Consultative Committee for Space Data Systems

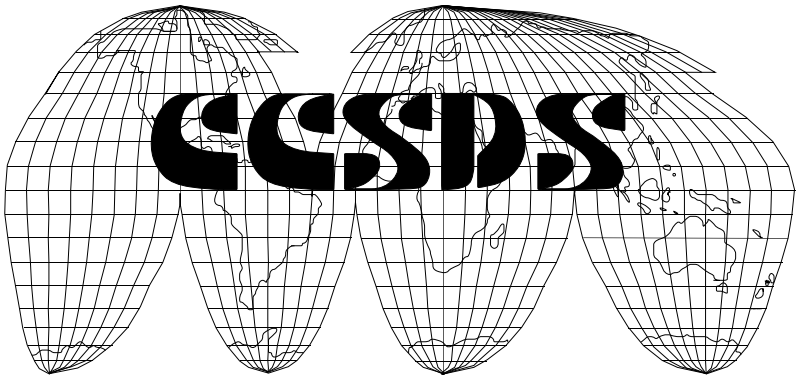
REPORT CONCERNING SPACE DATA SYSTEMS STANDARDS

STANDARD TERMINOLOGY, CONVENTIONS, AND METHODOLOGY (TCM) FOR DEFINING DATA SERVICES

CCSDS 910.2-G-1

GREEN BOOK

November 1994



AUTHORITY

Issue:	Green Book, Issue 1
Date:	November 1994
Location:	Greenbelt, Maryland, USA

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 authorization of CCSDS Reports is detailed in reference [1].

This document is published and maintained by:

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

FORWARD

The Standard Terminology, Conventions, and Methodology (TCM) for Defining Data Services is a summary of, and cross-reference to, internationally-adopted standards for defining data services. This Report is the result of a study of different data service definition conventions conducted in support of the definition of CCSDS Space Link Extension services. However, the material contained herein is not limited to Space Link Extension services and may be applicable to other data service definition activities of CCSDS and its member Agencies.

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

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

Member Agencies

- British National Space Centre (BNSC)/United Kingdom.
- Canadian Space Agency (CSA)/Canada.
- Central Research Institute of Machine Building (TsNIIMash)/Russian Federation.
- Centre National d'Etudes Spatiales (CNES)/France.
- Deutsche Forschungsanstalt für Luft- und Raumfahrt e.V. (DLR)/Germany.
- European Space Agency (ESA)/Europe.
- Instituto Nacional de Pesquisas Espaciais (INPE)/Brazil.
- National Aeronautics and Space Administration (NASA HQ)/USA.
- National Space Development Agency of Japan (NASDA)/Japan.

Observer Agencies

- Australian Space Office (ASO)/Australia.
- Austrian Space Agency (ASA)/Austria.
- Belgian Science Policy Office (SPO)/Belgium.
- Centro Tecnico Aeroespacial (CTA)/Brazil.
- Chinese Academy of Space Technology (CAST)/China.
- Communications Research Laboratory (CRL)/Japan.
- Danish Space Research Institute (DSRI)/Denmark.
- European Organization for the Exploitation of Meteorological Satellites (EUMETSAT)/Europe.
- European Telecommunications Satellite Organization (EUTELSAT)/Europe.
- Hellenic National Space Committee (HNSC)/Greece.
- Indian Space Research Organization (ISRO)/India.
- Industry Canada/Communications Research Centre (CRC)/Canada.
- Institute of Space and Astronautical Science (ISAS)/Japan.
- Institute of Space Research (IKI)/Russian Federation.
- KFKI Research Institute for Particle & Nuclear Physics (KFKI)/Hungary.
- MIKOMTEK: CSIR (CSIR)/Republic of South Africa.
- Ministry of Communications (MOC)/Israel.
- National Oceanic & Atmospheric Administration (NOAA)/USA.
- Swedish Space Corporation (SSC)/Sweden.
- United States Geological Survey (USGS)/USA.

DOCUMENT CONTROL

Document	Title	Date	Status and Substantive Changes
CCSDS 910.2-G-1	Report Concerning Space Data Systems Standards: Standard Terminology, Conventions, and Methodology (TCM) for Defining Data Services, Issue 1	November 1994	Original Issue

CONTENTS

Section Page

1 INTRODUCTION 1-1

1.1 PURPOSE AND SCOPE 1-1
1.2 DOCUMENT STRUCTURE 1-1
1.3 DEFINITIONS 1-2
1.4 REFERENCES 1-9

2 OVERVIEW OF STANDARD SERVICE TERMINOLOGY AND CONVENTIONS 2-1

3 OSI SERVICE MODELING CONVENTIONS 3-1

3.1 OSI SERVICE DEFINITION CONVENTIONS 3-1
3.2 OSI BRM DEFINITIONS 3-3
3.3 APPLICATION LAYER STRUCTURE DEFINITION 3-5

4 ABSTRACT SERVICE DEFINITION CONVENTIONS 4-1

4.1 ABSTRACT MODEL 4-1
4.2 ABSTRACT SERVICE 4-4

5 RELATIONSHIP BETWEEN THE OSI AND ABSTRACT SERVICE DEFINITION CONVENTIONS 5-1

6 OSI MANAGEMENT FRAMEWORK 6-1

6.1 MANAGED OBJECTS 6-1
6.2 OSI MANAGEMENT FUNCTIONAL AREAS 6-1
6.3 OSI MANAGEMENT OPERATION 6-2

ANNEX A ACRONYMS A-1

ANNEX B EXAMPLE APPLICATION OF ASDC TO THE DELTA/ SEDS MISSION B-1

CONTENTS (concluded)

Figure Page

3-1 OSI-Service Model 3-2
3-2 Example of a Peer-to-Peer Connection-Mode Service 3-3
3-3 OSI-Service Components on Open Systems 3-4
3-4 Application Layer Components 3-6
4-1 ASDC Example - Yellow Environment 4-3
4-2 ASDC Example - Yellow System Refinement 4-4
4-3 ASDC Example - Yellow Environment Operations 4-6
5-1 Mapping of ASDC Model to Application Layer Model 5-2
B-1 Delta/SEDS Mission Elements B-2
B-2 Delta/SEDS Mission Facility Abstract-Objects B-3
B-3 Delta/SEDS Mission Agency Abstract-Objects B-4
B-4 User Agency–Provider Agency Depiction of SEDS Objects B-5

1 INTRODUCTION

1.1 PURPOSE AND SCOPE

This document summarizes International Organization for Standardization (ISO) standard Terminology, Conventions, and Methodologies (TCM) applicable to the definition of data services.

The terminology, conventions, and methodologies summarized herein provide a common vocabulary for describing systems and their interactions, from the initial high-level (abstract) conceptual level through the level at which specific technologies, protocols, and applications are applied to the development of CCSDS recommendations. This basic, common vocabulary can be used as the foundation for expressing concepts of operation and architectural specifications, leading to the definition of specific data services and protocol specifications.

The original motivation for the creation of this document was for the purpose of defining Space Link Extension (SLE) services, and appropriate aspects of the TCM have been incorporated into the Space Link Extension Service Reference Model (reference [2]). However, the TCM is not in any way limited to this purpose, and is applicable to a wide variety of data services.

It is not the purpose of this Report to describe the standards for specific protocols and services. Rather, it describes the methodologies used to create the reference models that form the context in which the various specific protocols and service were defined.

It is not the purpose of this document to duplicate or replace the definitions and conventions found in the reference documents. The material provided in this document is intended to provide enough of an overview to help the reader gain a general understanding of the material presented in the references. Study of the reference documentation is required for a complete and detailed understanding of the terminology and conventions.

1.2 DOCUMENT STRUCTURE

Subsection 1.3 provides definitions of terms and conventions used in this Report.

Section 2 provides an overview of the standard service terminology and conventions that are presented in greater detail in later sections of this Report.

Section 3 describes the Open System Interconnection (OSI) service modeling conventions: the OSI service definition conventions, the OSI Basic Reference Model, and the application layer structure definitions. The OSI modeling conventions are associated with services for communicating data among open systems.

Section 4 presents the ISO Abstract Service Definition Conventions, used for describing peer-to-peer services in distributed processing environments.

Section 5 briefly discusses the relationship between the OSI service modeling conventions and the ASDC, and describes how both sets of conventions might be applied in a complementary fashion in the definition of standards leading to real implementations.

Section 6 summarizes the OSI management framework, which provides models and terminology associated with the management of open systems.

Annex A is a glossary of acronyms.

Annex B provides an extended example of the use of the ISO Abstract Service Definition Conventions (ASDC), to describe the ground systems involved in the Delta-launched Small Expendable-tether Deployer System mission.

1.3 DEFINITIONS

1.3.1 TERMS

For the purposes of this document, the following definitions apply. Where they are explicitly defined in the reference standards, the definitions are quoted directly from those standards. For each definition, the reference to the standard in which it is originally defined is indicated by a bracketed number following the definition. The bracketed numbers correspond to numbers in the list of references (1.4).

abstract syntax: The specification of Application Layer data or application-protocol-control-information by using notation rules which are independent of the encoding techniques used to represent them. [5]

abstract-association: The relationship that exists between two abstract-ports that are bound to each other. [3]

abstract-bind-operation: A task whose successful performance binds one or more pairs of abstract-ports. [3]

abstract-error: A predefined condition that, upon occurrence during an attempted invocation of an abstract-operation, causes that abstract-operation to fail. [3]

abstract-model: The macroscopic level of ASDC, composed of abstract-objects, abstract-ports, and abstract-services. [3]

abstract-object: A functional entity (e.g., a system) that interacts with one or more other abstract-objects to provide abstract-services to, and/or use the abstract-services of, those abstract-objects. [3]

abstract-operation: A task whose performance provides all or part of an abstract-service. An abstract-operation is invoked by an abstract-object and performed by another abstract-object via the abstract-association that exists between them. [3]

abstract-port: A point at which an abstract-object interacts with another abstract-object, via an abstract-port of the same type on the other abstract-object. [3]

abstract-port-type: The common identifier for a collection of abstract-ports that perform the same set of abstract-operations. There are two varieties of abstract-port-types, symmetric and asymmetric. [3]

abstract-refinement: The process of modeling a system as one or a few high-level abstract-objects and successively decomposing each of those abstract-objects into lower-level component-abstract-objects. [3]

abstract-service: A set of capabilities that one abstract-object offers to another by means of one or more of its abstract-ports. [3]

abstract-service-provider: An abstract-object that offers an abstract-service to another abstract-object. [3]

abstract-service-user: An abstract-object that uses an abstract-service of another abstract-object. [3]

abstract-unbind-operation: A task whose performance unbinds two abstract-objects. [3]

acceptor: In a particular instance of OSI-service-procedure, an OSI-service-user that receives a deliver primitive and as a result may issue one or more submit primitives. [6]

AE-invocation: A specific utilization of part or all of the capabilities of a given application-entity in support of the communications requirements of an application-process-invocation.

NOTE – This is a specific use of the ASO-invocation concept. [8]

application-association, association: A cooperative relationship between two ASO-invocations which governs their bilateral use of the Presentation Service for communication of information and coordination of their joint operation.

