The Consultative Committee for Space Data Systems

Draft Recommendation for Space Data System Standards

MISSION OPERATIONS— MISSION DATA PRODUCT DISTRIBUTION SERVICES

DRAFT RECOMMENDED STANDARD

CCSDS 522.2-R-2

RED BOOK

November 2024

The Consultative Committee for Space Data Systems

Draft Recommendation for Space Data System Standards

MISSION OPERATIONS— MISSION DATA PRODUCT DISTRIBUTION SERVICES

DRAFT RECOMMENDED STANDARD

CCSDS 522.2-R-2

RED BOOK

November 2024

AUTHORITY

Issue:	Red Book, Issue 2
Date:	November 2024
Location:	Not Applicable

(WHEN THIS RECOMMENDED STANDARD IS FINALIZED, 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 represents the consensus technical agreement of the participating CCSDS Member Agencies. The procedure for review and authorization of CCSDS documents is detailed in *Organization and Processes for the Consultative Committee for Space Data Systems* (CCSDS A02.1-Y-4), and the record of Agency participation in the authorization of this document can be obtained from the CCSDS Secretariat at the email address below.

This document is published and maintained by:

CCSDS Secretariat
National Aeronautics and Space Administration
Washington, DC, USA
Email: secretariat@mailman.ccsds.org

STATEMENT OF INTENT

(WHEN THIS RECOMMENDED STANDARD IS FINALIZED, IT WILL CONTAIN THE FOLLOWING STATEMENT OF INTENT:)

The Consultative Committee for Space Data Systems (CCSDS) is an organization officially established by the management of its members. The Committee meets periodically to address data systems problems that are common to all participants, and to formulate sound technical solutions to these problems. Inasmuch as participation in the CCSDS is completely voluntary, the results of Committee actions are termed **Recommended Standards** and are not considered binding on any Agency.

This **Recommended Standard** is issued by, and represents the consensus of, the CCSDS members. Endorsement of this **Recommendation** is entirely voluntary. Endorsement, however, indicates the following understandings:

- o Whenever a member establishes a CCSDS-related **standard**, this **standard** will be in accord with the relevant **Recommended Standard**. Establishing such a **standard** does not preclude other provisions which a member may develop.

- o Whenever a member establishes a CCSDS-related **standard**, that member will provide other CCSDS members with the following information:

    - -- The **standard** itself.

    - -- The anticipated date of initial operational capability.

    - -- The anticipated duration of operational service.

- o Specific service arrangements shall be made via memoranda of agreement. Neither this **Recommended Standard** nor any ensuing **standard** is a substitute for a memorandum of agreement.

No later than five years from its date of issuance, this **Recommended Standard** will be reviewed by the CCSDS to determine whether it should: (1) remain in effect without change; (2) be changed to reflect the impact of new technologies, new requirements, or new directions; or (3) be retired or canceled.

In those instances when a new version of a **Recommended Standard** is issued, existing CCSDS-related member standards and implementations are not negated or deemed to be non-CCSDS compatible. It is the responsibility of each member to determine when such standards or implementations are to be modified. Each member is, however, strongly encouraged to direct planning for its new standards and implementations towards the later version of the Recommended Standard.

FOREWORD

Through the process of normal evolution, it is expected that expansion, deletion, or modification of this document may occur. This Recommended Standard is therefore subject to CCSDS document management and change control procedures, which are defined in the *Organization and Processes for the Consultative Committee for Space Data Systems* (CCSDS A02.1-Y-4). 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 sent to the CCSDS Secretariat at the email 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.
– Canadian Space Agency (CSA)/Canada.
– Centre National d'Etudes Spatiales (CNES)/France.
– China National Space Administration (CNSA)/People's Republic of China.
– Deutsches Zentrum für Luft- und Raumfahrt (DLR)/Germany.
– European Space Agency (ESA)/Europe.
– Federal Space Agency (FSA)/Russian Federation.
– Instituto Nacional de Pesquisas Espaciais (INPE)/Brazil.
– Japan Aerospace Exploration Agency (JAXA)/Japan.
– National Aeronautics and Space Administration (NASA)/USA.
– UK Space Agency/United Kingdom.

Observer Agencies

– Austrian Space Agency (ASA)/Austria.
– Belgian Science Policy Office (BELSPO)/Belgium.
– Central Research Institute of Machine Building (TsNIIMash)/Russian Federation.
– China Satellite Launch and Tracking Control General, Beijing Institute of Tracking and Telecommunications Technology (CLTC/BITTT)/China.
– Chinese Academy of Sciences (CAS)/China.
– China Academy of Space Technology (CAST)/China.
– Commonwealth Scientific and Industrial Research Organization (CSIRO)/Australia.
– Danish National Space Center (DNSC)/Denmark.
– Departamento de Ciência e Tecnologia Aeroespacial (DCTA)/Brazil.
– Electronics and Telecommunications Research Institute (ETRI)/Korea.
– Egyptian Space Agency (EgSA)/Egypt.
– European Organization for the Exploitation of Meteorological Satellites (EUMETSAT)/Europe.
– European Telecommunications Satellite Organization (EUTELSAT)/Europe.
– Geo-Informatics and Space Technology Development Agency (GISTDA)/Thailand.
– Hellenic National Space Committee (HNSC)/Greece.
– Hellenic Space Agency (HSA)/Greece.
– Indian Space Research Organization (ISRO)/India.
– Institute of Space Research (IKI)/Russian Federation.
– Korea Aerospace Research Institute (KARI)/Korea.
– Ministry of Communications (MOC)/Israel.
– Mohammed Bin Rashid Space Centre (MBRSC)/United Arab Emirates.
– National Institute of Information and Communications Technology (NICT)/Japan.
– National Oceanic and Atmospheric Administration (NOAA)/USA.
– National Space Agency of the Republic of Kazakhstan (NSARK)/Kazakhstan.
– National Space Organization (NSPO)/Chinese Taipei.
– Naval Center for Space Technology (NCST)/USA.
– Netherlands Space Office (NSO)/The Netherlands.
– Research Institute for Particle & Nuclear Physics (KFKI)/Hungary.
– Scientific and Technological Research Council of Turkey (TUBITAK)/Turkey.
– South African National Space Agency (SANSA)/Republic of South Africa.
– Space and Upper Atmosphere Research Commission (SUPARCO)/Pakistan.
– Swedish Space Corporation (SSC)/Sweden.
– Swiss Space Office (SSO)/Switzerland.
– United States Geological Survey (USGS)/USA.

PREFACE

This document is a draft CCSDS Recommended Standard. Its ‘Red Book’ status indicates that the CCSDS believes the document to be technically mature and has released it for formal review by appropriate technical organizations. As such, its technical contents are not stable, and several iterations of it may occur in response to comments received during the review process.

Implementers are cautioned **not** to fabricate any final equipment in accordance with this document’s technical content.

Recipients of this draft are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

DOCUMENT CONTROL

Document	Title	Date	Status
CCSDS 522.2-R-1	Mission Operations—Mission Data Product Distribution Services, Draft Recommended Standard, Issue 1	November 2018	Original draft, discontinued
CCSDS 522.2-R-2	Mission Operations—Mission Data Product Distribution Services, Draft Recommended Standard, Issue 2	November 2024	Current draft

CONTENTS

CONTENTS (continued)

Section Page

Figure

Table

1 INTRODUCTION

1.1 GENERAL

Mission products can be any data set generated by the mission system for distribution to external (and internal) users. In the context of this standard, the term **product** is used to signify a mission data product. A **product** may correspond to, but is not restricted to, the following types of data:

– Mission Operations Data:

  - extracts of Monitoring & Control (M&C) data for a specified time period,
  - summary status reports,
  - Navigation Data Messages (orbit and attitude vectors, predicted geometric events, etc.);

– Mission Exploitation Data:

  - payload or science data,
  - images,
  - derived mission products.

A **product** comprises a set of metadata describing the product and a product body that has a structure known to the product's producer and consumers.

This Recommended Standard has the objective of specifying generic, interoperable Mission Product Distribution Services (MPDS), for typical space mission data sets, including the ones identified above. This standard focuses on the Mission Product Distribution (MPD) services identified for supporting interoperability and defines an information model that defines the data structures required by the operations of these services.

Mission product distribution is an integral part of Mission Operations (MO) and closely related to the other aspects of the overall monitoring and control of space missions. This close relationship is recognized in the context of the CCSDS Mission Operations and Information Management (MOIMS) Area by the fact that the MPD Services have been identified and included from the start among the envisaged Mission Operations services described in reference [D1], *Mission Operations Services Concept*. This standard defines the MPD Services in conformance with the CCSDS MO service framework.

The MPD Services are a set of services that support: interaction between a mission system and its users at the level of generated **products** and **standing orders** for their distribution.

The MPD Services are defined in terms of the Message Abstraction Layer (MAL) (see reference [2], *Mission Operations Message Abstraction Layer*), that is the core of the MO service framework.

1.2 PURPOSE AND SCOPE

This Recommended Standard defines, in an abstract manner, the MPD Services in terms of:

a) an information model that describes the structure of MPDS data, including **products**, **standing orders**, and supporting information objects that are referenced by the MPD Services;

b) service specifications that define the operations necessary to provide the services, together with their parameter data and required behaviour.

Not all aspects of this Recommended Standard need to be applied in the context of a specific MPDS system in order to support a conformant interface. The standard specifies a set of services, each of which is self-standing. A compliant system may implement any combination of these services and omit those that are not required.

This Recommended Standard does not specify:

a) individual implementations or products;

b) the implementation of entities or interfaces within real systems;

c) the methods or technologies required for communications;

d) how required MPDS service configuration data is made available to deployed MPDS functions;

e) the structure of the **product body** for individual product types: this is transparent to the service and must be known to both product producer and consumers. An out-of-band agreement is required to specify the available set of **product types**, associated metadata, and their structure.

1.3 APPLICABILITY

This specification is applicable to any mission operations component that provides mission data products and exposes interfaces for their distribution to users. This includes interfaces between:

– Mission users and the Mission Operations system;

– Mission users and the Mission Exploitation system.

This standard is intended to support ground to ground interfaces within a space system, although deployment in space to ground or space to space contexts is not precluded because of its specification in terms of the MO service framework.

1.4 RATIONALE

The primary goal is to increase the level of interoperability for mission product distribution among agencies and space system users at the level of exchanged mission **products**. The service specifications can also be used between systems within an agency and to promote the development of re-usable infrastructure for space systems and interoperability between missions.

Various use case scenarios applicable to mission product distribution standardization have been identified by performing a survey of a number of representative space missions and associated mission infrastructure of various CCSDS member agencies. A summary of the identified use cases is provided in section 2 of this document.

The use of the underlying MO framework enables abstract services to be implemented using appropriate encoding and information transfer technologies (file and/or message based) for the deployment context. An extensible set of MAL technology bindings exist to support:

- encoding of the service messages;
- binding of the service operations to a specific messaging technology.

It should be noted that security considerations are assumed to be handled at the MO framework layer or below.

1.5 DEFINITIONS

For the purposes of this document, the following definitions apply.

NOTE – Abbreviations are listed in annex C.

definition: The statically declared information associated with an **information object** or data item. This may, for example, include a description, set of defined arguments, or any other information that applies to all occurrences of the information object. There may be multiple definitions (**versions**) over the mission lifetime associated with the same **identity** (definitionID). MPDS **product type** is an example of a definition.

identity: A unique identity associated with an **MO object**, which comprises:

- the **domain** of the object;
- the **area** of the object;
- the type of the object;
- a **key** (identifier) for the object, unique within domain, area and type;
- the **version** of the object [optional].

information object: The set of information about a real-world entity that is exchanged across an interface. This may include static definitions, dynamic status, and metadata. MPDS information objects include mission **products** and **standing orders**.

instance: A dynamically created object representing each new occurrence of an **information object**. This includes a unique instanceID of the occurrence and any unchanging data associated with it as a set of static attributes. It may also includes the current status of the object as a set of dynamic attributes. An instance has a reference to its **definition**. MPDS **product** is an example of an instance.

metadata: Information associated with a product that describes the product content and can be used when searching and filtering available products for retrieval. The metadata includes:

- product type;
- creation date;
- time period to which the product content relates;
- a product Source reference [optional];
- **metadata parameters**;
- a free-text description.

metadata parameter: A set of parameters specific to the **product type** that can be used to characterise the product content. Metadata parameters may be of any MO attribute type and are represented as name-value pairs. Metadata parmeter definitions are contained within the **product type** definition.

MO instance: An MO object pattern for an **information object** that has separate **definition** and **instance** objects, the latter representing an individual occurrence of the object with an evolving status.

Instances reference the **definition** object.

product: Any data set generated by the mission system for distribution to external (and internal) users. This may correspond to, but is not restricted to:

- Mission Operations Data:
    - M&C data for a specified time period,
    - Summary status reports,
    - Navigation Data Messages (orbit and attitude vectors, predicted geometric events, etc.);
- Mission Exploitation Data:
    - Payload or Science Data,
    - Images,
    - Derived Mission Products.

A product comprises a set of metadata describing the product and a **product body** that has a structure known to the product's producer and consumers.

product body: The main content of a **product**, excluding the associated metadata. The structure of the product body is unknown to the MPDS service specification, which treats it as a binary data set or 'blob'. The structure must be known to both the product's producer and its consumers.

product type: The **definition** of a **product** that identifies the **product body** of being of a particular type and structure (without specifying its structure) and also defines a set of associated metadata parameters specific to the product type.

source: An optional reference to an external object responsible for triggering the generation of a **product**, for example, an MPS planning request, an M&C alert, or any other type of Event.

standing order: A subscription for **products** of a specified **product type**, optionally filtered by its metadata parameters or **source**, to be delivered to a specified address or MPDS service consumer. Standing orders are associated with the **user** that created them.

user: An identity associated with the MPDS service consumer that is linked to **standing orders** and may be used by the service provider to restrict access to certain **products**.

version: Part of the **identity** of an **MO object**, typically of a **definition** object, that represents a defined set of values for its static attributes. When a **definition** is updated, the version is incremented, but other elements of the object's identity, including its **key**, remain unchanged.

1.6 NOMENCLATURE

1.6.1 NORMATIVE TEXT

The following conventions apply for the normative specifications in this recommended standard:

a) the words ‘shall’ and ‘must’ imply a binding and verifiable specification;

b) the word ‘should’ implies an optional, but desirable, specification;

c) the word ‘may’ implies an optional specification;

d) the words ‘is’, ‘are’, and ‘will’ imply statements of fact.

