The Consultative Committee for Space Data Systems

Draft Report Concerning
Space Data System Standards

SPACECRAFT ONBOARD INTERFACE SERVICES

DRAFT INFORMATIONAL REPORT

CCSDS 850.0-G-0b

DRAFT GREEN BOOK
November 2006

AUTHORITY

Issue:	Draft Green Book, Issue 0b
Date:	November 2006
Location:	Not Applicable

(WHEN THIS INFORMATIONAL REPORT IS FINALISED, IT WILL CONTAIN THE FOLLOWING STATEMENT OF AUTHORITY:)

This document has been approved for publication by the Management Council of the Consultative Committee for Space Data Systems (CCSDS) and reflects the consensus of technical panel experts from CCSDS Member Agencies. The procedure for review and authorisation of CCSDS Reports is detailed in the *Procedures Manual for the Consultative Committee for Space Data Systems*.

This document is published and maintained by:

CCSDS Secretariat
Office of Space Communication (Code M-3)
National Aeronautics and Space Administration
Washington, DC 20546, USA

FOREWORD

This document is a CCSDS Informational Report to assist readers in understanding the Spacecraft Onboard Interface Services (SOIS) documentation. It has been prepared by the Consultative Committee for Space Data Systems (CCSDS). The concepts described herein are the baseline concepts for the CCSDS standardisation activities in respect of communication services and generic support services to be used in the flight segment of spacecraft systems.

This Report describes the challenges posed by spacecraft onboard interfaces, details the service architecture of the SOIS services, and elaborates on the goals and expected benefits of the of key SOIS services. It is intended to serve as a reference for both service users and service implementers in order to maximise the potential of standardised onboard interfaces with respect to re-use, interoperability, and inter-agency cross support.

Through the process of normal evolution, it is expected that expansion, deletion or modification to this document may occur. This Report is therefore subject to CCSDS document management and change control procedures that are defined in reference [1]. Current versions of CCSDS documents are maintained at the CCSDS Web site:

http://www.ccsds.org/

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

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

Member Agencies

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

Observer Agencies

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

DOCUMENT CONTROL

Document	Title	Date	Status
CCSDS 850.0-G-0b	Spacecraft Onboard Interface Services, Draft Informational Report, Issue 0b	November 2006	Current draft

CONTENTS

Section Page

1 INTRODUCTION 1-1

1.1 PURPOSE 1-1
1.2 SCOPE 1-1
1.3 APPLICABILITY 1-1
1.4 RATIONALE 1-2
1.5 APPROACH 1-4
1.6 TERMS AND DEFINITIONS 1-4
1.7 REFERENCES 1-5

2 SOIS CONCEPTS AND ARCHITECTURE 2-1

2.1 CONCEPTS 2-1
2.2 ARCHITECTURAL CONSIDERATIONS 2-3
2.3 SOIS SERVICE AND PROTOCOL ARCHITECTURE 2-3
2.4 SOIS NAMING AND ADDRESSING 2-7
2.5 MANAGEMENT CONCEPTS 2-7
2.6 SOIS COMPLIANCE 2-8
2.7 EXAMPLES 2-8
2.8 APPLICATION SUPPORT/SUBNETWORK LAYER INTERACTION 2-11

3 SOIS APPLICATION SUPPORT LAYER SERVICES 3-1

3.1 INTRODUCTION 3-1
3.2 COMMAND AND DATA ACQUISITION SERVICES 3-2
3.3 TIME ACCESS SERVICE 3-5
3.4 MESSAGE TRANSFER SERVICE 3-7
3.5 FILE SERVICES 3-8
3.6 DEVICE ENUMERATION 3-12

4 SOIS SUBNETWORK SERVICES 4-1

4.1 INTRODUCTION 4-1
4.2 DATA LINK LAYER FUNCTIONS 4-3
4.3 SOIS SUBNETWORK SERVICE DESCRIPTIONS 4-5

ANNEX A ACRONYMS AND ABBREVIATIONS A-1

CONTENTS (continued)

Figure Page

2-1 SOIS Architecture 2-2
2-2 SOIS Data Link Functions 2-6
2-3 Symmetrical Communication 2-9
2-4 Asymmetrical Communication 2-9
2-5 Simple Asymmetric Communication 2-10
2-6 Generic Application Support Layer Service 2-11
2-7 MTS over SOIS Packet Service and SpaceWire 2-12
2-8 DAS over SOIS Memory Access Service, RMAP and SpaceWire 2-13
2-9 Example of Application Support Layer Services Multiplexed onto Subnetwork 2-14
3-1 SOIS Application Support Layer Services 3-1
3-2 Time Access Services 3-5
3-3 File Services Overview 3-8
4-1 SOIS Subnetwork Decomposition 4-1

1 INTRODUCTION

1.1 PURPOSE

The purpose of this document is to describe the concept and supporting rationale for the Spacecraft Onboard Interface Services (SOIS) developed by the Consultative Committee for Space Data Systems (CCSDS). This document:

– provides an introduction and overview of the SOIS services concept upon which the detailed CCSDS SOIS recommendations are based;

– summarises the specific individual service recommendations and supplies the supporting rationale.

This document is a CCSDS Informational Report and is therefore not to be taken as a CCSDS Recommended Standard.

1.2 SCOPE

This document:

– describes the rational and approach of CCSDS SOIS standardisation,

– establishes the SOIS concepts and architecture (including the addressing strategy),

– provides an overview of the SOIS services and protocols,

– provides examples of the deployment of SOIS services and protocols.

The basic context of SOIS services is that of a single spacecraft within a single mission. Communications between elements outside of a single spacecraft, and communicating between multiple spacecraft falls outside the scope of SOIS. However, other CCSDS services exist that fulfil these external interfacing requirements, and the SOIS services are designed to be compatible with these.

A grey area exists in the case of application of wireless local area networking in swarms of spacecraft or proximal landed elements. An emerging activity within CCSDS will examine this area, and the outcome may or may not fall within the SOIS purview.

1.3 APPLICABILITY

The SOIS standardised services are intended to be applicable to all classes of missions, including scientific and commercial spacecraft, and manned and un-manned systems.

1.4 RATIONALE

CCSDS has enjoyed a great deal of success in standardisation of interfaces between spacecraft and ground systems and has managed to extend this success to areas such as lander to orbiter interfaces. Although CCSDS's authority derives from the requirement for interoperability between national space agencies, the primary benefit has been in cost and risk reduction internal to the agencies and to the individual missions. This manifests itself in:

- reuse of mission hardware and software;
- ready availability of space qualified components and subsystems;
- accumulated knowledge base within the agencies;
- reuse of standard EGSE;
- extensive validation of the operation and completeness of the standards.

In general, spacecraft interface development is based on unique designs which are specified and implemented on a project by project basis. Any reuse of these interfaces is usually a by-product of reuse of the whole spacecraft bus, with data handling interfaces having no self-sustaining level of reuse. While individual developers may have limited proprietary standards, these are generally closed and require significant adaptation across missions, particularly those involving inter-organisational cross-support.

Similarly, there exists little interface standardisation which can be used by individual equipment and instrument providers. While it is true that there are a limited number of physical interfaces applicable for use in the space environment, the services and access to these interfaces vary considerably between implementations.

At the international level there have so far been very few significant attempts at standardizing spacecraft onboard interfaces, and consequently incompatible interfacing solutions have evolved. Typically, the interfacing solutions that have been developed for spacecraft are rooted in 1960's technologies and bear very little resemblance to the 'plug-and-play' interfaces used to integrate computing devices in modern terrestrial systems.

The result is that a multitude of solutions are in place, with each mission either inheriting past solutions or developing new ones. However, an increase in the number and complexity of international missions and the cost of developing state-of-the-art high-speed data interfaces has led to significant impetus for pushing missions in the direction of using standards within and across programs.

CCSDS is perfectly placed to develop standards for agency adoption because:

- it can call on a multi-agency expertise base;
- it can offer global cost and risk reduction by nurturing suppliers on an international basis;
- it has the influence at mission and agency level to promote standards adoption.

Within CCSDS, the SOIS area has been charged with addressing the issue. Its solution lies in the development of a suite of open recommendations involving the complete spacecraft. The goal of the CCSDS SOIS standardisation activity is therefore to develop standards that will improve both the process of spacecraft development and integration as well as the quality of the finished product, and at the same time facilitate the adoption of promising new hardware and software technologies supporting international onboard interface interoperability.

The SOIS approach is to standardise the interfaces between items of spacecraft equipment by specifying well-defined standard service interfaces and protocols which allow standardised access to sensors, actuators, and generic spacecraft functions, allowing spacecraft applications to be developed independently of the mechanisms that provide these services. Applications are thus insulated from the specifics of a particular spacecraft implementation and may be reused across different spacecraft platforms with little regard of implementation details.

Service interface standardisation allows hardware interfaces to be accessed by flight software such that core spacecraft software may be reused on different underlying communications infrastructures with little or no change. The standard services could be implemented using a standard Application Programming Interface (API) that would enable portability and re-use of application software, and of service implementations.

The definition of the services makes no assumption about the implementation of the services in hardware or software or a mixture of both. In addition, SOIS aims to promote interoperability between software and hardware devices operating on various spacecraft communication buses. There are several benefits of this approach:

– as long as the subnetwork services remain stable, software and hardware may evolve independently;
– developers of core spacecraft software can rely on a standard set of services on which to base their design;
– requirements definition activities are reduced as direct reference may be made to the CCSDS Recommended Standards;
– a standard test suite may be used during qualification;
– costs are reduced by adhering to a single solution;
– risk is reduced through amortisation of development and testing across mission cost and time bases;
– subsystem and payload portability across missions is enabled;
– the possibility for reuse of both interfaces and core spacecraft software and the scope for further standardisation activities is significantly increased.

1.5 APPROACH

The process for SOIS standardisation is progressively:

– to identify and articulate a standard set of services which application software or higher-layer services can use to communicate between onboard components over a single Data Link;

– to provide standard mappings between service provision and various underlying Data Link communications media, recognizing that implementation of services is link-dependent;

– to provide a framework to allow various qualities of service to be supported over any underlying Data Link;

– finally, to develop protocols in support of the various SOIS services.

1.6 TERMS AND DEFINITIONS

With respect to service and protocol definition, SOIS, in general, uses terms and definitions defined within the ISO Open Systems Interconnect model defined in reference [2]. The following definitions are provided:

Best Effort—No guarantee of packet delivery or order of delivery of packets.

Heterogeneous network—A network that uses one or more underlying communications protocols, e.g., part SpaceWire and part Mil-Std 1553.

Packet—Delimited octet aligned data unit.

Priority—Identification of the transmit precedence of an SDU relative to other SDUs.

Protocol data unit (PDU)—A unit of data specified in a protocol and consisting of protocol-control-information and possibly user data.

Quality of Service (QoS)—The ability of a communication system to provide predictable and differentiated services. Quality of Service for a communication service may be characterised in terms of important features relevant to that communications service, for example: Reliability, Transmission rate, Effective Bandwidth and latency, Error rate.