NOTE – This is a specific use of the ASO-association concept. [8]

application-entity: An application element, within an application process, embodying a set of capabilities which is pertinent to OSI and which is defined for the Application Layer, that corresponds to a specific application-entity-type (without any extra capabilities being used). [5]

application-process: An element within a real open system which performs the information processing for a particular application. [5]

application-process-invocation: A specific utilization of part or all of the capabilities of a given application process in support of a specific occasion of information processing. [5]

application-service-element: A set of application-functions that provides a capability for the interworking of application entity-invocations for a specific purpose; application-service-elements are a component of application-service-objects.

NOTE – This definition refines the original definition of application-service-elements in ITU-T Rec. X.200 | ISO 7498-1. [8]

application-service-object: An active element within (or equivalent to the whole of) the application entity embodying a set of capabilities defined for the Application Layer that corresponds to a specific ASO-type (without any extra capabilities being used).

NOTE – This is a specific use of the (N)-entity concept defined in ITU-T Rec. X.200 | ISO 7498-1. [8]

ASO-association: A cooperative relationship among two or more ASO-invocations for the purpose of communication of information and the coordination of their joint operation.

NOTE – This is a specific use of the (N)-association concept. [8]

ASO-invocation: A specific utilization of part or all of the capabilities of a given ASO (without extra capabilities being used).

NOTE – This is a specific use of the (N)-entity-invocation concept defined in ITU-T Rec. X.200 | ISO 7498-1. [8]

ASO-type: A description of a class of ASOs in terms of a set of capabilities defined for the Application Layer.

NOTE – This is a specific use of the (N)-entity-type concepts defined in ITU-T Rec. X.200 | ISO 7498-1. [8]

association control service element: An ASE that provides the exclusive means for establishing and terminating all application-associations.

NOTE – The functionality of this ASE is defined in CCITT Rec. X.217 | ISO/IEC 8649. [8]

asymmetric: A variety of abstract-port-type that indicates that the two abstract-ports of the port pair invoke and perform different and complementary sets of abstract-operations. One asymmetric abstract-port is designated the consumer, and the other, the supplier. The consumer performs the set of operations invoked by the supplier, and the supplier performs the operations invoked by the consumer. [3]

asymmetrical service: An OSI-service for which the definitions of all OSI-local views are not all the same (i.e. there are several types of OSI-local view). [6]

bound: The state that exists between two abstract-ports as a result of a successful bind-operation. [3]

component-abstract-object: A lower-level abstract-object that results for the refinement of a higher-level abstract-object. [3]

confirm (primitive); acceptor.submit (primitive): A deliver primitive received by a requestor. [6]

consumer: The designation of one of the two port subtypes of an asymmetric abstract-port-type; the complement to the supplier subtype. [3]

control function: The component of an ASO that controls the interaction among the ASEs and/or ASOs within the containing ASO. [8]

deliver (primitive): An OSI-service primitive initiated by an OSI-service-provider. [6]

error-information: Information supplied by the responder as the consequence of an unsuccessful abstract-bind-operation or abstract-unbind-operation. [3]

indication (primitive); acceptor.deliver (primitive): A deliver primitive received by an acceptor. [6]

initiator: The abstract-object that issues the request to bind (abstract-bind-operation) or unbind (abstract-bind-operation). [3]

invoker: The abstract-object that issues the request to perform an abstract-operation. [3]

macro: A facility in ASN.1 to add semantic information to a collection of ASN.1 data types. [11]

managed object: The OSI management view of a resource within the OSI environment that may be managed through the use of OSI management protocols. [12]

management information base (MIB): the conceptual repository of management information within an open system. [12]

match: The condition that allows two abstract-ports to bind to each other. Symmetric abstract-ports match if they are of the same abstract-port-type. Asymmetric abstract-ports match if they are of the same abstract-port-type and one is a consumer and the other is a supplier. [3]

(N)-association: A cooperative relationship among (N)-entity-invocations. [5]

(N)-entity: An active element within an (N)-subsystem embodying a set of capabilities defined for the (N)-layer that corresponds to a specific (N)-entity-type (without any extra capabilities being used). [5]

(N)-entity-invocation: A specific utilization of part or all of the capabilities of a given (N)-entity (without any extra capabilities being used). [5]

(N)-entity-type: A description of a class of (N)-entities in terms of a set of capabilities defined for the (N)-layer. [5]

(N)-function: A part of the activity of (N)-entities. [5]

(N)-layer: A subdivision of the OSI architecture, constituted by subsystems of the same rank (N). [5]

(N)-layer management: Functions related to the management of the (N)-layer partly performed in the (N)-layer itself according to the (N)-protocol of the layer (activities such as activation and error control) and partly performed as a subset of systems management. [5]

(N)-layer operation: The monitoring and control of a single instance of communication. [12]

(N)-protocol: A set of rules and formats (semantic and syntactic) which determines the communication behavior of (N)-entities in the performance of (N)-functions. [5]

(**N)-protocol-control-information**: Information exchanged between (N)-entities, using an (N-1)-connection, to coordinate their joint operation. [5]

(N)-protocol-data-unit: A unit of data specified in an (N)-protocol and consisting of (N)-protocol-control-information and possibly (N)-user data. [5]

(N)-service: A capability of the (N)-layer and the layers beneath it, which is provided to (N+1)-entities at the boundary between the (N)-layer and the (N+1)-layer. [5]

(N)-service-access-point: the point at which (N)-services are provided by an (N)-entity to an (N+2)-entity. [5]

(N)-service-data-unit: an amount of information whose identity is preserved when transferred between peer-(N+1)-entities and which is not interpreted by the supporting (N)-entities. [5]

(N)-service-user: The user of the service provided by the (N)-service-provider.

(N)-subsystem: An element in hierarchical division of an open system which interacts directly only with elements in the next higher division or the next lower division of that open system. [5]

(N)-user-data: The data transferred between (N)-entities on behalf on the (N+1) entities for whom the (N)-entities are providing services. [5]

open system: The representation within the Reference Model of those aspects of a real open system that are pertinent to OSI. [5]

OSI-local view: the shared behaviour on an OSI-service-user and an OSI-service-provider in terms of their interactions at a service boundary. [6]

OSI-service: The capability of an OSI-service-provider to OSI-service-users at the boundary between the OSI-service-provider and the OSI-service-users.

NOTE – The OSI-service defines the external behavior of the OSI-service-provider independent of the mechanisms used to provide that behavior. (N)-layers, (N)-entities, application-service-elements, etc. are components of an OSI-service-provider. [6]

OSI-service primitive; primitive: An abstract, atomic, implementation-independent representation of an interaction between an OSI-service-user and its OSI-service-provider.

NOTE – The term “primitive” is used in some documents in place of the preferred form “OSI-service primitive”. [6]

OSI-service-procedure: Either a submit primitive together with the locally-resulting deliver primitive of primitives, if any, or a deliver primitive together with the locally-resulting submit primitive or primitives, if any, seen at an OSI-local view. [6]

OSI-service-provider: An abstract representation of the totality of those entities which provide an OSI-service to OSI-service-users. [6]

OSI-service-user: An entity in a single open system that makes use of an OSI-service.

NOTE – The OSI-service-user makes use of the OSI-service through a collection of OSI-service primitives defined for the OSI-service. [6]

parameter: An information object associated with an abstract-error. [3]

peer-(N)-entitles: Entities within the same (N)-layer. [5]

performer: The abstract-object that performs an abstract-operation. [3]

real open system: A real system which complies with the requirements of OSI standards in its communication with their real systems. [5]

real system: A set of one or more computers, the associated software, peripherals, terminals, human operators, physical processes, information transfer means, etc., that forms an autonomous whole capable of performing information processing and/or information transfer. [5]

Remote Operation Service Element: The application-service-element defined in ISO/IEC 9072. [9]

Remote Operations: (1) A concept and notation supporting the specification of interactive communication between application-entities. This includes the Remote Operation Service Element and the mapping of the notation onto the service primitives of used application-service-elements. (2) The set of bind-operations, unbind-operations and operations. [9]

request (primitive); requestor.submit (primitive): A submit primitive issued by a requestor. [6]

requestor: In a particular instance of OSI-service-procedure, an OSI-service-user that issues a submit primitive and as a result may receive one or more deliver primitives. [6]

responder: The abstract-object the performs an abstract-bind-operation or an abstract-unbind-operation. [3]

response (primitive); acceptor.submit (primitive): A submit primitive issued by an acceptor. [6]

result: An optional information object provided by the performer of a successful abstract-operation. [3]

submit (primitive): An OSI-service primitive initiated by an OSI-service-user. [6]

supplier: The designation of one of the two subtypes of an asymmetric abstract-port-type; the complement to the consumer subtype. [3]

symmetric: A characteristic of an abstract-port-type that indicates that either one of the two abstract-ports of the port pair may invoke any of the abstract-operations associated with that port pair. [3]

symmetrical service: An OSI-service for which definitions of all OSI-local views are the same (i.e. there is only one type of OSI-local view). [6]

systems management: Functions in the Application layer related to the management of various OSI resources and their status across all layers of the OSI architecture. [5]

systems management application entity (SMAE): An application-entity for the purposes of systems management communications. [5]

1.3.2 CONVENTIONS

Defined terms are identified by **bold text** in this Report.

1.4 REFERENCES

The following documents contain provisions which, through reference in this text, constitute provisions 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-6. Yellow Book. Issue 6. Washington, D.C.: CCSDS, May 1994.

[2] *Space Link Extension — Cross Support Reference Model*. Draft Recommendation for Space Data Systems Standards, CCSDS 910.4-R-1. Red Book. Issue 1. Washington, D.C.: CCSDS, November 1995.

[3] *Information Technology—Text Communication—Message-Oriented Text Interchange Systems (MOTIS)—Part 3: Abstract Service Definition Conventions*. International Standard, ISO/IEC 10021-3:1990 (E). 1st ed. Geneva: ISO, 1990. As updated by Technical Corrigendum 1, ISO/IEC 10021-3:1990/Cor. 1:1992 (E). Geneva: ISO, 1992.

[4] *Cross Support Concept — Part 1: Space Link Extension Services*. Report Concerning Space Data Systems Standards, CCSDS 910.3-G-1. Green Book. Issue 1. Washington, D.C.: CCSDS, May 1995.

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

[6] *Information Technology—Open Systems Interconnection—Basic Reference Model—Conventions for the Definition of OSI Services*. Draft International Standard, ISO/IEC DIS 10731. Geneva: ISO, 1991.

[7] Rose, Marshall T. *The Open Book: A Practical Perspective on OSI*. Englewood Cliffs, NJ: Prentice-Hall, 1990.

[8] *Information Technology—Open Systems Interconnection—Application Layer Structure*. International Standard, ISO/IEC 9545. 2nd ed. Geneva: ISO, 1994.

[9] *Information Processing Systems—Text Communication—Remote Operations—Part 1: Model, Notation and Service Definition*. International Standard, ISO/IEC 9072-1. 1st ed. Geneva: ISO, 1989.

[10] "Proposed Draft Amendment 1: Enhancements to Service Definition." ISO/IEC JTC 1/SC 21 N 6717. *Information Processing Systems—Text Communication—Remote Operations—Part 1: Model, Notation and Service Definition*. International Standard, ISO/IEC 9072-1. 1st ed. Geneva: ISO, 1992

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

[12] *Information Processing Systems—Open Systems Interconnection—Basic Reference Model—Part 4: Management Framework*. International Standard, ISO/IEC 7498-4. 1st ed. Geneva: ISO, 1989.