NOTE – These conventions do not imply constraints on diction in text that is clearly informative in nature.

1.6.2 INFORMATIVE TEXT

In the normative sections of this document, informative text is set off from the normative specifications either in notes or under one of the following subsection headings:

– Overview;

– Background;

– Rationale;

– Discussion.

1.7 CONVENTIONS

1.7.1 DIAGRAMS

Unified Modelling Language (UML) (reference [6]) notation is used for diagrams representing the MPDS information model in section 4, and illustrating the service specifications in section 3.

In the MPDS service specifications, UML sequence diagrams are used to illustrate the sequence of operations and their constituent messages (following MAL interaction patterns).

In the MPDS information model, UML class diagrams are used to illustrate data structures and their relationships.

1.7.2 TABLES

The formal normative definitions of data structures, services, and service operations are presented in an abstract tabular format in this document. This is consistent with that specified in the MO MAL standard (reference [2]).[1]

1.8 REFERENCES

The following publications contain provisions which, through reference in this text, constitute provisions of this document. At the time of publication, the editions indicated were valid. All publications are subject to revision, and users of this Recommended Standard are encouraged to investigate the possibility of applying the most recent editions of the publications indicated below. The CCSDS secretariat maintains a register of currently valid CCSDS publications.

[1] *Mission Operations Reference Model*. Issue 1. Recommendation for Space Data System Practices (Magenta Book), CCSDS 520.1-M-1. Washington, D.C.: CCSDS, July 2010.

[2] *Mission Operations Message Abstraction Layer*. Issue 3. Recommendation for Space Data System Standards (Blue Book), CCSDS 521.0-B-3. Washington, D.C.: CCSDS, March 2024.

[3] T. Berners-Lee, R. Fielding, and L. Masinter. *Uniform Resource Identifier (URI): Generic Syntax*. STD 66. Reston, Virginia: ISOC, January 2005.

[4] H. Thompson and C. Lilley. *XML Media Types*. RFC 7303. Reston, Virginia: ISOC, July 2014.

[5] “Registries.” Space Assigned Numbers Authority. https://sanaregistry.org/r.

NOTE – The specific SANA registries applicable to this specification are detailed in section 6.

[6] *OMG® Unified Modeling Language (OMG® UML)*. Version 2.5.1. formal/2017-12-05. Needham, Massachusetts: Object Management Group, December 2017.

NOTE – Informative references are listed in annex D.

[1] Minor differences in style/layout are to enable autogeneration of the XML specification from the tables.

2 OVERVIEW

2.1 GENERAL

This section introduces the concepts behind the Mission Product Distribution (MPD) Services. It has the following main sections:

- MPDS Use Cases;
- MPDS Concept;
- MPDS Information Model Overview;
- MPDS Services Overview;
- Relationship To Mission Operations Services;
- Optional Elements of the Standard.

2.2 MISSION PRODUCT DISTRIBUTION USE CASES

2.2.1 GENERAL

This section provides the set of operational use cases for the Mission Product Distribution Services (MPDS).

In the context of this specification, no distinction is made regarding whether the **products** requested or delivered through the MPD service interface are pre-existing, having been automatically generated, or are produced on-the-fly when requested. Automatically generated products may, for example, be produced according to a pre-defined schedule or in response to a specific event, but this is an internal implementation choice not addressed by the standard.

The MPD services provide a standardized interface between the mission system that generates and stores the products and any user system that requires access to those products.

In inter-operability scenarios, the list of types of product, subject to exchange, are typically predefined and captured in out-of-band agreements (Interface Control Documents [ICDs]). Event files, consolidated telemetry (TM) packets, command histories and images are a few examples of such products. Each product type may be further categorized by a set of parameters specific to the **product type** that may be used to filter or further specify the products to be distributed.

How the data is organized in one or multiple products depends on the implementation of a mission product distribution system (mission choice). To take the example of TM Packets, the product could be a single packet or a set of packets.

Products may be retrieved (pulled) by the service consumer, or distributed (pushed) by the service provider. In the latter case, the service consumer must have explicitly (through the service) or implicitly (through out-of-band agreement) subscribed for the receipt of specific product types through a **standing order**.

Products may be delivered off-line by file transfer, or through the exchange of messages within the MPD service itself.

The following generic use cases are identified:

1) Retrieval (Pull) of distinct **products**

2) Recurrent Retrieval (Pull) of distinct **products**

3) Delivery (Push) of products subscribed to by **standing order**

These use cases are elaborated in the following subsections, together with typical examples of how they relate to real-world scenarios.

It should be noted that an actual MPDS deployment may not need, and is not required to support, all the above use cases.

2.2.2 USE CASE 1: RETRIEVAL (PULL) OF DISTINCT PRODUCTS

In this scenario the client requests a particular **product** to be delivered. It can specify generic filters such as time or product specific filters based on the metadata parameters associated with the **product type**. The delivery of the product can then be either by file transfer or by means of messages exchanged within the service interface.

Example 1: The client requests a daily summary of specified CCSDS Space Packets (reference [D3]) in telemetry. A **product type** exists for a daily TM packet extract, for which the Application Process ID (APID) is a metadata parameter. The client requests a **product** of this type, specifying the time period and APID of the required packets.

Example 2: The client requests an image of a specified type. A **product type** exists for images, which has an associated set of metadata parameters, including the image type (for example spectral band) and the subject of the image (as a named target, or a set of coordinates defining the location or direction of the target). The client requests a **product** of this type, specifying the time period, image type, and coordinates.

Example 3: The client has previously requested an observation or other operation to be performed that results in the generation of a **product** of a particular **product type**: the **source** of the product references the original request. Alternatively, the provider generates a **product** in response to the occurrence of an event, such as an astronomical Target Of Opportunity event notified by another observatory: the **source** of the product references the event. The client requests a **product** of this type, specifying the **source** of the product (request or event), optionally also specifying the time period and other metadata parameters.

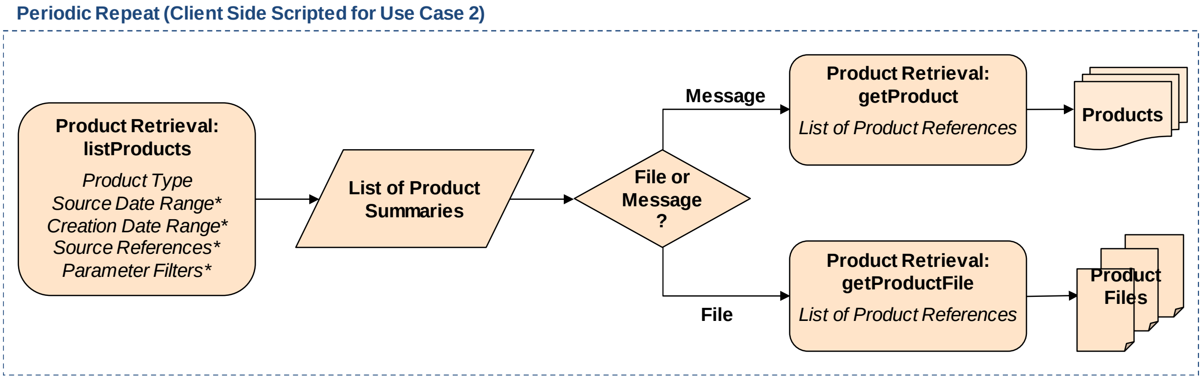
Periodic Repeat (Client Side Scripted for Use Case 2)
Product Retrieval: listProducts
Product Type
Source Date Range*
Creation Date Range*
Source References*
Parameter Filters*
List of Product Summaries
File or Message ?
Message
File
Product Retrieval: getProduct
List of Product References
Products
Product Retrieval: getProductFile
List of Product References
Product Files

Figure 2-1: Flowchart of Service Operations for Use Cases 1 and 2

The flowchart above shows how the MPDS Product Retrieval service operations can be used to fulfil the use case. First the client uses the listProduct operation to request a list of products satisfying a specified set of filter criteria (fields with a * are optional). This returns a list of product summaries including the product reference, source and creation date of the product, and any metadata parameters. Depending on whether the client requires product delivery by service messages or file transfer, the client uses the getProduct or getProductFile operation for the required list of product references (which may be one, all, or a selected subset of the product references returned in the listProducts operation).

2.2.3 USE CASE 2: RECURRENT RETRIEVAL (PULL) OF DISTINCT PRODUCTS

In this scenario, the client requests recurrent, periodic retrieval of a **product** of a specified **product type**.

Example: The client requests a daily summary of specified CCSDS Space Packets (reference [D3]) contained in telemetry at midnight each day, or requests a weekly summary every Monday at 12:00.

The use case of periodical retrieval is covered in this specification by means of client side automation/scripting of use case 1, and is equally applicable to all examples listed there.

2.2.4 USE CASE 3: DELIVERY (PUSH) OF PRODUCTS SUBSCRIBED TO BY STANDING ORDER

The client requests recurrent delivery of a **product** of a specified **product type** when it becomes available at the provider. In this case, the client sets up a **standing order** to express its interest in the products.

Example 1: The client requires a daily summary of specified TM Packets. A **product type** exists for a daily TM packet extract, for which the APID is a metadata parameter. The client sets up a **standing order** for **products** of this type, specifying the APID of the required packets.

Example 2: The client requires images of a specified type. A **product type** exists for images, which has an associated set of metadata parameters, including the image type (for example spectral band) and the subject of the image (as a named target, or a set of coordinates defining the location or direction of the target). The client sets up a **standing order** for **products** of this type, specifying the image type and optionally coordinates.

Example 3: The client requests an observation or other operation to be performed that results in the generation of a **product** of a particular **product type**: the **source** of the product references the user. Alternatively, the provider generates a **product** in response to the occurrence of an event, such as an astronomical Target Of Opportunity event notified by another observatory: the **source** of the product references the event. The client sets up a **standing order** for **products** of this type, specifying the **source** of the product (user or event), optionally also specifying other metadata parameters.

In each case, the mission product distribution system (provider) delivers the **products** automatically, when they become available. The delivery of the **product** can be either as a file or by means of messages exchange.

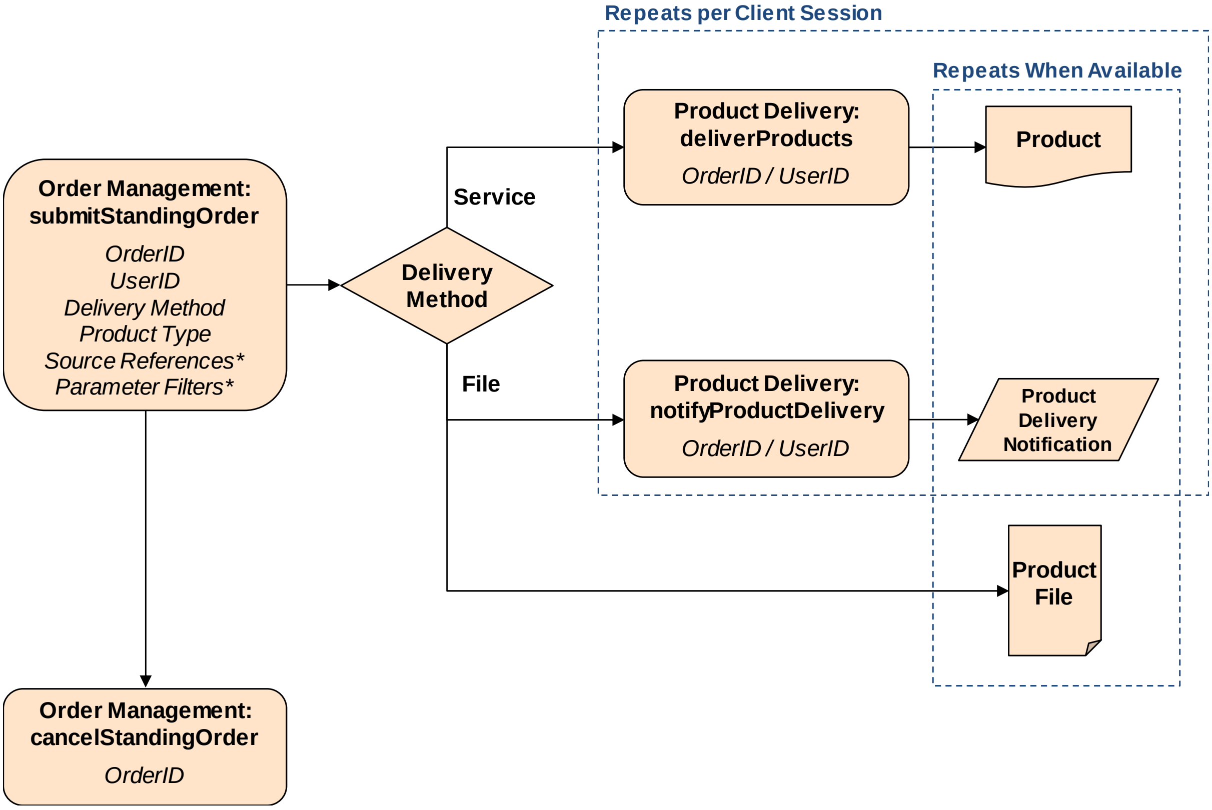
Repeats per Client Session
Repeats When Available
Order Management: submitStandingOrder
OrderID
UserID
Delivery Method
Product Type
Source References*
Parameter Filters*
Delivery Method
Service
File
Product Delivery: deliverProducts
OrderID / UserID
Product
Product Delivery: notifyProductDelivery
OrderID / UserID
Product Delivery Notification
Product File
Order Management: cancelStandingOrder
OrderID

Figure 2-2: Flowchart of Service Operations for Use Case 3

The flowchart above shows how the MPDS Order Management and Product Delivery services can be used to fulfil the use case.

First the Order Management service submitStandingOrder operation is used to create a **standing order** for the automatic delivery of **products** that satisfy a specified set of filter criteria (fields with a * are optional), using the specified delivery method (Service or File) as they become available. This standing order can then be cancelled at any time using the cancelStandingOrder operation.

Once a standing order has been established, the Product Delivery service can be used.

If the delivery method ‘Service’ was specified in the standing order, then to receive the **products** via the service interface, then the deliverProducts operation must be used, referencing either the OrderID or a UserID (for all orders belonging to that User). **Products** will then be delivered via this operation as long as the client session remains active. The deliverProducts operation must be repeated each time the client session is restarted.