Reliability—A QoS parameter indicating whether or not a Data Link function will attempt to acknowledge the successful receipt of a packet and possibly retry sending a PDU if no acknowledge is received by the sender.

Service Data Unit (SDU)—A unit of data passed into or out of a service interface.

Time Critical—Necessity to deliver a packet within a certain period of time or treat preferentially with respect to other packets.

Traffic Class—A traffic class is a category of traffic on a subnetwork distinguished by its QoS.

1.7 REFERENCES

The following documents are referenced in the text of this Report. At the time of publication, the editions indicated were valid. All documents are subject to revision, and users of this Report are encouraged to investigate the possibility of applying the most recent editions of the documents indicated below. The CCSDS Secretariat maintains a register of currently valid CCSDS Recommendations.

[1] *Procedures Manual for the Consultative Committee for Space Data Systems*. CCSDS A00.0-Y-9. Yellow Book. Issue 9. Washington, D.C.: CCSDS, November 2003.

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

[3] *Space Communications Protocol Specification (SCPS)—Network Protocol (SCPS-NP)*. Recommendation for Space Data System Standards, CCSDS 713.0-B-1. Blue Book. Issue 1. Washington, D.C.: CCSDS, May 1999.

[4] *Space Packet Protocol*. Recommendation for Space Data System Standards, CCSDS 133.0-B-1. Blue Book. Issue 1. Washington, D.C.: CCSDS, September 2003.

[5] J. Postel. *Internet Protocol*. STD 5, September 1981. [RFC 791, RFC 950, RFC 919, RFC 922, RFC 792, RFC 1112][1]

[6] J. Postel. *Transmission Control Protocol*. STD 7, September 1981. [RFC 793]

[7] SOIS Quality of Service Green Book [To Be Supplied].

[8] *CCSDS File Delivery Protocol (CFDP)*. Recommendation for Space Data System Standards, CCSDS 727.0-B-3. Blue Book. Issue 3. Washington, D.C.: CCSDS, June 2005.

[1] Internet Request for Comments (RFC) texts are available on line in various locations (e.g., http://ietf.org/rfc/); Internet standards are made up of one or more RFCs, which are identified in square brackets following the entry.

2 SOIS CONCEPTS AND ARCHITECTURE

2.1 CONCEPTS

On any given spacecraft, several interface types may be used between specific entities. The actual type of interface used is determined by the required characteristics of the interaction between those entities. These may typically be categorised as:

– **Multidrop Buses** providing connection to a central bus master and a number of slaves. Communication is generally asymmetrical and often involves low-level read and write access to slaves. The central control of bus traffic results in a highly stochastic traffic profile well suited to applications requiring bounded communications timing.

– **Point-to point serial interfaces** used for instrument connection, possibly for bulk data transfer but also combined with instrument control. Again, these interfaces usually operate in a master/slave mode.

– **LANs** used on larger infrastructures where hosts have generally equal computing power and have a diversity of communication requirements. Communication is on a peer-to-peer basis with a level of variability in delay due to resource queuing.

– **Point-to-point sensor and actuator interfaces** used for gathering sensor readings or controlling spacecraft equipment.

Onboard applications should not be concerned with these interfaces, and so the SOIS concept aims to provide a solution by recommending that applications interact only with a well-defined set of standard onboard data services. In addition, given the disparity of functionality supported by different SOIS Data Link layers, the SOIS subnetwork services provide a common interface and a convergence of common services to the upper-layer applications for communicating over any single SOIS link. Together, the application and subnetwork services provide a standard means to communicate between virtually all spacecraft components.

Figure 2-1 shows a layered view of the recommended services and their associated access points. **User Applications** are the mission-dependent applications that make use of the SOIS defined services. The **Transfer Layer** provides Transport- and Network-layer services based on existing protocols either defined or adopted by CCSDS (e.g., IP, SCPS, Space Packet Protocol). In many cases the Transfer Layer will not be required and is therefore considered optional. The **Subnetwork Layer** provides access to the Data Link medium and provides a set of SOIS-defined services over the subnetwork defined by that medium.

Network management services and plug-and-play services, while part of the architecture, are for future development and will only be discussed briefly in this document. Network management aspects are implemented for all services and functions and will be accessed by a variety of methods. SOIS makes no recommendations concerning these access methods. However, it is incumbent on SOIS to detail the Management Information Bases (MIBs) for all of its recommendations. The plug-and-play services make use of the device discovery service in the subnetwork and the device enumeration application service.

Details of the services, the architecture, and associated protocols are given in subsequent subsections.

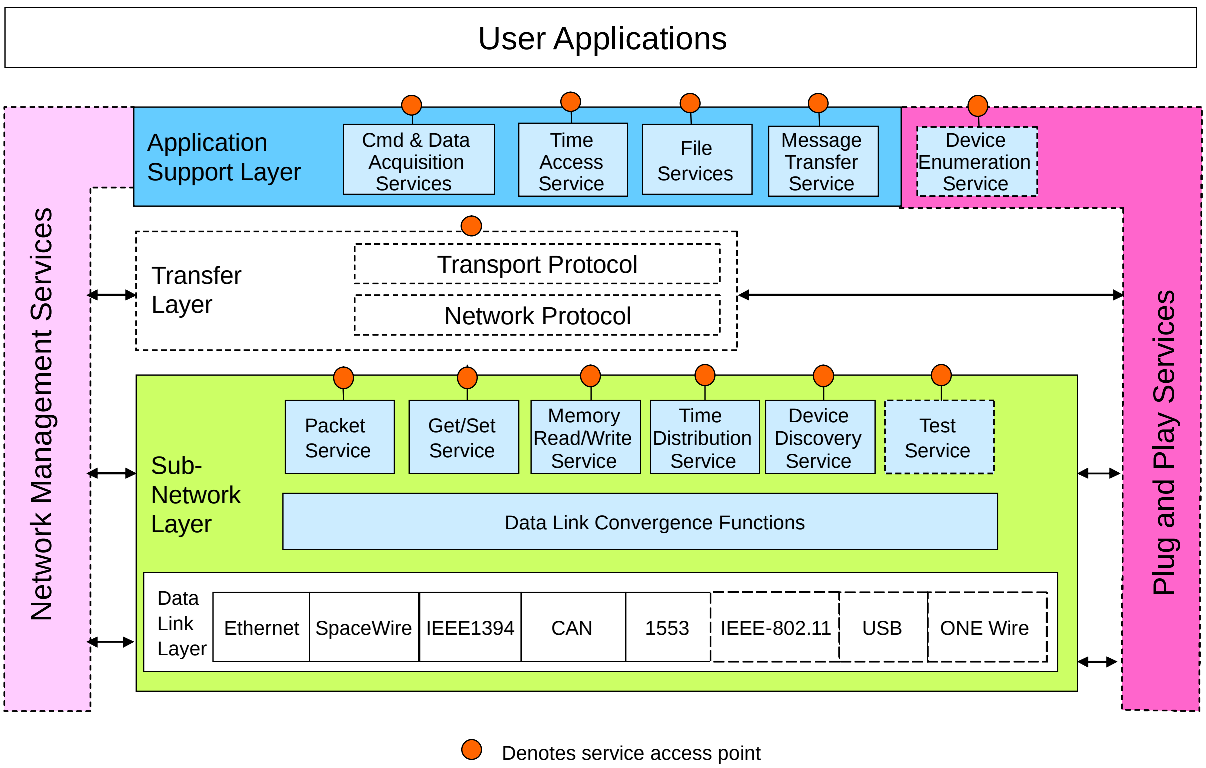

Figure 2-1: SOIS Architecture

Note that all service access points, whether they be at Subnetwork, Transfer, or Application Support Layer, are accessible by user applications.

2.2 ARCHITECTURAL CONSIDERATIONS

In arriving at the overall SOIS architecture a number of fundamental issues must be tackled:

– Spacecraft do not uniformly use a single underlying data communications medium; instead, a single spacecraft will most likely implement one or more busses and point to point links.

– The choice of underlying data communications media although limited, will vary across spacecraft dependent on mission needs.

– Communicating devices are often unbalanced in their capabilities. Typically, a spacecraft design will use a limited number of processors having sufficient resources to implement a full protocol stack but the communications will also require access to sensors and actuators that have little or no computing capabilities.

2.3 SOIS SERVICE AND PROTOCOL ARCHITECTURE

2.3.1 GENERAL

SOIS services, as shown earlier, exist at three service interfaces:

– an Application Support Layer service interface;

– a Transfer Layer Service interface;

– a Subnetwork Layer service interface.

2.3.2 APPLICATION SUPPORT LAYER SERVICES

The Application Support Layer services provide a number of capabilities commonly required onboard a spacecraft, which need not be limited to communications. The Application Support Layer services make use of the Subnetwork Layer services either locally or remotely over a network. The services are defined in terms of protocols, procedures, protocol data units and a Management Information Base (MIB). The Application Support Layer services identified are:

– Command and Data Acquisition—typically used to access spacecraft hardware devices such as sensors and actuators;

– Time Access—providing access to a local time source;

– Message Transfer—providing application-to-application message exchange;

– File Services—providing access to the spacecraft storage system;

– Device Enumeration—providing support for dynamic spacecraft configuration.

SOIS Application Support Layer services may be provided across a spacecraft network consisting of a number of heterogeneous or homogeneous SOIS subnetworks in conjunction with a Network layer such as the Space Packet or IP protocol. In cases where time criticality does not allow this approach, the SOIS Application Support Layer services may directly use SOIS Subnetwork Layer services over a single subnetwork.

Application Support Layer services are more fully addressed in section 3 of this document.

2.3.3 TRANSFER LAYER SERVICES

At present, the Transfer Layer is assumed to be composed of extant CCSDS recognised protocols and services. Examples of these are:

– TCP/UDP/IP;

– SCPS NP/TP;

– Space Packet Protocol.

2.3.4 SUBNETWORK SERVICES

The SOIS Subnetwork provides a set of SOIS-defined services which support upper-layer Application-Support and Transfer-layer entities. The subnetwork services which are provided are independent of the underlying Data Link in that the service primitives and associated parameters are the same regardless of the underlying Data Link. However, any given instance of a SOIS subnetwork service is associated with a single underlying Data Link. Data Link selection is achieved by selection of a service instance by the entity making use of subnetwork services.

The services provided by the underlying Data Link need to be matched to those required by the subnetwork service, and this may require the provision of convergence functions. Convergence functions add the necessary functionality to that inherently provided by the Data Link. The work needed to map a particular set of services to a Data Link depends on the services provided by the underlying Data Link. Where a Data Link already provides a SOIS Data Link service, it will be used to the maximum extent practical. In some cases the inherent Data Link service will be equivalent to that required by the subnetwork, in which case no convergence function will be required. In others, the inherent Data Link service will not match that required by the subnetwork, in which case a convergence function will be required.

