to one airport (i.e. it contains the ATIS information of that airport only),
2. a centralized FIS(ATIS) ground system is tied to several airports (of a country or a region),
3. the ground FIS system providing the ATIS service can be used for providing other types of FIS information.

[1] In this document, "FIS Application" is a generic term referring to all FIS(XXX) Data Link Applications whereas "FIS(ATIS)" Application refers to the specific instance of the FIS Application developed for providing the ATIS service.



2.2.7 How the infor ; of the
FIS information (e.ş of the
responsibility of the . y or by
bilateral agreements

2.3.1 The architectu (ULA)
SARPs [2]:

- One Application (ATIS)
  Application Entit
- Each AE contai lement
  responsible for t Service
  Element. There i e same
  FIS(ATIS) AE. T ver the
  lifetime of the ASE.
- The internal abstract structure of the ATN ASE is not standardised across applications. The FIS(ATIS) ASE contains four modules which are described later.
- The dialogue service is the ATN ASE's view of the ATN Upper Layer Architecture. The modules mentioned above use the Dialogue Service for communication with the peer ASE through the ATN.

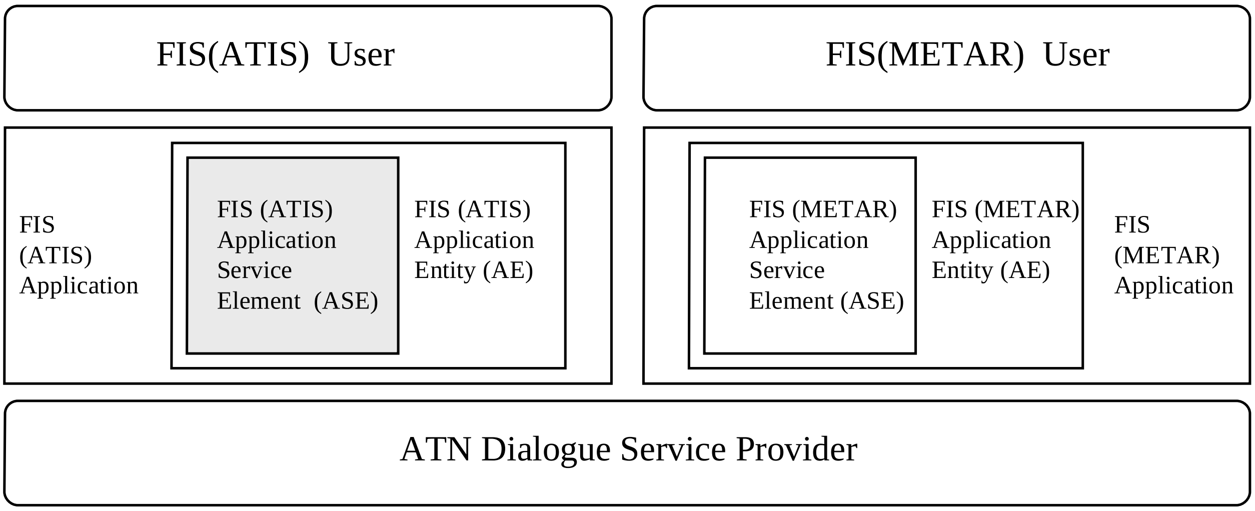

FIS(ATIS) User
FIS(METAR) User
FIS (ATIS) Application
FIS (ATIS) Application Service Element (ASE)
FIS (ATIS) Application Entity (AE)
FIS (METAR) Application Service Element (ASE)
FIS (METAR) Application Entity (AE)
FIS (METAR) Application
ATN Dialogue Service Provider

Figure 0-2: Abstract Structure of the FIS Applications

2.3.2 The FIS(ATIS) SARPs describe the FIS(ATIS) Application Service Element. The service provided by the FIS(ATIS) ASE at its top interface is provided as suchby the FIS(ATIS) AE to the FIS(ATIS)-users.

2.3.3 The FIS(ATIS) ASE handles the FIS(ATIS) protocol. If individual functions are not supported as allowed by the subsetting rules, the ASE will ensure that the protocol will handle the remaining subset of the FIS(ATIS) functionality.

2.3.4 The FIS(ATIS)-air-user is responsible for initiating the FIS contracts and making the FIS-reports available to the end user, i.e. the pilot or the crew members. The FIS(ATIS)-ground-user is responsible for the operation of the FIS contracts. On receipt of the contract request, it is responsible for responding to the request and then creating and submitting FIS reports in line with the contract. The functionality of the FIS(ATIS)-users is described in SARPs section 2.4.7.

2.3.5 The internal structure of the FIS(ATIS) ASE has been defined based on the following principle: the basic procedures used to support the operation of the FIS contracts are common to most DFIS services. The major difference amongst FIS contracts is the nature of the operational data exchanged during the contract lifetime, not the nature of the procedures. It is therefore possible to define generic ASE modules common to several FIS Applications. Those generic FIS ASE modules are integrated in a specific FIS Application and are slightly customised to support the specific DFIS service.

2.3.6 As far as the FIS(ATIS) Application is concerned, four generic FIS ASE modules have been defined for handling the ATIS protocol:

- the FIS Demand Contract (DC) module,
- the FIS Update Contract (UC) module,
- the FIS Cancel Contract (CL) module, and
- the FIS Abort Contract (AB) module.

2.3.7 The abstract internal structure of both air and ground FIS(ATIS) ASEs is illustrated in figure 2-3. Two interface modules have been defined to cope with the interface with the ASE-users (High Interface (HI) module) and with the Dialogue Service Interface (Low Interface (LI) module).

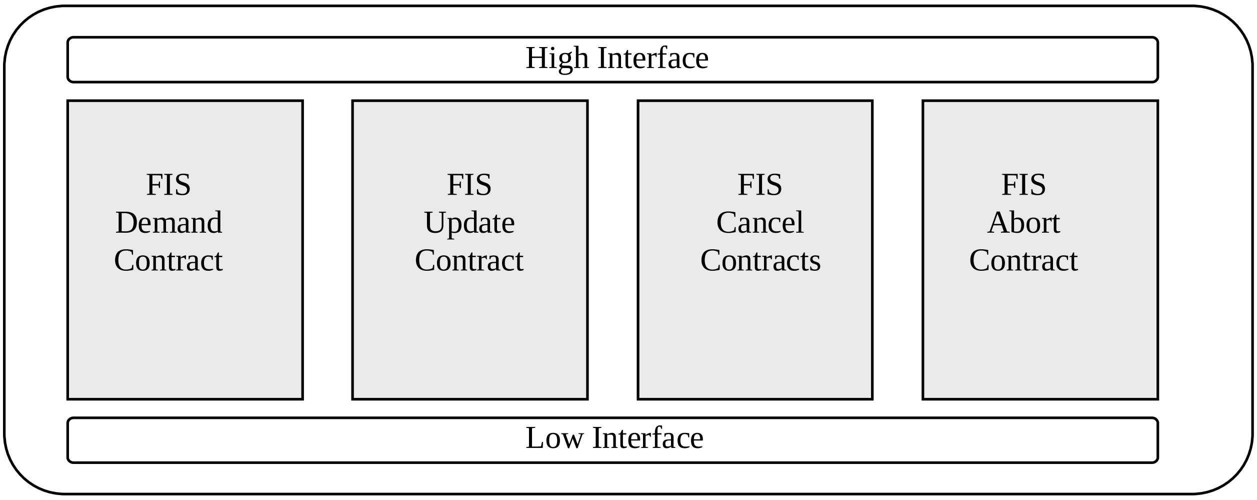


Figure 0-3: Abstract Structure of the FIS(ATIS) ASE

2.3.8 This flexible architecture allows for the reuse of common FIS ASE modules in FIS Applications that will be specified in the future. In addition, additional ASE modules could be added in the ASE to provide additional services

2.3.9 It is strongly emphasized that there is no requirement on implementers to build the interface defined in the SARPs between the ASE and the user. If it is convenient, from an engineering perspective, to build an interface between two modules that embody the functionality of the ASE and the user, then the implementers are free to do so. However, if it is more convenient to build the system with interfaces in other places, then that is also acceptable. In testing a product to see if it conforms to the SARPs, no test can be made to test internal interfaces within the system.

2.4 Implementation Dependent Functionality

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

2.4.2 The definition of the ATIS reports in the SARPs is quite open. It has not been possible to specify with the ASN.1 notation all the constraints and relationships between the ATIS fields. The rules for building a consistent ATIS report are not checked by the FIS ASEs and have to be implemented by the FIS-users.

2.5 Rationale for ASE / Users Split

2.5.1 The rationale for the split in functionality between the FIS ASEs and the FIS-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 dialogue service primitive and also ensure that the peer ASE is behaving according to the requirements in the SARPs. The ASE thus ensures interoperability.

2.5.1.2 Chapter 7 of the SARPs define some requirements for the users. These are minimum user requirements necessary to ensure the semantic interoperability of the two peers. Thus it explains how each FIS contract is interpreted and how it should operate.

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 FIS 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 FIS(ATIS) Application and the other CNS/ATM-1 Data Link applications. The FIS(ATIS) 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 FIS(ATIS) dialogue, some naming and addressing information have to be known from the dialogue initiator:

- the ATN address of the ground FIS(ATIS) system must be present somewhere in the data link communication system. Without this information, the dialogue with the ground FIS system can not be established.
- the version number of the FIS(ATIS) protocol run in the ground FIS system must be available to the FIS-air-user. The protocol version negotiation is not performed by the FIS(ATIS) Application Entities and the FIS-air-user is responsible for checking the version compatibility before establishing a FIS Contract.

2.6.3 The Context Management (CM) Application is the most natural way to exchange this addressing information. 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. A CM exchange is therefore required between the aircraft and the ground CM system administrating the contacted FIS system. There is no direct interaction between the CM Application and the FIS(ATIS) Application (in fact there are not running at the same time), but the addressing information exchanged by the CM Application is required to establish a FIS contract.

2.6.4 In the hypothesis the CM Application is run, the definition of the ground topology for CM shall take into account the presence of the FIS ground systems as follows:

1. a CM area can not contain more than one FIS ground system of a same version number, and
2. a FIS ground system can belong to more than one CM area.

2.6.5 Two pieces of addressing information must be available before a FIS Contract can be established:

1. the identification of the ground FIS system which holds the desired information,
2. the identification and the address of the CM Application attached to the ground FIS system identified in 1/ above. The CM Logon procedure with this ground CM system is run to retrieve the address of the ground FIS system.

2.7 Ground FIS Exchanges

2.7.1 No operational requirement related to the ground FIS exchanges have been specified in [3]. As a consequence, there is no ground FIS protocol defined between FIS-ground-users in the first version of the FIS(ATIS) Application.