If the deliver method ‘File’ was specified in the standing order, then **products** will be automatically delivered to the specified address as they become available. No further service operations are required, but the client can optionally register to receive notifications of file delivery via the notifyProductDelivery operation.

2.3 MISSION PRODUCT DISTRIBUTION CONCEPT

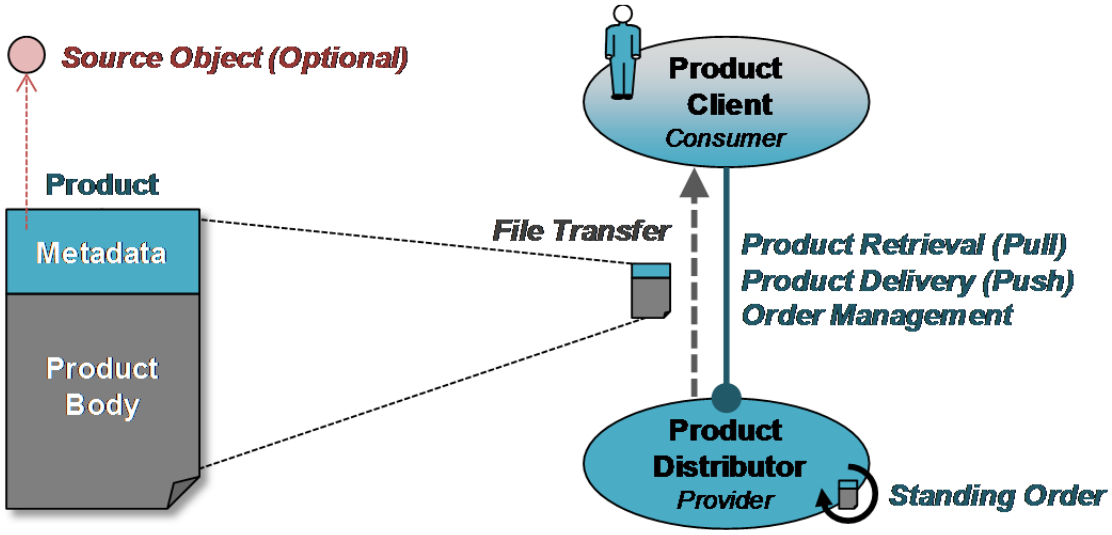

Figure 2-3: Mission Product Distribution Concept

Two functions are identified: the Product Distributor and the Product Client.

The Product Client is the consumer of the MPD service and receives mission **products**, either by file transfer of via service messages.

Products are automatically generated, in advance or on-demand, and made available to the Product Distributor function that acts as the MPD service provider.

The product itself comprises a standardized header containing metadata and a product body whose structure is opaque to the standardized distribution service itself. The information regarding how to interpret the content of the product body is subject to out-of-band agreement. The product header contains metadata that describes or categorizes the product without defining its structure. This includes the date of generation, the time period covered by the product, a free text description, a set of metadata parameters, and an optional reference to a source object. Metadata parameters provide additional information categorizing the product. The source object may be used to reference something that triggered the generation of the product, such as a planning request or event, and can be used by the consumer to close the loop (for example a planning request for an operation might trigger the generation of an output product from that operation).

Multiple **product types** may be available for distribution. Each product type has a name that can be used to identify the content and structure of the product. The set of available metadata parameters are also specific to and defined by the product type.

Products may be delivered to service consumers on the basis of a standing order or subscription. The term **standing order** is used to differentiate it from the publish-subscribe interaction pattern used by the services. The standing order may include a product filter that specifies a subset of products of a named type, based on the value of their metadata parameters, product source, generation time, or coverage period.

2.4 MPDS INFORMATION MODEL OVERVIEW

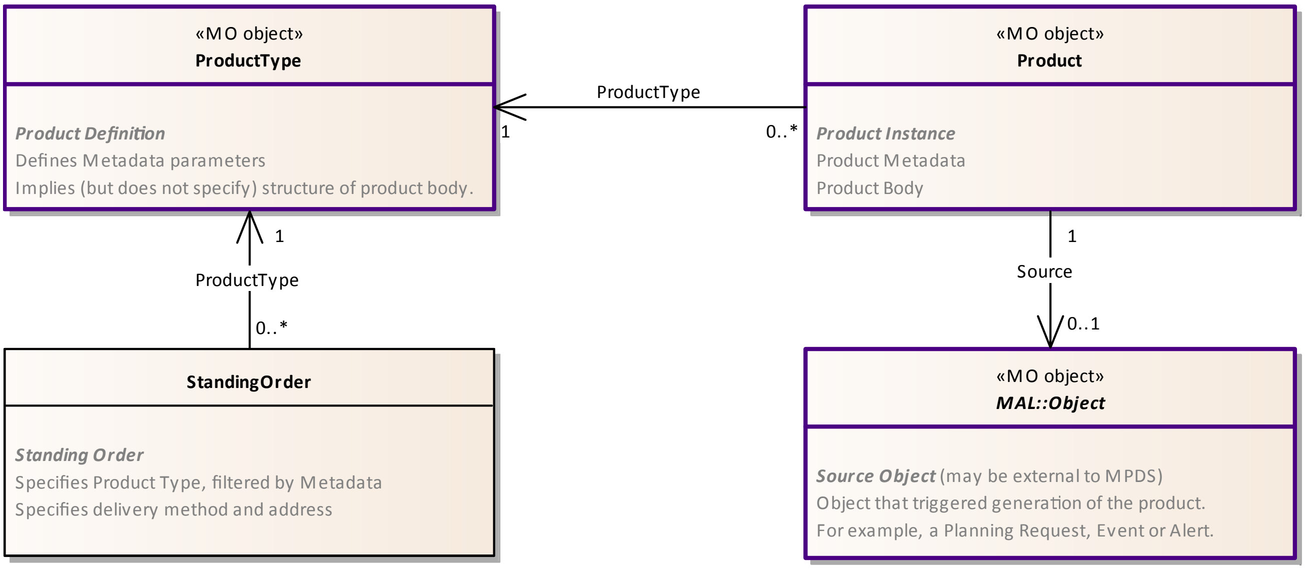
«MO object»
ProductType
Product Definition
Defines Metadata parameters
Implies (but does not specify) structure of product body.
ProductType
1
0..*
«MO object»
Product
Product Instance
Product Metadata
Product Body
1
ProductType
0..*
StandingOrder
Standing Order
Specifies Product Type, filtered by Metadata
Specifies delivery method and address
1
Source
0..1
«MO object»
MAL::Object
Source Object (may be external to MPDS)
Object that triggered generation of the product.
For example, a Planning Request, Event or Alert.

Figure 2-4: MPDS Information Model Overview

The concept for the MPD services has been introduced in 2.3 above. Three main information objects are identified:

– ProductType;

– Product;

– StandingOrder.

ProductType and Product are represented as MO Objects, while StandingOrder is a composite data structure whose identity is represented by a simple MAL::Identifier.

The ProductType defines a type of product that is available for distribution. It comprises a description of the product type and specification of the associated metadata parameters. It also implies, but does not specify, the structure of the product body.

The Product corresponds to an **instance** of a particular **product type**. It comprises metadata and the product body. The metadata includes:

– Product Type reference;

– Date of Production;

– Period of Source Data to which the product relates;

– Product Source (Optional) reference to an MO Object that triggered product generation;

– Product Parameters, as defined by the Product Type.

The StandingOrder specifies a standing order to deliver **products** of a specified **product type** that pass a defined filter on the product’s metadata to an identified user. The **standing order** also specifies the method of **product** delivery and any required destination address.

The optional source of a **product** can be any type of MO Object and can be used to indicate that it is associated with triggering the generation of the product, for example MPS Planning Requests or M&C Alerts. This can be useful for a user to close the loop between different MO services.

2.5 MPDS SERVICES OVERVIEW

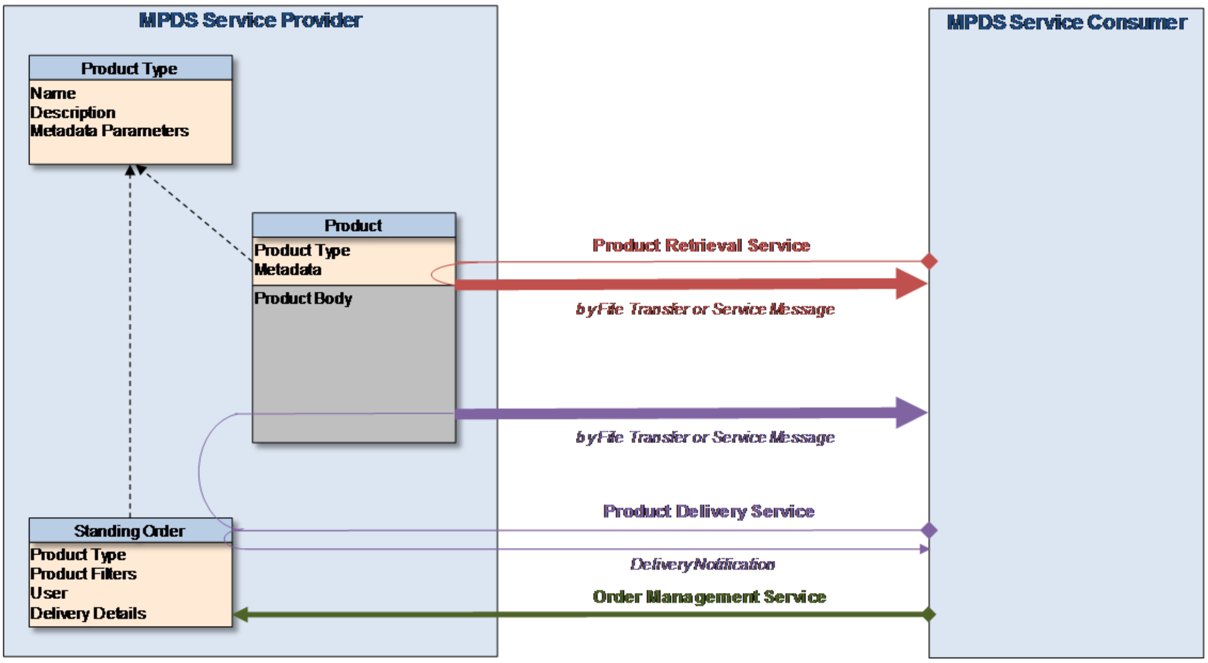

Figure 2-5: MPDS Services Overview

The MPDS service interface allows for the following interactions between the service consumer and service provider, each of which is supported by a separate MPD service:

- Product Retrieval Service:
  Product Retrieval by the Consumer (no **standing order** is required);
- **Order Management Service:**
  Submission of a new or cancellation of an existing **standing order** by the Consumer;
- **Product Delivery Service**:
  Product Delivery by the Provider in accordance with an existing **standing order**.

Product delivery can be made via service messages or through an external file delivery protocol such as FTP.

Where products are delivered as a result of an existing standing order, file delivery can proceed without intervention by the service consumer. However, the Product Delivery service also enables the receipt of notifications that a file has been delivered, or the receipt of the product via service messages.

The Order Management service enables a user to directly set up and cancel **standing orders**. However, the Product Delivery service can be used without this if the product orders are managed locally by the service provider.

The Product Retrieval service does not require implementation of the **standing order** concept.

2.6 RELATIONSHIP TO MISSION OPERATIONS SERVICES

The CCSDS Mission Operations (MO) Services Concept provides a standard framework for the specification of end-to-end services between mission operations applications (reference [D1]). MO services are defined in terms of a Message Abstraction Layer (MAL) (reference [2]), which provides a means of specifying data and service interfaces in an implementation-, encoding-, and communication-agnostic manner.

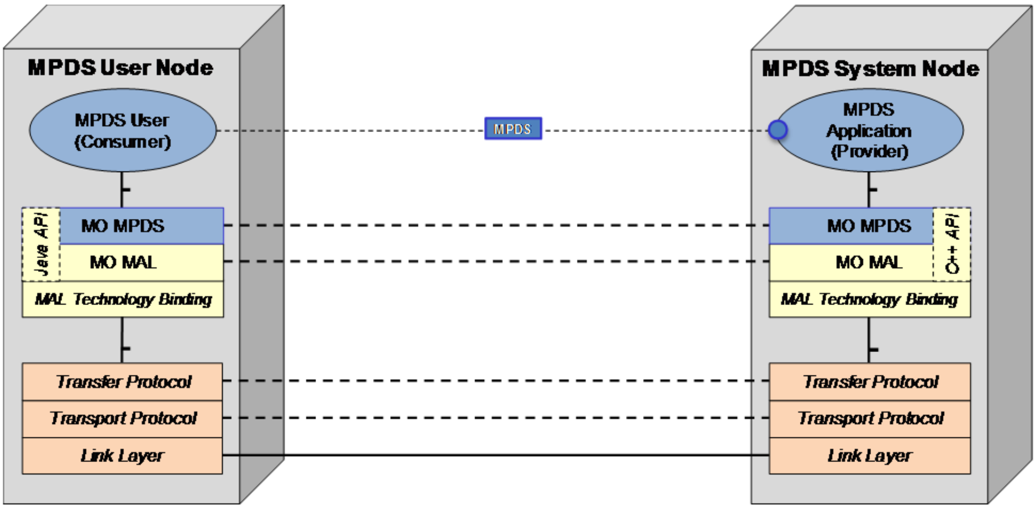

Figure 2-6: MO MPDS Services Generic Protocol Stack

The MO MAL defines:

- A set of MAL Attribute data types that can be used to represent the individual data fields of message structures;
- A set of MAL Interaction patterns that correspond to the message exchange behavior of individual service operations.

The abstract specification of the service interfaces and data can be mapped to a concrete implementation through:

a) a technology binding that defines how the abstract messages (composed of a sequence of MAL attributes) are encoded in a concrete format (e.g., binary, XML, or ASCII);

b) a technology binding that defines how the resulting messages are carried over a concrete message transport protocol by mapping the standard MAL interaction patterns to that protocol;

c) a language binding that transforms the abstract service interface into a concrete API for a given programming language (e.g., Java, C++ or Python).