The services identified at the Subnetwork Layer are:

- Get/Set Parameter—providing low-level access to sensors and actuators;
- Memory Access (memory location read/write, includes block move)—providing direct access to device memory;
- Time Distribution—providing transmission and reception of spacecraft time;
- Packet—providing packet delivery over a single subnetwork;
- Device Discovery—providing dynamic device recognition;
- Test Service—providing establishment of subnetwork functionality and availability.

SOIS will define a standard subnetwork service interface for each of the services outlined above.

For each SOIS-compliant Data Link there will need to be a mapping of subnetwork services to actual Data Link implementation, including the provision of convergence functions where required. This mapping is left to dedicated groups and possibly SOIS (where no dedicated group is available). SOIS/CCSDS may adopt as recommendations the mappings performed by dedicated groups. The functions which have been identified as necessary to support SOIS subnetwork, either inherently by the Data Link or by the addition of convergence functions, are:

- redundancy;
- retry;
- segmentation;
- resource reservation;
- prioritisation;
- protocol multiplexing.

Figure 2-2 shows the Data Link layer convergence functions required with reference to examples of Data Link types. Each of these functions is described in 4.2.

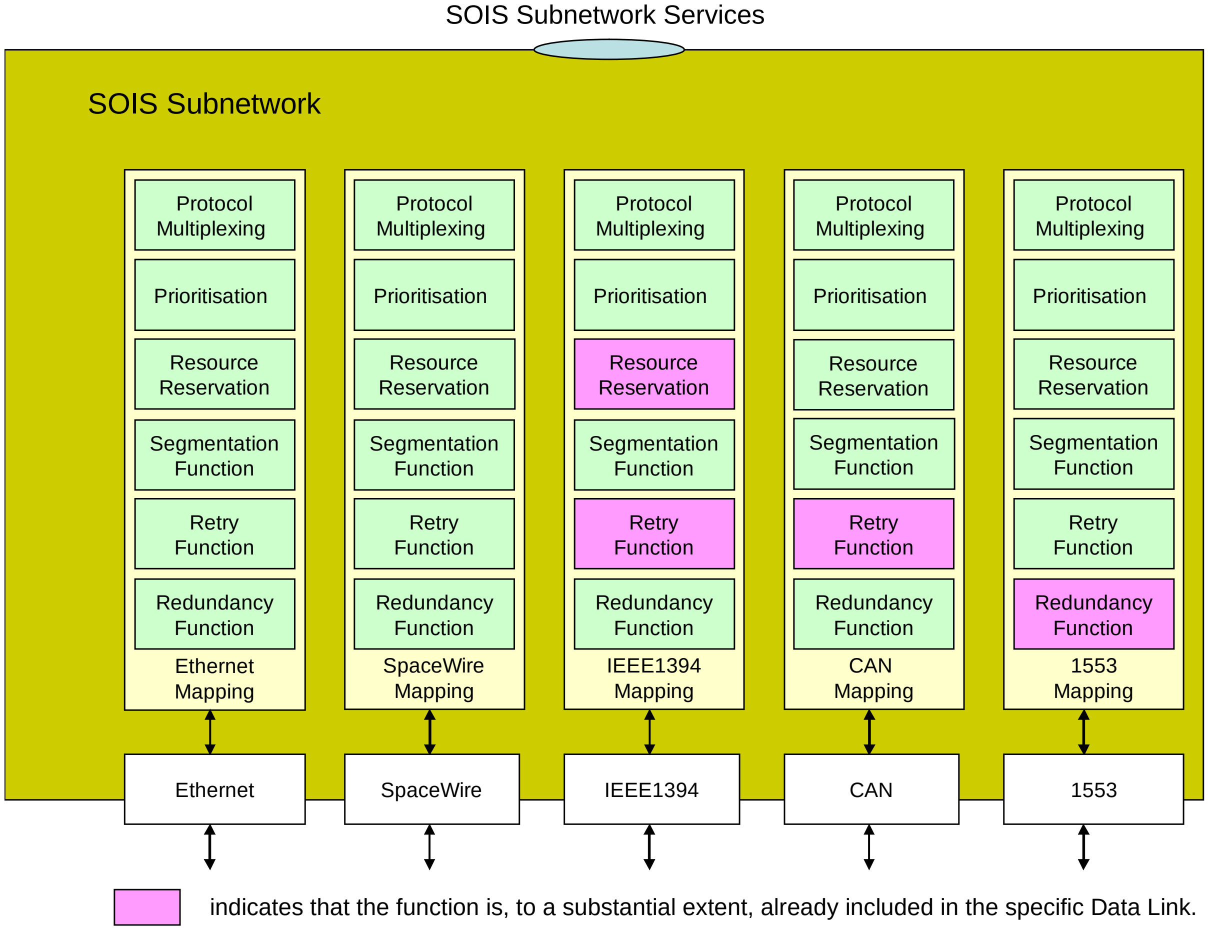

Figure 2-2: SOIS Data Link Functions

Each Data Link must support at least one QoS defined by SOIS (see reference [7]). The QoS relates to priority and delivery conditions and these will be made visible via the subnetwork service interface as parameters associated with each service request.

Note that segmentation is used in the general sense as is prevalent within CCSDS rather than in the specific sense used by TCP to indicate stream segmentation.

2.4 SOIS NAMING AND ADDRESSING

SOIS requires a consistent and integrated approach to a number of addressing regimes:

- the addressing of devices and, for each service, the device independent representation of Data Link service parameters (e.g., memory addresses, parameter names) and application service;
- translation of addresses by the SOIS application functions and device-dependent drivers;
- the addressing of Data Link service users by data system, service, and user entity;
- the addressing of application service users by data system, service, and user entity;
- the assignment of an address range to a particular Physical layer interface.

The particular challenge of the SOIS asymmetrical communications model is in ensuring that connection-oriented transactions remain possible given the disjoint addressing capabilities of the two ends of a transaction. This is typified by a low-level device having no cognisance of the addressing to an Application Support Layer entity, requiring that address translation occur through the system and that it be reversible, such that return data is sent to the correct data requestor.

The scope of addressing in the SOIS layers is:

- Application Support Layer—abstract global device identification;
- Transfer Layer—global addressing of data systems, and identification of lower-layer application entities, routing to multiple subnetworks;
- Subnetwork Layer—subnetwork-specific addressing of subnetwork data systems, services, and Network- or higher-layer entities.

2.5 MANAGEMENT CONCEPTS

SOIS conforms to the established consensus within CCSDS regarding management concepts.

The services and protocols defined by SOIS will include an abstract definition of the Management Information Base (MIB), which will include parameters, databases, and actions necessary to inform operation of the services and protocols. The method of access to the MIB by the management system is undefined and may be a combination of preconfigured code, local configuration, or remote management via management protocol and local agent.

The MIB associated with the Application Support Layer will inevitably be more extensive than that associated with the Data Link layer because both services and protocols are SOIS responsibilities in the Application Support Layer.

2.6 SOIS COMPLIANCE

The SOIS initiative attempts to standardise the way in which commonly required communication functions are implemented onboard a spacecraft. It does this by defining standard services at the Subnetwork Layer upon which upper-layer applications may rely. The subnetwork services are independent of the underlying hardware and thus insulate upper layers from implementation details. Further standardisation is provided by defining a common set of Application Support Layer services upon which spacecraft dependent applications may be built.

Ideally, any SOIS compliant implementation should implement the full set of Subnetwork and Application Support Layer services, but in reality implementers must have the freedom to select only those services which are required and to omit others in accordance with implementation requirements. To aid this selection process while still allowing implementers to claim SOIS compliance, the SOIS services will be specified using mandatory and optional compliance statements. These conformance statements will accompany the SOIS service definition and will be required to be completed by any developer claiming compliance to SOIS services. The conformance statements will allow for the necessary balance between mandatory and optional capabilities. For example, all implementations using SOIS must implement the packet service as a fundamental capability, but the four classes of delivery will be defined as optional, with only a single delivery class required for SOIS compliance.

To be SOIS compliant the Data Link-dependent and Data Link-independent functions shall be defined in a Data Link mapping document that includes sufficient detail and references to allow an independent implementation to be built.

SOIS will endeavour to restrict options to a bare minimum in the interests of promoting interoperability at both protocol and service levels. However, because of the varying capabilities of interfaces and the varying requirements of applications, some profiling will be defined, and appropriate conformance-statement pro formae will be included in the service/protocol definitions.

2.7 EXAMPLES

The following examples illustrate the SOIS approach.

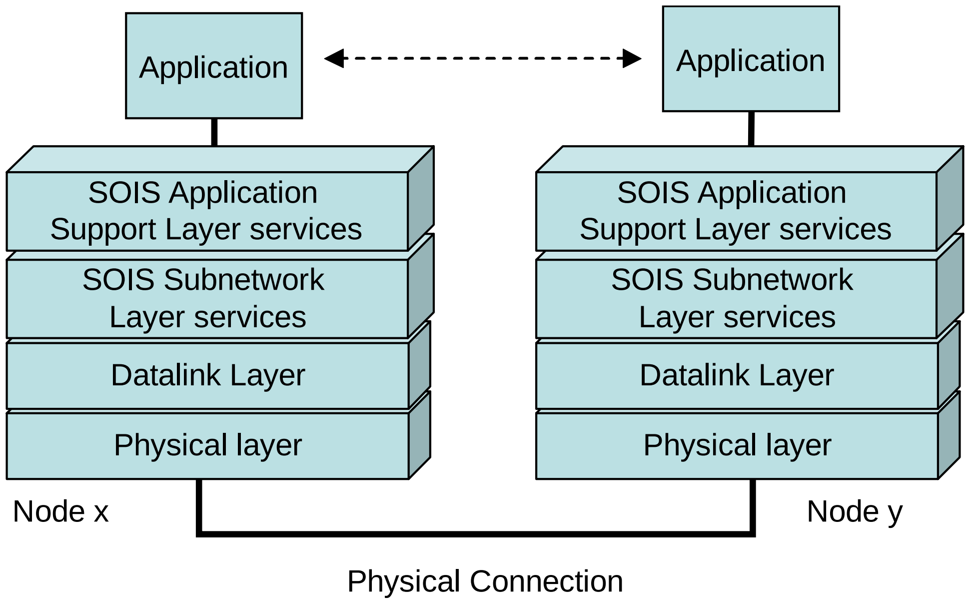

Figure 2-3: Symmetrical Communication

In figure 2-3 it is assumed that all communications are supported by intelligent nodes able to implement a full SOIS protocol stack. This scenario is typically used for communication between application software supported by a processor and associated resources, for example for message exchange using the SOIS message transfer service.