[13] *Information Processing Systems—Open Systems Interconnection Reference—Basic Reference Model—Part 3: Naming and Addressing*. International Standard, ISO 7498-3. 1st ed. Geneva: ISO, 1989.

2 OVERVIEW OF STANDARD SERVICE TERMINOLOGY AND CONVENTIONS

In creating Recommendations for Space Data Systems Standards, a common approach used by CCSDS is to document those Recommendations in the form of specifications of data services. These services generally take one of two forms, data communication services or application services.

Data communication services provide for the exchange of data between *two or more* service users. Over time, different data communication architectures have arisen, each with their own models, methodologies, sets of terminology, and conventions. However, one of these architectures, the Open Systems Interconnection (OSI) architecture developed by the International Organization for Standardization (ISO), has become the standard for modeling data communication systems, even though the particular services and protocols that have resulted from the OSI initiative have yet to become de-facto universal standards in their own right. In the OSI model, the users of OSI services are computer programs (or protocol entities operating on their behalf, using the layering principle of the now-famous "seven layer" model). The provider of OSI services is an abstract machine that exists in part in the same computers that host the user programs/protocol entities, and in part in the transmission media that interconnect the computers. An OSI service cannot have a single user, since its only purpose is to connect users to each other. In recognition of their wide acceptance, the OSI service definition conventions and models are the logical choice for use in modeling data communication systems. Section 3 of this Report summarizes the OSI service modeling conventions.

Application services are data-processing services that are performed by one entity on behalf of another. Unlike the OSI service, the fundamental nature of an application service is such that it has a *single* user: each user of the service in effect interacts only with the provider of the service, and not with other users who may also happen to be obtaining service from the same service provider. The service provider is not a mere intermediary for transferring data from a distant user; rather, it "does work" for the user on its own. Note that this does not imply that the application service provider cannot receive data from a third entity necessary to perform its service for the user, but it indicates the provider does perform some "value-added" processing on said data beyond merely transporting the data from a remote location.

As with the case of data communications services, many different approaches to modeling application services have been developed over the years. Unlike the case of data communication services, there is no one methodology for application services that achieved anything like the universal acceptance and application of the OSI service conventions. This is due primarily to the more complicated nature of application services, with some methodologies more suited to some classes of applications than others. The service conventions selected for description in this document are the OSI Abstract Service Definition Conventions (ASDC). The ASDC were developed as part of the Message-Oriented Text Interchange System (MOTIS) standard (reference [3]), but they are a general set of conventions suitable to a broad range of peer-to-peer interactions in a distributed processing environment. The ASDC were specifically designed to complement the OSI conventions.

Section 4 of this Report summarizes ASDC. Annex B of this Report provides an example of the use of ASDC to describe the ground systems involved in the Delta-launched Small Expendable-tether Deployer System mission.

The ASDC have been selected for use in the definition of CCSDS Space Link Extension (SLE) services. At first glance, the communication-oriented OSI conventions seem to be most suitable for the SLE services, which deal with the transfer of spacecraft data to and from ground destinations remote from the ground station (reference [4]). However, on further investigation, several aspects of the OSI model are found to be unsuitable for the cross-support environment envisioned for the provision of SLE services. For example, as stated above, in the OSI model the service provider is an abstract machine that exists in part on *all* of the systems involved in the interconnection. This is counter-intuitive in an environment where one "system" (one agency's resources) is considered the provider of services to another "system" (the user agency). Another model mismatch results from the fact that the OSI services are inherently symmetrical: the service data that is input by the sending user to the provider is the same service data that is output from the provider to the receiving user. This does not readily support cross-support scenarios such as those in which the service provider (one agency's ground station) receives a radio-frequency-modulated signal and provides CCSDS packets to the service user (another space agency's mission ground facility). The ASDC can nevertheless be used to model service in this cross-support environment, and thus it was selected for the SLE service definitions.

Selection of ASDC as the basis of SLE service modeling and definition does not mean that the OSI conventions have no role in SLE service development. Real distributed systems implementing the SLE services will need to transfer the resulting data products among themselves. The services used to effect these transfers will be precisely the data communication services that are best modeled using the OSI conventions. Section 5 describes how the ASDC and the OSI service conventions can be applied in a complementary fashion to the definition of the various services needed to create real implementations.

In the current environment of cross support among space agencies, the ability for one agency to provide service (communication or application) to another depends not only on the standardization of the service interface, but also on the ability of the two agencies' management systems to interoperate, at least to the degree necessary to coordinate the provision of service and monitor its delivery. ISO has developed a management framework for OSI systems as a precursor to the standardization of tools to allow management systems to interoperate with managed systems in an open, non-propriety manner. Although the OSI management framework was developed with OSI services (that is, data communication services) in mind, many aspects of the framework can be applied, either directly or by modification, to the management of application services. Section 6 of this Report briefly summarizes the OSI management framework.

3 OSI SERVICE MODELING CONVENTIONS

The OSI Basic Reference Model (BRM) (reference [5]) introduces the OSI Environment, which is the set of concepts, elements, services, protocols, etc., that allow communication among **open systems**. An **open system** is the representation of those aspects a **real open system** that are pertinent to interconnection with other open (real) systems. A **real open system** is a **real system** that complies with OSI standards, where a **real system** is "a set of one or more computers, the associated software, peripherals, terminals, human operators, physical processes, information transfer means, etc., that forms an autonomous whole capable of performing information processing and/or information transfer."

Implied but not explicitly stated in the OSI documentation is the fact that real (open) systems exist to support applications. The aspect of the application that is of interest with respect to OSI is the **application process**, which is the "element within a **real open system** which performs the information processing for a particular application."

Although OSI facilitates the interconnection of open systems, and thus the interactions of application processes, OSI is not concerned with the specification of those application processes. According to reference [5]:

> "OSI is concerned with the exchange of information between open systems (and not the internal functions of each individual real open system)." (4.2.7)
>
> "OSI is concerned only with the interconnection of systems. All other aspects of systems which are not related to interconnection are outside the scope of OSI." (4.2.9)
>
> "OSI is concerned not only with the transfer of information between systems, i.e., transmission, but also with their capability to interwork to achieve a common (distributed) task. In other words, OSI is concerned with the interconnection aspects of cooperation between systems, which is implied by the expression 'open systems interconnection.'" (4.2.10)

The OSI documents address a range of topics regarding OSI services. For the purposes of the TCM, these topics fall into four categories: the OSI service definition conventions, the OSI BRM definitions, the application layer structure definitions, and the OSI management framework.

3.1 OSI SERVICE DEFINITION CONVENTIONS

Since OSI is concerned with providing interconnections between (or among) application processes, it follows that, from the OSI perspective, *service* is defined in terms of providing the connection (and not, for example, in terms of what one application process on one open system does to support the application process on another open system). The formal definition of **OSI-service** and related terms is not found in the BRM proper but rather in the OSI service definitions convention standard (reference [6]). Figure 3-1, which is a copy of figure 1 in reference [6], illustrates the OSI service model. As shown, three **OSI-service-**

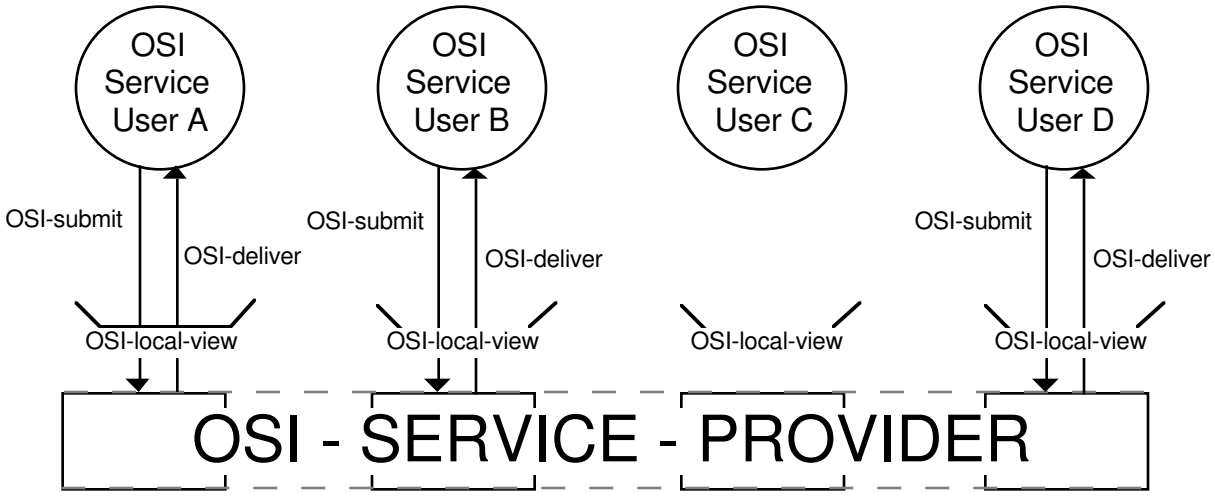


Figure 3-1: OSI-Service Model

users exchange **OSI-service-primitives** (**submit** primitives and **deliver** primitives) with the **OSI-service-provider.** A **submit** primitive is an **OSI-service-primitive** initiated by an **OSI-service-user**. A **deliver** primitive is an **OSI-service-primitive** initiated by an **OSI-service-provider**. The shared behavior of an **OSI-service-user** and an **OSI-service-provider** in terms of their interactions at the service boundary is called the **OSI-local-view**. When an **OSI-service** is defined such that all of its **OSI-local-views** are the same (i.e., there is only one type of **OSI-local-view** for the service), the **OSI-service** is said to be a **symmetrical-service**. When an **OSI-service** is defined such that all of its **OSI-local-views** are the not the same (i.e., there are several types of **OSI-local-view**s for the service), the **OSI-service** is said to be an **asymmetrical-service**.

At an **OSI-local-view,** related **submit** and **deliver** primitives form **OSI-service-procedures**. An **OSI-service-procedure** is "either a **submit** primitive together with the locally-resulting **deliver** primitive or primitives, if any, or a **deliver** primitive together with the locally-resulting **submit** primitive or primitives, if any, seen at an **OSI-local-view**."

The OSI service model provides limited terminology for describing the roles of the two (or more) users of an OSI service. These roles are **requestor** and **acceptor**. In the context of a particular instance of **OSI-service-procedure**, a **requestor** is "an **OSI-service-user** that issues a **submit** primitive and as a result may receive one or more **deliver** primitives." In the context of a particular instance of **OSI-service-procedure**, an **acceptor** is "an **OSI-service-user** that receives a **deliver** primitive and as a result may issue one or more **submit** primitives." Names are defined for the **submit** and **deliver** primitives used by the **requestor** and **acceptor**: **request**, **indication**, **response**, and **confirm**.

- A **request** is a **submit** primitive issued by a **requestor**.

- An **indication** is a **deliver** primitive received by an **acceptor**. The **indication** is related (in a way that is specific to the **OSI-service**) to the **request** issued by the **requestor**.

- A **response** is a **submit** primitive issued by an **acceptor** as a result of the **indication** received.

- A **confirm** is a **deliver** primitive received by the **requestor**. The **confirm** is related (in a way that is specific to the **OSI-service**) to the **response** issued by the **acceptor**.

For connectionless-mode OSI-services, only the request and indication have meaning, since a connectionless-mode OSI service does not maintain the state (that is, track the relationship) between data flowing from **requestor** to **acceptor** and data flowing from **acceptor** to **requestor**. Figure 3-2, derived from figure 3 of reference [6], illustrates the user roles and primitives present for a connection-mode **OSI-service**.