Figure 2-6 illustrates a generic deployment of MPD services using the MO service framework with service consumer and provider functions hosted on different deployment nodes. MPDS specific functions and protocol layers are shown in blue; elements of the ‘vanilla’ MO framework in yellow; and underlying communications infrastructure layers in tan (light orange). The application level MPDS service interaction is shown by the direct interface between service provider and consumer functions, carrying MPDS service messages defined in terms of data structures specified in the MPDS information model.

Adopting a single MAL technology binding in any specific deployment ensures on-the-wire interoperability. Transfer protocol equates to the messaging or file transfer service used over the underlying transport and physical link layers. The diagram illustrates how different language bindings can be used by provider and consumer for the service API, as this does not affect the wire level protocol.

It is noted that while the MAL may be implemented as a specific software layer for reasons of maintainability and reusability, it is not a requirement to do so. The MAL may simply be used as an abstract specification that enables transformation of the service specification by the applied technology bindings into a concrete implementation of that service with no distinct MAL layer. The MO MPDS, MO MAL, and MAL technology binding layers in the diagram are effectively combined into a single software component. This is an important distinction for deployment contexts where the implementation is required to be both compact and efficient, such as on-board a spacecraft.

It is also noted that the MPD services are defined in terms of the generic concept of an **MO object**. An MO object is an entity defined within the information model of an MO compliant service specification that has a unique identity enabling it to be referenced by other MO objects and in the body of MO service messages. The **identity** of an MO object is defined by its type and unique **key**, scoped by its **domain**, and optionally by a **version**. The specification of a service-specific MO object class includes a custom set of references to other MO objects that capture the relationships between those objects. In the context of the MPD services, MO objects are defined to represent Mission Data **products**, **product types**, and **standing orders**.

An MO application-level service specification comprises a set of operations that the service consumer may invoke on the service provider. Each operation is mapped to a standard interaction pattern defined by the MAL and provides the service-specific body of the constituent messages.

In addition to the MAL and technology bindings, the MO service framework includes a set of Common Services (reference [D2]) that can be used in conjunction with any MO application level service. These include a Directory, Login, and Configuration services.

2.7 OPTIONAL ELEMENTS OF THE STANDARD

The MPD services specification defines an MPDS information model and three services. Compliance of an individual system deployment to the MPD service specification does not imply that either the complete set of services specified here or the full MPDS information model has to be supported.

The level of compliance of a specific deployment to the standard can be selected in terms of the set of MPD services supported. Each service is self-standing and they can be deployed in any combination. A compliant system may implement only those services it requires and omit the others.

The supported capabilities of a service provider can be made available to consumers via the MO Common Directory Service (reference [D2]). An entry in the directory is made for each MPD service provided, which lists the supported capability sets for that service. For the MPD services, each service comprises a single capability set.

The following table summarizes the MPD services and information model elements used by each service. Only those parts of the information model used by a given service need to be implemented.

For each service, applicability of information model elements are shown as follows:

✓ Required;
R the item is only referenced by the service and does not appear as a data structure in service messages;
- not applicable.

Table 2-1: Optional Service Capabilities

Services		Information Model					
	Service Name	Product Type	Product	Standing Order	Additional Types	Parameters	Service Data Structures
PRS	Product Retrieval Service	R	✓	-	✓	✓	Product, Product Filter, Product Summary
PDS	Product Delivery Service	R	✓	R	✓	-	Product
OMS	Order Management Service	R	-	✓	✓	✓	Standing Order, Product Filter

3 MPD SERVICE SPECIFICATIONS

3.1 GENERAL

This section contains the service specifications for the MPD services. An overview of the services has been given previously in 2.5 above. Each of the following services is defined in a separate subsection below:

a) Product Retrieval Service;

b) Order Management Service;

c) Product Delivery Service.

There is no requirement to implement all services. Any subset of the services may be supported by an MPDS deployment.

Each service specification comprises the following parts:

a) Summary: summary of service, a table listing the service operations and grouping them into capability sets, and UML sequence diagrams illustrating the service operations.

b) MO Objects and other data items: the set of MO objects and other data items (defined in section 4) that are operated on by the service or referenced by service operations.

c) High-Level Requirements: general requirements what is made available to both provider and consumers in a deployment of a service, such as the set of configuration data required by the service.

d) Functional Requirements: specific requirements on the behaviour of the service provider and consumer.

e) Operations: specification of each service operation.

All services defined in this document are part of the **MPD** Area, Area number **9**.

3.2 SERVICE: PRODUCT RETRIEVAL

3.2.1 SUMMARY

The Product Retrieval Service, introduced in 2.5, enables browsing and retrieval of existing **products**.

Area Identifier	Service Identifier	Area Number	Service Number	Area Version
MPD	ProductRetrieval	6	1	1
Interaction Pattern	Operation Identifier		Operation Number	Capability Set
REQUEST	listProducts		1	
PROGRESS	getProducts		2	1
PROGRESS	getProductFiles		3	

The listProducts operation enables the service consumer to obtain a filtered list of available **products** of a specified **product type**.

The getProduct operation can then be used to retrieve one or more **products** using the identity of those products returned by the listProducts operation. There are two variants of this operation: getProduct returns the requested products via the service interface itself; getProductFile delivers the requested products by file transfer and notifies successful delivery via the service interface.

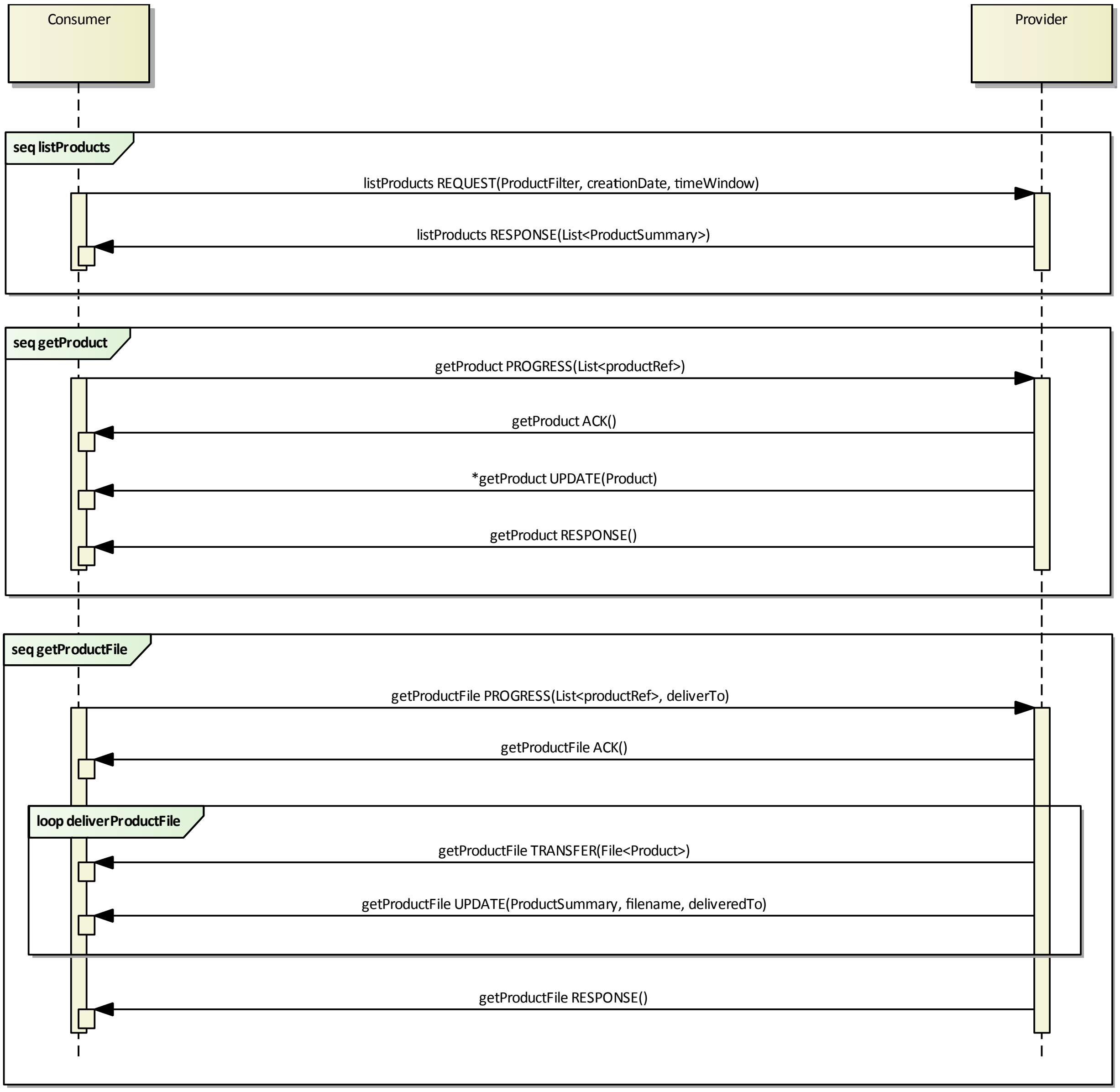
Consumer
Provider
seq listProducts
listProducts REQUEST(ProductFilter, creationDate, timeWindow)
listProducts RESPONSE(List<ProductSummary>)
seq getProduct
getProduct PROGRESS(List<productRef>)
getProduct ACK()
*getProduct UPDATE(Product)
getProduct RESPONSE()
seq getProductFile
getProductFile PROGRESS(List<productRef>, deliverTo)
getProductFile ACK()
loop deliverProductFile
getProductFile TRANSFER(File<Product>)
getProductFile UPDATE(ProductSummary, filename, deliveredTo)
getProductFile RESPONSE()

Figure 3-1: Product Retrieval Operations

3.2.2 MPDS DATA ITEMS

MPDS data items and their relationships relevant to the product retrieval service are defined within the MPDS information model in 4.2.

The following MO objects are directly applicable to the service (see figure 4-2):

– Product.

Products comprise metadata that describes the product (but not its structure) and the product body. The structure of the product body is unknown to the service interface, but must be known to the service user.

The following MO objects may be referenced in the context of the product retrieval service:

– ProductType: contains the definition of metadata parameters specific to a type of mission data product.

3.2.3 HIGH-LEVEL REQUIREMENTS

The following set of mission data product configuration data shall be available to both provider and consumers in any deployment of the product retrieval service:

– Product definitions.

3.2.4 FUNCTIONAL REQUIREMENTS

3.2.4.1 In response to a listProducts operation, the service provider shall compile a list of Products that satisfy the specified criteria and return this list (as object references) to the service consumer.

NOTE – It is implementation dependent whether the Products listed are restricted to those already available or may also be generated on-the-fly in response to the listProducts operation.

3.2.4.2 In response to a getProduct operation, the service provider shall return each referenced product to the service consumer in a separate Update message, the final, empty Response message signalling that all products have been returned.

3.2.4.3 In response to a getProductFile operation, the service provider shall return each referenced product by file transfer to the specified Uniform Resource Identifier (URI) (see reference [3]) and, following completion of the transfer, notify the service consumer of the delivery of each product file in a separate Update message, the final, empty Response message signalling that all product files have been transferred.

3.2.5 OPERATION: LISTPRODUCTS

3.2.5.1 Overview

The listProducts operation can be used to obtain a list of available **products** of a specified **product type** that pass a set filter criteria. The service consumer sends the required filter criteria to the service provider, which then builds a list of products that pass the filter and returns this to the consumer.

The filter comprises a ProductFilter structure that specifies the product type and required filters by source and metadata parameter, together with two additional filter criteria relating to the creation and/or source data dates of the products. To pass the filter a **product** must match all of the individual filter criteria specified (source, metadata parameters, creation date, and time window).

The response consists of a list of ProductSummaries that pass the specified filter. These contain the metadata of the products but not the product body itself.

Operation Identifier	listProducts		
Interaction Pattern	REQUEST		
Pattern Sequence	Message	Nullable	Body Signature
IN	REQUEST	No Yes Yes	productFilter : (ProductFilter) creationDate : (TimeWindow) timeWindow : (TimeWindow)
OUT	RESPONSE	No	productList : (List<ProductSummary>)

3.2.5.2 Errors

In addition to standard MAL errors, the operation may return the following MPDS errors. (See section 5 for details of the secondary error code.)

Error	Error #	ExtraInfo Type	ExtraInfo Comment
Invalid	1	SecondaryErrorCodeEnum	A UInteger giving a secondary error code that details the reason for invalidity.

3.2.6 OPERATION: GETPRODUCTS

3.2.6.1 Overview

The getProduct operation allows consumers to retrieve one or more mission data **products**. The consumer sends a list of one or more product references (as ObjectRefs) to the service provider, their identity having previously been established using the listProducts operation.

Each requested product is then returned in a separate Update message, with the final Response message indicating that all requested products have been delivered.

Operation Identifier	getProducts		
Interaction Pattern	PROGRESS		
Pattern Sequence	Message	Nullable	Body Signature
IN	PROGRESS	No	productRefs : (List <MAL::ObjectRef<Product>>)
OUT	ACK	No	Empty
OUT	UPDATE	No	product : (Product)
OUT	RESPONSE	No	Empty

3.2.6.2 Errors

In addition to standard MAL errors, the operation may return the following MPDS errors. (See section 5 for details of the secondary error code.)

Error	Error #	ExtraInfo Type	ExtraInfo Comment
Invalid	1	SecondaryErrorCodeEnum	A UInteger giving a secondary error code that details the reason for invalidity.

3.2.7 OPERATION: GETPRODUCTFILES

3.2.7.1 Overview

This variant of the getProduct operation allows consumers to retrieve one or more mission data **products** with the delivery of the products being effected by file transfer outside the service interface.

The consumer sends a list of one or more product references (as ObjectRefs) to the service provider, their identity having previously been established using the listProducts operation. The message also contains the file delivery address as a URI.

After each requested product has been delivered by file transfer, a separate Update message is returned with a product delivery notification comprising ProductSummary, filename, and delivery address as a URI, with the final Response message indicating that all requested products have been delivered.

Operation Identifier	getProductFiles		
Interaction Pattern	PROGRESS		
Pattern Sequence	Message	Nullable	Body Signature
IN	PROGRESS	No No	productRefs : (List <MAL::ObjectRef<Product>>) deliverTo : (MAL::URI)
OUT	ACK	No	Empty
OUT	UPDATE	No No No	productSummary : (ProductSummary) filename : (MAL::String) deliveredTo : (MAL::URI)
OUT	RESPONSE	No	Empty

3.2.7.2 Errors

In addition to standard MAL errors, the operation may return the following MPDS errors. (See section 5 for details of the secondary error code.)