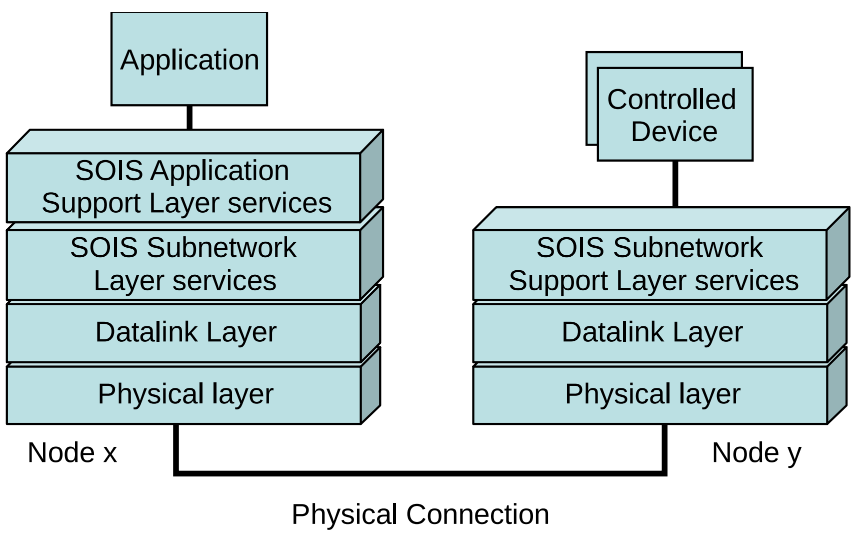

Figure 2-4: Asymmetrical Communication

In figure 2-4 it is assumed that a the controlled device is directly connected to a standard spacecraft Physical and Data Link layer and has sufficient capability to implement SOIS-defined Subnetwork Layer services. The subnetwork services implemented by the controlled device may range from a simple memory access service to a packet service. It is most likely utilised for devices supporting packet based communication.

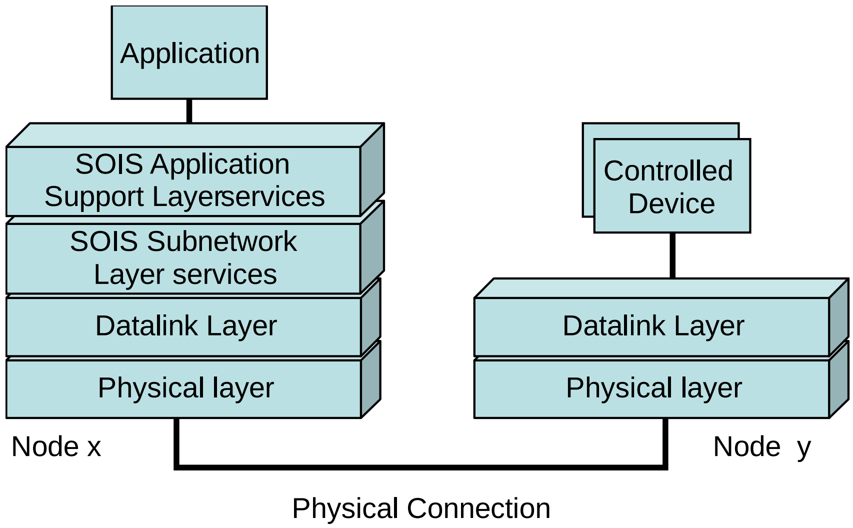

Figure 2-5: Simple Asymmetric Communication

In figure 2-5 it is assumed that the controlled device is connected directly to a Data Link and has no capability to implement the SOIS protocol stack. This is typical of sensors and actuators where the interface provides only for reading and writing register values. This example requires an unbalanced method of communication whereby the SOIS capabilities resident in the controlling node must take full responsibility for the controlled device. Unlike a conventional peer-to-peer data communications scenario, the service is provided at only one side of the interface, and consequently simple devices can be supported at the other side.

2.8 APPLICATION SUPPORT/SUBNETWORK LAYER INTERACTION

Figure 2-6 shows a generic view of the Application Support Layer services defined by SOIS.

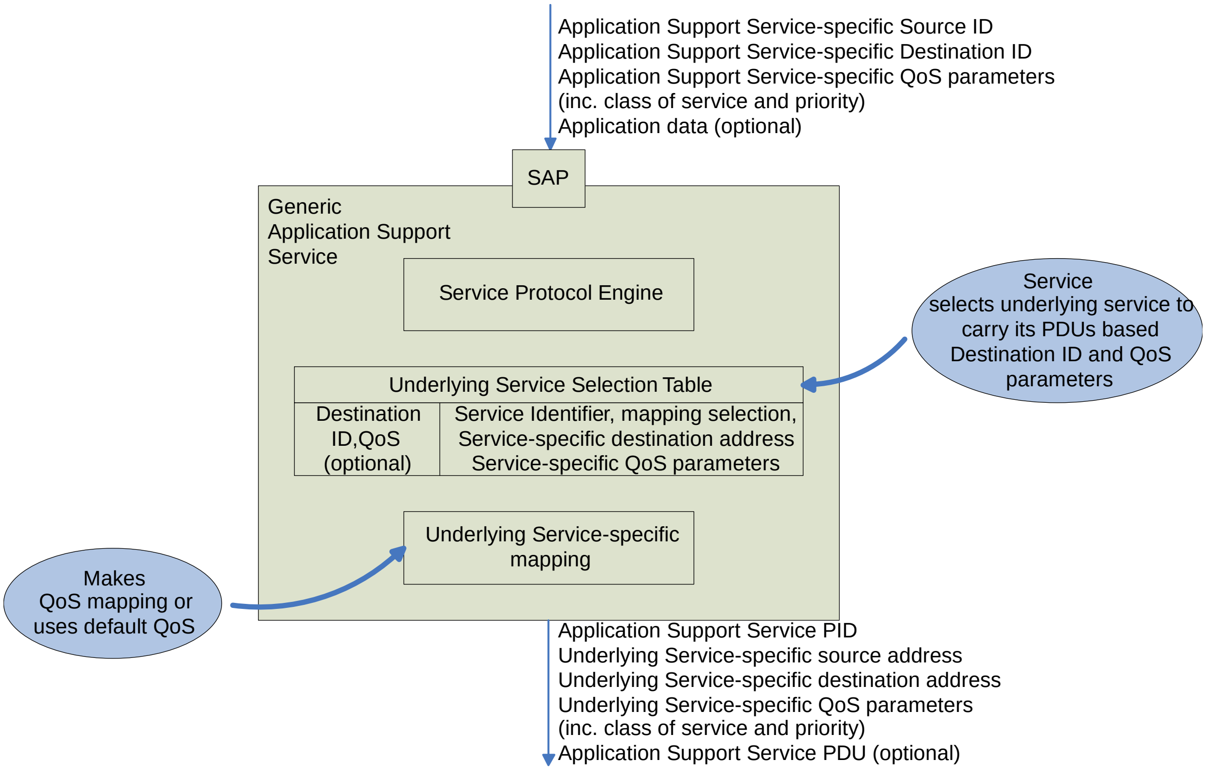

Figure 2-6 : Generic Application Support Layer Service

The figure shows the generic mechanism used to decide which service and underlying Data Link to use to satisfy a particular combination of required destination and QoS, and also the translation, if required, between addressing and QoS domains. The Application Support Layer service adapts to invoke the underlying services whether they be Transfer or Subnetwork Layer services. The subnetwork, on the other hand, provides the same services irrespective of whether the service user is a Transfer or Application Support Layer entity.

Note that:

– The underlying service can be provided by either Transfer or Subnetwork Layers. The source and destination addresses in the underlying subnetwork may therefore refer to network or subnetwork addresses.

– Application Support Layer service QoS semantics and parameters (‘App QoS’) are distinct from underlying ‘transfer’ service QoS semantics and parameters.

– The Application Support Layer service selects the appropriate ‘transfer’ service based on destination identifier and Application Support Layer service QoS, together with associated destination address and QoS parameters (where they can vary).

– Type of destination address and QoS parameters are ‘tailorable’ depending upon system engineering choice of direct to subnetwork or use of Transport/Network layers.

– The Application Support Layer service uses its appropriate mapping onto the underlying service (which may translate Application Support Layer service QoS to underlying service QoS or use default, ‘hard-coded’, QoS) to interface with service’s SAP.

– The Application Support Layer service performs address translation between Application Support Layer service source and destination IDs and the underlying service source and destination IDs.

– The nature of the Subnetwork service interface, including the service primitives and parameters, is independent of the entity using the service, be that entity at Transfer or Application Support Layer.

Figure 2-7 shows an example of the Message Transfer service operating over the SOIS packet service provided by a SpaceWire Data Link.

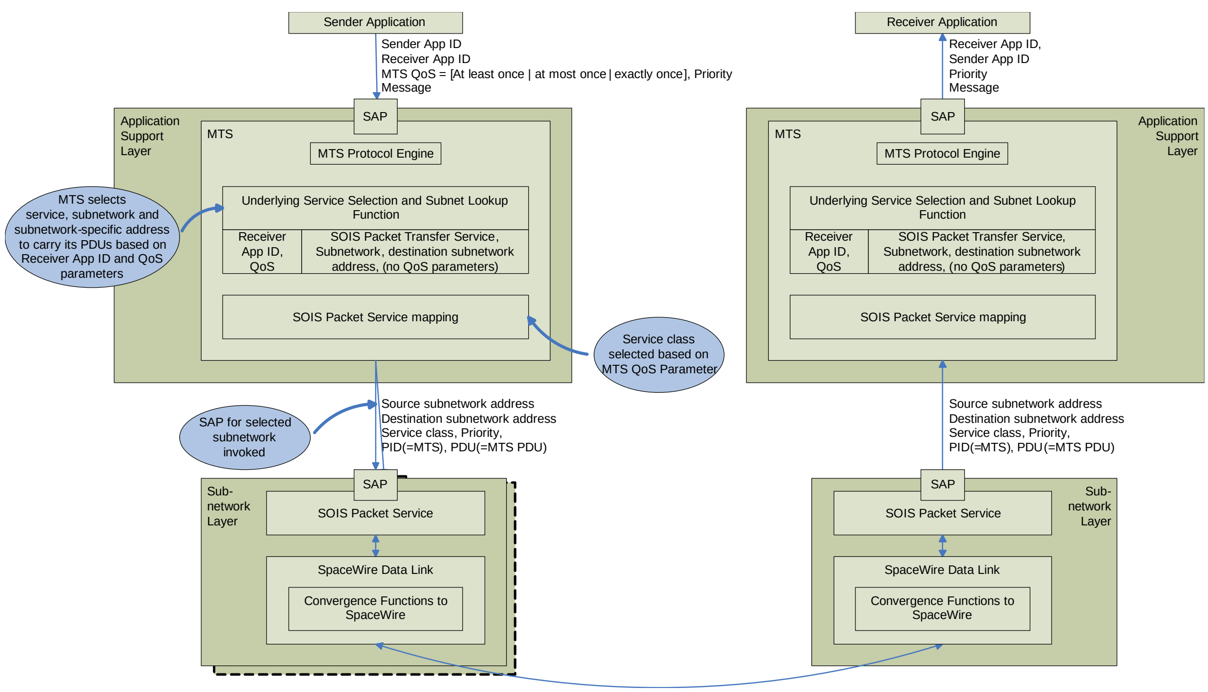

Figure 2-7: MTS over SOIS Packet Service and SpaceWire

A salient feature of this example is the selection of subnetwork performed by the Application Support layer. Alternate subnetwork selections may or may not be SpaceWire subnetworks.