2.7.2 As a consequence of the non availability of ground communications between FIS ground systems, a request sent by an aircraft to a FIS ground system cannot be forwarded to an other FIS ground system. Thus, FIS(ATIS) requests must be sent in the first place to the appropriate ground FIS system, i.e. the one which holds the requested information. If the requested information is not maintained by the FIS ground system receiving the request, the request is rejected.

2.7.3 Standardised FIS ground forwarding exchanges will be considered by ICAO in the scope of CNS/ATM-2 Package. However, similar ground exchanges may be implemented by some states (based on bilateral agreements) between FIS ground systems allowing the forwarding of both FIS requests and reports. This topology is outside the scope of the FIS(ATIS) SARPs.

2.8 Dialogue Management

2.8.1 General

2.8.1.1 The dialogue is always initiated by the airborne system, by the process of setting up a FIS contract. In like manner, the airborne system is always responsible for closing the dialogue.

2.8.1.2 The dialogue service provider is used by the FIS-ASEs for the following purposes:

- establishment, graceful release and abort of a dialogue,
- transfer of unstructured data,
- support for quality of service and version number negotiation, and
- application naming.

2.8.2 Optimisation of the use of dialogues

2.8.2.1 The FIS(ATIS) service hides the use of the underlying communication service to the FIS-users: the FIS service users do not realise when the dialogue is being established or released.

2.8.2.2 Multiplexing over a single dialogue of FIS contracts set up between an aircraft and a ground FIS system is supported by the FIS protocol. Up to 256 FIS contracts can be multiplexed in parallel on a single dialogue. The time required to establish the dialogue before any operational data can be exchanged penalises therefore only the first FIS contract. The dialogue multiplexing is performed by the Low Interface Module.

2.8.3 Dialogue Establishment

2.8.3.1 If no dialogue was established with the ground FIS system (i.e. if no connection has already been established), a dialogue is established first, then data related to the FIS contract are then exchanged. During the time of the dialogue establishment, no new request can be accepted by the FIS-service provider.

2.8.3.2 On receipt of a FIS contract establishment request, if a dialogue is still in place and was maintained for a future use, it is used immediately for data transmission.

2.8.4 Dialogue Release

2.8.4.1 The dialogue is closed when no activity takes place during a certain period of time. Without explicit action from the pilot or a ground operator, the air FIS system triggers the release of the dialogue in place. The value for the inactivity timer is not standardised in the FIS(ATIS) SARPs. It could be customised for each airborne system.

2.9 Protocol Monitoring

2.9.1 The FIS(ATIS) ASE controls that the protocol is correctly handled by the peer and can be correctly operated locally.

2.9.2 In case a serious error is detected, the dialogue in place with the peer FIS system is aborted and all contracts in place or being established are cancelled. Active FIS-users are informed of this situation by a FIS-provider-abort indication received at both side and providing a reason for the abort.

2.9.3 Possible causes of dialogue abort are listed below:

2.9.3.1 **Timer Expiration**: in case of a confirmed service (FIS-demand-contract, FIS-update-contract, FIS-cancel-update-contract and FIS-cancel-contracts), the generation and the transmission of the response expected to be issued by the remote FIS-user or FIS-ASE are monitored by the FIS ASEs by activating a timer.

2.9.3.2 **Protocol Error**: on receipt of a dialogue service indication or confirmation, the ASE checks that the APDU transmitted in the user data parameter of this primitive are authorised.

2.9.3.3 **Sequence Error**: on receipt of a dialogue service indication or confirmation, the ASE checks that the invoked dialogue service is authorised.

2.9.3.4 **Decoding Error**: if the received PDU cannot be decoded, a decoding error is raised highlighting a transmission error or an encoding error by the APDU sender.

2.9.3.5 **Unrecoverable Internal Error**: the capacity of the system running the application may be temporarily limited as such that the FIS function can not be provided anymore. This could be the case if the system get short of memory. If the system has enough resources to do it, it will abort the connection and inform both the local user and the peer of the situation.

2.9.3.6 **Invalid Contract Number**: the contract number supplied by the FIS-users must be unallocated when establishing a new contract. Otherwise the contract number must correspond to an existing contract.

2.9.3.7 **Dialogue End Not Supported**: the release of the dialogue must always be accepted by the peer.

2.9.3.8 **Invalid QOS parameter**: the values for the priority and the residual error rate parameters used by the peer are checked on reception of a D-START indication.

2.10 Version Number Negotiation

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

2.10.2 The version numbers supported by the air and ground systems for the FIS(ATIS) application are exchanged during the CM procedures. The FIS-air-user must check whether the ASE implemented onboard is compatible with one of the versions implemented on the ground.

2.11 Number of Concurrent FIS(ATIS) Contracts

2.11.1 In theory, there is no lower and upper limits specified in the SARPs for the number of concurrent FIS(ATIS) contracts.

2.11.2 However, dimensioning and cost consideration should be given when allowing an airborne system to run multiple FIS contracts with several ground systems. There will be as many transport connections set up in parallel, as there are contacted ground systems. When parallel contracts are set up with the same ground system, only one transport connection is used and all contracts are multiplexed over this dialogue.

3.

FUNCTIONALITY OF SERVICES

3.1 Concepts

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

3.1.2 This section describes first the information provided by the ASEs to the FIS-users or supplied by the FIS-users to the ASEs when FIS service primitives are invoked. Then, it considers FIS Service in turn and provides a overview of the data flow within the ASE which handles the service primitives. The primitives are grouped according to the services they provide: setting a FIS Demand Contract, setting a FIS Update Contract, cancelling and aborting FIS contracts.

3.2 FIS Service Parameters

3.2.1 FIS Contract Number

3.2.1.1 The *FISContractNumber* parameter identifies unambiguously any contract in place between a peer of FIS-users.

3.2.1.2 This identifier is allocated by the FIS-air-user when establishing a new FIS contract and supplied to the FIS-air-ASE as the *FISContractNumber* parameter of a FIS-demand-contract request or a FIS-update-contract request primitive. It is then supplied by the FIS-users in any subsequent request or response primitives or provided by the ASE in any subsequent indication or confirmation primitives related to a specific FIS contract.

3.2.1.3 Unallocated number must be used when invoking a FIS-demand/update-contract request. The invocation of terminal primitives (such as FIS-demand/update-contract confirmation (accepted) or FIS-user-abort request/indication) causes the de-allocation of the current contract number(s) which could be used for a new FIS contract.

3.2.1.4 The FIS contract number supplied by the FIS-ground-ASE when invoking the FIS-report service or the FIS-cancel-update-contract service must have been assigned previously to an existing FIS contract.

3.2.2 ICAO Facility Designation

3.2.2.1 The *ICAOFacilityDesignation* parameter is used to indicate the four to eight characters facility designation of the called ground FIS system.

3.2.3 Class Of Communication Service

3.2.3.1 The *ClassofCommunicationService* parameter is a means for the FIS-air-user to indicate the required performance in terms of end-to-end transit delay. Values for these transit delays are given in Sub-Volume 1 of the ATN SARPs.

3.2.3.2 The *ClassOfCommunicationService* parameter provided by the FIS-air-user is passed to the Transport Service Provider (TSP) in the T-CONNECT *SecurityLabel* parameter when the connection is established

3.2.3.3 The value of the Class of Communication requested by a FIS-air-user for a dialogue is not transmitted to the FIS-ground-user.

3.2.3.4 The Class of Communication Service is not guaranteed nor a degradation of the provided class is indicated to the users. It is of the responsibility of the application users to determine the actual transit delay achieved by local means such as time stamping.

3.2.3.5 There is no negotiation of the Class of Communication Service between air and ground FIS-users.

3.2.3.6 If the FIS-air-user does not require a particular Class of Communication, the *ClassofCommunicationService* parameter may be left blank indicating no routing preference. That means that the Class of Communication Service is chosen by the Dialogue Service Provider.

3.2.4 FIS Contract Details

3.2.4.1 The *FISContractDetails* parameter contains the details of the FIS Contract requested by the FIS-air-user (i.e. the contract number, the contract type, the FIS service type, the airport and the ATIS type requested). In version 1 of the FIS protocol only ATIS information may be requested.

3.2.4.2 The *FISContractDetails* parameter is supplied by the FIS-air-user when requesting the establishment of a FIS demand contract or a FIS update contract.

3.2.4.3 The *FISContractDetails* parameter is passed transparently to the FIS-ground-user in the corresponding indication. However, the FIS service provider checks before the data encoding that the values supplied by the user are compatible with the syntactical description of the data types.

3.2.5 FIS Information

3.2.5.1 The *FISInformation* parameter contains the FIS information as requested by the FIS-air-user. In version 1 of the FIS protocol, only ATIS data element may be included in this parameter: arrivalATIS, departureATIS, combined ATIS or both arrival and departure ATIS.

3.2.5.2 The *FISInformation* parameter is supplied by the FIS-ground-user when accepting a FIS contract and when sending an update of the requested FIS information. It may also optionally be used by an FIS-ground-user not supporting the update functionality when rejecting the FIS update contract.

3.2.5.3 The *FISInformation* parameter is passed transparently to the FIS-air-user. However, the FIS service provider checks before the data encoding that the values supplied by the user are compatible with the syntactical description of the data types.

3.2.6 Result

3.2.6.1 The *Result* parameter is used by the FIS-ground-user to indicate in a FIS-demand/update-contract response whether it accepts or rejects the FIS demand/update contract.

3.2.6.2 The *Result* parameter is passed transparently to the FIS-air-user in the corresponding FIS-demand/update-contract confirmation.

3.2.7 Reject Reason

3.2.7.1 The *RejectReason* parameter is used by the FIS-ground-user to indicate why it rejects the FIS-demand/update contract.

3.2.7.2 The abstract value "canNotComply" is used when the request is valid but the requested information is not available (e.g. the ATIS request for a particular airport is not sent to the appropriate FIS ground server).

3.2.7.3 The abstract value "FISServiceUnavailable" is used when the service requested is noty supported by the FIS ground server (e.g. a METAR request received by a FIS ground server dedicated to ATIS).

3.2.7.4 The abstract value "erroInRequest" is used when the request can be decoded but the values are wrong.

3.2.7.5 The abstract value "undefined" is used for all other cases of rejection.

3.2.7.6 The *RejectReason* parameter can be provided by the FIS-ground-user only in that case it rejects the establishment of the FIS contract.

3.2.7.7 The *RejectReason* parameter is passed transparently to the FIS-air-user in the corresponding FIS-demand/update-contract confirmation.

3.2.8 Reason

3.2.8.1 The *Reason* parameter contains the reason for an abort initiated by the FIS service provider. It is indicated to the users in the FIS-abort indication primitive.