The Delivery_Failed error occurs where an attempt to deliver a product file to the nominated address failed. ExtraInfo provides additional information on the reason for failure as a free format string.

Error	Error #	ExtraInfo Type	ExtraInfo Comment
Invalid	1	SecondaryErrorCodeEnum	A UInteger giving a secondary error code that details the reason for invalidity.
Delivery Failed	2	MAL::String	ExtraInfo provides additional information on the reason for failure as a free format string.

3.3 SERVICE: ORDER MANAGEMENT

3.3.1 SUMMARY

The Order Management Service, introduced in 2.5, is used to submit and manage **standing orders**. Standing orders specify a product filter to subscribe to a subset of automatically generated products and define the method and destination for product delivery.

.

Area Identifier	Service Identifier	Area Number	Service Number	Area Version
MPD	OrderManagement	6	2	1
Interaction Pattern	Operation Identifier		Operation Number	Capability Set
REQUEST	listStandingOrders		1	
REQUEST	submitStandingOrder		2	1
SUBMIT	cancelStandingOrder		3	

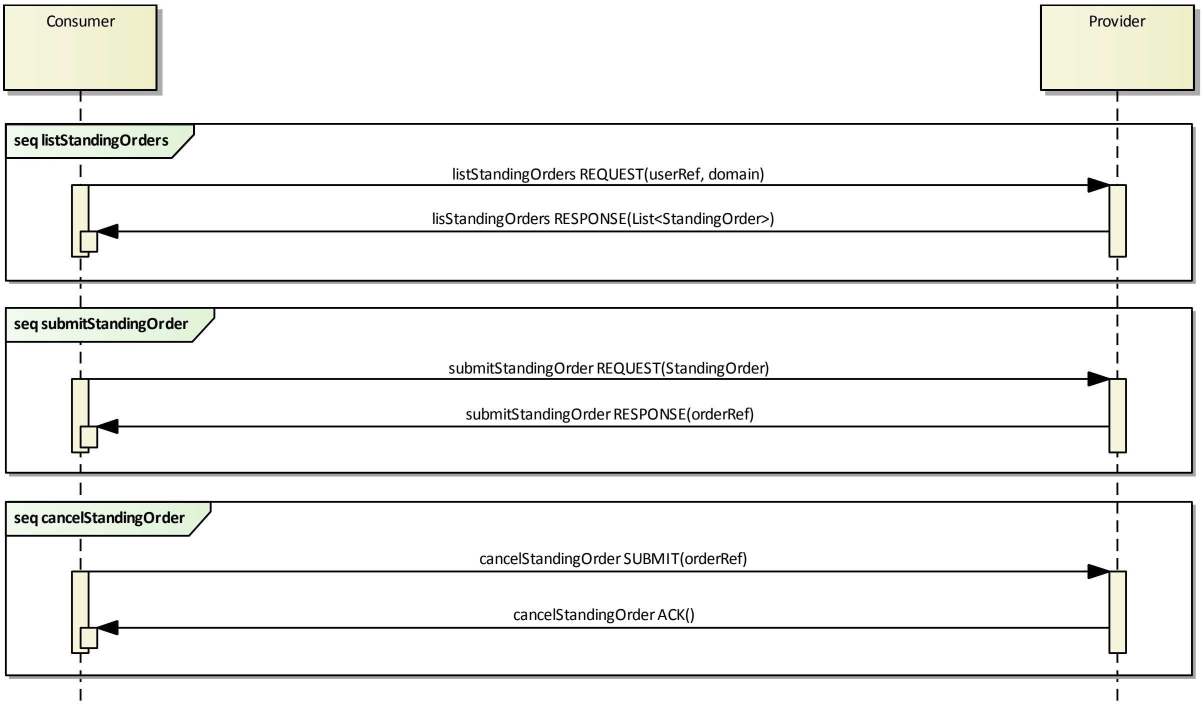
Consumer
Provider
seq listStandingOrders
listStandingOrders REQUEST(userRef, domain)
lisStandingOrders RESPONSE(List<StandingOrder>)
seq submitStandingOrder
submitStandingOrder REQUEST(StandingOrder)
submitStandingOrder RESPONSE(orderRef)
seq cancelStandingOrder
cancelStandingOrder SUBMIT(orderRef)
cancelStandingOrder ACK()

Figure 3-2: Order Management Operations

The listStandingOrders operation enables the service consumer to obtain a list of existing **standing orders** for a specified **domain** and/or product **user**.

The submitProductOrder operation enables the service consumer to create a new product order.

The cancelProductOrder operation enables the service consumer to cancel an existing product order.

NOTE – The service provider may choose to restrict the listProductOrders and cancelProductOrder operations to product orders that are owned by the user requesting these operations in order to protect the data of other service users.

3.3.2 MPDS DATA ITEMS

MPDS data items and their relationships relevant to the product order management service are defined within the MPDS information model in 4.2.

The following data items are directly applicable to the service (see figure 4-4):

- StandingOrder.

StandingOrders specify a **standing order** for a filtered set of **products** of a specified **product type** to be delivered when they become available, together with the method and destination for delivery.

The following MO objects and other data items may be referenced in the context of the order management service:

- ProductType: contains the definition of metadata parameters specific to a type of mission data product;
- User: identifies the owner of a **standing order**.

3.3.3 HIGH-LEVEL REQUIREMENTS

The following set of mission data product configuration data shall be available to both provider and consumers in a any deployment of the product order management service:

Product definitions (as ProductType objects [4.2.2]).

3.3.4 FUNCTIONAL REQUIREMENTS

3.3.4.1 In response to a listStandingOrders operation, the service provider shall compile a list of **standing orders** that satisfy the specified criteria and return this list (as StandingOrder data structures) to the service consumer.

NOTE – Depending on any mission specific confidentiality policy, the list of **standing orders** may be restricted to those belonging to the user associated with the service consumer.

3.3.4.2 On receipt of a Standing Order through a submitStandingOrder operation that has an empty orderID, the service provider shall create a new **standing** order, assign a unique orderID to this, and return the orderID to the service consumer.

3.3.4.3 On receipt of a Standing Order through a submitStandingOrder operation that has an specified orderID, the service provider shall verify the uniqueness of this orderID, before creating a new **standing order**. If the orderID already exists, an error is returned to the service consumer.

3.3.4.4 Following successful creation of a new **standing order** in response to a submitStandingOrder operation, the delivery of requested products shall be automatically commenced by the service provider from the start of the specified validity window. Where the specified delivery method is by file transfer, the delivery proceeds without further intervention from the service consumer. Where the specified delivery method is by service, delivery is subject to the operations of the ProductDelivery service.

3.3.4.5 If the end of the specified validity window is reached, the service provider shall cease delivery of products associated with the standing order.

3.3.4.6 In response to a cancelStandingOrder operation, the service provider shall immediately cease delivery of **products** associated with the **standing order** and shall delete the standing order.

3.3.5 OPERATION: LISTSTANDINGORDERS

3.3.5.1 Overview

The listStandingOrders operation can be used to obtain a list of existing **standing orders** relating to a specified **domain** and/or product **user**.

The response consists of a list of **standing orders** that satisfy the specified criteria. This may be restricted to those belonging to the service user requesting them.

The domain field in the request refers to the domain of the **products** to which the **standing order** relates. The standing order itself is not associated with a domain.

Operation Identifier	listStandingOrders		
Interaction Pattern	REQUEST		
Pattern Sequence	Message	Nullable	Body Signature
IN	REQUEST	Yes	user : (MAL::Identifier)
		Yes	domain : (List <MAL::Identifier>)
OUT	RESPONSE	No	standingOrders : (List <StandingOrder>)

3.3.5.2 Errors

In addition to standard MAL errors, the operation may return the following MPDS errors.

If the referenced item (user or domain) does not exist, the Unknown error is returned. ExtraInfo contains the specified ID that does not exist.

Error	Error #	ExtraInfo Type	ExtraInfo Comment
Unknown	4	MAL::Identifier	ExtraInfo contains the specified user or domain ID that does not exist.

3.3.6 OPERATION: SUBMITSTANDINGORDER

3.3.6.1 Overview

The submitStandingOrder operation can be used by the service consumer to create a new **standing order**.

The service consumer sends an StandingOrder structure in the Request message to the service.

In most implementations, the service provider is responsible for assigning a unique ID for the **standing order**. In this case, the orderID field of the submitted StandingOrder structure shall be set to an empty string. The orderID assigned by the service provider is then returned in the Response message as a MAL::Identifier.

In some implementations, the service consumer may be responsible for assigning a unique ID for the **standing order**. In this case, the orderID field of the submitted StandingOrder structure is set to the required identifier. The service provider checks the uniqueness of the submitted orderID and where this already exists returns an ID_Already_Used error.

Operation Identifier	submitStandingOrder		
Interaction Pattern	REQUEST		
Pattern Sequence	Message	Nullable	Body Signature
IN	REQUEST	No	orderDetails : (StandingOrder)
OUT	RESPONSE	No	orderRef : (MAL::Identifier)

3.3.6.2 Errors

In addition to standard MAL errors, the operation may return the following MPDS errors. (See section 5 for details of the secondary error code.)

The Order_Failed error occurs where creation of a new product order failed. ExtraInfo provides additional information on the reason for failure as a free format string. If the order could not be created because a non-unique orderID was specified then the ExtraInfo field is set to ‘ID Already Exists’.

Error	Error #	ExtraInfo Type	ExtraInfo Comment
Invalid	1	SecondaryErrorCodeEnum	A UInteger giving a secondary error code that details the reason for invalidity.
Order Failed	3	MAL::String	ExtraInfo provides additional information on the reason for failure as a free format string. If the order could not be created because a non-unique orderID was specified then the ExtraInfo field is set to ‘ID Already Exists’.

3.3.7 OPERATION: CANCELSTANDINGORDER

3.3.7.1 Overview

The cancelStandingOrder operation is used by the service consumer to cancel an existing **standing order**.

The service consumer sends the orderID of the **standing order** to be cancelled to the service provider.

The authorization aspects of who is allowed to cancel which **standing orders** is not in the scope of this service specification, as this service specifies only the interface of a product distribution system. In a full system implementation the authorization is either handled by the implementation of respective rules in the Access Control element of MAL or as part of the service provider software implementation.

<table>
<tr><td colspan="2">Operation Identifier</td><td colspan="3">cancelStandingOrder</td></tr>
<tr><td colspan="2">Interaction Pattern</td><td colspan="3">SUBMIT</td></tr>
<tr><td>Pattern Sequence</td><td>Message</td><td>Nullable</td><td colspan="2">Body Signature</td></tr>
<tr><td>IN</td><td>SUBMIT</td><td>No</td><td colspan="2">orderRef : (MAL::Identifier)</td></tr>
</table>

3.3.7.2 Errors

In addition to standard MAL errors, the operation may return the following MPDS errors.

If the referenced orderID does not exist, then the Unknown error is returned. ExtraInfo contains the specified ID that does not exist.

Error	Error #	ExtraInfo Type	ExtraInfo Comment
Unknown	4	MAL::Identifier	ExtraInfo contains the specified order ID that does not exist.

3.4 SERVICE: PRODUCT DELIVERY

3.4.1 SUMMARY

The Product Delivery Service, introduced in 2.5, is used in conjunction with existing **standing orders** (for the delivery of selected **products**). File delivery of products, if requested in the standing order, will be delivered without the need to use these service operations.

The service provides two capabilities to the consumer:

- to receive notifications each time their ordered products have been delivered as files;
- to receive the ordered products via message transfer rather than file delivery.

Area Identifier	Service Identifier	Area Number	Service Number	Area Version
MPD	ProductOrderDelivery	6	3	1
Interaction Pattern	Operation Identifier		Operation Number	Capability Set
PUBSUB	notifyProductDelivery		1	1
PUBSUB	deliverProducts		2	

The notifyProductDelivery operation enables the service consumer to obtain notification following the delivery of a product by file transfer.

The deliverProducts operation is for use in conjunction with an existing standing order that has specified product delivery via service messages. Once subscribed, the service consumer will receive products via service messages.