Figure 2-8 shows the realisation of the arrangement put forward in figure 2-5 where asymmetric interaction occurs directly between application and a device using the memory access service.

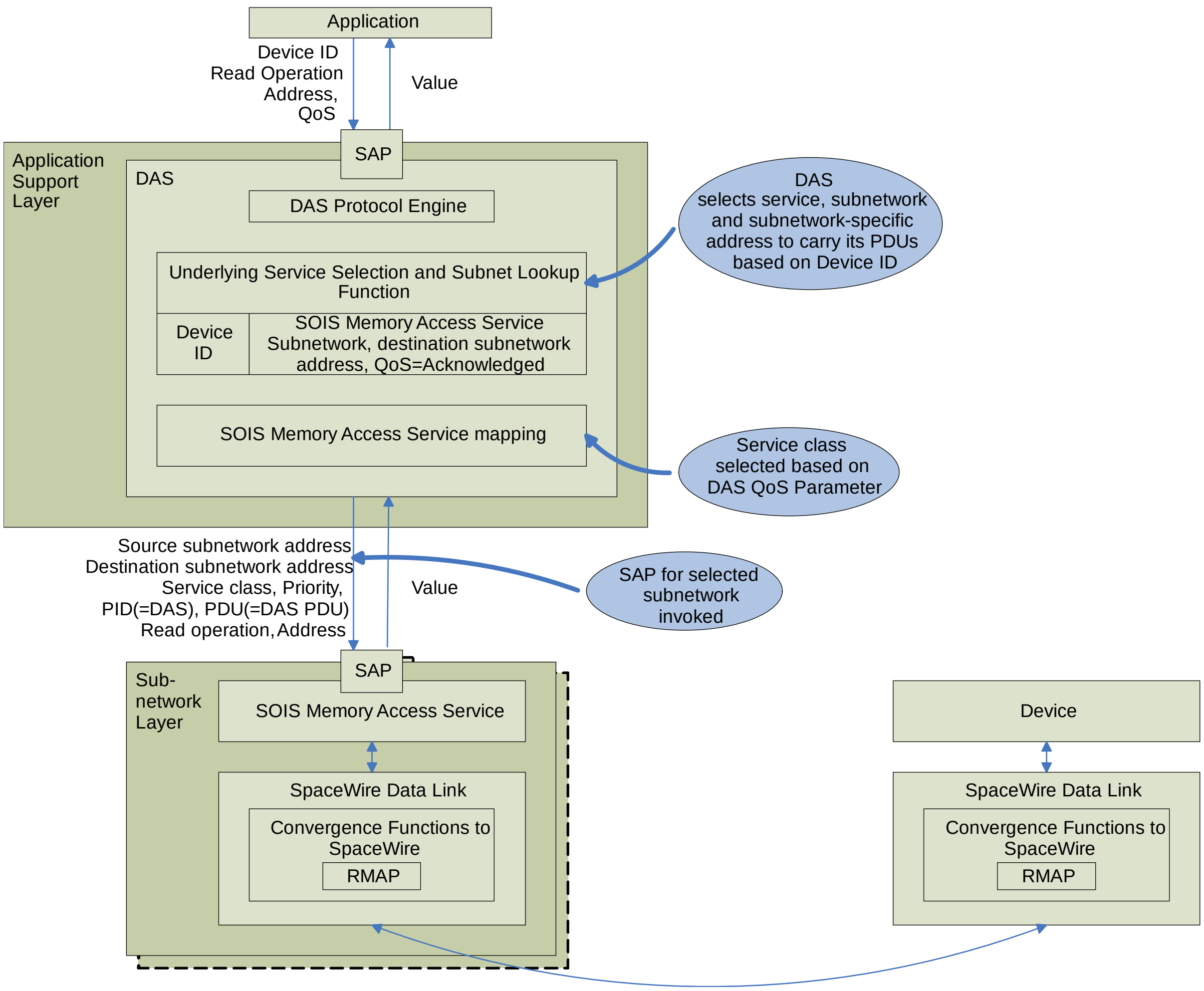

Figure 2-8: DAS over SOIS Memory Access Service, RMAP and SpaceWire

Finally, figure 2-9 illustrates the multiplexing of Application Support Layer services over a subnetwork service as well as the concurrent operation of both different Application Support Layer services and of different subnetwork services. The two SOIS packet service SAPs are differentiated by the protocol ID parameter present in the service primitives and carried in Data Link-specific protocol elements.

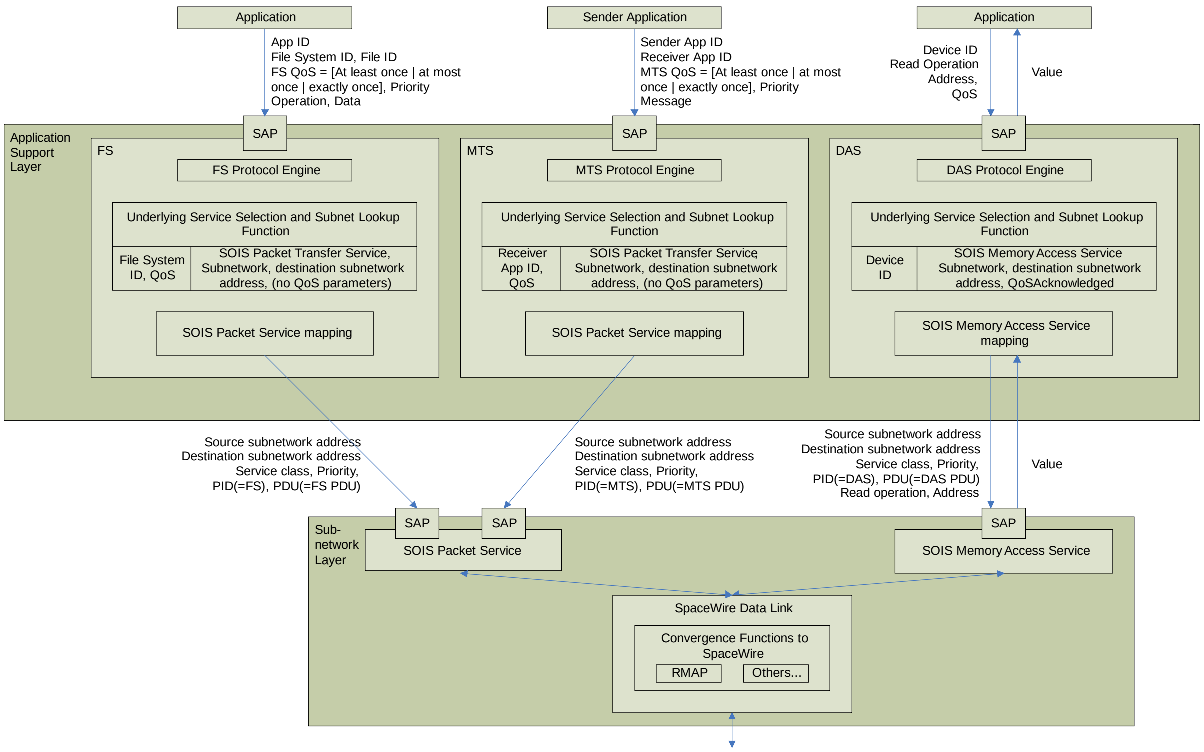

Figure 2-9: Example of Application Support Layer Services Multiplexed onto Subnetwork

3 SOIS APPLICATION SUPPORT LAYER SERVICES

3.1 INTRODUCTION

This section describes the suite of services in the SOIS Application Support layer which are designed to provide common services required by applications on any processing node of the spacecraft. They isolate the applications from the underlying topology and communications architecture of the spacecraft. These services then rely upon services provided by the underlying SOIS layers to carry the services' protocols over the spacecraft communications architecture.

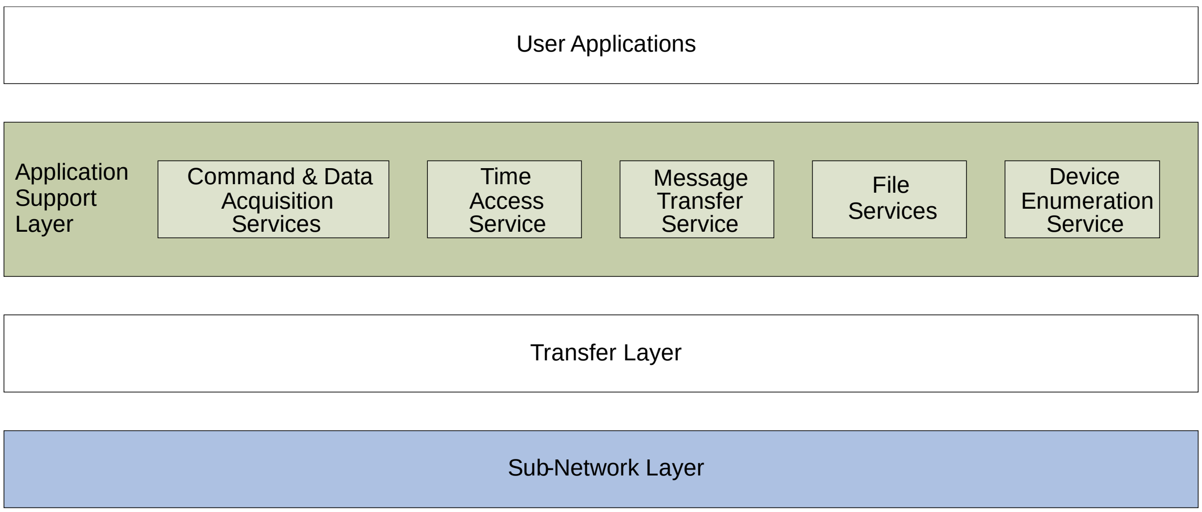

Figure 3-1: SOIS Application Support Layer Services

The suite of SOIS Application Support Layer services is as follows:

– Command and Data Acquisition Services (CDAS)—commanding and data acquisition by applications for transducers and simple instruments independent of their locations;

– Time Access Service (TAS)—access for applications to the onboard time with bounded accuracy independent of their locations;

– Message Transfer Service (MTS)—enables applications hosted onboard a spacecraft to communicate with each other using asynchronous, ad-hoc, discrete messaging with a bounded latency, including multicast and broadcast, independent of their locations;

– File Services (FS)—access by applications to, management of, and transfer of files within a (nominal) global onboard file store;

– Device Enumeration Service (DES)—support for dynamic spacecraft configuration.

This section provides an overview of each of the Application Support Layer services (a complete description of each of these services can be found in the relevant CCSDS Red Books).

3.2 COMMAND AND DATA ACQUISITION SERVICES

3.2.1 GENERAL

The Command and Data Acquisition Services (CDAS) are used to provide a low-overhead access method for spacecraft hardware devices such as sensors and actuators, regardless of location.