3.2.8.2 The possible abstract values for the *Reason* parameter are described in section 2.9 of this document.

3.2.9 FIS Service Type

3.2.9.1 The *FISServiceType* parameter is used by the FIS-users to indicate which type of FIS contracts has to be cancelled when a multiple contracts cancellation is requested. In version 1 of the FIS protocol, only FIS(ATIS) contracts can be cancelled with the FIS-cancel-contracts services.

3.3 Setting a FIS Demand Contract

3.3.1 The *FIS-demand-contract* service is used to set up a FIS Demand Contract between a FIS-air-user and a FIS-ground-user. It is a confirmed service, initiated by the FIS-air-user.

- The FIS-demand-contract request is passed to the air HI module which examines it to see which other module to pass it on to. The request is passed to the air DC module. The FIS Contract Number attached to the new demand contract is stored by the air HI module.

- The air DC module generates a FISDownlinkAPDU [FISRequest] APDU. It passes the APDU to the air LI module with the ground system identifier and the class of communication supplied by the user.

Timers t-DC-1 and t-DC-2 are started to monitor the reception of the reply from the ground ([FISAccept] APDU or [FISReject] APDU) and the reception of the FIS report when the response is postponed.

- The air LI module decides how to use the dialogue service. If a dialogue exists already, it makes use of that dialogue and uses the D-DATA service to pass the APDU to the ground system. If no dialogue exist, it uses the D-START service to pass the APDU to the ground system. The APDU is passed to the ground system by way of the upper and lower layers and emerges in the ground LI module.

- The ground LI module examines the APDU in order to decide which module to pass it to. The APDU is passed to the ground DC module.

- The ground DC module recognises the APDU as FISDownlinkAPDU [FISRequest] APDU and passes a FIS-demand-contract indication to the ground HI module.

- The ground HI module passes it to the FIS-ground-user.

3.3.2 The FIS demand contract may be either accepted (in that case, a FIS report is enclosed in the response message), rejected (a reject reason is provided) or postponed (the FIS report is sent later using the FIS-report service) by the ground FIS system.

- The FIS-demand-contract response is passed to the ground HI module which examines it to see which other module to pass it on to. The request is passed to the ground DC module.

- The ground DC module generates a FISUplinkAPDU [FISAccept] or [FISReject] APDU based on the reply supplied by the FIS-ground-user. It passes the APDU to the ground LI module.

- The ground LI module then passes the APDU to the air using a D-DATA request (if a dialogue is already open) or a D-START response (if no D-START response has yet been given). The APDU is passed to the airborne system by way of the upper and lower layers and emerges in the air LI module.

- The air LI module examines the APDU in order to decide which module to pass it to. The APDU is passed to the air DC module.

- The air DC module recognises the APDU as FISUplinkAPDU [FISAccept] or [FISReject] APDU and accordingly passes a positive or negative FIS-demand-contract confirmation to the air HI module. If the response is not postponed and if there is no FIS contract in place, the timer t-inactivity is started.

- The air HI module passes it to the FIS-air-user.

3.3.3 The *FIS-report* service is then invoked by the FIS-ground-user to both send a FIS report and close a previously postponed FIS Demand Contract. It is an unconfirmed service initiated by the FIS-ground-user.

- The FIS-report request is passed to the ground HI module which examines it to see which other module to pass it on to. The decision is taken based on the FIS contract number supplied by the user and the contracts already managed and identified by the FIS ASE. The request is passed to the ground DC module.

- The ground DC module generates a FISUplinkAPDU [FISReport]. It passes the APDU to the ground LI module.

- The ground LI module then passes the APDU to the air using a D-DATA request. The APDU is passed to the airborne system by way of the upper and lower layers and emerges in the air LI module.

- The air LI module examines the APDU in order to decide which module to pass it to. The APDU is passed to the air DC module.

- The air DC module recognises the APDU as FISUplinkAPDU [FISReport] APDU and passes a FIS-report indication to the air HI module. If there is no FIS contract in place, the timer t-inactivity is started.

- The air HI module passes it to the FIS-air-user.

3.4 Setting a FIS Update Contract

3.4.1 The *FIS-update-contract* service sets up a FIS Update Contract between a FIS-air-user and a FIS-ground-user. It is a confirmed service, initiated by the FIS-air-user.

- The FIS-update-contract request is passed to the air HI module which examines it to see which other module to pass it on to. The request is passed to the air UC module. The FIS Contract Number attached to the new demand contract is stored by the air HI module.

- The air UC module generates a FISDownlinkAPDU [FISRequest] APDU. It passes the APDU to the air LI module with the ground system identifier and the class of communication supplied by the user. Timers t-UC-1 and t-UC-2 are started to monitor the reception of the reply from the ground ([FISAccept] APDU or [FISReject] APDU) and the reception of the FIS report when the response is postponed.

- The air LI module decides how to use the dialogue service. If a dialogue exists already, it makes use of that dialogue and uses the D-DATA service to pass the APDU to the ground system. If no dialogue exist, it uses the D-START service to pass the APDU to the ground system. The APDU is passed to the ground system by way of the upper and lower layers and emerges in the ground LI module.

- The ground LI module examines the APDU in order to decide which module to pass it to. The APDU is passed to the ground UC module.

- The ground UC module recognises the APDU as FISDownlinkAPDU [FISRequest] APDU and passes a FIS-update-contract indication to the ground HI module.

- The ground HI module passes it to the FIS-ground-user.

3.4.2 The update contract may be either accepted (in that case, the FIS report is enclosed in the response message), rejected (a reject reason is provided, optionally the FIS report is enclosed in the response message) or postponed (the FIS report is sent later) by the ground FIS system. A FIS Update Contract must be explicitly closed by the FIS-users.

- The FIS-update-contract response is passed to the ground HI module which examines it to see which other module to pass it on to. The request is passed to the ground UC module.

- The ground UC module generates a FISUplinkAPDU [FISAccept] or [FISReject] APDU based on the reply supplied by the FIS-ground-user. It passes the APDU to the ground LI module.

- The ground LI module then passes the APDU to the air using a D-DATA request (if a dialogue is already open) or a D-START response (if no D-START response has yet been given). The APDU is passed to the airborne system by way of the upper and lower layers and emerges in the air LI module.

- The air LI module examines the APDU in order to decide which module to pass it to. The APDU is passed to the air UC module.

- The air UC module recognises the APDU as FISUplinkAPDU [FISAccept] or [FISReject] APDU and accordingly passes a positive or negative FIS-update-contract confirmation to the air HI module.

- The air HI module passes it to the FIS-air-user.

3.4.3 The *FIS-report* service is then invoked by the FIS-ground-user to activate a postponed FIS Update Contract and to uplink any update of the FIS information. It is an unconfirmed service initiated by the FIS-ground-user. FIS reports are delivered to the FIS-air-user in the order they have been supplied by the FIS-ground-user.

- The FIS-report request is passed to the ground HI module which examines it to see which other module to pass it on to. The decision is taken based on the FIS contract number supplied by the user and the contracts already managed and identified by the FIS ASE. The request is passed to the ground UC module.

- The ground UC module generates a FISUplinkAPDU [FISReport]. It passes the APDU to the ground LI module.

- The ground LI module then passes the APDU to the air using a D-DATA request. The APDU is passed to the airborne system by way of the upper and lower layers and emerges in the air LI module.

- The air LI module examines the APDU in order to decide which module to pass it to based on the FIs contract number. The APDU is passed to the air UC module.

- The air UC module recognises the APDU as FISUplinkAPDU [FISReport] APDU and passes a FIS-report indication to the air HI module.

- The air HI module passes it to the FIS-air-user.

3.5 Cancelling FIS Contracts

3.5.1 Cancelling several contracts

3.5.1.1 The *FIS-cancel-contracts* service allows the FIS-air-user to cancel all FIS demand and update contracts of the same type with a particular FIS-ground-user. This service does not affect the FIS contracts of other types still in place.

- The FIS-cancel-contracts request is passed to the air HI module which examines it to see which other module to pass it on to. The request is passed to the air CL module.

- The air CL module generates a FISDownlinkAPDU [FISCancelContracts] APDU. It requests the air DC and UC modules to stop operation. It passes the APDU to the air LI module. Timer t-CL-1 is started to monitor the reception of the reply from the ground ([FISCancelContractsAccept] APDU.

- The air LI module then passes the APDU to the ground using a D-DATA request. The APDU is passed to the ground system by way of the upper and lower layers and emerges in the ground LI module.

- The ground LI module examines the APDU in order to decide which module to pass it to. The APDU is passed to the ground CL module.

- The ground CL module recognises the APDU as FISDownlinkAPDU [FISCancelContracts] APDU and passes a FIS-cancel-contracts indication to the ground HI module. It requests the ground DC and UC modules to stop operation.

- The ground HI module passes the FIS-cancel-contracts indication to the FIS-ground-user.

3.5.1.2 The FIS-cancel-contracts service is a confirmed service initiated by the FIS-air-user. As the cancellation can not be rejected by the FIS-ground-user, no response is requested. The confirmation APDU is sent automatically by the FIS-ground-ASE.

- The ground CL module generates a FISUplinkAPDU [FISCancelContractsAccept] APDU. It passes the APDU to the ground LI module.

- The ground LI module then passes the APDU to the air using the D-DATA request. The APDU is passed to the airborne system by way of the upper and lower layers and emerges in the air LI module.

- The air LI module examines the APDU in order to decide which module to pass it to. The APDU is passed to the air CL module.

- The air CL module recognises the APDU as FISDownlinkAPDU [FISCancelContractsAccept] APDU and passes a FIS-cancel-contracts confirmation to the air HI module. The t-inactivity timer is started.

- The air HI module passes the FIS-cancel-contracts confirmation to the FIS-air-user.

3.5.1.3 The FIS-air-user specifies the type of FIS contracts it wants to cancel.

3.5.2 Cancelling a particular Update Contract

3.5.2.1 The *FIS-cancel-update-contract* service allows the FIS-air-user or the FIS-ground-user to cancel in an orderly manner an active FIS update contract. The FIS reports in transit in the communication system are delivered before the FIS update contract is cancelled. This service does not affect the other FIS contracts in place.

- The FIS-cancel-update-contract request is passed to the local HI module which examines it to see which other module to pass it on to. The request is passed to the local UC module.

- The local UC module generates a FISDownlinkAPDU or FISUplinkAPDU [FISCancelUpdateContract] APDU. It passes the APDU to the local LI module. Timer t-UC-3 is started to monitor the reception of the reply from the peer ([FISCancelUpdateContractAccept] APDU).

- The local LI module then passes the APDU to the peer using a D-DATA request. The APDU is passed to the peer system by way of the upper and lower layers and emerges in the remote LI module.

- The remote LI module examines the APDU in order to decide which module to pass it to. The APDU is passed to the remote UC module.