Figure 3-2: Example of a Peer-to-Peer Connection-Mode Service

3.2 OSI BRM DEFINITIONS

The OSI service definition conventions provide a general frame of reference for describing the roles and relationships between a user of an interconnection service and a provider of such a service. The OSI-BRM builds upon these ISO service definition conventions to address modeling of interconnections between open systems. In an open systems environment the **OSI-service-provider** is realized through the interaction of service entities residing in the open systems. Furthermore, the OSI-BRM employs the concept of a layered architecture. In this architecture, each open system is viewed as being logically composed of an ordered set of **(N)-subsystems**, where *N* denotes the number of the layer (the **(N)-layer**) in which the logical subsystem exists. Figure 3-3, adapted from figure 2.2 in reference [7], illustrates the realization of an OSI-service-provider in the layered open system environment, using key terminology.

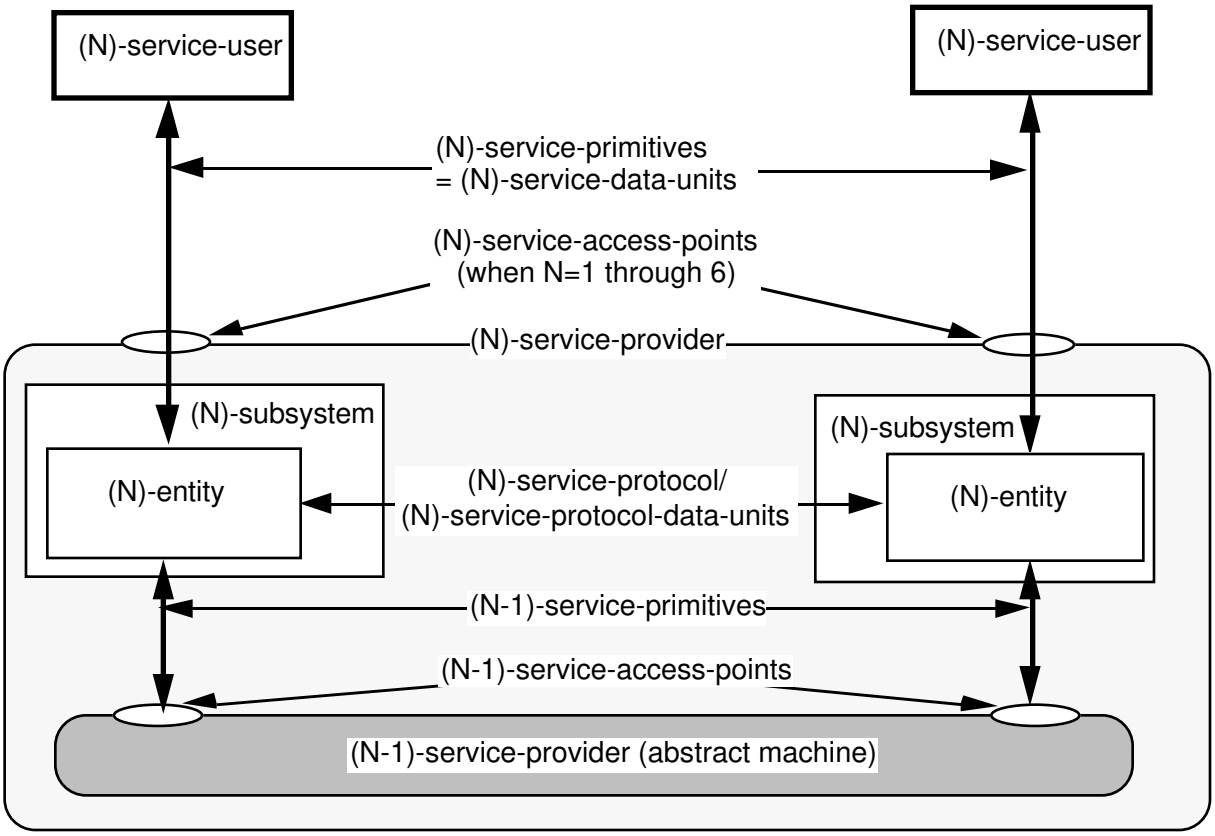


Figure 3-3: OSI-Service Components on Open Systems

As shown in figure 3-3, one or more **(N)-entities** exist within each **(N)-subsystem**, where the **(N)-entity** is an active element that embodies a set of capabilities defined for the **(N)-layer**. **Peer-(N)-entities** within each **(N)-layer** interact according to **(N)-protocols** to form the **(N)-service-provider**, which is the **OSI-service-provider** at the **(N)-layer**. The **peer-(N)-entities** interact via the exchange of **(N)-protocol-data-units**. The primitives exchanged between the **(N)-service-user** and its local **(N)-entity** contain **(N)-user-data** in the form of **(N)-service-data-units**.[1]

In the OSI-BRM layered architecture, the exchange of **(N)-protocol-data-units** between the **(N)-entities** is carried out by employing the **OSI-service** of the layer below. With respect to an **(N-1)-service-provider**, the **(N)-entity** is also the **(N-1)-service-user**.

For the OSI physical through presentation layer services (i.e., N = 1 through 6), the **(N)-entity** provides the **(N)-service** to the **(N)-service-user** at the **(N)-service-access-point**. The concept of service access point does not apply to services provided by the application layer.

[1] In addition to the (N)-user-data, the service primitives contain information needed by the (N)-entity to properly execute the service. In previous versions of ISO/OSI documentation this other information was formally named interface-control-information, but that term appears to have been discarded in the most recent versions of documents. The topic of the other information is no longer addressed in the OSI-BRM.

In addition to defining how layered services are realized on distributed open systems, the BRM defines a specific set of layers, the famous seven layers, and defines the functionality of each of those layers.

3.3 APPLICATION LAYER STRUCTURE DEFINITION

The application layer is the layer at which the OSI Environment intersects the **application-process**. The intersection is realized as one or more **application-entities**. **Application-entity** is defined in reference [5] as "an active element, within an **application process**, embodying a set of capabilities which is pertinent to OSI and which is defined for the Application Layer, that corresponds to a specific **application-entity-type** (without any extra capabilities being used)." Note that because the **application-entity** resides within the **application-process**, there is no concept of an application service access point, because such a service access point would be internal to the **application-process**. That is, the mechanism by which the service primitives are exchanged between the **application-entity** and the remainder of the **application-process** is a local matter.

According to reference [8], "when communication is required between two **application-entities** (**AE**s) to meet the needs of an application, one or more **application-associations** are established between **AE-invocations** of the **AE**s." "An **application-association** is a cooperative relationship between two **AE-invocations** for the purpose of communication of information and coordination of their joint operation."[2] The actual communication between the **AE-invocations** is carried out, of course, by a presentation service underlying the **AE**s. The extended application layer structure standard (reference [8]) provides terminology and conventions for describing the internal composition of **AE**s in terms of **application-service-objects** (**ASO**s). An **ASO** is "an active element within (or equivalent to the whole of) the application-entity embodying a set of capabilities defined for the Application Layer that corresponds to a specific **ASO-type** (without any extra capabilities being used)". According to reference [8], "an **ASO** is a composition of:

a) one or more **ASE**s [**application-service-elements**] and a **CF** [**control-function**], or
b) one or more **ASO**s and a **CF**, or
c) one or more **ASE**s and one or more **ASO**s and a **CF**."

An **ASE** is an indivisible combination of application communication functions that is distinguished for purposes of OSI service and protocol specification. Thus **ASE**s are the building blocks of **AE**s. Much of the ISO standardization activity at the application layer deals with defining **ASE**s of general use, such as the **Association Control Service Element** and the **Remote Operations Service Element**. However, generally at least several such **ASE**s are needed to provide all of the application communication functions required of an **AE**, and it is often convenient (for reasons of organization or reuse) to combine the **ASE**s into intermediate groupings. The **ASO**s are those intermediate groupings. Combination of

[2]In general, the definitions for application layer terminology are taken from [5], the extended and updated replacement for [8]. However, with respect to application-association, [8] provides a more straightforward definition. The definition in [5] is technically the same as in [8], but is phrased in terms of ASOs and in general is less intuitive.

ASOs and **ASE**s to form new **ASO**s may be performed recursively to an arbitrary degree. The **CF** of an **ASO** provides the necessary integration to make the component **ASO**s and **ASE**s operate as a single functional entity. Since the **AE** is the ultimate combination of **ASO**s and **ASE**s within an **application-process**, the **AE** itself is an **ASO**.

Figure 3-4 illustrates the relationships between the various components at the application layer, and the relationship between the application layer and the services provided by the underlying layers. In this example each **AE** is an **ASO** comprising an **ASE**, a **CF**, and a lower-level **ASO** which itself comprises two **ASE**s and a **CF**. Note that the presentation service provider is recursively constructed of the Presentation-entities interoperating through an underlying session service, which is itself recursively constructed of Session-entities interoperating through an underlying transport service, etc., according to the layered service model.

When communication is required between two or more **ASO-invocations** (**ASOI**s), one or more **ASO-associations** are established between the **ASOI**s. According to reference [8], "an **ASO-association** is a cooperative relationship between two or more **AE-invocations** for the purpose of communication of information and coordination of their joint operation." When an **ASO** is an **AE** itself, each corresponding **ASO-association** is known as an **application-association**.

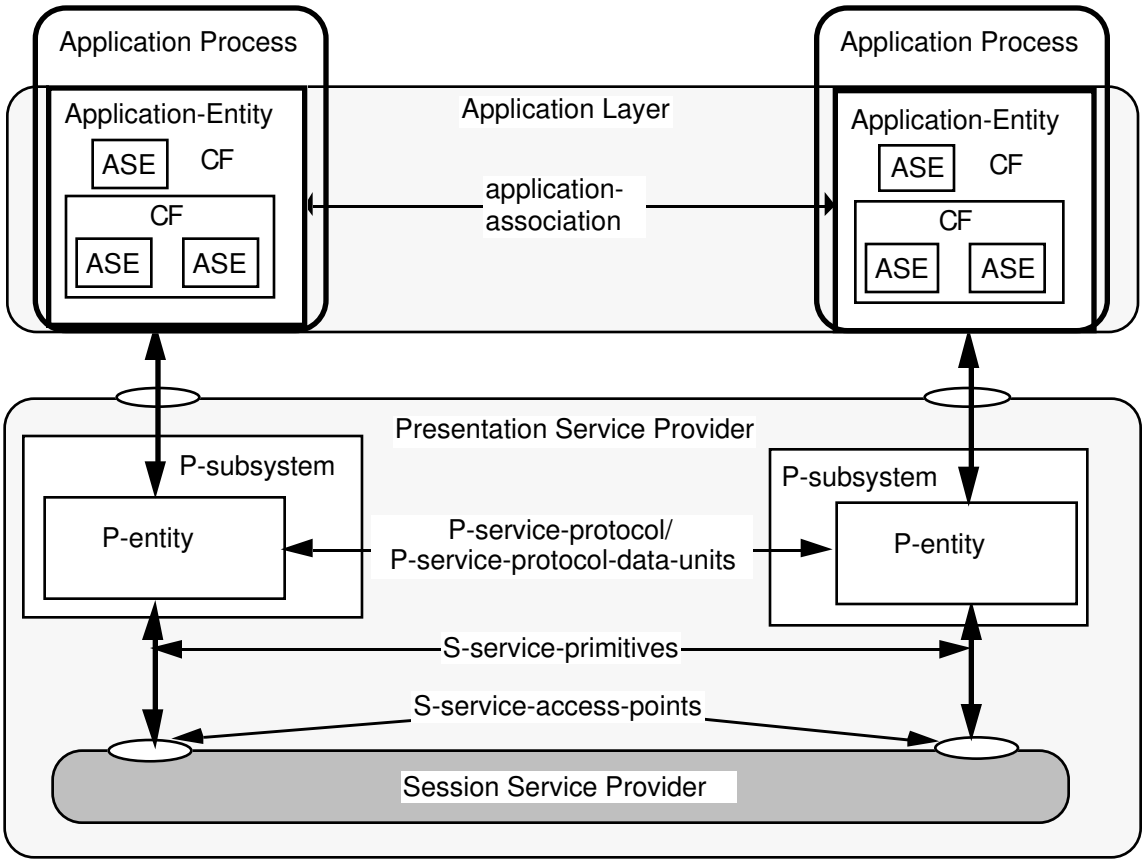


Figure 3-4: Application Layer Components

4 ABSTRACT SERVICE DEFINITION CONVENTIONS

The Abstract Service Definition Conventions (ASDC) (reference [3]) were devised for the purpose of describing and specifying complex distributed information processing systems in abstract, rather than concrete, terms. Although developed as part of the Message-Oriented Text Interchange Systems standardization effort, ASDC are applicable beyond that domain.