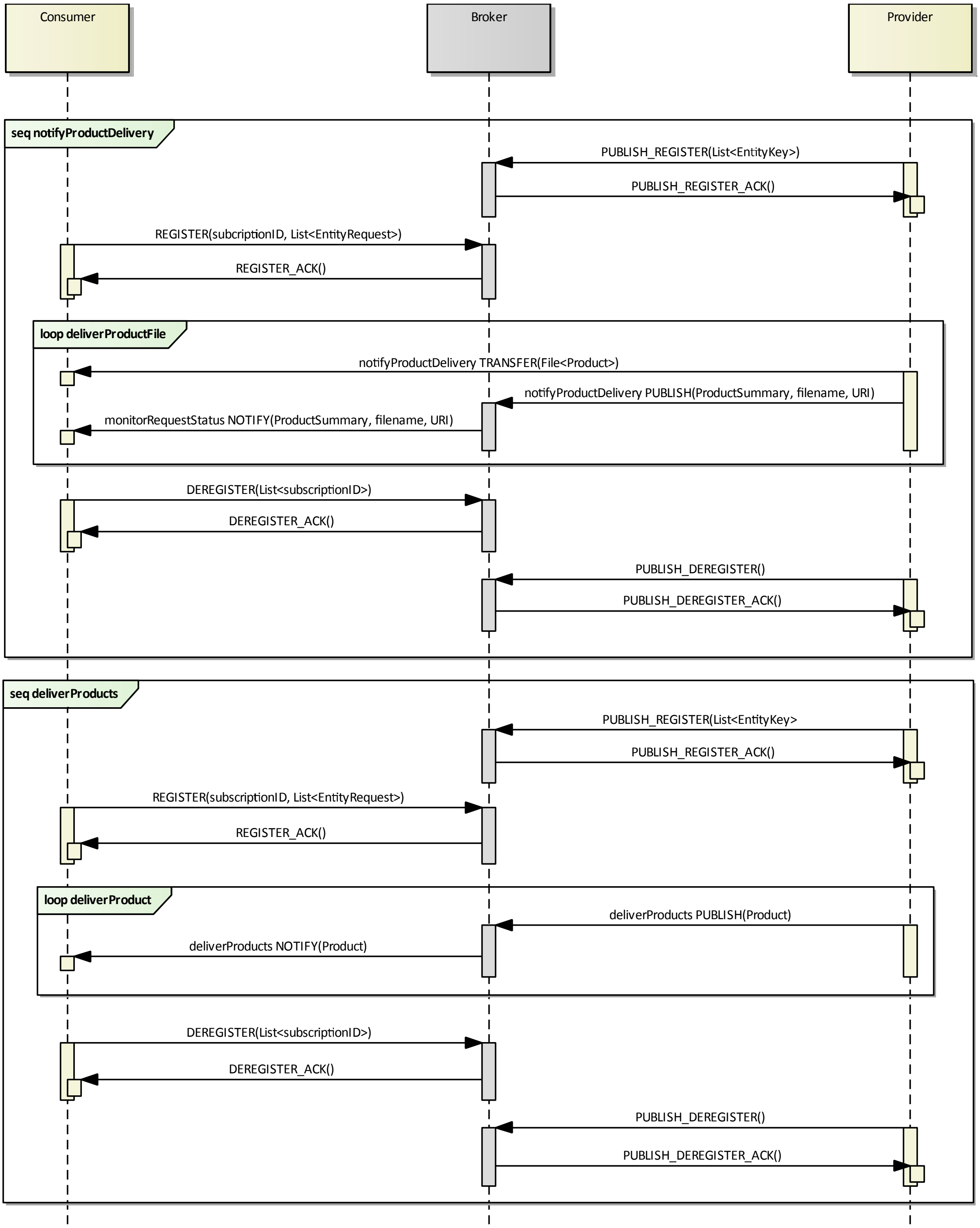
Consumer
Broker
Provider
seq notifyProductDelivery
PUBLISH_REGISTER(List<EntityKey>)
PUBLISH_REGISTER_ACK()
REGISTER(subcriptionID, List<EntityRequest>)
REGISTER_ACK()
loop deliverProductFile
notifyProductDelivery TRANSFER(File<Product>)
notifyProductDelivery PUBLISH(ProductSummary, filename, URI)
monitorRequestStatus NOTIFY(ProductSummary, filename, URI)
DEREGISTER(List<subscriptionID>)
DEREGISTER_ACK()
PUBLISH_DEREGISTER()
PUBLISH_DEREGISTER_ACK()
seq deliverProducts
PUBLISH_REGISTER(List<EntityKey>
PUBLISH_REGISTER_ACK()
REGISTER(subcriptionID, List<EntityRequest>)
REGISTER_ACK()
loop deliverProduct
deliverProducts PUBLISH(Product)
deliverProducts NOTIFY(Product)
DEREGISTER(List<subscriptionID>)
DEREGISTER_ACK()
PUBLISH_DEREGISTER()
PUBLISH_DEREGISTER_ACK()

Figure 3-3: Product Delivery Operations

3.4.2 MPDS DATA ITEMS

MPDS data items and their relationships relevant to the product order delivery service are defined within the MPDS information model in 4.2.

The following MO objects are directly applicable to the service (see figure 4-2):

- Product.

Products comprise metadata that describes the product (but not its structure) and the product body. The structure of the product body is unknown to the service interface, but must be known to the service user.

The following MO objects and other data items may be referenced in the context of the product order delivery service:

- StandingOrder: specifies a standing order for a filtered set of products of a specified product type to be delivered when they become available, together with the method and destination for delivery.
- ProductType: contains the definition of metadata parameters specific to a type of mission data product.
- User: identifies the owner of a **standing order**

3.4.3 HIGH-LEVEL REQUIREMENTS

The following set of mission data product configuration data shall be available to both provider and consumers in a any deployment of the product order delivery service:

Product definitions (as ProductType objects [4.2.2]).

NOTE – The service consumer only needs to be aware of its own User identity.

3.4.4 FUNCTIONAL REQUIREMENTS

3.4.4.1 In response to subscription using the notifyProductDelivery operation, the service provider shall forward delivery notifications to the service consumer following the successful transfer of each product file that is associated with the specified **standing order,** or all **standing orders** associated with the specified **user**.

NOTE – The provision of delivery notifications continues until the service consumer deregisters or the service session is terminated. It is implementation dependent whether notifications relating to product deliveries that occur when the service consumer is not subscribed are queued awaiting subscription or discarded.

3.4.4.2 In response to subscription using the deliverProducts operation, the service provider shall forward **products** associated with the specified **standing order,** or all **standing orders** associated with the specified **user,** to the service consumer as they become available.

NOTE – The provision of delivery notifications continues until the service consumer deregisters or the service session is terminated. It is implementation dependent whether **products** that become available when the service consumer is not subscribed are queued for delivery awaiting subscription or ignored.

3.4.5 OPERATION: NOTIFYPRODUCTDELIVERY

3.4.5.1 Overview

The notifyProductDelivery operation can be used to receive notification when **products** have been delivered by file transfer or email in accordance with an existing **product order**.

The service provider publishes the delivery notification together with subscription keys corresponding to the identities of the associated user and **standing order**. The service consumer can subscribe to receive notifications for individual standing orders, or for all product orders belonging to a user.

The service consumer then receives a notification, in the form of a productSummary together with the filename and URI to which it was transferred, each time delivery of a product file has been completed.

Operation Identifier	notifyProductDelivery
Interaction Pattern	PUBLISH-SUBSCRIBE
Subscription Keys	userID : (MAL::Identifier) orderID : (MAL::Identifier)

Pattern Sequence	Message	Nullable	Body Signature
OUT	PUBLISH/NOTIFY	No	productSummary : (ProductSummary)
		No	filename : (MAL::String)
		No	deliveredTo : (MAL::URI)

3.4.5.2 Errors

In addition to standard MAL errors, the operation may return the following MPDS errors. (See section 5 for details of the secondary error code.)

The Delivery_Failed error occurs where an attempt to deliver a product file to the nominated address failed. ExtraInfo provides additional information on the reason for failure as a free format string. The Unknown error occurs if the supplied userID or orderID as a subscription key does not exist. The unrecognized Identifier is returned in the ExtraInfo field.

Error	Error #	ExtraInfo Type	ExtraInfo Comment
Invalid	1	SecondaryErrorCodeEnum	A UInteger giving a secondary error code that details the reason for invalidity.
Delivery Failed	2	MAL::String	ExtraInfo provides additional information on the reason for failure as a free format string.
Unknown	4	MAL::Identifier	ExtraInfo contains the specified user or order ID that does not exist.

3.4.6 OPERATION: DELIVERPRODUCTS

3.4.6.1 Overview

The deliverProducts operation can be used to receive **products** via the service interface where this has been specified as the delivery method for an existing **standing order**.

The service provider publishes the **product** together with subscription keys corresponding to the identifiers of the associated user and **standing order**. The service consumer can subscribe to receive products relating to individual standing orders, or to all product orders belonging to a user.

The service consumer then receives the product each time it becomes available.

Operation Identifier	deliverProducts
Interaction Pattern	PUBLISH-SUBSCRIBE
Subscription Keys	userID : (MAL::Identifier) orderID : (MAL::Identifier)

Pattern Sequence	Message	Nullable	Body Signature
OUT	PUBLISH/NOTIFY	No	product : (Product)

3.4.6.2 Errors

In addition to standard MAL errors, the operation may return the following MPDS errors. (See section 5 for details of the secondary error code.)

The Unknown error occurs if the supplied userID or orderID as a subscription key does not exist. The unrecognized Identifier is returned in the ExtraInfo field.

Error	Error #	ExtraInfo Type	ExtraInfo Comment
Invalid	1	SecondaryErrorCodeEnum	A UInteger giving a secondary error code that details the reason for invalidity.
Unknown	4	MAL::Identifier	ExtraInfo contains the specified user or order ID that does not exist.

4 MPDS INFORMATION MODEL

4.1 GENERAL

This section defines the information model applicable to the Mission Product Distribution Services standard, introducing and defining major data elements in line with the terminology used throughout this document. An overview of the MPDS information model has been given in 2.4 above.

The MPDS information model has been defined in terms of the CCSDS Mission Operations (MO) framework, specifically the MO Message Abstraction Layer (MAL) and the associated set of MAL Attribute types. This is to enable the specification of MO compliant data formats and services that reference elements of the information model.

While the information model itself is not normative, the data formats derived from it are normative and specified in this section as MPD data type definition tables consistent with the MO framework and any associated requirements. The information model diagrams contained in this section are expressed in UML (reference [6]) and are informative. The diagrams and tables follow the conventions introduced in 1.7.

The MPDS information model describes both the data actively exchanged by MPD Services and that required as common configuration data by service providers and users. The structure of MPDS MO objects is fully defined within this document, together with their relationships.

The section is organized into the following main sections:

a) this introduction;

b) MPDS Data Items;

c) MPDS Data Types.

The principle MPDS Data Items defined are those introduced in 2.4 that correspond to a set of MO objects and other data items that can be directly referenced in the context of the MPD services:

- **Products**;
- **Standing Orders**.

MPDS Data Types are supporting data structures used in the context of MPDS Data Items and MPDS service messages:

- MO MAL Base Data Types;
- additional MPDS Data Types;
- Parameters.

4.2 MPDS DATA ITEMS

4.2.1 GENERAL

MPDS data items are the principal elements of the MPDS information model that are represented as MO objects. These have been introduced in 2.4 and are shown in figure 4-1 below. The following data items are defined in the MPS information model:

– Products;

– Standing Orders.

Each of these is described in turn in the following subsections.

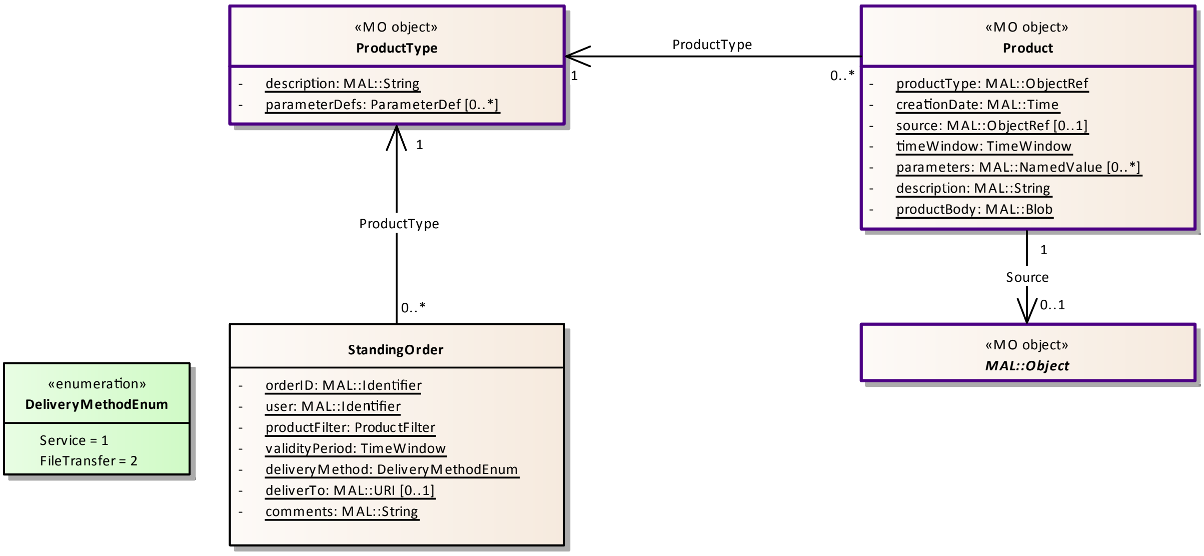
«MO object»
ProductType
- description: MAL::String
- parameterDefs: ParameterDef [0..*]
ProductType
1
0..*
«MO object»
Product
- productType: MAL::ObjectRef
- creationDate: MAL::Time
- source: MAL::ObjectRef [0..1]
- timeWindow: TimeWindow
- parameters: MAL::NamedValue [0..*]
- description: MAL::String
- productBody: MAL::Blob
1
ProductType
0..*
1
Source
0..1
«MO object»
MAL::Object
«enumeration»
DeliveryMethodEnum
Service = 1
FileTransfer = 2
StandingOrder
- orderID: MAL::Identifier
- user: MAL::Identifier
- productFilter: ProductFilter
- validityPeriod: TimeWindow
- deliveryMethod: DeliveryMethodEnum
- deliverTo: MAL::URI [0..1]
- comments: MAL::String

Figure 4-1: MPDS Data Items

Mission Data **products** comprise a common pattern of two distinct MO Object types: a **definition** ProductType and an **instance** Product.

Each MO object has a unique object **identity**, which includes an unchanging key (Identifier) and optionally a **version**. When a **definition** object is updated, it retains the same key, but its **version** is updated to uniquely identify a specific version of the definition.

The **definition** objects associated with **products** (ProductType) form the configuration data that must be available to both communicating parties that exchange products. Definition objects comprise only static attributes. The MPD services defined in this standard do not address the bulk transfer of MPDS configuration data between communicating parties.

The **instance** objects associated with **products** (Product) are the actual mission data products exchanged between communicating parties. Although dynamically created, these comprise only static attributes assigned at generation time.

Standing orders are represented by a single data structure: StandingOrder. These may be dynamically created, either locally at the service provider or by a consumer of the Order Management service.

The following subsections contain the definition of each MPDS Data Item and are structured as follows:

- MO Objects or other data structures defined for the Data Item itself;
- Service-specific Data Types defined for the Data Item (these may be used in the context of MPDS service messages).