- The remote UC module recognises the APDU as FISDownlinkAPDU or FISUplinkAPDU [FISCancelUpdateContract] APDU and passes a FIS-cancel-update-contract indication to the remote HI module. If the remote system is air based, the t-inactivity timer is started.

- The remote HI module passes the FIS-cancel-update-contract indication to the remote FIS-user.

3.5.2.2 The FIS-cancel-update-contract service is a confirmed service, initiated by the FIS-air-user or the FIS-ground-user. As the cancellation can not be rejected by the FIS-ground-user, no response is requested. The confirmation APDU is sent automatically by the FIS-ASE.

- The remote UC module generates a FISUplinkAPDU or FISDownlink [FISCancelUpdateContractAccept] APDU. It passes the APDU to the remote LI module.
- The remote LI module then passes the APDU to the peer using the D-DATA request. The APDU is passed to the peer system by way of the upper and lower layers and emerges in the local LI module.
- The local LI module examines the APDU in order to decide which module to pass it to. The APDU is passed to the local CL module.
- The local CL module recognises the APDU as FISDownlinkAPDU or FISUplinkAPDU [FISCancelContractsAccept] APDU and passes a FIS-cancel-contracts confirmation to the local HI module. If the local system is air based, the t-inactivity timer is started.
- The local HI module passes the FIS-cancel-contracts confirmation to the local FIS-user.

3.6 Aborting a FIS Contract

3.6.1 User-Initiated Abort

3.6.1.1 When a FIS-user detects an error it views as catastrophic, the *FIS-user-abort* service is invoked. All active FIS demand and update contracts processed by the ASE are cancelled. Messages in transit may be lost during this operation. This is an unconfirmed service without parameter.

3.6.1.2 The FIS-user-abort service can be invoked at any time that the FIS-user is aware that any FIS service is in operation. After this primitive is invoked, no further primitives may be invoked for the current FIS contracts.

- The FIS-user-abort request is passed to the local HI module which examines it to see which other module to pass it on to. The request is passed to the local AB module.
- The local AB module requests all running local ASE modules to stop their current activities. It request the local LI module to abort the dialogue.
- The local LI module abort the dialogue using a D-ABORT request if the dialogue is still open. The abort originator parameter is set to "user". A D-ABORT indication emerges in the remote LI module.
- The remote LI module passes the D-ABORT indication to the remote AB module.
- The remote AB module recognises a user abort situation (the abort originator of the D-ABORT indication is set to "user"). It requests all running remote ASE modules to stop their current activities. It passes a FIS-user-abort indication to the remote HI module.
- The remote HI module passes it to the remote FIS-user if still active.

3.6.2 Provider-Initiated Abort

3.6.2.1 When the FIS ASE detects an error from which it cannot recover, the *FIS-provider-abort* service is invoked. This is a FIS provider-initiated service with a single, mandatory parameter telling why the FIS Contracts in progress with a ground FIS system were lost. Messages in transit may be lost during this operation.

- The local ASE module detecting the error situation requests the local AB module to abort the communication with a reason value.

- The local AB module requests all running local ASE modules to stop their current activities. It passes a FIS-provider-abort indication to the local HI module. It generates a FISUplinkAPDU or FISDownlinkAPDU [FISAbort]. It passes the APDU to the local LI module.
- The local HI module passes the FIS-provider-abort indication to the local FIS-user if he is still active,
- The local LI module passes the APDU to the remote system using a D-ABORT 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 remote LI module.
- The remote LI module examines the APDU in order to decide which module to pass it to. The APDU is passed to the remote AB module.
- The remote AB module recognises a provider abort situation (the abort originator of the D-ABORT indication is set to "provider"). It requests all running remote ASE modules to stop their current activities. It decodes the FISUplinkAPDU [FISAbort] APDU and passes a FIS-provider-abort indication to the remote HI module.
- The remote HI module passes it to the remote FIS-user if still active.

3.6.2.2 Errors detected by the ASE are the following:

- *timer expiration*: a timer set previously by the ASE elapses.
- *protocol error*: the ASE detects that the peer ASE does not respect the rules of the protocol.
- *sequence error*: a not expected dialogue service primitive has been received by the ASE.
- *decoding error*: the ASE is not able to decode the data packet received in the user data parameter of a dialogue service primitive.
- *unrecoverable internal error*: a system function call has failed, for instance a memory allocation/release function, an I/O operation, etc...
- *invalid contract number*: the contract number received from the peer does not correspond to an existing contract.
- *dialogue end not supported*: A D-END response negative has been received, but this is not allowed by the FIS protocol.
- *undefined error*: any other error.

4.

FIS(ATIS) SECTION DESCRIPTION

4.1 SARPs Section 2.4.2: GENERAL REQUIREMENTS

4.1.1 FIS(ATIS) ASE Version Number

4.1.1.1 This section is included to allow the Context Management (CM) application to exchange version numbers of the FIS(ATIS) Application. It is necessary to allow for future versions of the FIS protocol to be exchanged by the CM Application. It has no effect on the FIS 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 the ASE is in an inappropriate 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 such 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 SARPs Section 2.4.3: THE ABSTRACT SERVICE

4.2.1 The Concept of Abstract Service

4.2.1.1 SARPs Section 2.4.3 concerns the FIS abstract service. There has been a lot of confusion over the term 'abstract service' and so it deserves some explanation.

4.2.1.2 In order to define the FIS ASE (i.e. the module that contains the protocol machine - see SARPs section 2.4.5), it is necessary to describe its reactions to both PDUs arriving from the peer ASE, and the actions from the local FIS-user. The PDUs are well defined in the protocol. The actions of the FIS-user, however, are not. The SARPs do not attempt to dictate the actions of the FIS-user except where absolutely necessary. Despite this, in order to define the ASE it is necessary to have a clear definition of user actions.

4.2.1.3 In order to get around this conundrum, an "application abstract service" is defined. The FIS abstract service is a technical description of the interactions between the FIS-user and the ASE. These interactions are precisely defined in SARPs section 2.4.3. Having this definition allows the ASE to be specified precisely in terms of its reactions to the arrival of PDUs and the invocation of the service primitives by the user. This, therefore, is the reason for defining the abstract service.

4.2.1.4 The abstract service is defined as being the description of the interface between the ASE and the ASE service-user. These are known as the FIS ASE and the FIS-user. The FIS user is generally not the human user; it is that part of the system that uses the FIS ASE.

4.2.2 The Concept of APIs

4.2.2.1 If one was to buy an FIS application, one would be buying a suite of executable code. From the code itself it is impossible to know whether or not the abstract service has actually been implemented. Therefore the FIS SARPs do not require that the abstract service has to be build has a real interface. It only requires that, when one examines it from an external point of view, it behaves in the same way as if it had been built. This is the explanation of SARPs requirement 2.4.3.1.1.

4.2.2.2 Thus the implementers may choose to build a FIS application with an Application Programmatic Interface (API) that corresponds to the FIS application abstract service, or they may choose not to - it is entirely up to them. However, it should be realized that there are a number of good reasons why one might not want to build a system with an API exactly like the abstract service. Examples include:

- there may be a more efficient way of building the software.
- the abstract service does not include parameters that are needed locally, but do not effect the state machine; for example, an API might include an indication of which ground system a FIS report has come from.
- it may not be easy to build the abstract service from the development tools that are used being used.
- the abstract service does not have any programming language bindings. An API would require an interface defined in a particular programming language.

4.2.2.3 The question then arises: since implementation of the abstract service is not mandated, what API should the implementers build? The FIS SARPs do not answer this question. The requirements for FIS set out by ICAO are limited in scope - they are designed only to ensure interoperability between air and ground systems, and to ensure that they meet the stated functionality requirements. The FIS SARPs do not specify the nature of any API within the software, nor does it specify the human interface.

4.2.2.4 In summary, an application abstract service is defined in the SARPs in order to be able to define the ASE protocol machine. It does not have to be built as an API in any implementation, and there are several good reasons why it should not be implemented exactly as defined. A real implementation of the FIS SARPs would normally be expected to define its own APIs.

4.2.3 The FIS Functional Model

4.2.3.1 SARPs Figure 2.4.3-1 shows an abstract model for the FIS application. Just as with the abstract service, this model shows a design of the FIS application, breaking it down into modules. However, there is no requirement that an implementation actually builds it this way. The figure is presented here in order to explain the terms that are used throughout the document. It is not required that the design of an implementation follows this structure.

4.2.3.2 The figure shows three modules:

- The FIS-user (which could be a FIS air user or a FIS ground user),
- The control function, and
- The FIS-ASE (application service element – which could be a FIS-air-ASE or a FIS-ground-ASE).

4.2.3.3 In addition, it defines the FIS application entity as the control function together with the FIS ASE.

4.2.3.4 Abstract interfaces are shown between the different modules:

- The FIS application entity service interface – which is the same as the abstract service interface defined in SARPs section 2.4.3,
- The FIS application service element service interface – which is also the same as the abstract service interface defined in SARPs section 2.4.3,
- The dialogue service interface – which is defined in the Upper Layer Communications SARPs, and is identical for all air/ground applications.

4.2.3.5 Since the FIS application entity service interface is identical with the FIS application service element service interface, it is easy to see that the control function module passes called directly from one to the other without interference.

4.2.4 Conventions

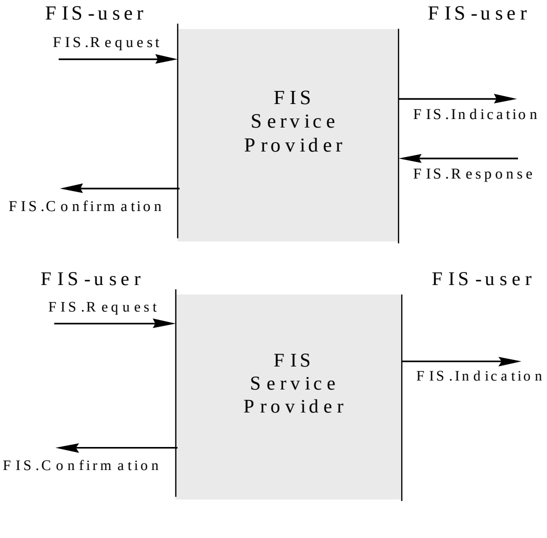
4.2.4.1 Service Primitives

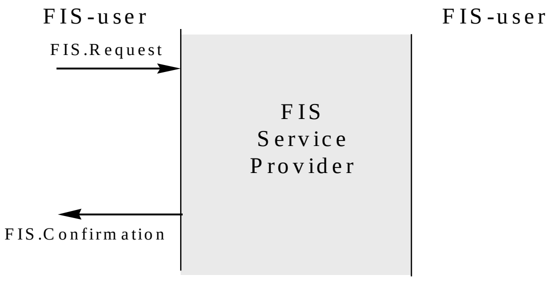
4.2.4.1.1 The FIS SARPs define seven FIS services. A FIS service consists of one to four primitives. Each primitive consists of the name of the FIS service (FIS-demand-contract, FIS-update-contract, FIS-cancel-contracts, FIS-cancel-update-contract, FIS-user-abort and FIS-provider-abort) and a suffix that indicates at what point in the service the primitive occurs (request, indication, response, confirmation):

- the FIS-user that initiates the service calls on the FIS ASE to perform an action - this is called the "request",
- after the request is passed to the FIS ASE on the other side of the communication link, it uses the service to pass the information on to its FIS user - this is called the "indication",
- the FIS-user that has received the indication may choose to respond to it, in which case it calls upon its FIS ASE to send a reply – this is called the "response",
- finally, the FIS ASE receiving the response provides its FIS-user (which started the sequence of events) with the information – this is called the "confirmation".