Almost every term in the ASDC lexicon is prefixed with the word *abstract*, e.g. **abstract-object**, **abstract-port**, **abstract-service**, and **abstract-operation**. The relentless use of *abstract* is not primarily intended to remind the reader that what is being described is an abstract version of the service, although that is certainly a byproduct of the terminology. Rather, the main reason for prefixing the terms with abstract is that almost all (if not all) of the ASDC terms are generalizations (abstractions) of the notation for Remote Operations Service (ROS) (references [9] and [10]). Although ASDC notation is based on ROS notation, the ASDC standard emphasizes that no assumptions are made about the interconnections that will eventually be used to realize the abstract services and systems being described using ASDC. That is, the interconnections may or may not be OSI, and even if they are OSI, they do not have to use ROS. However, since the ASDC elements are almost one-for-one abstractions of ROS elements, it is, according to reference [3], "trivial" to realize ASDC-defined abstract services using ROS.

The abstract service definition conventions are described at two scales. The "macroscopic level" is described in terms of the **abstract model**. The "microscopic level" is described in terms of the **abstract service**.

4.1 ABSTRACT MODEL

The abstract model is composed of abstract-objects, abstract-ports, and abstract-services. The abstract model employs the concept of abstract-refinement.

4.1.1 ABSTRACT-OBJECT

An **abstract-object** is a functional entity (e.g., a system) that interacts with one or more other **abstract-objects**. An **abstract-object** possesses one or more **abstract-ports**.

The ASDC standard provides an Abstract Syntax Notation One (ASN.1) **macro** for specifying **abstract-object**s. ASN.1 is an abstract syntax language created by ISO for use in describing arbitrarily-complex data types in a machine-independent fashion (reference [11]). An ASN.1 macro is used to add semantic information to a collection of ASN.1 data types. In ASDC, macros are used to organize the data types associated with the various elements of the **abstract model**.

4.1.2 ABSTRACT PORT

An **abstract-port** is a point at which an **abstract-object** interacts with another **abstract-object**. **Abstract-ports** are of different types which determine the kinds of interaction they enable. An **abstract-port type** is defined in terms of the **abstract-operations** that are offered via that **abstract-port type**. There are two varieties of **abstract-port types**:

- **Symmetric**: Each **abstract-port** offers all of the **abstract-operations** associated with the **abstract-port type**;

- **Asymmetric**: Each **abstract-port** is either a **consumer** or **supplier**, each of which invokes different **abstract-operations**. The ASDC standard points out that while the terms **consumer** and **supplier** may seem to imply certain roles, in ASDC they actually have no semantic significance beyond differentiating between the two **abstract-ports**. They could have just as easily been named "X-port" and "not-X-port".

Two **abstract-objects** interact only when one **abstract-port** in one **abstract-object** is **bound** to (in contact with) an **abstract-port** in the other **abstract-object**. When two **abstract-ports** are **bound**, an **abstract-association** exists between them. **Abstract-port**s can be **bound** only if they **match**. **Symmetric abstract-port**s **match** if they are of the same **abstract-port type**. **Asymmetric abstract-port**s **match** if they are of the same **abstract-port type** and one is a **consumer** and the other a **supplier**.

The ASDC standard provides an ASN.1 **macro** for specifying **abstract-ports**.

4.1.3 ABSTRACT-SERVICE

An **abstract-service** is a set of capabilities that one **abstract-object** offers to another by means of one or more of its **abstract-port**s. An **abstract-object** has one of two roles with respect to a particular **abstract-service**. An **abstract-object** that offers the **abstract-service** is called the **abstract-service-provider**. An **abstract-object** that uses the **abstract-service** is called the **abstract-service-user**.

The ASDC standard provides an ASN.1 **macro** for specifying **abstract-services**.

4.1.4 ABSTRACT-REFINEMENT

The process of modeling a system as one or a few high-level **abstract-object**s and successively decomposing each of those **abstract-object**s into multiple lower-level **abstract-object**s (called **component abstract-objects**) is known as **abstract-refinement**. Abstract-**refinement** can be performed recursively to whatever level of detail necessary to support the realization of the elements of the abstract model as concrete systems, services, and protocols.

The ASDC standard provides an ASN.1 **macro** for specifying **abstract-refinements**.

4.1.5 EXAMPLE OF AN ABSTRACT MODEL

A graphical example of an abstract model is useful at this point. Figure 4-1, adapted from figure A.1 in reference [3], illustrates the Yellow Environment. Figure 4-1 shows the Yellow Environment **abstract-object** as a composition of three **abstract-objects**; two Yellow Users, which are users of the yellow-service; and the Yellow System, which provides the yellow-service. The Yellow Users and Yellow System interact via yellow-use **abstract-ports**, with the Yellow Users having **consumer abstract-ports** and the Yellow System having **supplier abstract-ports**. Although reference [3] defines no special meaning for "environment" **abstract-objects**, it is useful to assign to them the special property of serving as context **abstract-objects**, containing all **abstract-objects** of interest. An environment **abstract-object** should be a closed system, i.e., no **abstract-ports** on its boundary.

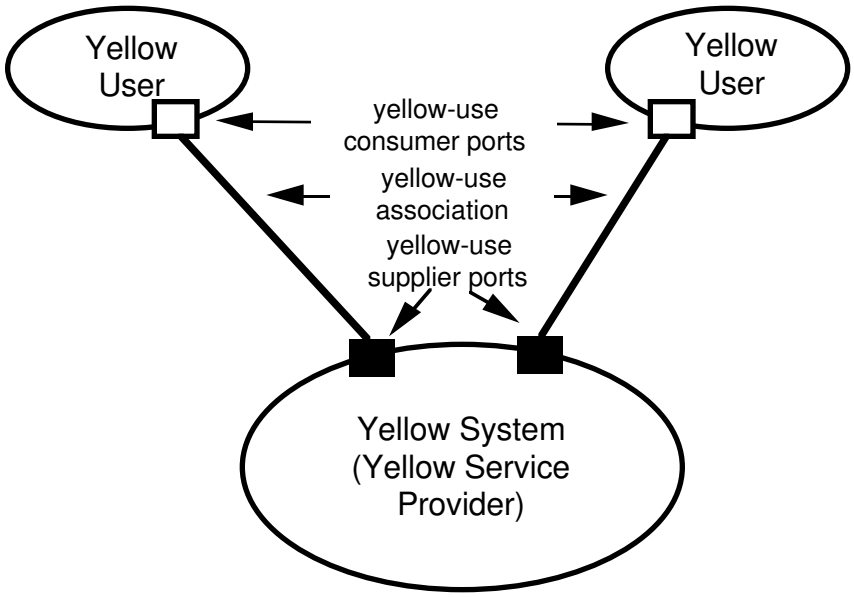


Figure 4-1: ASDC Example—Yellow Environment

Figure 4-2, adapted from figure A.3 in reference [3], is a refinement of the Yellow System in figure 4-1. The single Yellow System is refined to four component **abstract-objects**, two Yellow-Use Agents, a Green System, and a Green Archive.[3] The Green System provides the green-service to the Yellow-Use Agents and the Green Archive via the green-use **abstract-ports**. The Yellow-Use Agents are users of the green-service and providers of the yellow-service. In addition to being a user of the green-service, the Green Archive also interacts with the Green System via green-retrieval **abstract-ports**.

[3]The ASDC specification (reference [5]) labels the Green Archive object the Green Manager, and the associated ports the green-management ports in figure A.3. In order to avoid confusion between this example and the modeling of CCSDS management functions, the names have been changed.

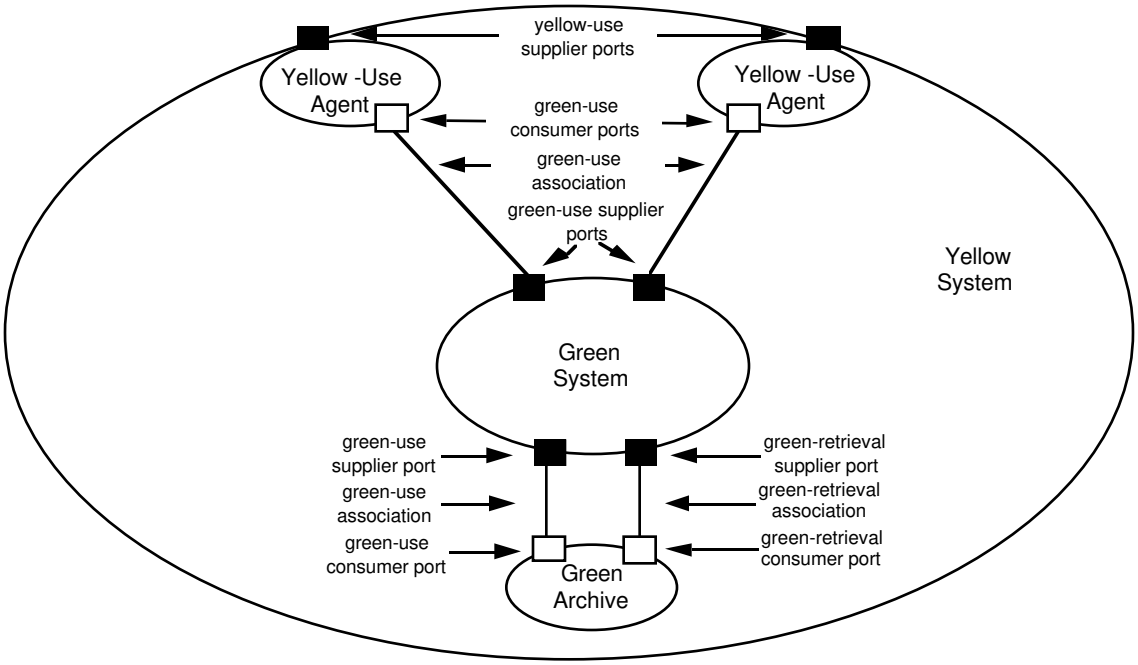


Figure 4-2: ASDC Example—Yellow System Refinement

4.2 ABSTRACT SERVICE

The specification of an **abstract-service** defines how a distributed information processing task is initiated, controlled, and terminated. An **abstract-service** is defined in terms of its **abstract-bind-operations**, **abstract-unbind-operations**, **abstract-operations**, and **abstract-errors**.