4.2.2 MISSION PRODUCTS

4.2.2.1 Mission Product Objects

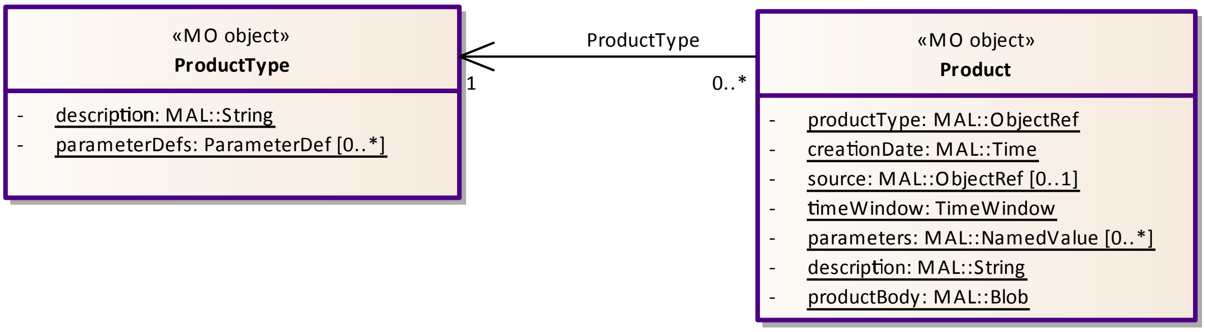
«MO object»
ProductType
- description: MAL::String
- parameterDefs: ParameterDef [0..*]
ProductType
1
0..*
«MO object»
Product
- productType: MAL::ObjectRef
- creationDate: MAL::Time
- source: MAL::ObjectRef [0..1]
- timeWindow: TimeWindow
- parameters: MAL::NamedValue [0..*]
- description: MAL::String
- productBody: MAL::Blob

Figure 4-2: Mission Product Objects

Mission **products** are the main data item of the information model, comprising product metadata, and a product body that has a structure that must be known to both product provider and consumer, but is opaque to the service itself. They follow the MO Instance object pattern, comprising **definition,** and **instance** classes of MO object.

ProductTypes provide the **definition** of the product metadata. They imply the structure of the product body and define a set of metadata parameters that are associated with each **instance** of the product type.

Products correspond to the product **instances** of a defined product type and relate to a specific occurrence of a generated product. They comprise a set of metadata, including parameters specific to the product type, and the product body.

ProductType

A ProductType is an MO object that contains the static definition associated with multiple occurrences of a mission data **product**. Its **identity** includes a constant **key** and an evolving **version** that is updated each time the definition is revised. The ProductType defines the metadata parameters associated with the product and implies (but does not specify) the structure of the product body.

NOTES

1 The **domain** of the ProductType and associated Products may not be the same. Typically the domain of generic ProductTypes (for example an orbit file) may be defined at agency level, while the generated Products may be associated with a mission specific domain.

2 The ProductTypes are not contained within the service messages of any operation defined in this specification but are referenced by Products and other service data structures. A future service may support the distribution and access to mission product configuration data, including ProductTypes.

Name	**ProductType**		Extends	MAL::Object	SFP	101
Attribute	Type	Nullable	Description			
identity	MAL::ObjectIdentity	No	Identity of the ProductType, including version.			
description	MAL::String	No	Description of the Product Type.			
parameterDefs	List <ParameterDef>	Yes	List of metadata Parameter Definitions.			

Product

A Product is an MO object that corresponds to a specific occurrence of a generated mission data **product**. It comprises both metadata about the product and the product body itself, whose structure is opaque to the MPD services but known to both the product generator and consumer, given the referenced ProductType.

Name	**Product**		Extends	MAL::Object	SFP	102
Attribute	Type	Nullable	Description			
identity	MAL::ObjectIdentity	No	Identity of the Product.			
productType	MAL::ObjectRef <ProductType>	No	Reference to the ProductType.			
creationDate	MAL::Time	No	Date and time at which the product was generated.			
source	MAL::ObjectRef	Yes	Reference to an MO Object that is associated with triggering the generation of the product, for example an MPS planning request, an event, or an M&C alert.			
timeWindow	TimeWindow	No	Period of time to which the source data used to generate the product relates.			
parameters	List<MAL::NamedValue>	Yes	Named values for metadata parameters whose name and type correspond to those defined in the referenced ProductType.			
description	MAL::String	No	Free text description of this specific occurrence of the product.			
productBody	MAL::Blob	No	Product body of opaque structure known to both product generator and product user.			

4.2.2.2 Mission Product Service Structures

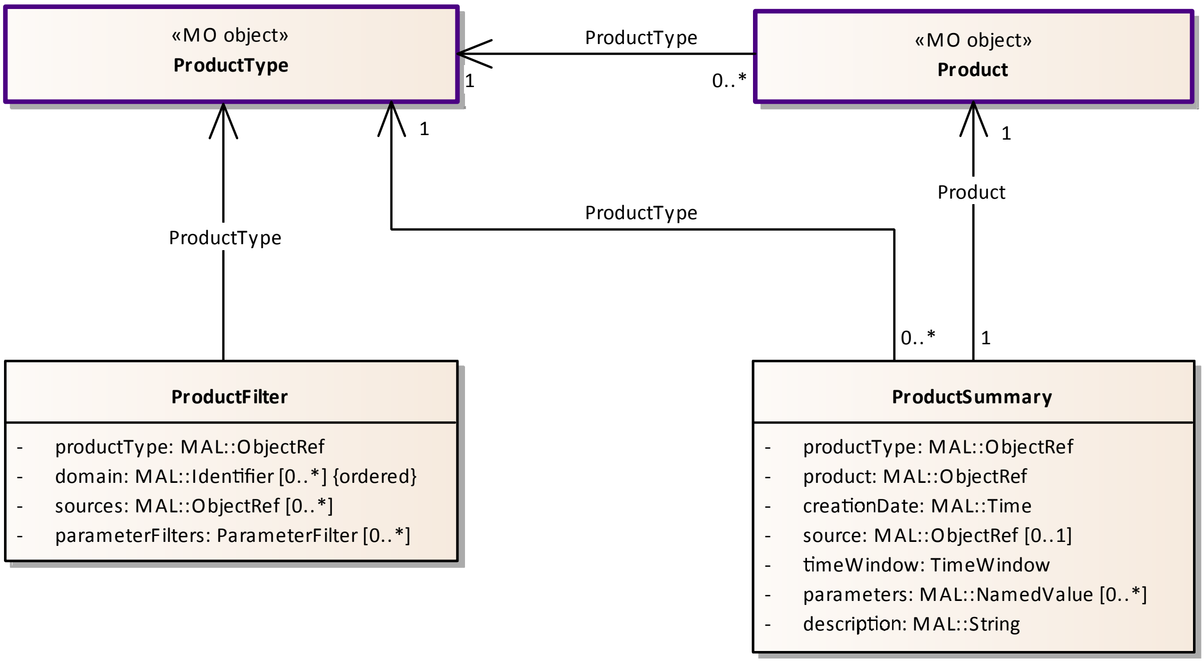
«MO object»
ProductType
ProductType
1
0..*
«MO object»
Product
1
1
Product
ProductType
ProductType
0..*
1
ProductFilter
- productType: MAL::ObjectRef
- domain: MAL::Identifier [0..*] {ordered}
- sources: MAL::ObjectRef [0..*]
- parameterFilters: ParameterFilter [0..*]
ProductSummary
- productType: MAL::ObjectRef
- product: MAL::ObjectRef
- creationDate: MAL::Time
- source: MAL::ObjectRef [0..1]
- timeWindow: TimeWindow
- parameters: MAL::NamedValue [0..*]
- description: MAL::String

Figure 4-3: Mission Product Service Structures

In addition to the MO Objects, a set of data structures are defined that reference these objects and are used to convey information about them in the context of service messages.

ProductSummary

The ProductSummary comprises the product metadata without the product body and is used when returning a list of available **products** for retrieval.

A ProductSummary relates to a specific Product instance.

Name	**ProductSummary**		Extends	MAL::Composite	SFP	103
Attribute	Type	Nullable	Description			
productType	MAL::ObjectRef <ProductType>	No	Reference to the ProductType definition			
product	MAL::ObjectRef <Product>	No	Reference to the Product instance			
creationDate	MAL::Time	No	Date and time at which the product was generated.			
source	MAL::ObjectRef	Yes	Reference to an MO Object that is associated with triggering the generation of the product, for example an MPS planning request, an event, or an M&C alert.			
timeWindow	TimeWindow	No	Period of time to which the source data used to generate the product relates.			

Attribute	Type	Nullable	Description
parameters	List<MAL::NamedValue>	Yes	Named values for metadata parameters whose name and type correspond to those defined in the referenced ProductType.
description	MAL::String	No	Free text description of this specific occurrence of the product.

ProductFilter

The ProductFilter is used in the context of **standing orders** and service operations requesting a filtered list of available **products**. It specifies a filter in terms of productType, **domain**, sources, and metadata parameter values. To pass the filter, the product must satisfy all criteria specified: the productType, one of any listed source objects, and all specified metadata parameter filters. If no filter is specified for source or parameterFilters then all products of the specified type pass the filter.

A ProductFilter relates to a specific ProductType. It should be noted that the domain filter relates to the **domain** of the **product** and not of the **product type**. The domain of a generic product type (for example an orbit file) may be defined at agency level to avoid the need to define new types for each mission, but the generated product will be associated with a specific mission domain.

It should be noted that in the case of requesting a list of available products, filtering by a range of product creation or time window dates may also be required. These are specified as additional fields of the corresponding service message structure.

Name	**ProductFilter**		Extends	MAL::Composite	SFP	104
Attribute	Type	Nullable	Description			
productType	MAL::ObjectRef <ProductType>	Yes	Reference to the ProductType definition			
domain	List <MAL::Identifier>	Yes	An ordered list of hierarchical domain Identifiers, ‘*’ may be used as a wildcard.			
sources	List <MAL::ObjectRef>	Yes	Set of references to MO Objects associated with triggering the generation of filtered products, for example an MPS planning request, an event, or an M&C alert. If the product source matches one of the listed sources, then it passes the filter.			
parameterFilter	List <ParameterFilter>	Yes	Set of parameterFilters defining the desired values of product metadata parameters. If multiple parameterFilters are defined then the product metadata must match all specified criteria to pass the filter.			

4.2.3 STANDING ORDERS

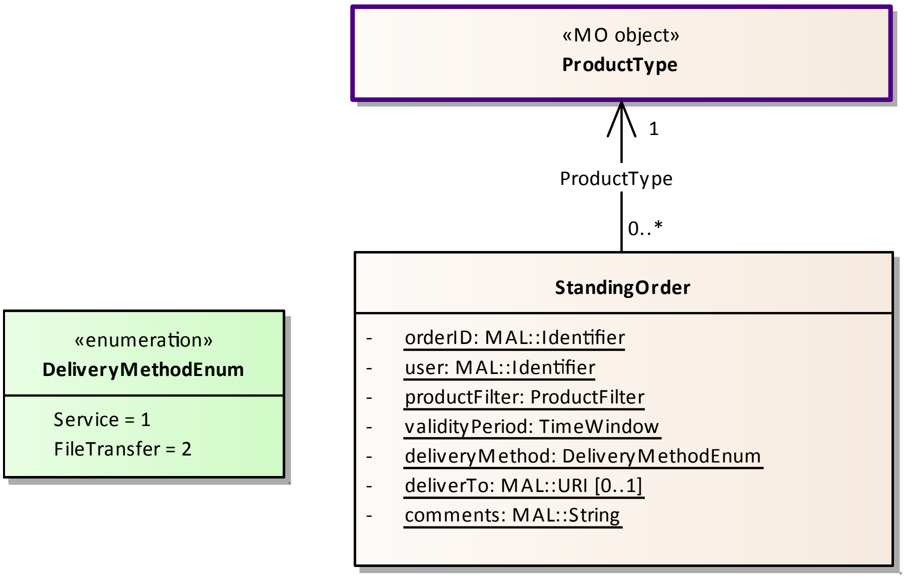

Figure 4-4: Standing Order Data Structures

Standing orders hold the specification for the automatic delivery of selected mission **products** as they become available. Each standing order relates to a specific **user** and a specific **product type.** Standing orders are represented by a single MAL::Composite data structure StandingOrder.

The identity of the StandingOrder is represented as a simple MAL::Identifier and assigned by the consumer.

StandingOrder

A StandingOrder is a data structure that holds the details of a **standing order**. This includes a reference to the **user** who owns the product order, the product filter, how the products are to be delivered to the user, and the current status of the order.

Name	**StandingOrder**		Extends	MAL::Composite	SFP	201
Attribute	Type	Nullable	Description			
orderID	MAL::Identifier	No	Identity of the StandingOrder.			
user	MAL::Identifier	No	Reference to the user that is the owner of the product order.			
productFilter	ProductFilter	Yes	Specifies the filter criteria for the standing order, including the product type and optional filters on product source and metadata parameters.			
validityPeriod	TimeWindow	Yes	Period of time over which the standing order is required to be active.			

Attribute	Type	Nullable	Description
deliveryMethod	DeliveryMethodEnum	No	Specifies how the product is to be delivered: via the service interface or by file transfer.
deliverTo	MAL::URI	Yes	Delivery address for file transfer. Not required if deliveryMethod is SERVICE.
comments	MAL::String	No	Any additional notes.

DeliveryMethodEnum

The current delivery methods are defined for mission products:

Name	**DeliveryMethodEnum**	SFP	202
Status	Value	Description	
SERVICE	1	The mission product is to be delivered via service messages. This requires the service consumer to initiate a service operation to subscribe to the products via the existing standing order.	
FILETRANSFER	2	The mission product is to be delivered by file transfer to the address contained in the deliverTo field.	

4.3 MPDS DATA TYPES

4.3.1 MPDS BASE DATA TYPES

MPDS data structures are defined in terms of the MO MAL. All data structures defined in this standard are ultimately derived from one of the following MAL data types:

- MAL::Object;
- MAL::Composite;
- MAL::Enumeration.

Fields within MPDS data structures are defined as one (or a list) of the following:

- MAL::Attribute;
- MAL defined data structure;
- MPDS defined data structure.