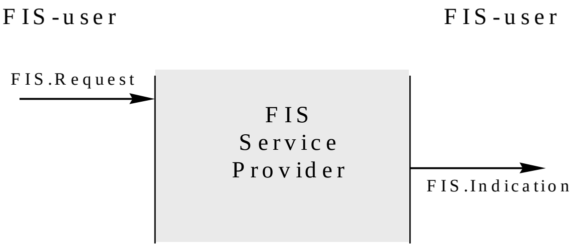
4.2.4.1.2 The terms "request", "indication", "response" and "confirmation" come from the field of communication protocols. A given service need not use all four primitives. Some FIS services make use of only one (indication), of two (request and indication, request and confirmation), some three (request, indication and confirmation) and some all four (request, indication, response and confirmation).

4.2.4.1.3 A *confirmed FIS 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:

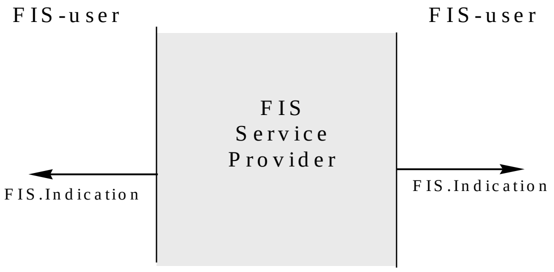




4.2.4.1.4 An *unconfirmed FIS service* involves no handshake.



4.2.4.1.5 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 FIS service provider.



4.2.4.2 Service Primitives and Parameter Tables

4.2.4.2.1 The services are depicted in the SARPs by primitive and parameter tables. Not all services require each primitive to be used for each point in the service. That is, if indication of a particular primitive is not needed due to reasons like redundant information being relayed, then the parameter column for that primitive is omitted.

4.2.4.2.2 Also, not all parameters need to be present for all indications of a particular primitive.

4.2.4.2.3 For a specific primitive, each parameter is described by a value that dictates the terms under which that parameter is used. If the use of any parameter does not follow the rules as set forth by the primitive and parameter tables, there is an error in the implementation. The abbreviations used in the primitive and parameter sections are described in the following sections.

4.2.4.2.3.1 A "C" is used to indicate that the parameter is conditional upon some state. A "C" differs from a "U" (User Option) due to the fact that if the stated condition exists, the parameter must be supplied, while a "U" means that the parameter’s use is wholly up to the user.

4.2.4.2.3.2 A "C(=)" is used to show that a parameter is conditional upon the value of the parameter to the left of it in the table being present, and if it is present, the "C(=)" parameter will retain the parameter to the left in the tables’s value.

4.2.4.2.3.3 A "M" is used to indicate that the use of that parameter is mandatory, and no option not to use it exists.

4.2.4.2.3.4 A "M(=)" is used to indicate that te parameter will always take the value to the left of it in the table; its presence is not conditional.

4.2.4.2.3.5 A "U" is used to indicate that the use of the parameter is a user option. Therefore the presence of the parameter is completely optional, and will be used based upon a user's particular need.

4.2.4.2.3.6 A blank in the table is used to indicate that the specific parameter will not be used.

4.2.4.2.3.7 When there is no parameter in the table, that means that the primitive is relevant but does not contain any parameter.

4.2.4.3 Service Parameters

4.2.4.3.1 Throughout the service descriptions every parameter is described. In particular, there is a note that explains the purpose of the parameter, and a mandatory statement that states what values it may contain.

4.2.4.3.2 In many cases, the primitive parameters have the same contents than APDU fields in the ASN.1 description. The FIS-ASE is tasked to copy the parameter value within the APDU field without modification. In order to avoid confusion by defining twice identical data structures, 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.

4.2.4.3.3 For the other parameters, the syntax is described by enumerating the authorised abstract values

4.3 SARPs Section 2.4.4: FORMAL DEFINITION OF MESSAGES

4.3.1 Encoding/Decoding Rules

4.3.1.1 SARPs section 2.4.4.1 simply identifies the PDUs which the FIS systems must be able to encode and decode.

4.3.1.2 A FIS system is not required to be able to encode and decode all messages specified in the ASN.1 description. For instance, an air FIS system is not required to be able to encode FISUplinkAPDU nor to decode FISDownlinkAPDU.

4.3.2 ASN.1 Abstract Syntax

4.3.2.1 General

4.3.2.1.1 SARPs section 2.4.4.2 defines the abstract syntax of the protocol. That is it defines the structure of the PDUs that are to be sent between aircraft and the ground systems. It is written in a notation that is called ASN.1. It is strongly recommended that the reader is familiar with ASN.1 before attempting to understand the detail of this SARPs section.

4.3.2.1.2 Data types exchanged by FIS ASEs are described in the FIS 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.

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

4.3.2.2 ASN.1 Organization

4.3.2.2.1 The ASN.1 itself is organized into a number of different sections:

- The FIS messages – top level,
- The FIS messages – second level,
- The ATIS messages, and
- The ATIS fields.

4.3.2.2.2 The top level (which will typically be used as entry point in any ASN.1 compiler) consists of two main structures: the FISDownlinkAPDU is a choice of time-stamped PDUs generated by the aircraft, and FISUplinkAPDU is a choice of time-stamped PDUs generated by the ground system.

4.3.2.2.3 An index of the types defined in the ASN.1 is given in section 6 of this document.

4.3.2.3 ASN.1 Tags

4.3.2.3.1 Tags are used in ASN.1 to allow distinguishing 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 FIS 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 an 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 FIS service are made available, the extensibility ASN.1 feature (ellipse) has been used in each data type identifying the DFIS service.

4.3.2.4.2 For instance, extensions have been allowed in the following data types:

- DownlinkAPDU to allow the definition of new FIS downlink APDUs,
- UplinkAPDU to allow the definition of new FIS uplink APDUs,
- FISAbort to allow the definition of abort data for new FIS services,
- FISCancelAcceptData to allow the definition of cancel accept data for new FIS services, etc.

4.3.2.4.3 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 via INTEGER data type.