The CDAS are split into a number of capability sets, each served by a distinct service:

– Device Access—Device Dependant Driver providing basic reading from and writing to devices regardless of location;

– Device Virtualisation—Standard Device Driver providing reference to a device using a virtual, i.e., generic, image of a physical device;

– Device Data Pooling—maintaining an image of the states of a number of devices.

Each device is identified by the user’s using an abstract device identifier. The CDAS are responsible for mapping this device identifier onto an access method and appropriate addressing scheme, e.g., a subnetwork service and address.

3.2.2 DEVICE ACCESS SERVICE

3.2.2.1 Overview

The Device Access Service (DAS) provides a very basic device read and write capability that can be used directly by software applications, or can be used as the basis for more capable services, such as those performing engineering unit conversions on raw data, or monitoring services. The service user is isolated from the physical location or the detailed knowledge of the electrical interface (be it accessed via direct IO; analogue, digital, pulsed, etc., across a subnetwork or full network). It may be thought of as a **Device Dependant Driver**.

The benefit of the service is that the service user is no longer concerned with the details of the location of the sensor, its physical interface or how it is accessed. As a result, configuration changes involving a change in the physical location of a device or changes to its electrical interface do not require changes to the application software using that device.

Although isolated from the details of device location and interface type, the service user must still know the format of commands written to and data read from the device, and the user remains responsible for correctly composing and interpreting those formats.

3.2.2.2 Functions Performed

The functions performed by the DAS are:

– Read value from device: to read a value from a device, a service user provides a logical identifier, which the service resolves in order to determine the device location and the interface through which it is accessed. The service then reads the device and returns the value.

– Write value to device: to write a value to a device, a service user provides a logical identifier together with the value to be written. The service resolves the logical identifier in order to determine the device location and the interface through which it is accessed and then writes the value to it.

The DAS must deal with two scenarios when communicating with devices:

– client-server—there may be an imbalance in capability between the processor upon which the DAS is implemented and the device being controlled, requiring simple communications driven by a processor, e.g., an RS232 driver;

– peer-to-peer—other devices may be more sophisticated such that they implement protocol engines, e.g., a SpaceWire RMAP FPGA or a processor implementing full SOIS protocol.

The DAS must be able to select the appropriate access mechanism to each device based upon the device identifier. Because each access mechanism may support multiple devices, an address associated with the device must also be identified. Thus the DAS must provide the following mapping mechanism:

device identifier -> <access mechanism, address>

This service may be extended in future to offer a more device-driven service where values are returned from the device in response to an event or value change at the device.

3.2.2.3 Requirements for Underlying Services

The access mechanisms themselves are provided by underlying services. They depend upon the different devices, but typically take the form of one of the following:

– packet;

– get/set parameter;

– memory/register read/write.

Where the same access mechanism may be used to access different devices, the underlying service must provide a unique device identifier, e.g., **address**.

Where there is a common resource used for accessing multiple devices, e.g., an ADC multiplexer or a subnetwork, **contention** for the resource may occur. To support the real-time properties required by onboard applications, the underlying services must provide **bounded worst-case access times** and **prioritised access**.

3.2.3 DEVICE VIRTUALISATION SERVICE

3.2.3.1 Overview

The Device Virtualisation Service (DVS) provides for the service user to refer to a device using a virtual, i.e., generic, image of a physical device. The service user interacts with the virtual image of the physical device and the DVS service handles the translation of commands to the virtual image into commands to the physical device, and vice versa for data. A simple example of virtualisation is use of the virtual image like a disk drive for a physical device like flash memory. The user does not need to know the physical access mechanism of a flash memory and accesses it by sending commands to access a disk; such commands get translated into appropriate accesses to a flash memory. It may be thought of as a **Standard Device Driver**.

The benefit of using this service comes from the service user's being isolated from the physical characteristics of the device so that a class of devices can be commanded in the same manner, by interacting with the virtual device.

3.2.3.2 Functions Performed

The functions performed by the DVS are dependant upon the virtual device. However, they fall into two broad categories:

– Commanding: to command a device, a service user provides a device identifier, command identifier, and command parameters. The service initiates the command and returns a status indicating the outcome of the command.

– Data Acquisition: to acquire data from a device, a service user provides a device identifier and a data identifier. The service typically initiates the acquisition of the identified data from the device and returns the data.

3.2.3.3 Requirements for Underlying Services

The DVS typically uses the DAS to command and acquire data from a device.

3.2.4 DEVICE DATA POOLING SERVICE

3.2.4.1 Overview

The Device Data Pooling Service (DDPS) maintains an image of the states of a number of devices. A service user can access the state of a device in data pool without having to generate an explicit read request for the real device. The DDPS will periodically sample the devices at a determined sampling rate or cache state from devices that generate interrupts. It provides guarantees on the maximum age of each parameter in the pool and the accuracy of the software images. The DDPS may be implemented on top of a DAS implementation.

The benefit of using this service comes from the avoidance of the repetition of multiple users sending the same sampling commands to devices, thereby reducing traffic on the data bus.

3.2.4.2 Functions Performed

The functions performed by the DDPS are:

– Acquire data from a device and store in data pool. This can take two forms:
  - DDPS periodically acquires data from device;
  - device periodically or sporadically sends data to DDPS.

– Read value from data pool; allow service users to read values from the data pool.

3.2.4.3 Requirements for Underlying Services

The DDPS typically uses either the DVS or the DAS to acquire data from a device.

3.3 TIME ACCESS SERVICE

3.3.1 GENERAL

The Time Access Service (TAS) provides service users with a consistent interface to a local time source that is correlated to some centrally maintained master onboard time source by some mission-defined underlying service. The time values provided by this service might typically be used by the application to schedule some operation, such as the acquisition of an image or to time stamp locally generated telemetry data.

The need to provide a local, correlated time source in onboard processor nodes is common to all spacecraft that have more than one processing node connected to a subnetwork or a network. A typical architectural scenario is shown in figure 3-2. Note that the TAS is concerned only with providing the interface to the local time source. Time distribution is addressed by the SOIS Time Distribution Service described in 4.3.6.

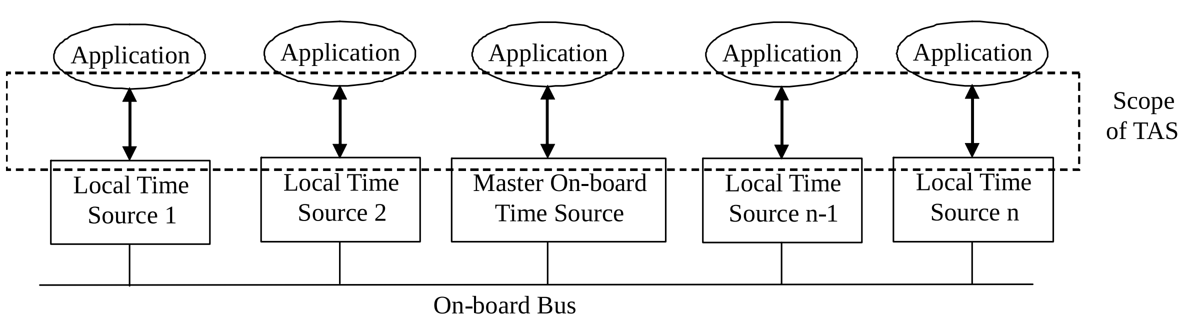

Figure 3-2: Time Access Services

In this architecture the local time sources are typically free-running hardware counters accumulating seconds and sub-seconds of elapsed time. Each of these counters is driven by its own oscillator, and the absolute frequency and frequency stability of each oscillator are different in each node. The master onboard time source, the reference for onboard time for all onboard mission operations, is usually a similar free-running counter driven by an oscillator with precise absolute frequency and high stability. The value provided by this time source is usually called the Mission Elapsed Time (MET).

The TAS defines a standard interface between applications hosted on each node and the local time source for that node (the scope of TAS is indicated by the dashed box in figure 3-2).

The benefit is that all service users have a uniform interface to the local time source, regardless of their location on the spacecraft, and have no need to access local hardware directly. This simplifies the development of applications and allows them to be relocated if necessary and re-used in other missions.

3.3.2 FUNCTIONS PERFORMED

The basic capability provided by the TAS is:

– ‘wall clock’ capability, which enables the application to read the time on demand.

Two optional extensions that reflect common requirements for onboard software systems are also defined:

– ‘alarm clock’ capability, which enables the application to request notification at a particular time;

– ‘metronome’ capability, which enables the application to request periodic notifications with a specified interval and starting at a particular time.

3.3.3 REQUIREMENTS FOR UNDERLYING SERVICES

The TAS requires access to a local time source. Where the spacecraft has more than one processing node connected to a subnetwork or a network, an underlying service shall provide correlation of the local time source to a master onboard time source.

3.4 MESSAGE TRANSFER SERVICE

3.4.1 GENERAL

The Message Transfer Service (MTS) enables applications hosted onboard a spacecraft to communicate with each other using asynchronous, ad-hoc, discrete messaging with a bounded latency, including multicast and broadcast, independent of their locations. The MTS can be used for direct exchange of information or synchronisation between applications or as the basis of higher-level services, e.g., application frameworks such as the European PUS services. It is a goal of MTS to be efficient and predictable so as to support modelling of onboard software task and communication scheduling.

Note that each service user must be uniquely identified by an 'MTS User Identifier'. The transfer of messages between service users may be determined by QoS parameters that, in conjunction with the location of the service users, will determine the underlying service used and its associated QoS parameters.

3.4.2 FUNCTIONS PERFORMED

The MTS offers the following functions:

– Send a discrete message to another application at a particular priority;

– Receive the next queued discrete message: this is the next queued message waiting to be received, the next message being determined in priority order, FIFO with a priority level.

Two optional extensions are also defined:

– Broadcast a discrete message to all applications;

– Multicast a discrete message to all applications within a defined group.

Where MTS is implemented as a peer-to-peer communication between MTS implementations where sender and receiver reside on different nodes, an MTS protocol will be required. This will be carried by the underlying services.

3.4.3 REQUIREMENTS FOR UNDERLYING SERVICES

Where the sending and receiving applications reside on different processors, the underlying services shall provide a service to **transfer discrete messages** between MTS implementations on the different processors. The underlying service must provide for uniquely identifying the source and destination processors, e.g., **addresses**. To support the real-time properties required by onboard applications, the underlying services must provide **bounded worst-case delivery times** and **prioritised delivery** of the discrete messages.

The MTS must provide a **queuing mechanism** for the messages awaiting reception by a receiving application. This mechanism must deliver the messages in priority order, FIFO within a priority level. Such a mechanism may be provided by an underlying RTOS.

3.5 FILE SERVICES

3.5.1 GENERAL

The File Services (FS) are for use by service users to access, manage, and transfer files that could contain any type of data, including telemetry, commands and command sequences, software updates, imagery and other science observations. In addition to the general SOIS goal of more reusable applications and tolerance for changes in the spacecraft hardware configuration, use of the file services will make it easier to control access and management of shared hardware resources (e.g., mass memories).