4.2.1 ABSTRACT-BIND-OPERATION

An **abstract-bind-operation** is a task whose successful performance binds one or more pairs of **abstract-ports**. Each of the two **abstract-objects** involved in an **abstract-bind-operation** assumes one of two roles. The **initiator** is the **abstract-object** that issues the request to bind. The **responder** is the **abstract-object** that performs the abstract bind.

For the binding of asymmetric ports, the ability to initiate an abstract-bind is prescribed by the definition of the abstract-port type. Initiation may be constrained to the **consumer** or **supplier** port, or permitted for either.

An **abstract-bind-operation** succeeds if it is carried out in full and fails otherwise. An **abstract-bind-operation** may (but need not) require that the **responder** apprise the **initiator** of success or failure. If the bind is successful, the **abstract-bind-operation** may (but need not) require that the **responder** provide the **initiator** with an information object called **result**. If the bind fails, the **abstract-bind-operation** may (but need not) require that the **responder** provide the **initiator** with an information object called **error information**.

The ASDC standard provides an ASN.1 **macro** for specifying **abstract-bind-operations**.

4.2.2 ABSTRACT-UNBIND-OPERATION

An **abstract-unbind-operation** is a task whose performance, *successful or not,* unbinds two **abstract-ports**. The **initiator** of the **abstract-unbind-operation** must be the same **abstract-object** that initiated the original binding. The **responder** is the **abstract-object** that performs the abstract unbind.

An **abstract-unbind-operation** may (but need not) require that the **responder** apprise the **initiator** of success or failure. If the unbind is successful, the **abstract-unbind-operation** may (but need not) require that the **responder** provide the **initiator** with an information object called **result**. If the unbind fails, the **abstract-unbind-operation** may (but need not) require that the **responder** provide the **initiator** with an information object called **error information**.

The ASDC standard provides an ASN.1 **macro** for specifying **abstract-unbind-operations.**

4.2.3 ABSTRACT-OPERATION

An **abstract-operation** is a task whose performance provides all or part of the **abstract-service**. Each of the two objects involved in an **abstract-operation** assumes one of two roles. The **invoker** of the **abstract-operation** is the **abstract-object** that issues the request to perform the task. The **performer** is the **abstract-object** that performs the **abstract-operation**. For **symmetric abstract-ports**, invocation may be from either **abstract-port**. For **asymmetric abstract-ports**, the role of **invoker** is prescribed to either the **consumer** or **supplier** in the definition of the **abstract-port type.**

An **abstract-operation** succeeds if it is carried out in full and fails when it encounters an **abstract-error**. **Abstract-errors** are prescribed for each **abstract-operation**.

An **abstract-operation** may (but need not) require that the **performer** apprise the **invoker** of success or failure. If the **abstract-operation** is successful, the **abstract-operation** may (but need not) require that the **performer** provide the **invoker** with an information object called **result**. If the **abstract-operation** fails, the **abstract-unbind-operation** may (but need not) require that the **performer** provide the **invoker** with the identity of the **abstract-error** and an information object called the error **parameter**.

The ASDC standard specifies that the ASN.1 **macro** for **abstract-operations** is the same as the Operation **macro** found in reference [9].

Figure 4-3 is an enhanced version of figure 4-1, showing the addition of **abstract-bind-operation**, **abstract-unbind-operation**, and **abstract-operation** information to the basic Yellow Environment diagram. As shown in the figure, the yellow-use **consumer abstract-ports** are assigned the role of **initiator** of the yellow-use **abstract-bind-operation** and **abstract-unbind-operation**, and **invoker** of the yellow-operation-1 **abstract-operation**. Implicit in this designation is the complementary designation of the yellow-use **supplier abstract-ports** assigned the role of **responder** to the yellow-use **abstract-bind-operation** and **abstract-unbind-operation**, and **performer** of the yellow-operation-1 **abstract-operation**.

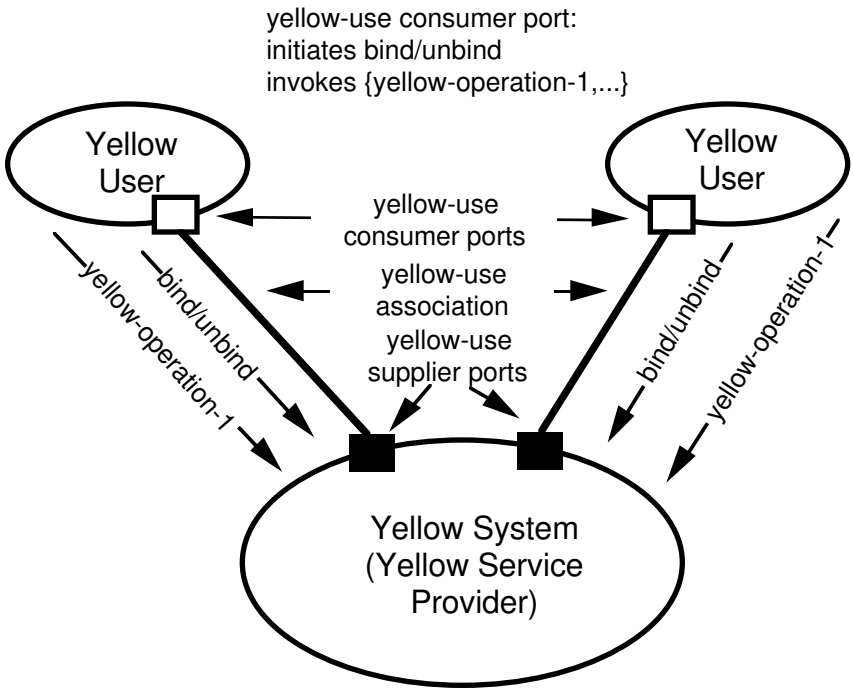


Figure 4-3: ASDC Example—Yellow Environment Operations

5 RELATIONSHIP BETWEEN THE OSI AND ABSTRACT SERVICE DEFINITION CONVENTIONS

Sections 3 and 4 summarize two sets of terminology and conventions, OSI service modeling conventions and ASDC, respectively. This section discusses the different areas of applicability of these two sets, and describes how both sets might be applied in the definition of standards leading to real implementations.

The OSI service modeling conventions are targeted at the definition of communication services that connect the users of said services. In the OSI model, the service provider is an abstract machine that resides partially in every open system in the environment. If the purpose of a particular standardization activity is to develop a data communication standard, the OSI service modeling conventions are appropriate.

The ASDC, on the other hand, are targeted at the definition of peer-to-peer interactions in a distributed processing environment. In the ASDC model, the service provider is an abstract-object which may be and is often realized as a single system. This case seems particularly appropriate to the CCSDS notion of cross support, in which one space agency performs services on behalf of another. It is the ability of the ASDC to model this kind of situation that resulted in its selection for use in defining CCSDS Space Link Extension Services (references [2] and [4].

The ASDC concentrate on the **abstract-operations** between **abstract-objects**. The "model" for communication in ASDC is implied and almost non-existent: abstract-ports are bound and unbound, and abstract-operations are effected, without reference to the ugly details of how those things are accomplished. Of course, those communication details cannot be ignored when the **abstract-objects** are realized as real applications operating on real systems. The **abstract-objects** must be mapped onto **application processes**, **AE**s, **ASO**s, and **ASE**s. This mapping is performed by defining a pair of **application processes** for each pair of **abstract-objects** that share **abstract-associations**. The **application processes** are defined in such a way that the **abstract-operations** across the **abstract-association** are supported by the **application-association** formed between the **AE**s contained within the **application processes**. Figure 5-1 illustrates the mapping among the **abstract-objects**, **abstract-associations**, **application processes**, **AE**s, and **application-associations**.

In defining the **AE**s, the preference is to select existing **AE**s that have already been defined by ISO or other standardization body. If there is no pre-existing **AE** that fits the requirements of the cross-support-abstract-association, then the next-most-desirable approach is to construct the **AE** from already-defined **ASO**s. If no suitable predefined **ASO**s exist, the approach is to construct the **ASO**s from defined **ASE**s. Simply put, existing application elements are used to the maximum extent possible, at the highest level of composition possible, before an attempt is made to create a CCSDS-unique application element.

The final step in the process is to define the layered communication services supporting the application layer. Again, the preference is to select services, protocols, and profiles standardized by ISO or other international bodies, and create communication services only as a last resort if no suitable service/protocol exists. If any such CCSDS-unique communication services are created, they would be created in accordance with the ISO service modeling conventions.

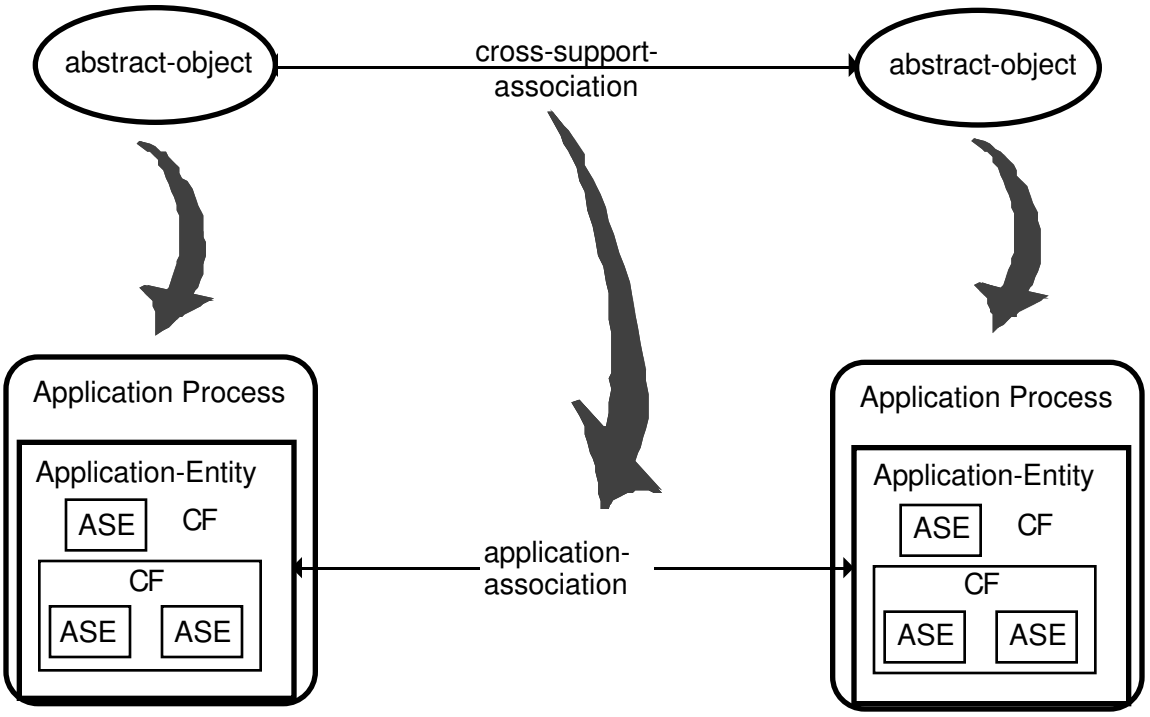


Figure 5-1: Mapping of ASDC Model to Application Layer Model

6 OSI MANAGEMENT FRAMEWORK

The OSI Management Framework (reference [12], part 4 of the OSI BRM) contains general terminology and models for describing management of open systems. The information contained in the Management Framework addresses topics that are appropriate to the management of both application services and data communication services (that is, OSI services). This section summarizes those concepts contained in the OSI Management Framework that appear to be most relevant to the management of Space Link Extension services and cross-support services in general.