4.3.2.4.4 ASN.1 constraints are defined for INTEGER data types 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.

4.3.2.4.5 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
}
```

4.3.2.5 Time Representation

4.3.2.5.1 Data types have been specified for time indication (Date, DateTimeGroup, Year, Month, Hours, Minutes, and 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.

4.4 SARPs Section 2.4.5: PROTOCOL DEFINITION

4.4.1 Message Sequence Diagrams

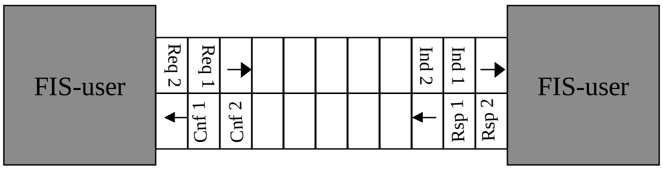
4.4.1.1 Time sequence diagrams or message sequence diagrams are used to denote the relationship between the primitives that form a FIS service and the order in which they occur.

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

4.4.1.3 Inherent to the service model is the notion of queuing. The FIS-service indications and confirmations are delivered to the FIS-users in the order that the corresponding FIS-service requests and responses were issued. One exception to the notion of queuing is the abortive services (FIS-user-abort and FIS-provider-abort services) which may overtake other primitives and empty the primitives in the queue.

4.4.1.4 Each fi re four
vertical lines that sep

- The FIS-ground-
- The FIS-ground-
- The dialogue ser gether,
  provide the dial(n strip
  down the middle
- The FIS-air-ASE
- The FIS-air-user



4.4.1.5 The m 'IS-air-
ASE) together form t

4.4.1.6 The ou ent the
invocation of one of of the

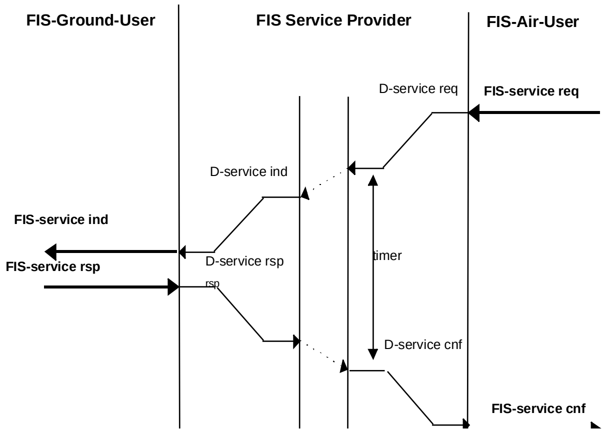


figure.

4.4.1.7 The inner two vertical lines represent the Dialogue service. Any lines crossing them represent the invocation of one of the Dialogue service primitives. The Dialogue service primitives are labelled in the FIS ASE part of the figure.

4.4.1.8 The diagrams represent a sequence of events. Time is always considered to run down the figure from the top (representing the earliest time) to the bottom (representing the latest time).

4.4.1.9 If the ASEs set timers, there are marked on the figures by vertical lines with arrows at both ends.

4.4.1.10 It should be noted that the last figures representing an abort situation can be overlaid on top of any of the other figures to represent an abort in action.

4.4.2 FIS Service Provider Timers

4.4.2.1 SARPs section 2.4.5.2 lists the technical timers that are defined in the protocol, and suggests values for them.

4.4.2.2 The purpose of the technical timers in the FIS service provider is not operational. For operational reasons, there may be a requirement to have other timers that are shorter than the ones described here. The purpose of these timers is only to ensure that the ASE protects itself when communicating with a system that has failed to respond for some reason.

4.4.2.3 The assignment of values for timers must be optimised based on operational testing of the application. In such testing, incompatible timer values and optimum combinations can be identified. Implementations of FIS 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.

4.4.3 FIS-ASE Protocol Description

4.4.3.1 The FIS 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 FIS-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 FIS-users and/or the underlying Dialogue Service provider).

4.4.3.2 The protocol description explains the rules by which the ASEs work. There is a detailed specification of actions taken by the ASEs when triggered by certain events:

- The arrival of a PDU through the dialogue service,
- The invocation of one of the service primitives by the user,
- The expiration of one of the internal timer, and
- The occurrence of an unrecoverable error.

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

4.4.3.4 Functional Model

4.4.3.4.1 In order to resolve a somewhat complex specification, both the ground and the air ASE have been broken up into a number of internal modules. The behavior of each is specified separately. In addition to the four triggers specified above, individual modules may also be triggered into action by the other modules within the same

ASE.

4.4.3.4.2 Both air and ground ASEs have modules that mirror each other. However, apart from the AB and LI modules, the actions of the air and ground mirrored modules are different but complementary. The modules are:

- HI module – the main job of the HI module is to select which module should handle the primitives that are invoked by the user.
- LI module – the main jobs of the LI module are a) to select which module handle the PDU passed to it frm the dialogue service, and b) to manage the dialogue, selecting which dialogue service should be used at any given time.
- DC module – the job of the DC module is to manage demand contracts,
- UC modules – the job of the UC module is to manage update contracts,
- CL module – the job of the CL module is t manage the cancellation of contracts, and
- AB module – the job of the AB module is to handle abort situations.

4.4.3.4.3 There is a requirement that the ASE does not accept the invocation of primitives when no actions are described or that primitive in that state (2.4.5.3.2). Some explanation of this statement is needed. Each module has several different states. When a module is in a particular state, only some primitives are permitted to invoke. For example, if a FIS-update-contract request is invoked, it is not sensible to invoke a second FIS-update-contract(request until a reply has been received for the first one. There is no statement in the description of the protocol that explains what the ground ASE should do if it receives a second FIS-update-contract request before the reply to the forst one hs been received. The SARPs therefore require (by statement 2.4.5.3.2), that the FIS-air-user must not invoke a FIS-update-contract request during this period.

4.4.3.4.4 Thus only actions which are permitted are described. If an action is not described, then it is not permitted.

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

4.4.4 State Tables

4.4.4.1 State tables are provided in SARPs section 2.4.5.5. These should be an exact reflection of the FIS protocol description, in a condensed form. The state tables are only presented for guidance, since the textual protocol description always takes precedence.

4.5 SARPs Section 2.4.6: COMMUNICATION REQUIREMENTS

4.5.1 Encoding Rules

4.5.1.1 SARPs section 2.4.6.1 states that PER (Packed Encoding Rules) must be used to encode the PDUs. PER is an ISO standard and is particularly efficient at encoding data. Implementers may use ASN.1 compilers to generate code that encode and decode PER automatically.

4.5.2 Dialogue Service Requirements

4.5.2.1 The dialogue service requires a number of parameters to operate. SARPs section 2.4.6.2 defines those parameters that are not defined elsewhere.

4.5.2.2 In addition to the *ClassOfCommunication* parameter provided by the contract initiator, 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 FIS(ATIS) application and therefore are not requested to the users.

4.5.2.3 The Application Priority is set by default to the abstract value "Aeronautical Information Service messages".

4.5.2.4 The Residual Error Rate is set by default to the abstract value "low".

4.6 SARPs Section 2.4.7: FIS USER REQUIREMENTS

4.6.1 Jane Justification of chapter 7. If there is restrriction in the SARPs compared with the operation requirements for CNS/AZTM-1, it is justify in this chapter.

4.6.2 one section per function (2 sections max.), reference section overview, operational issues raised in chapter 7.

4.7 SARPs Section 2.4.8: SUBSETTING RULES

4.7.1 General

4.7.1.1 There is some functionality within the FIS SARPs that implementers may choose not to incorporate. For example, if a particular implementation of an air system is designed always to use the demand contract, then there is no requirement for it to have the code for update contract implemented.

4.7.1.2 There are some combinations of functionality that are, therefore, sensible to build, and some that are not. SARPs section 2.4.8 defines the combinations of functionality that are sensible.

4.7.1.3 The options are defined in a set of tables.

- Version number – only one version is defined. This is a placeholder for when future versions are defined.
- Protocol Options – this defines a number of options for parts of the protocol that may be implemented. The options may be implemented together. Each has a name associated with it – the predicate.
- FIS-ground-ASE configurations – this defines two combinations of protocol options, each of which yields a coherent protocol.
- FIS-air-ASE configurations – this defines two combinations of protocol options, each of which yields a coherent protocol.
- Supported FIS service primitives – this defines the conditions under which the service primitives are applicable.
- Supported FIS APDUs – this defines the conditions under which the PDUs are applicable.

4.7.1.4 Functionality supported by an air-based FIS system is different from the functionality supported by a ground-based FIS system. The air-based FIS system acts as the originator of the FIS contracts whereas the ground-based FIS system is the provider of the FIS information.

4.7.1.5 Any aircraft can independently select a subset and be inter-operable with any ground system independently selecting a subset.

4.7.2 Mandatory Functionality

4.7.2.1 The FIS demand contract service is always supported by both air and ground systems. These functions constitute the core functionality of the FIS protocol. The FIS-demand-contract is therefore always provided in full to both air and ground users.

4.7.2.2 The FIS-report service is part of the core functionality of the FIS systems. It must be supported by any configuration of airborne and ground FIS systems.

4.7.3 Optional Functionality

4.7.3.1 The FIS update contract service is optionally supported by the FIS systems:

4.7.3.1.1 The FIS update contract is a facility provided to the air-users upon the choice of the implementers. When the FIS-update-contract service is not available to the pilots, they are not automatically warned of the update of the information they have received previously. They have therefore to request the update of the FIS information by invoking the FIS-demand-contract service several times. Another consequence is that the underlying dialogue is not maintained between the aircraft and the FIS ground system and each new FIS demand request causes the establishment of a new dialogue.

4.7.3.1.2 The ground FIS systems provide the FIS update contract service upon the choice of the FIS service provider. However, the ground system shall support a stub of the FIS update contract function to enable rejection of the request for an update contract.

4.7.3.1.3 In the case the update contract service is not provided by the ground FIS system, upon receipt of a FIS-update-contract request from an aircraft, the ground ASE forwards the request to the FIS-ground-user. The FIS-update-contract response primitive allows the FIS-ground-user to indicate that it can not provide updates of the FIS information and optionally to uplink a single FIS report.

4.7.3.2 The FIS-cancel-contracts service is optionally offered to the air-users:

4.7.3.2.1 Only when the update contract is supported, the FIS-cancel-update service is supported.

4.7.3.2.2 If this service is not available, the air users must cancel individually each FIS contract in sequence to get the same result. A FIS ground system – supporting the FIS update contract or not – is always able to receive a request of cancellation of all contracts.

5.

DIMENSIONS

5.1 PDU Size

5.1.1 Theoretical Limits

5.1.1.1 Theoretically, the FIS ground system could be requested to provide a FIS report with all possible types of information, including all fields of arrival and departure ATIS related to a given airport. The largest PDU that the ground system will be required to produce will thus depend on the maximum number of runways available and the level of use if the free text capability; Where the 36 arrival runways and 36 departure runways authorized by the ASN.1 is used and all free text fields are used at the maximum length (2047 octets), then the largest PDU that the ground system could generate is approximately 620 kilo-bytes. The ground system should be capable of managing a PDU of this size. It is unlikely that the air systems will be capable of managing such PDUs. The limiting factor with PDUs of this size is likely to be the bandwidth of the air-ground link. The size of other air and ground initiated PDUs are small by comparison, and all ground systems should be built with the capability to receive all requests from the aircraft.

5.1.2 Error Handling

5.1.2.1 Should either airborne or ground system receive a PDU that is too large for it to manage, it will be unable to decode it.

5.1.2.1.1 If it cannot decode it because it does not have the resources when the PDU arrives, the system should abort the connection under these circumstances (with abort reason set to "unrecoverable error").

5.1.2.1.2 If it cannot decode it because it exceeds the limits of the system, the PDU should be ignored.

5.2 Rate of Message Transmission

5.2.1 Limits

5.2.1.1 <TBD>

5.2.2 Error Handling

5.2.2.1 <TBD>

5.3 Number of FIS Connections

5.3.1 Limits

5.3.1.1 <TBD>

5.3.2 Error Handling

5.3.2.1 <TBD>

6.

ASN.1 INDEX

6.1 ASN.1 Section Index

6.1.1 Within the main body of the ASN.1, there are six sections divided up by comments crossing the page. There are four sections. These have been given a section number S1 - S4, and are listed in the table below. These section numbers are not in the SARPs themselves, but have been included here to make the reading of the table in the following section easier. They can be used to determine the location of the ASN.1 type definition within the SARPs.

Section Reference	*ASN.1 Section*
S1	FIS messages (top level)
S2	FIS messages (second level)
S3	ATIS messages
S4	ATIS fields

6.2 ASN.1 Type Index

6.2.1 The following table lists each ASN.1 type defined in the FIS SARPs in alphabetical order. In order to enable the definition to be found more easily, a cross reference to the ASN.1 section is given.

6.2.2 The third column lists those ASN.1 types that are used in the definition of the ASN.1 type in the first column, and the fourth column lists those ASN.1 types that use it. The third and fourth columns are therefore inverse references.

6.2.3 <Add range and resolution for leaf fields>

Type	*ASN.1 Section*	*Type used by this type*	*Types using this type*
Airport	S4	-	ATISReport ATISRequest
AltimeterSetting	S4	PressureMeasure RunwayQFE	CommonATISInformation
AltitudeFlightLevel	S4	-	Level
AltitudeFlightLevelMetric	S4	-	Level
Approach	S4	-	ApproachType
ApproachType	S4	Approach FreeText	SpecificATISArrivalInfo
ArrivalAndDepartureATIS	S4	ArrivalATIS	ATISInformation

		DepartureATIS	
ArrivalATIS	S4	ArrivalRunway ATISCode CommonATISInformation SpecificATISArrivalInfo Time	ArrivalAndDepartureATIS ATISInformation