A basic concept of this service is the definition of a file store. A file is a named data set residing in a file store. A file store comprises:

– A memory element in which files reside. This may be implemented in, e.g., local processor memory, hard disk stores, or a dedicated mass memory subsystem;

– An associated file system providing functionality for managing the files.

Note that no assumption is made about persistence of files in the file store or any file replication strategies.

As indicated in figure 3-3, the FS comprises the following categories of service:

– File Access—allows access to files and portions of their contents;

– File Management—allows manipulation of existing files on local or remote file stores;

– File Transfer—allows a service user to transfer files between file stores residing on the same spacecraft.

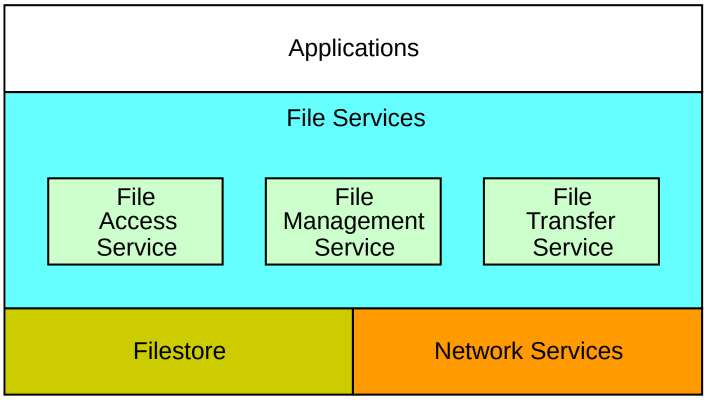

Figure 3-3: File Services Overview

Each of these distinct services and required underlying services is described in more detail in the following subsections.

3.5.2 FILE ACCESS SERVICE

3.5.2.1 Overview

The File Access Service (FAS) provides for the service user to access files and portions of their contents in a file store regardless of its location, i.e., the accessed files can reside on file stores local or remote to the service user on the spacecraft.

3.5.2.2 Functions Performed

The FAS provides the following capabilities:

- open file;
- close file;
- read from file;
- write to file;
- append to file;
- insert into file;
- remove from file;
- find first in file;
- find next in file;
- find last in file.

3.5.2.3 Requirements for Underlying Services

Where the file store is located on a different processor from that of the service user, the FAS requires access to a network file access protocol.

Where the file store is located on the same processor to the service user, the FAS requires the provision of a Minimum File Store service, as defined in 3.5.4.3.

3.5.3 FILE MANAGEMENT SERVICE

3.5.3.1 Overview

The File Management Service (FMS) allows service users to manipulate existing files in a file store regardless of location, i.e., the accessed files can reside on file stores local or remote to the service user on the spacecraft.

3.5.3.2 Functions Performed

The FMS provides the following capabilities:

- make directory;
- destroy directory;
- rename directory;
- lock directory;
- unlock directory;
- create file in directory;
- delete file from directory;
- delete all types of file in directory;
- copy file (within a file store);
- concatenate two files;
- rename file;
- lock file;
- unlock file;
- find file in file store;
- list directory contents;
- operate specified algorithm on file.

3.5.3.3 Requirements for Underlying Services

Where the file store is located on a different processor from that of the service user, the FAS requires access to a network file access protocol.

Where the file store is located on the same processor as the service user, the FAS requires the provision of a Minimum File Store service, as defined in 3.5.4.3.

3.5.4 FILE TRANSFER SERVICE

The File Transfer Service (FTS) allows a service user to transfer files between file stores residing on the same spacecraft.

NOTE – The FTS does NOT allow a service user to transfer files between file stores residing on different nearby networked spacecrafts as part of a constellation, a formation or a planet exploration fleet, or between the spacecraft and ground; this capability is provided by the CCSDS File Delivery Protocol (CFDP) (reference [8]).

3.5.4.1 Functions Performed

The FTS provides the following capabilities:

– initiating the transmission of files between file stores;

– writing produced data into a file in the local or in a remote file store;

– initiating file store operations on local and remote file stores;

– receiving events related to the operation of current transactions;

– requesting status information related to the current transactions;

– sending or receiving messages associated with the current transactions;

– suspending, resuming, or cancelling the transmission of a current transaction.

3.5.4.2 Requirements for Underlying Services

The FAS requires access to a network file transfer protocol and a Minimum File Store service, as defined in 3.5.4.3.

3.5.4.3 Minimum File System

To enable interoperability, the FS requires a file system (interfacing to a file store) to provide the following minimum set of capabilities to be used to access, manage, and transfer files:

– create file;

– read from file;

– write to file;

– delete file;

– rename file;

– append file;
– replace file;
– create directory;
– delete directory;
– rename directory;
– list contents of directory.

In an FS implementation, the service primitives must then be mapped to and from the actual file system available. This approach allows complete independence from the technology used to implement the file store. Of course, the way in which this mapping is performed is implementation specific.

3.6 DEVICE ENUMERATION

The device enumeration service is intended to be used in support of future plug-and-play applications This service is to be defined.

4 SOIS SUBNETWORK SERVICES

4.1 INTRODUCTION

The SOIS subnetwork is decomposed into a number of layers and sublayers as illustrated in figure 4-1. The uppermost sublayer is formed by the subnetwork service sublayer. This sublayer exposes a standard set of SOIS services to upper-layer applications. These services are supported by two underlying mechanisms:

– Where the capabilities provided by the underlying Physical and Data Link layers already support the required SOIS service, a direct mapping of the Data Link service to the SOIS subnetwork is performed.

– Where the capabilities of the underlying Physical and Data Link layers do not support the required SOIS service, additional functionally must be provided by the SOIS convergence sublayer in order to achieve the required SOIS subnetwork service. The convergence sublayer may require the provision of a protocol to provide the required functionality. While it is tempting to assume that a single protocol could be developed to operate over all types of Data Link layers, in reality the requirements of the convergence sublayer protocol will be highly dependent on the diverse services provided by individual Data Link protocols.

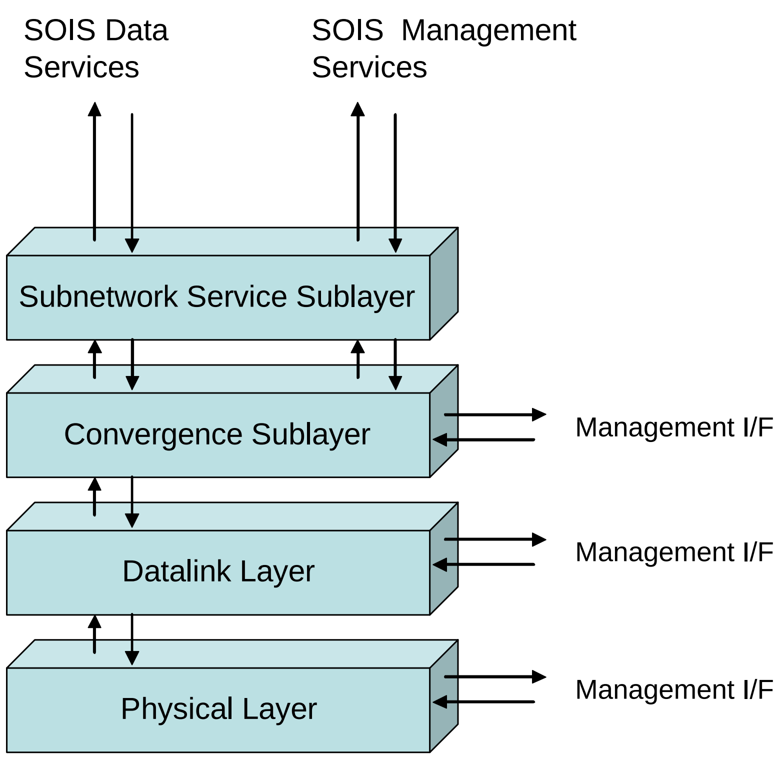

Figure 4-1: SOIS Subnetwork Decomposition

It should be noted that it may not be feasible, or desirable, to provide all SOIS subnetwork services over all Physical and Data Link layer combinations. For example, a simple sensor interface is not likely to require or support packet or time distribution. To support such variation in capability, the SOIS services will be divided into classes with each class having a defined set of capabilities.

To provide a set of services that are feasible and realistic to support, account must be taken of implementation constraints and variability in design choice. For example:

– Priority of transfer may be supported by multiple queues at the Data Link layer, with each queue assigned a certain priority. It could equally be implemented using a single queue at the Data Link layer and assigning priority at a high layer of protocol. Alternatively, priority may be on a first-come-first-served basis.

– Data transfer may be guaranteed by the Data Link layer protocol; alternatively, the Data Link layer may provide a best-effort service which may be optionally improved by Transfer Layer retransmission protocols. Furthermore, for time-critical data transfers it may be undesirable to perform any retransmission, although retransmission may still be utilised with a bounded time after which the data is considered useless and retries are abandoned.

To account for this variability in implementation, the SOIS services will include optional parameters as part of the service primitive. It will be up to the implementer to decide which optional parameters will be used.

The following set of services has been selected for SOIS subnetwork standardisation:

– Packet;

– Get Parameter;

– Set Parameter;

– Write Location;

– Read Location;

– Time Distribution;

– Device Discovery;

– Test.

4.2 DATA LINK LAYER FUNCTIONS

4.2.1 GENERAL

Data Link functions provide a consistent interface to the subnetwork services by implementing a common set of functionality, described in the following subsections. Not all buses/subnetworks provide the full set of required Data Link functionality. The SOIS Data Link layer provides the missing functionality for each of the buses/subnetworks. The Data Link layer sends and receives delimited octet-aligned data across a single Data Link. It provides optional link-level services such as link-level retry, link-level redundancy management, and link-level scheduling.

4.2.2 REDUNDANCY FUNCTION

The redundancy model adopted by SOIS is that of equivalent Data Links that provide alternative paths from a source end-point to a destination end-point on a single subnetwork. The architecture supports autonomous switching between equivalent Data Links. Non-autonomous switching of paths can be done by an application specifying the bus/link/path to be used (using different destination addresses) or reconfiguring routing tables used to select which link is used for a particular address. Applications may control autonomous redundancy and link-level retry by using a management parameter associated with the transmit traffic class. It should be noted that system management policy might uniformly dictate a redundancy policy which applications must use.

Link equivalence requires two independent paths to a destination. These redundant paths may be used in one of three ways:

a) sending data over both paths at the same time;

b) sending over the prime link and then if there is a failure using the redundant link (often used for MIL-STD-1553B bus);

c) sending over either link, and then if there is failure of one link all traffic goes over the remaining link.

The link redundancy function is bus/subnetwork specific.

4.2.3 RETRY FUNCTION

The link-level retry function provides a mechanism for resending PDUs that are not received at the other end of the Data Link, either through data unit loss or data unit discard because of errors. The retry mechanism itself is subnetwork specific. Consideration should be given to any QoS parameters when setting the time-out and retry values, i.e., packets that have bounded latency requirements. If multiple copies of the same PDU arrive at the destination, i.e., the first PDU arrived after the initial time-out, any duplicates are discarded.