Beyond the OSI Management Framework, ISO has promulgated numerous standards regarding OSI management, such as those for the Common Management Information Service (CMIS) and its associated Common Management Information Protocol (CMIP). These standards are currently beyond the scope of this Report.

6.1 MANAGED OBJECTS

A basic concept in the OSI Management Framework is that of the **managed object**, which is the OSI Management view of a resource within the OSI Environment that may be managed through the use of OSI management protocols. Note that the managed object is a view of a resource, not the resource itself. This is important from the perspective of CCSDS cross support, because it permits management interactions to deal with something other than the actual physical resources of a cross-supporting entity (e.g., space agency). For example, the managed objects created for cross-support management can be defined in terms of the services that are being provided and the parameters (attributes) needed to support those services, thus preserving the sovereignty of each agency over its physical resources. The collection of all **managed objects** within a given system constitutes the **management information base** (**MIB**) for that system. The **MIB** is the conceptual repository of management information within an open system.

6.2 OSI MANAGEMENT FUNCTIONAL AREAS

The Management Framework organizes the requirements for ISO management into five functional areas: fault management, accounting management, configuration management, performance management, and security management.

The following definitions of the functional areas are quoted from reference [12]:

> Fault management encompasses fault detection, isolation, and correction of abnormal operation in the OSI environment.
>
> Accounting management enables charges to be established for the use of resources in the OSI environment, and for costs to be identified for the use of those resources.

Configuration management identifies, exercises control over, collects data from and provides data to open systems for the purpose of preparing for, initialising, starting, providing the continuous operation of, and terminating interconnection services.

Performance management enables the behaviour of resources in the OSI environment and the effectiveness of communication activities to be evaluated.

Security management supports the application of security policies by means of functions which include

a) the creation, deletion, and control of security services and mechanisms;

b) the distribution of security-relevant information, and;

c) the reporting of security-relevant events.

6.3 OSI MANAGEMENT OPERATION

The OSI Management Framework defines three approaches to OSI management: **systems management**, **(N)-layer management**, and **(N)-layer operation**.

Systems management provides mechanisms for the monitoring, control, and coordination of all managed objects within open systems. **Systems management** can be used to manage any managed objects within a managed system. **Systems management** is carried out through a pair of communicating **systems management application-entities** (SMAEs), one of which resides on the managing system and the other on the managed system. Communication between the SMAEs is accomplished using a general management protocol. In the Space Link Extension (SLE) service environment, systems management is the basic notion underlying the management relationship between the Space Link Extension Utilization Management function of the Mission and the Complex Management function of the service complex (reference [2]).

(N)-layer management provides mechanisms for the monitoring, control, and coordination of **managed objects** within the **(N)-layer**, through the use of special-purpose management protocols unique to the **(N)-layer**. **(N)-layer management** is available for systems that do not employ systems management. A typical example of the use of (N)-layer management is the case of a simple network-layer router which does not need higher-layer protocols or application processes to perform its functions. Rather than implementing an upper stack and an SMAE purely for the purpose of allowing the router to exchange management information through standard **system management** protocols, the router could instead use a special network-layer management protocol that rides directly on the network protocol. The manager of the router would use that protocol to download routing tables, retrieve status, etc. In the SLE service environment, there is no equivalent to **(N)-layer management**, since all managed objects in all layers are accessible via the equivalent of **systems management**.

(N)-layer operation provides the mechanisms for the monitoring and control of a single instance of communication. For example, a retransmission-based protocol such as the Transmission Control Protocol (TCP) has the receiver control the flow of data from the sender through parameters contained within TCP itself. Unlike **(N)-layer management, (N)-layer operation** can only affect the operation of the single instance of communication (e.g., connection) with which it is associated. In the SLE service environment, **(N)-layer operation** is incorporated in the service data interface between the service provider and the service user.

ANNEX A

ACRONYMS

This Annex expands the acronyms used throughout this Report.

AE	application-entity
ASDC	Abstract Service Definition Conventions
ASE	application-service-element
ASN.1	Abstract Syntax Notation One
ASO	application-service-object
ASOI	application service object invocation
BRM	Basic Reference Model
CCSDS	Consultative Committee for Space Data Systems
CF	control function
ISO	International Organization for Standardization
MIB	management information base
MOTIS	Message-Oriented Text Interchange System
OSI	Open System Interconnection
ROS	Remote Operations Service
SLE	Space Link Extension
SMAE	systems management application entity
TCM	terminology, conventions, and methodology
TCP	Transmission Control Protocol

ANNEX B

EXAMPLE APPLICATION OF ASDC TO THE DELTA/SEDS MISSION

This annex provides an example of the use of the ISO Abstract Service Definition Conventions (ASDC) to describe the ground systems involved in the Delta-launched Small Expendable-tether Deployer System (Delta/SEDS) mission. The refinement of the Delta/SEDS environment presented here is only one of many possible refinements. The example provided here is informal: abstract-objects, abstract-ports, and abstract-associations are presented graphically, and no ASN.1 macros are provided. The purpose of this exercise is to convey the notions of ASDC, and not to provide a complete specification.

Delta/SEDS was a 14-minute experiment involving deployment at the end of a 12-mile tether. It flew in late March 1993. The Delta/SEDS mission is an example of a cross-support mission, where CNES provides ground station, orbit determination, and communication cross support. Figure B-1 is a reproduction of the vugraph "Mission DELTA/SEDS - 2" taken from the CNES presentation to the P3CG meeting at Darmstadt on 8 March 1993. It shows facilities at four locations: the CNES ground station at Hartebeesthoek; the CNES network control center (COR) and orbit determination center (COO) in Toulouse, France; communication facilities at the Goddard Space Flight Center (GSFC) in Greenbelt, Maryland; and Hangar AE and a flight dynamics facility at Merritt Island, Florida.[4] Hangar AE serves as the end user for the Delta/SEDS mission.

The first step in applying ASDC concepts to the Delta/SEDS environment is to abstract the elements of figure B-1 into ASDC objects. Figure B-2 shows a level of abstraction in which the key facilities involved in the mission are depicted as **abstract-objects**: the Hartebeesthoek Ground Station, COR, COO, Hangar AE, MILA Telemetry Processing, and Flight Dynamics. These **abstract-objects** are shown in heavier outline, and the **abstract-associations** between them are shown in heavier lines also. The **abstract-associations** connect the various pairs of **abstract-ports** on the **abstract-objects**. The **abstract-ports** are depicted as heavier-line squares. Note that since the abstract-associations are logical connections, they do not follow the routing through the intermediate message switch in Toulouse. Note also that there are no standard graphical conventions associated with ASDC: the ones used in this example are strictly local to this annex.

[4] The flight dynamics facility may have actually been located at GSFC and not at Merritt Island, but for purposes of this example it is not important.

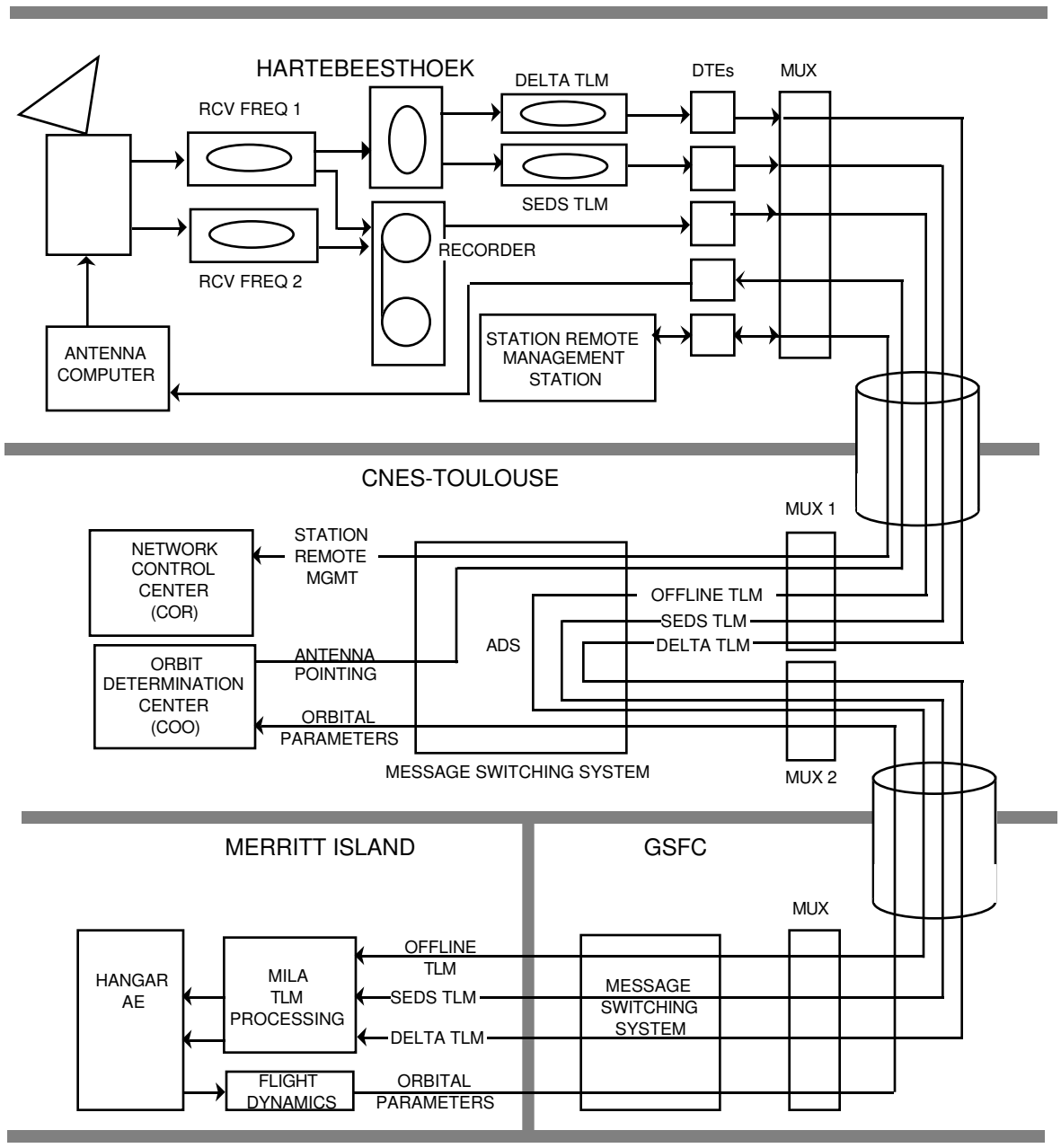


Figure B-1: Delta/SEDS Mission Elements

Figure B-3 shows another level of abstraction in which the facility-level **abstract-objects** are collected into two agency-level **abstract-objects**, CNES and NASA. The agency objects are outlined in heaviest lines, as are the **abstract-ports** on those objects and the **abstract-associations** that connect them. The message switches and multiplexers have been removed from the figure to simplify it. Note that Station Remote Management and Antenna Pointing **abstract-ports** and **abstract-associations** are still depicted in the medium-weight line weight, because they are internal to CNES.