ArrivalDepartureIndicator	S4	-	ATISRequest
ArrivalRunway	S4	Runway	ArrivalATIS RunwayType
ATISCode	S4	-	ArrivalATIS CombinedATIS DepartureATIS
ATISInformation	S3	ArrivalAndDeparture ArrivalATIS CombinedATIS DepartureATIS	ATISReport
ATISReport	S3	Airport ATISInformation	FISReportData
ATISRequest	S3	Airport ArrivalDepartureIndicator	FISRequestData
BrakingAction	S4	BrakingActionDescription	Runway
BrakingActionDescription	S4	BrakingActionQuality FreeText	BrakingAction
BrakingActionQuality	S4	-	BrakingActionDescription
CloudAmount	S4	-	CloudInformation CloudSkyCoverGroup
CloudHeight	S4	-	CloudInformation CloudSkyCoverGroup
CloudInformation	S4	CloudAmount CloudHeight CloudType	SignificantMetPhenomena
CloudSkyCoverGroup	S4	CloudAmount CloudHeight	CommonATISInformation
CloudType	S4	-	CloudInformation
ContractNumber	S4	-	FISAccept FISCancelUpdateAccept FISCancelUpdateContract FISReject FISReport FISRequest
ContractType	S4	-	FISRequest
CombinedATIS	S4	ATISCode CommonATISInformation RunwayType	ATISInformation

		SpecificATISArrivalInfo Time	
CommonATISInformation	S4	AltitudeSetting CloudSkyCoverGroup FreeText Level RVR SignificantMetInfo SurfaceWinds Temperature Time VisualVisibility	ArrivalATIS CombinedATIS DepartureATIS
Date	S4	Day Month Year	DateTimeGroup
DateTimeGroup	S4	Date HHMMSS	FISDownlinkAPDU FISUplinkAPDU
Day	S4	-	Date
DepartureATIS	S4	ATISCode CommonATISInformation DepartureRunway Time	ArrivalAndDepartureATIS ATISInformation
DepartureRunway	S4	Runway	DepartureATIS RunwayType
DownlinkAPDU	S1	FISAbort FISCancelContracts FISCancelUpdateAccept FISCancelUpdateContract FISRequest ...	FISDownlinkAPDU
FreeText	S4	-	ApproachType BrakingActionDescription CommonATISInformation Runway SpecificATISArrivalInfo SignificantMetPhenomena
FISAbort	S2	FISProtocolErrorDiag ...	DownlinkAPDU UplinkAPDU
FISAccept	S2	ContractNumber FISAcceptData	UplinkAPDU
FISAcceptData	S2	FISReportData	FISAccept
FISCancelAcceptData	S2	...	FISCancelUpdateAccept
FISCancelContracts	S2	FISServiceType	DownlinkAPDU
FISCancelContractsAccept	S2	FISServiceType	UplinkAPDU
FISCancelUpdateAccept	S2	ContractNumber FISCancelAcceptData	DownlinkAPDU UplinkAPDU

FISCancelUpdateContract	S2	ContractNumber FISCancelUpdateData	DownlinkAPDU UplinkAPDU
FISCancelUpdateData	S2	...	FISCancelUpdateContract
FISDownlinkAPDU	S1	DateTimeGrounp DownlinkAPDU	-
FISProtocolErrorDiag	S2	...	FISAbort
FISUplinkAPDU	S1	DateTimeGrounp UplinkAPDU	-
FISReject	S2	ContractNumber FISRejectData	UplinkAPDU
FISRejectData	S2	FISRejectReason FISReportData	FISReject
FISRejectReason	S2	...	FISRejectData
FISReport	S2	ContractNumber FISReportData	UplinkAPDU
FISReportData	S2	ATISReport ...	FISAcceptData FISRejectData FISReport
FISRequest	S2	ContractNumber ContractType FISRequestData	DownlinkAPDU
FISRequestData	S2	ATISRequest ...	FISRequest
FISServiceType	S2	...	FISCancelContracts FISCancelContractsAccept
HHMMSS	S4	TimeMinutes TimeHours TimeSeconds	DateTimeGroup
Level	S4	AltitudeFlightLevel AltitudeFlightLevelMetric	CommonATISInformation
MeasuredWindVariations	S4	WindDirection WindSpeed	WindVariations
Month	S4	-	Date
PressureMeasure	S4	-	AltimeterSetting RunwayQFE
Runway	S4	BrakingAction FreeText RunwayId	ArrivalRunway DepartureRunway RunwayType
RunwayId	S4	RunwayLetter RunwayNumber	Runway RunwayQFE RunwayVisibility
RunwayLetter	S4	-	RunwayId

RunwayNumber	S4	-	RunwayId
RunwayQFE	S4	PressureMeasure RunwayId	AltimeterSetting
RunwayType	S4	ArrivalRunway DepartureRunway Runway	CombinedATIS
RunwayVisibility	S4	RunwayId Visibility	RVR
RVR	S4	RunwayVisibility	CommonATISInformation
SignificantMetInfo	S4	SignificantMetPhenomena	CommonATISInformation
SignificantMetPhenomena	S4	CloudInformation FreeText	SignificantMetInfo
SpecificATISArrivalInfo	S4	ApproachType FreeText	ArrivalATIS CombinedATIS
SurfaceWinds	S4	Winds	CommonATISInformation
Temperature	S4	-	CommonATISInformation
Time	S4	TimeHours TimeMinutes	ArrivalATIS CombinedATIS CommonATISInformation DepartureATIS
TimeHours	S4	-	HHMMSS Time
TimeMinutes	S4	-	HHMMSS Time
TimeSeconds	S4	-	HHMMSS
UplinkAPDU	S1	FISAbort FISAccept FISCancelContractsAccept FISCancelUpdateAccept FISCancelUpdateContract FISReject FISReport ...	FISUplinkAPDU
VerticalVisibility	S4	Visibility	
Visibility	S4	-	RunwayVisibility VerticalVisibility
VisibilityDirection	S4	-	VisualVisibility
VisibilityNautMiles	S4	-	VisibilityValue
VisibilityValue	S4	VisibilityNautMiles	VisualVisibility
VisualVisibility	S4	VisibilityDirection VisibilityValue	CommonATISInformation

Winds	S4	WindDirection WindSpeed WindVariations	SurfaceWinds
WindDirection	S4	-	MeasuredWindVariations Winds
WindSpeed	S4	-	MeasuredWindVariations Winds WindVariations
WindVariations	S4	MeasuredWindVariations WindSpeed	Winds
Year		-	Date
... (extensibility marker)	-	-	DownlinkAPDU FISAbort FISCancelAcceptData FISCancelUpdateData FISProtocolErrorDiag FISRejectReason FISReportData FISRequestData FISServiceType UplinkAPDU

6.3 ASN.1 Glossary

6.3.1.1 Two sets of data types are defined in the FIS(ATIS) ASN.1 description. The first set (FIS level defined in ASN.1 sections S1 and S2) contains information common to all type of DFIS services. The second set (ATIS level defined in ASN.1 sections S3 and S4) is contains information related to a specific DFIS service, here the ATIS service.

6.3.1.2 FIS Protocol data

6.3.1.2.1 These data are used to co-ordinate the processing of remote FIS ASEs. They are Protocol Control Information which do not carry operational data but are used for protocol processing purpose. They are used:

1. to identify the type of the APDU transmitted on the line (e.g. FISDemandContract Downlink APDU or FISAbort Uplink APDU),
2. to identify the type of DFIS service the contents of the APDU is related to (e.g. ATIS, METAR, etc...), and
3. to identify the contract the APDU is related to.

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

ContractNumber Identifies the contract to which the APDU is related to.

DownlinkAPDU Contains the identification and the actual contents of the downlink FIS messages.

UplinkAPDU Contains the identification and the actual contents of the uplink FIS messages.

FISAbort Specifies the reason of an abort caused by the FIS ASE.

FISAccept Identifies the FIS contract being accepted and optionally a FIS report.

FISAcceptData Contains data sent back with a contract acceptation message.

FISCancelContracts Identifies the types of FIS contracts being requested to be cancelled.

FISCancelContractsAccept Identifies the types of FIS contracts the cancellation of which is accepted.

FISCancelAcceptData Contains data sent back with a contract cancellation message.

FISCancelUpdateAccept Identifies the FIS update contract being cancelled.

FISCancelUpdateContract Identifies the FIS update contract being requested to be cancelled.

FISCancelUpdateData Contains data sent back with the request of a contract cancellation.

FISDownlinkAPDU Contains any FIS downlink message with a time stamp.

FISProtocolErrorDiag Specifies the reason of the abort by either of the FIS-ASEs.

FISReject Identifies the FIS contract being rejected with a reason for rejection.

FISRejectData Contains data sent back with a contract rejection.

FISRejectReason Identifies the reason of the contract rejection.

FISReport Identifies the FIS contract and contains the fields of the FIS report.

FISReportData Contains the fields of a FIS report.

FISRequest Identifies the FIS contract being established and contains the parameters of the contract.

FISRequestData Contains the parameters of the FIS contract being established.

FISServiceType Identifies the type of a FIS contract.

FISUplinkAPDU Contains any uplink FIS message with a time stamp.

6.3.1.3 ATIS Protocol data

6.3.1.3.1 These data are used to co-ordinate the processing between FIS ASEs providing the ATIS service. They are used to convey the operational information likely to be generated by ATIS users.

6.3.1.3.2 Some types are common to several data link applications. However, even if the semantic is the same, the range and resolution required by each application may be specific. Therefore it has been decided to specify stand-alone ASN.1 descriptions completely independent from the others.

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

Airport Contains four characters that specifies the ICAO four-letter identifier of the airport.

AltimeterSetting Gives QNH of the aerodrome and the runway QFE when available (in hectopascal or inches of mercury)

AltitudeFlightLevel Specifies the altitude expressed in feet flight level.

AltitudeFlightLevelMetric Specifies the altitude expressed in meter flight level.

Approach Specifies standardised approach type.

ApproachType Provides the approach type as a standardised value or a free text .

ArrivalAndDepartureATIS Provides both arrival and departure ATIS fields.

ArrivalDepartureIndicator Indicates which type of ATIS information is provided (i.e. arrival, departure or both).

ArrivalATIS Provides arrival ATIS fields.

ArrivalRunway Specifies an arrival runway.

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

ATISInformation Contains the fields of an arrival, departure, combined or both arrival and departure ATIS.

ATISReport Provides the airport identifier and the ATIS related to it.

ATISRequest Identifies the airport and the type of ATIS requested.

BrakingAction Provides braking action for both first half and second half of the runway in case of contaminated runway conditions.

BrakingAction Description Specifies the braking action as a standardised value or a free text.

BrakingActionQuality Specifies standardised braking action: poor, fair, good or nil.

CloundAmount Specifies cloud amount: sky clear, scattered, broken or overcast.

CloudHeight Specifies cloud height expressed either in meters or feet.

CloudInformation Provides cloud information: amount, height and type.

CloudSkyCoverGroup Provides up to 3 levels of cloud coverage: amount and height.

CloudType Specifies the cloud type: cumulonimbus or towering cumulus.

ContractType Specifies the FIS contract type: demand or update.

CombinedATIS Provides the field of a combined ATIS

CommonATISInformation Provides the ATIS fields common to both departure and arrival.

Date Provides date expressed in year, month and day.

DateTimeGroup Provides date and time expressed in hours, minutes and seconds.

Day Specifies the day of the month.

DepartureATIS Provides the fields of a departure ATIS.

DepartureRunway Identifies a departure runway.

FreeText Contains an ASCII string up to 2047 characters.

HHMMSS Provides the time in hours, minutes and seconds.

Level Provides the level in feet or metric level.

MeasuredWindVariations Provides the direction and the speed of the winds.

Month Specifies the month of the year.

PressureMeasure Provides the pressure expressed in hecto pascal or inches of mercury.

RunwayId Provides the number and the letter attached to a runway.

Runway Provides the surface conditions, the braking action and the arresting system of a runway.

RynwayLetter Identifies the runway expressed as left, centre or right when there is more than one.

RunwayNumber Identifies the runway.

RunwayQFE Provides the qFE measured for a runway.

RunwayType Identifies an arrival, departure and combined runway.

RunwayVisibility Provides the RVRs of a runway:

RVR Provides the RVRs of several runways.

SpecificATISArrival Info Contains ATIS fields specific to the arrival ATIS.

SignificantMetInfo Provides significant meteorological phenomena when available.

SignificantMet Phenomena Gives significant meteorological phenomena occurring in the approach, take-off and climb-out areas.

SurfaceWinds Indicates surface wind direction (in degrees) and speed (in km/hour or knots) from the aerodrome reference point.

Time Provides the time in hours and minutes.

TimeHours Specifies time in hours of a day.

TimeMinutes Specifies time in minutes of an hour.

TimeSeconds Specifies time in seconds of a minute.

Temperature Specifies temperature expressed in either degrees Celsius or degrees Fahrenheit.

VerticalVisibility Specifies the visibility when possible.

Visibility Indicate the ability, as determined by atmospheric conditions and expressed in units of distance (in km, m, nautical miles or 1/16 of nautical miles) to see and identify prominent unlighted objects by day and prominent objects by night.

VisibilityDirection Specifies a direction: North, Northwest, etc.

VisibilityNautMiles Specifies a distance in nautical miles or 1/16 of nautical miles.

VisibilityValue Specifies a distance in meters or kilometres.

VisualVisibility Specifies the visual visibility in terms of distance and direction.

Winds Provides winds characteristics: wind direction, wind variation, wind speed and gust speed.

WindDirection Specifies the direction of the wind expressed in degrees.

WindSpeed Specifies the speed of the wind expressed in kilometre per hour mile per hour or knots.

WindVariations Specifies the wind variations expressed as direction and speed or speed only.

Year Specifies the year between 1996 and 2095.

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 some of the scenarios that are theoretically possible using FIS. 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.
2. Due the potential for last minute weather changes, the pilot initiates a request for an FIS(ATIS) Update Contract for the departure airport. The ground FIS system accepts the contract and sends the current ATIS to the aircraft.
3. An indication alerts the flight crew to an updated ATIS. This ATIS indicates that the forecasted weather has arrived in the airport environment, and warns the crew of the approaching thunderstorms.
4. The rain has arrived at the airport now, and the flight crew receives another update from the FIS(ATIS) contract. Although conditions are deteriorating, the RVR is acceptable for departure.

7.2.3 After take-off

1. The aircraft is now leaving the TMA environment and the pilot elects to cancel the FIS(ATIS) contract for the departure airport.
2. The pilot initiate the CM-Logon process with the ground CM covering the destination airport.
3. The pilot initiates a FIS(ATIS) Demand Contract for the destination airport. He informs the passengers about the weather conditions at the arrival airport.
4. Times to times, the pilot initiates a FIS(ATIS) Demand Contract for the arrival airport.
5. One hour before arrival time, the pilots establishes a FIS(ATIS) Update Contract for the destination airport. The ATIS information indicates that conditions at the airport are conducive to an approach to the alternate arrival runway.

7.2.4 After landing

1. The pilot requests the cancellation of all FIS contracts still pending.

8.

EXAMPLE ENCODING

8.1 Encoding/Decoding Rules

8.1.1 Samples of PER encoded FIS 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.

8.1.2 <TBD: bit-oriented presentation>

8.1.3 FIS Downlink APDU [FISRequest]