4.2.4 SEGMENTATION FUNCTION

Segmentation is needed if the underlying bus/subnetwork cannot support the maximum SDU size in a single packet on the bus/subnetwork. Segmentation may also be necessary to reduce latency caused by large data units in the Data Link queues. It is the Data Link's responsibility to segment the PDUs if necessary and to reassemble them at the other end of the Data Link to reform the original PDUs before they are passed to the user of the Subnetwork Layer service. The segmentation function is bus/subnetwork specific.

4.2.5 RESOURCE RESERVATION FUNCTION

Resource reservation assigns Data Link resources to specific traffic classes. The Resource reservation function provides support for the following four traffic classes:

The **Best Effort traffic class** provides for non-reserved, try once communication. It makes no promises about the time of delivery, the network bandwidth available, or the error rate of the traffic. Several priority levels are provided for Best Effort traffic. Traffic with a higher priority level is treated preferentially compared to traffic with a lower priority level.

The **Assured traffic class** provides for non-reserved communication with retries. It tries to ensure that the traffic arrives at the intended destination. If the data does not arrive safely at the destination then it is resent. To support this, the destination acknowledges the receipt of Assured traffic. Several priority levels are provided for Assured traffic, which are the same levels as those for Best Effort traffic.

The **Reserved traffic class** provides for best-effort communication over a resource reserved logical link. It is able to ensure the time of delivery and the network bandwidth available, but makes no promises about the error rate of the traffic. Traffic may be lost but if it does arrive at the destination it will do so in a timely manner.

The **Guaranteed traffic class** provides for resource reserved communications with retries. It is able to ensure the time of delivery, the network bandwidth available, and tries to ensure that the traffic arrives at the intended destination without error.

Any combination or indeed all of these traffic classes may be used in the same SOIS system.

The resource reservation function is bus/subnetwork specific.

4.2.6 PRIORITISATION FUNCTION

The Prioritisation function queues incoming SDUs for transmission in priority order, that is to say, highest priority first, with FIFO ordering of SDUs within the same priority level. It is informed by the priority parameters in the subnetwork data service request.

4.2.7 PROTOCOL MULTIPLEXING FUNCTION

The protocol multiplexing function allows multiple Network or higher-layer entities to access Subnetwork Layer services. This is achieved via a protocol identification capability specific to the subnetwork. This capability is used to direct service indications to the appropriate Network or higher-layer entities. Note that the prioritisation and resource reservation functions need not be informed by the protocol identification since these functions are informed directly by appropriate service parameters.

4.3 SOIS SUBNETWORK SERVICE DESCRIPTIONS

4.3.1 PACKET SERVICE

4.3.1.1 Overview

The SOIS packet service supports the transfer of data packets over a single bus/subnetwork while presenting a consistent, uniform interface to the service users. It enables the multiplexing of multiple network protocols with a range of QoS support over underlying Data Links. QoS is provided by means of prioritisation of PDUs and resource reservation.

4.3.1.2 Functions

An implementation of the packet service must provide functions that are necessary to transfer data across a single bus/subnetwork. Required functionality of the underlying Data Link layer may vary according to the requested QoS. For example, an implementation of this service over MIL-STD-1553 requires the following functions:

– Subnetwork address translation, which translates the data destination address provided by the service user into a MIL-STD-1553B bus terminal address;

– Protocol Identification that ensures that incoming data are directed to the appropriate entity in the Transfer or Application Support layers;

– Segmentation, that breaks large data units into segments of data that can be transferred using a sequence of MIL-STD-1553B messages, each with a maximum size of thirty-two sixteen-bit data words.

By contrast, the implementation of this service to carry, for instance, IP packets reliably over a SpaceWire bus requires the following functions:

– protocol multiplexing, if other types of protocols and services, such as SCPS packets or DAS PDUs, share the bus;

– address translation, which translates the data destination address provided by the service user into a node address that is recognised on the SpaceWire bus;

– resource reservation, which assures that the packet has the bandwidth or slot allocation required;

– retry function, such as LLC, which will guarantee delivery or send back status to the service user if delivery could not be completed;

– redundancy function, which will provide an alternate path on the Data Link if the primary path is not available or healthy.

A major function of the packet service, and an essential one in time critical installations, is to provide the QoS function: prioritisation of PDUs and resource reservation. The transmit interface to the packet service accepts the PDU to be transmitted along with parameters for the protocol, source address, destination address and traffic class. Together, these parameters supply all the information needed for transporting the PDU across a specific bus/LAN, to its destination, with a given QoS.

Traffic Class defines the type of traffic being passed to the packet service and how that traffic will be treated as it passes through a particular subnetwork. Traffic Class encompasses priority, retry, redundancy, and resource reservation in a single QoS identifier that specifies to the packet service how an SDU is to be handled. There are four principal traffic classes supported: Best-Effort, Assured, Reserved, Guaranteed. The SOIS Green Book on the QoS (see reference [7]) should be consulted for more details.

The protocol multiplexing function supports the multiplexing and de-multiplexing of the different network protocols over the underlying Data Link links. Examples of protocols to be supported are IPv4, IPv6, and SCPS-NP. Abstract protocol identification is translated to subnetwork-specific capabilities in the Data Link layer.

Addressing is performed using system-wide logical addressing which is translated to the physical addresses used in the Data Link layer.

4.3.2 GET PARAMETER SERVICE

4.3.2.1 Overview

The SOIS get parameter service provides the capability to read the value of a parameter from a device. This is an asymmetric service that is provided in the node on which the service user is hosted. The service can acquire data from devices that are:

– directly connected to the node, e.g., through an analogue or digital interface;

– directly connected to a bus that is attached to the node, e.g., a MIL-STD-1553B data bus.

NOTE – This service does not acquire data from devices that are attached to remote nodes. SOIS Application Support Layer services are provided for this.

The service user must know the details of the physical port and/or channel to which the device is connected.

4.3.2.2 Functions

An implementation of the get parameter service must provide functions to read data from locally connected devices. These functions are specific to the actual device interfaces that are used.

4.3.3 SET PARAMETER SERVICE

4.3.3.1 Overview

The SOIS set parameter service provides the capability to write a parameter value to a device. This is an asymmetric service that is provided in the node on which the service user is hosted. The service can write data to devices that are:

– directly connected to the node, e.g., through an analogue or digital interface;

– directly connected to a bus that is attached to the node, e.g., a MIL-STD-1553B data bus.

NOTE – This service does not write data to devices that are attached to remote nodes. SOIS Application Support Layer services are provided for this.

The service user must know the details of the physical port and/or channel to which the device is connected.

4.3.3.2 Functions

An implementation of the set parameter service must provide functions to write data from locally connected devices. These functions are specific to the actual device interfaces that are used.

4.3.4 WRITE LOCATION SERVICE

4.3.4.1 Overview

The SOIS write location service provides the capability to write data into a memory or register location in a device. Data can be written one word at a time, or as a block of words that are loaded into contiguous memory locations on the target device.

This is an asymmetric service that is provided in the node on which the service user is hosted. The service can write data to devices that are:

- directly connected to the node, e.g., through a serial digital interface or a SpaceWire link;
- directly connected to a bus that is attached to the node, e.g., a MIL-STD-1553B data bus.

The service user must know the details of the physical port and/or channel to which the device is connected, and the memory/register mapping of the device to which data is to be written.

4.3.4.2 Functions

An implementation of the write location service must provide functions to write data to locally connected devices. These functions are specific to the actual device interfaces that are used.

4.3.5 READ LOCATION SERVICE

4.3.5.1 Overview

The SOIS read location service provides the capability to read data from a memory or register location in a device. Data can be read one word at a time, or as a block of words read from contiguous memory locations on the target device.

This is an asymmetric service that is provided in the node on which the service user is hosted. The service can read data from devices that are:

- directly connected to the node, e.g., through a serial digital interface or a SpaceWire link;
- directly connected to a bus that is attached to the node, e.g., a MIL-STD-1553B data bus.

The service user must know the details of the physical port and/or channel to which the device is connected, and the memory/register mapping of the device to which data is to be written.

4.3.5.2 Functions

An implementation of the read location service must provide functions to read data from locally connected devices. These functions are specific to the actual device interfaces that are used.

4.3.6 TIME DISTRIBUTION SERVICE

4.3.6.1 Overview

The SOIS time distribution service provides the capability to distribute a centrally maintained reference time to multiple users throughout the spacecraft. This is an asymmetric producer/consumer service. The producer service capability is provided on one master node (which normally also hosts the reference time clock) at any given time (thus not precluding redundant master nodes), and the consumer service capability is provided on all other nodes.

This service can be used to correlate the locally maintained time sources used by the SOIS time access service that is provided at the Application Support Layer.

4.3.6.2 Functions

An implementation of the producer part of the time distribution service must provide functions to distribute time information and time synchronisation to other nodes in the spacecraft. An implementation of the consumer part of the service must provide functions to receive time information from the consumer, and to update locally maintained time sources synchronously with the synchronisation pulses.

4.3.7 DEVICE DISCOVERY SERVICE

4.3.7.1 Overview

The SOIS device discovery service provides the capability to detect devices becoming active following a change in the hardware configuration of the spacecraft. This may occur when a cold redundant device is powered on, for example. This service discovers devices that are:

– directly connected to the node, e.g., through an analogue or digital interface;

– directly connected to a bus that is attached to the node, e.g., a MIL-STD-1553B data bus.

4.3.7.2 Functions

An implementation of the device discovery service must provide functions to detect hardware configurations, e.g., by detecting hardware events, or by periodically scanning for attached devices. These functions are specific to the Data Link to which devices may be attached.

4.3.8 TEST SERVICE

The Test Service is intended to be used for checking the status of the interface. The return parameters may be dependent on the inherent capabilities of the interface and that provided by the Data Link layer protocol. As a minimum the service should return a go/no-go status but this may be augmented by error codes, bit rate selection, prime/redundant media active, etc. The service does not indicate the correct operation of other subnetwork services but allows for the reporting of subnetwork-specific status.

ANNEX A

ACRONYMS AND ABBREVIATIONS

This annex identifies and defines the acronyms and abbreviations used in this Report.

API	Application Programming Interface
CDAS	Command and Data Acquisition Services
CFDP	CCSDS File Delivery Protocol
DAS	Device Access Service
DDPS	Device Data Pooling Service
DES	Device Enumeration Service
DVS	Device Virtualisation Service
EGSE	Electrical Ground Support Equipment
FAS	File Access Service
FMS	File Management Service
FPGA	Field Programmable Gate Array
FS	File Services
FTS	File Transfer Service
LAN	Local Area Network
MIB	Management Information Base
MTS	Message Transfer Service
NP	Networking Protocol
PDU	Protocol Data Unit
QoS	Quality of Service
RMAP	Remote Memory Access Protocol
RTOS	Real Time Operating System

SAP	Service Access Point
SCPS	Space Communications Protocol Specification
SDU	Service Data Unit
TAS	Time Access Service
TCP	Transmission Control Protocol
UDP	User Datagram Protocol