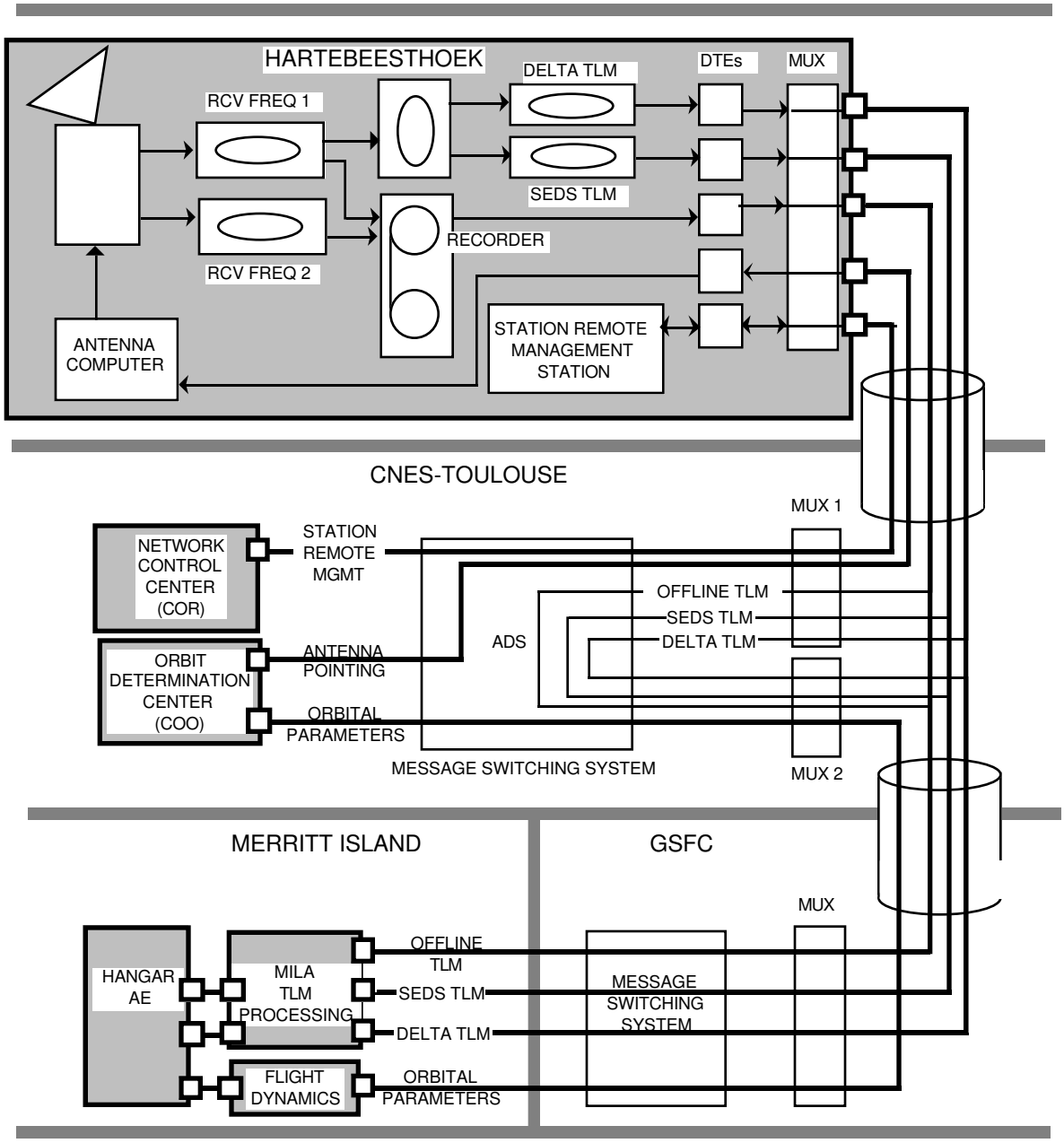


Figure B-2: Delta/SEDS Mission Facility Abstract-Objects

Likewise the abstract-ports and abstract-associations between the Hangar AE, MILA Telemetry Processing, and Flight Dynamics objects are depicted with medium-weight lines. The use of medium-weight lines for abstract-objects, ports, and associations is to indicate that they are internal components of the refinements of the agency-level abstract-objects.

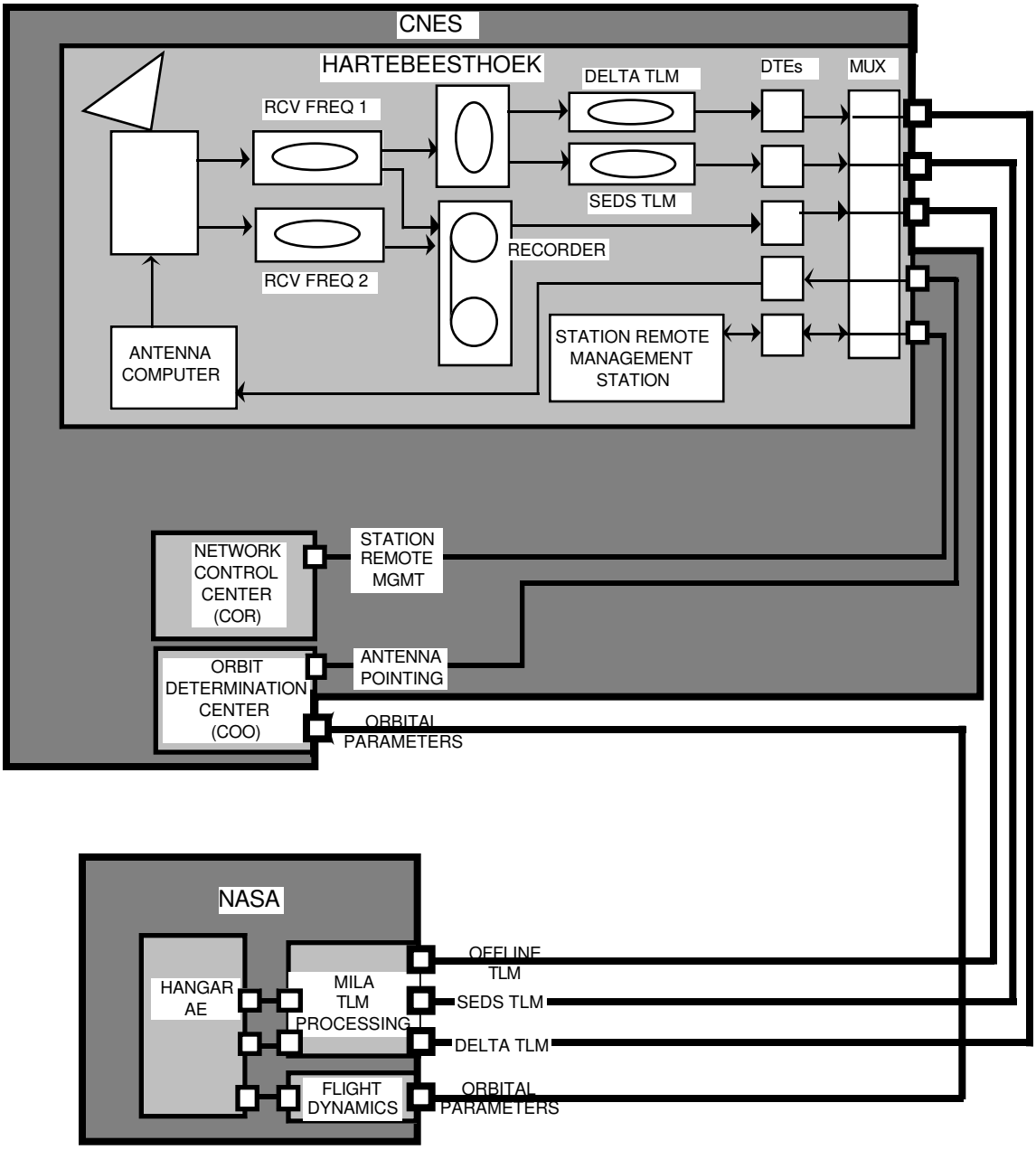


Figure B-3: Delta/SEDS Mission Agency Abstract-Objects

Figure B-3 can be redrawn for simplicity and to reveal more of the peer-to-peer, user-provider relationship that exists between the CNES and NASA objects. Figure B-4 is topologically equivalent to figure B-3, with the addition of the SEDS Mission Manager **abstract-object**, which is not found on the original SEDS diagram. The SEDS Mission Manager interacts with the COR to schedule CNES services, send real-time service reconfiguration requests (if any), and receive service accounting information (if any).

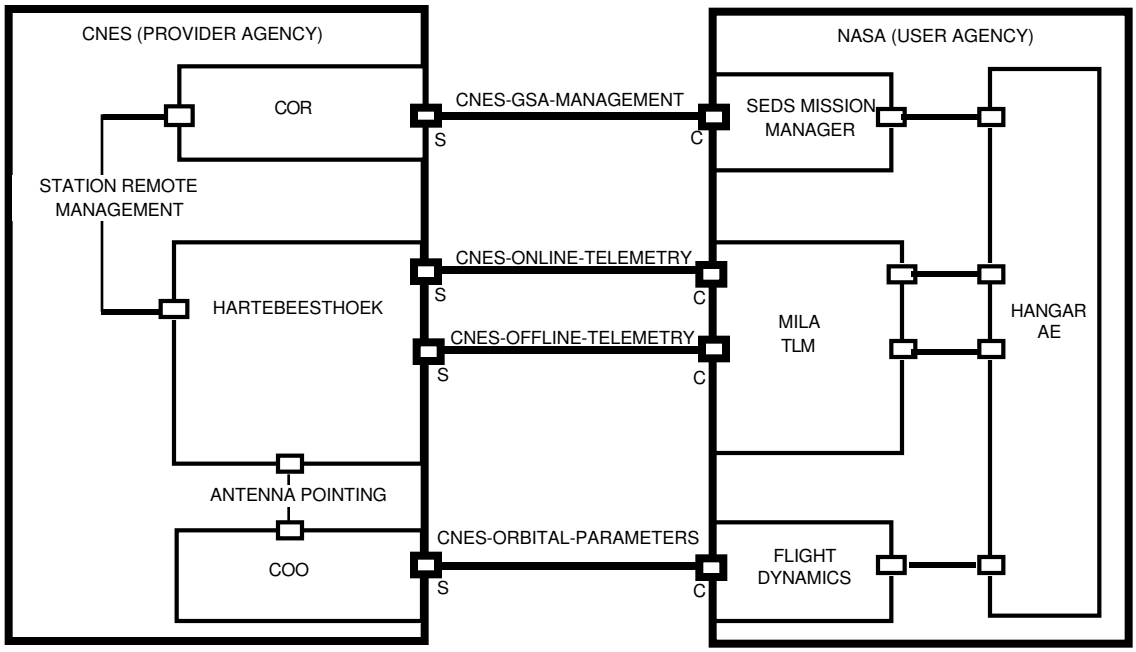


Figure B-4: User Agency–Provider Agency Depiction of SEDS Objects