```
********************************************************************************
*                             FIS Request APDU                                 *
********************************************************************************
FISDownlinkAPDU SEQUENCE COMPLETE
 {
    time SEQUENCE
     {
        date SEQUENCE
         {
            year INTEGER  = 1996
            month INTEGER  = 12
            day INTEGER  = 12
         }
        time SEQUENCE
         {
            timeHours INTEGER  = 11
            timeMinutes INTEGER  = 29
            timeSeconds INTEGER  = 0
         }
     }
    fisDownlinkAPDU CHOICE
     {
        fISRequest SEQUENCE
         {
            contractNumber INTEGER = 1
            contractType ENUMERATED VAL_DEF = demandContract(0)
            fISRequestdata CHOICE
             {
                aTISRequest SEQUENCE
                 {
                    airportID OCTET STRING = 0x4C46424F
                    arrivalDepartureIndicator ENUMERATED VAL_DEF = arrival(0)
                 }
             }
         }
     }
 }


********************************************************************************
*                                 PER STREAM                                   *
*                                ( 10 octets )                                 *
********************************************************************************
0000: 01 6b 5b a0 04 00 cc 8d 0a 78                   .k[......x
```

8.1.4

FIS Downlink APDU [FISAbort]

```
********************************************************************************
*                            FIS Downlink ABORT APDU                           *
********************************************************************************
FISDownlinkAPDU SEQUENCE COMPLETE
 {
    time SEQUENCE
     {
        date SEQUENCE
         {
            year INTEGER  = 1996
            month INTEGER  = 12
            day INTEGER  = 12
         }
        time SEQUENCE
         {
            timeHours INTEGER  = 11
            timeMinutes INTEGER  = 29
            timeSeconds INTEGER  = 0
         }
     }
    fisDownlinkAPDU CHOICE
     {
        fISAbort CHOICE
         {
            atis ENUMERATED = protocolError(1)
         }
     }
 }


********************************************************************************
*                                 PER STREAM                                   *
*                               ( 6 octets )                                   *
********************************************************************************
0000: 01 6b 5b a0 20 40                               .k[. @
```

8.1.5

FIS Uplink APDU [FISAccept(positive-acknowledgement)]

```
********************************************************************************
*                    FIS Accept - Positive Acknowledgement                     *
********************************************************************************
FISUplinkAPDU SEQUENCE COMPLETE
 {
    time SEQUENCE
     {
        date SEQUENCE
         {
            year INTEGER  = 1996
            month INTEGER  = 12
            day INTEGER  = 12
         }
        time SEQUENCE
         {
            timeHours INTEGER  = 11
            timeMinutes INTEGER  = 29
            timeSeconds INTEGER  = 0
         }
     }
    fisUplinkAPDU CHOICE
     {
        fISAccept SEQUENCE
         {
            contractNumber INTEGER = 1
            fISAcceptData CHOICE
             {
                positiveAcknowledgement NULL = nil
             }
         }
     }
 }

********************************************************************************
*                                 PER STREAM                                   *
*                                ( 6 octets )                                  *
********************************************************************************
0000: 01 6b 5b a0 00 04                               .k[...
```

8.1.6

FIS Uplink APDU [FISReport]

```
********************************************************************************
*                             FIS Uplink Report APDU                           *
********************************************************************************
FISUplinkAPDU SEQUENCE COMPLETE
 {
    time SEQUENCE
     {
        date SEQUENCE
         {
            year INTEGER  = 1996
            month INTEGER  = 12
            day INTEGER  = 12
         }
        time SEQUENCE
         {
            timeHours INTEGER  = 11
            timeMinutes INTEGER  = 29
            timeSeconds INTEGER  = 0
         }
     }
    fisUplinkAPDU CHOICE
     {
        fISReport SEQUENCE
         {
            contractNumber INTEGER = 1
            fISReportData CHOICE
             {
                atis SEQUENCE
                 {
                    airportId OCTET STRING = 0x4C46424F
                    aTISInformation CHOICE
                     {
                        arrivalATIS SEQUENCE
                         {
                            aTISCode OCTET STRING = 0x41
                            aTISTimeofObservation SEQUENCE OPT
                             {
                                timeHours INTEGER = 14
                                timeMinutes INTEGER = 13
                             }
                            arrivalRunwaysinUse SEQUENCE OF  SEQUENCE (1)
                             {
                                Runway SEQUENCE
                                 {
                                    runwayId SEQUENCE
                                     {
                                        runwayNumber INTEGER = 3
                                        runwayLetter ENUMERATED OPT = runwayleft(1)
                                     }
                                    brakingAction SEQUENCE OPT
                                     {
                                        brakingActionFirstHalf CHOICE
                                         {
                                            brakingActionQuality ENUMERATED = good(2)
                                         }
                                        brakingActionSecondHalf CHOICE OPT
                                         {
                                            brakingActionQuality ENUMERATED = poor(0)
                                         }
                                     }
                                 }
                             }
                            commonATISInfo SEQUENCE
                             {
                                surfaceWinds CHOICE
                                 {
                                    calmIndicator NULL = nil
                                 }
                                visibility SEQUENCE
                                 {
                                    value CHOICE
                                     {
                                        meters INTEGER = 30
                                     }
                                 }
                                cloudSkyCoverGroup SEQUENCE OF  SEQUENCE (1)
```

```
                                 {
                                    -------- SEQUENCE
                                     {
                                        cloudAmount ENUMERATED = skyclear(0)
                                        cloudHeight CHOICE
                                         {
                                            cloudHeightMeters INTEGER = 1000
                                         }
                                     }
                                 }
                                airTemperature CHOICE
                                 {
                                    temperatureC INTEGER = 10
                                 }
                                dewPointTemperature CHOICE
                                 {
                                    temperatureC INTEGER = 0
                                 }
                                altimeterSetting SEQUENCE
                                 {
                                    qNH CHOICE
                                     {
                                        hPa INTEGER = 10000
                                     }
                                 }
                                significantMetInformation CHOICE
                                 {
                                    nosig NULL = nil
                                 }
                                transitionLevel CHOICE OPT
                                 {
                                    altitudeFlightLevel INTEGER = 350
                                 }
                             }
                            arrivalATISInfo SEQUENCE
                             {
                                approachType CHOICE OPT
                                 {
                                    approachType ENUMERATED = ils(0)
                                 }
                             }
                         }
                     }
                 }
             }
         }
     }
 }

********************************************************************************
*                                 PER STREAM                                   *
*                               ( 26 octets )                                  *
********************************************************************************
0000: 01 6b 5b a0 10 02 64 68 53 cc 17 1a 02 84 68 02 .k[...dhS.....h.
0010: 2f 01 f4 16 8a 02 71 05 02 00                   /.....q...
```