MAL::Attributes are the base data types on which all others are built. The following types are supported:

- Boolean;
- Float *32-bit floating point number*;

- Double*64-bit floating point number*;
- Octet *8-bit signed integer*;
- UOctet*8-bit unsigned integer*;
- Short *16-bit signed integer*;
- UShort*16-bit unsigned integer*;
- Integer*32-bit signed integer*;
- UInteger *32-bit unsigned integer*;
- Long *64-bit signed integer*;
- ULong*64-bit unsigned integer*;
- Duration *Length of time in seconds (may include a fractional component)*;
- Time *Absolute date and time to millisecond resolution*;
- FineTime *Absolute date and time to picosend resolution*;
- String *Text as a variable-length, unbounded, Unicode string*;
- Blob *Stores binary objects, it is a variable-length, unbounded, octet array*;
- Identifier *Stores Identifiers, a variable-length, unbounded, Unicode string*;
- URI *Stores URI addresses, a variable-length, unbounded, Unicode string*;
- ObjectRef *Reference to an MO object.*

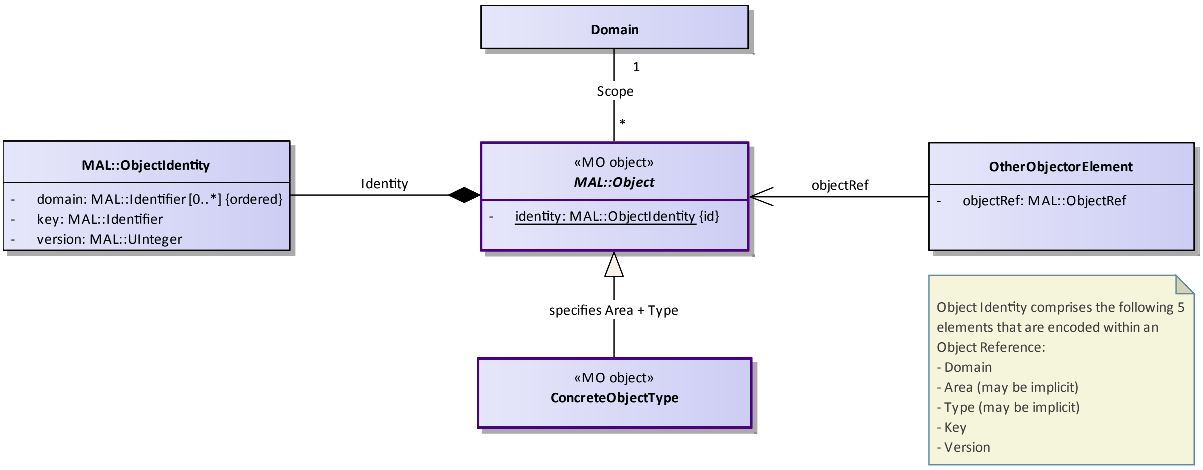

Figure 4-5: MO Objects

MAL::Object is defined as an abstract extension of MAL::Composite that contains a representation of the data structure of an MO object. It has one attribute, that is the **identity** of the MO object, represented as a MAL::ObjectIdentity composite data structure.

MAL::ObjectIdentity is defined as an extension of MAL::Composite that contains the **identity** of an MO object, including its **domain**, **key**, and **version**. The **area** and type of the MO object are implied by the derived concrete type of the object.

The MAL::ObjectRef attribute type has a similar conceptual structure but includes the **area** and type of the referenced object; its encoding is dependent the MAL binding used. MAL::ObjectRef can be used in two ways:

ObjectRef without a defined object type, in which case the **area** and type of the referenced MO object are explicitly encoded within the reference.

ObjectRef<T> with a defined object type, in which case the **area** and type of the referenced MO object are implicit and not encoded within the reference.

The latter allows for more compact encoding of the reference and should be used where the context only permits a single type of object to be referenced. If the reference can point to objects of multiple types or to an abstract type that has multiple concrete sub-types, then an untyped ObjectRef should be used.

The MAL also defines NamedValue as an extension of MAL::Composite that allows the specification of a name-value pair.

4.3.2 ADDITIONAL MPDS DATA TYPES

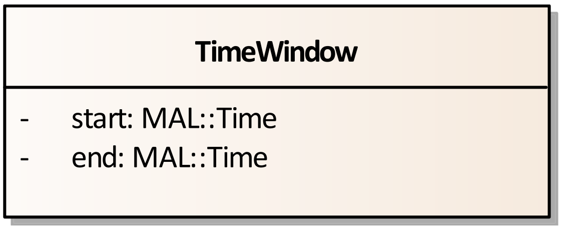

Figure 4-6: MPDS Additional Data Types

These correspond to general purpose data structures used within the MPDS information model.

TimeWindow

Represents a specific period of time, specified as two fields of type Time defining the start and end of the TimeWindow.

Name	**TimeWindow**		Extends	MAL::Composite	SFP	1
Attribute	Type	Nullable	Description			
start	MAL::Time	No	Start time of the time window.			

Attribute	Type	Nullable	Description
end	MAL::Time	No	End time of the time window.

4.3.3 MPDS PARAMETERS

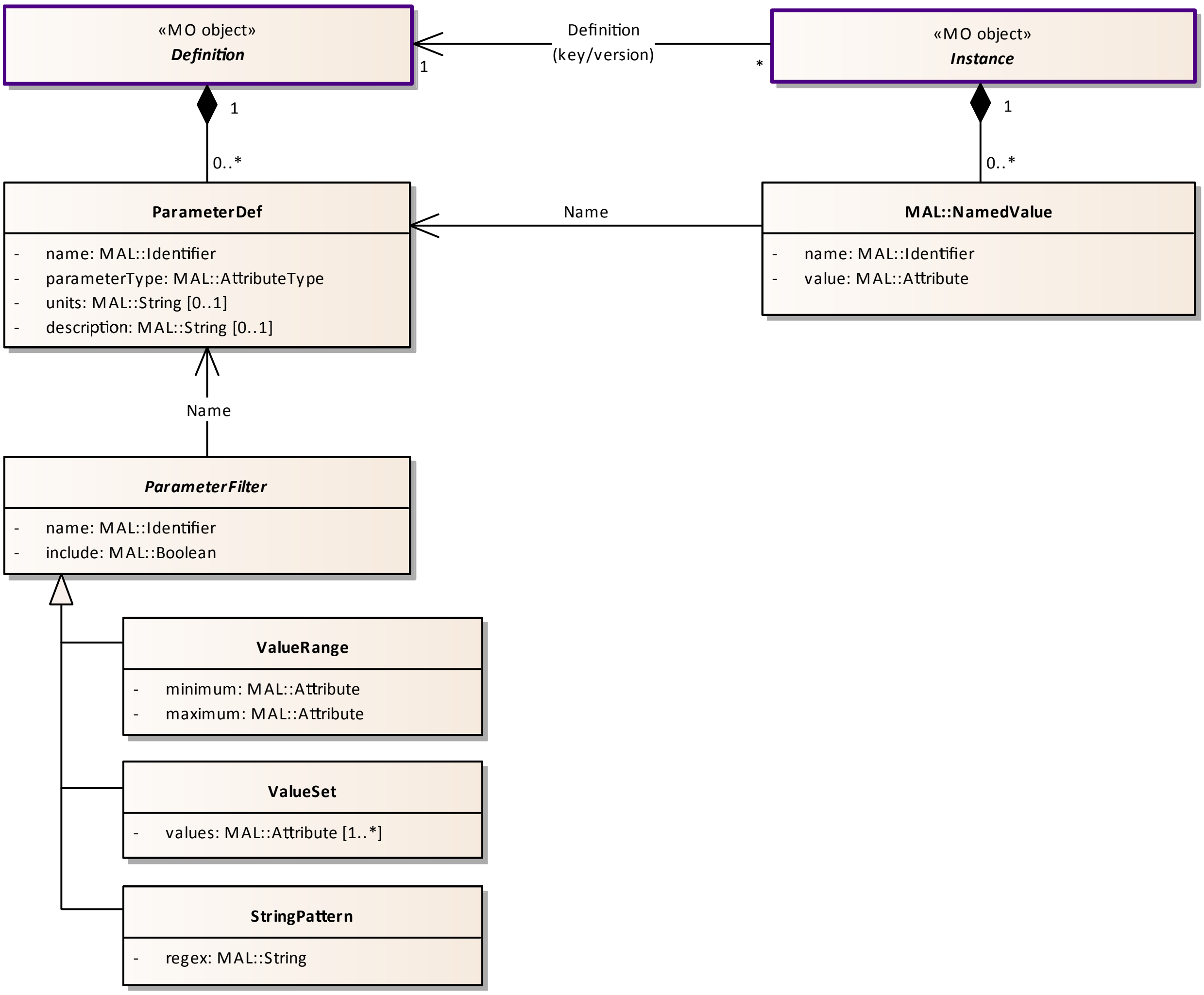

Figure 4-7: MPDS Parameters

MPDS Parameters are defined as a general feature that allows a user definable set of metadata parameters to be associated with a data item that follows the MO Instance object pattern, comprising **definition** and **instance** classes of MO object. In practice this only applies to mission **products**.

The **definition** class ProductType includes the definition of the set of metadata parameters associated with all **instances** (Product) of that **product type**. The metadata parameters are defined as a set of ParameterDefs that associate a name, MAL attribute type, and optionally units for each parameter.

The **instance** class Product includes the set of metadata parameters as standard MAL::NamedValue pairs.

A ParameterFilter is also defined to allow the specification of a filter based on the value of a metadata parameter. Different sub-types of ParameterFilter enable filtering by value range, a set of allowed values, or matching a string pattern.

ParameterDef

A ParameterDef specifies a metadata parameter in terms of its name, attribute type, optionally units, and a free text description.

It should be noted that as ParameterDef is only used in the context of ProductType, which is not contained in the service messages of any operation defined in this specification, it also is not contained in any currently defined service message. A future service may support the distribution and access to mission product configuration data, including ProductTypes and their contained ParameterDefs.

Name	**ParameterDef**		Extends	MAL::Composite	SFP	2
Attribute	Type	Nullable	Description			
name	MAL::Identifier	No	Assigns a name to the metadata parameter.			
parameterType	MAL::AttributeType	No	Specifies the MAL attribute type of the metadata parameter.			
units	MAL::String	Yes	Units associated with the metadata parameter (optional).			
description	MAL::String	Yes	Description of the metadata parameter (optional).			

ParameterFilter

A ParameterFilter enables specification of a filter based on the value of a metadata parameter. It is used in the context of selecting a subset of mission data products. It is an abstract data type, with concrete subtypes for specific types of filter.

Name	***ParameterFilter***		Extends	MAL::Composite	SFP	Abstract
Attribute	Type	Nullable	Description			
name	MAL::Identifier	No	References the name of a defined metadata parameter.			
include	MAL::Boolean	No	Indicates whether the filter is to include (TRUE) or exclude (FALSE) parameter values that match the filter.			

ValueRange

A ValueRange is a concrete subtype of ParameterFilter that allows the specification of an allowed (or disallowed) value range for a metadata parameter.

Name	**ValueRange**		Extends	ParameterFilter	SFP	3
Attribute	Type	Nullable	Description			
minimum	MAL::Attribute	Yes	Minimum value of the value range (greater than or equal to).			
maximum	MAL::Attribute	Yes	Maximum value of the value range (less than or equal to).			

ValueSet

A ValueSet is a concrete subtype of ParameterFilter that allows the specification of a set of allowed (or disallowed) values for a metadata parameter.

Name	**ValueSet**		Extends	ParameterFilter	SFP	4
Attribute	Type	Nullable	Description			
values	List <MAL::Attribute>	No	Set of allowed (or disallowed) values for the metadata parameter.			

StringPattern (Optional)

A StringPattern is a concrete subtype of ParameterFilter that allows the specification of a regular expression (or match pattern) to be searched for in the value of a text type metadata parameter.

Name	**StringPattern**		Extends	ParameterFilter	SFP	5
Attribute	Type	Nullable	Description			
regex	MAL::String	No	Regular expression - a sequence of characters that specifies a match pattern to be searched for in a text type metadata parameter (Sting, Identifier, or URI).			

5 ERROR CODES

Standard error codes defined by the MAL are applicable to the MPS service operations. In particular it is noted that this includes errors associated with delivery issues and authorization failure.

The following error codes shall apply to this specification.

Error	#	Description
Invalid	1	A field in the message contains an invalid value. If there are multiple errors, the first invalid field is reported.
Delivery Failed	2	An attempt to deliver a product file to the nominated address failed.
Order Failed	3	Creation of a new product order failed.
Unknown	4	The referenced item (user or orderID) does not exist.

SecondaryErrorCodeEnum

For the Invalid error, the extraInfo field contains a secondary error code defined as a UInteger that allows for deployment specific extensibility. The following standard secondary error codes are defined:

Name	**SecondaryErrorCodeEnum**	SFP	6
Error	Error #	Description	
UNKNOWN	0	Referenced MO object is not available to the service provider.	
UNDEFINED	1	Undefined value for enumeration field.	
OUT_OF_RANGE	2	A numeric value is outside the supported range.	
UNRECOGNIZED	3	Value of type MAL::Identifier or MAL::String (referencing a named item) does not correspond to a known item.	
BAD_TIME	4	A date-time value is outside the supported time period.	
INCONSISTENT	5	A value is inconsistent with that of another field within the message. This indicates violation of a constraint rule.	

6 MPDS XML SPECIFICATION

6.1 OVERVIEW

The MO MAL specification (reference [2]) defines a normative XML Schema Definition (XSD) for validating MO service specifications and the MAL XML specification. The use of XML for service specification provides a machine-readable format rather than the text-based document format (reference [4]).

The MPD service specification defined in this document is also represented as an XML specification that follows the MAL defined schema.

The published XML Schema Definition (XSD) and the service specifications are held in an online SANA registry (reference [5]) located at:

http://sanaregistry.org/r/moschemas/

6.2 XML SCHEMA DEFINITION (XSD) FOR MO SERVICES

6.2.1 The XML Schema Definition (XSD) that is used to validate the actual XML service specifications has a filename with the structure ‘ServiceSchema-vBBB.xsd’, where ‘BBB’ is replaced with the issue number of the corresponding document.

6.2.2 The normative XML for an MO service specification has a filename with the structure ‘areaAAA-vBBB-AREA’ where ‘AAA’ is replaced with the area number, ‘BBB’ is replaced with the area version which shall match the issue number of the corresponding document and ‘AREA’ is replaced by the area name.

6.2.3 For this specification the following version of the XML Schema Definition (XSD) is applicable:

https://sanaregistry.org/r/moschemas/ServiceSchema-v003.xsd

6.2.4 The latest version of the XML Schema Definition is directly available from the address:

https://sanaregistry.org/r/moschemas/ServiceSchema.xsd

6.3 MAL XML

6.3.1 The normative XML for the MAL specification, validated against the XML Schema Definition (XSD), is located at:

https://sanaregistry.org/r/moschemas/area001-v003-MAL.xml where 001 corresponds to the area number for MAL and 003 corresponds to the area version on which this specification is based.

6.3.2 The latest version of the MAL specification is directly available from the address:

https://sanaregistry.org/r/moschemas/ServiceDefMAL.xml

6.4 MPD XML

6.4.1 The normative XML for the MPD specification, validated against the XML Schema Definition (XSD), is located at[2]:

https://beta.sanaregistry.org/r/moschemas/area009-v001-MPD.xml

where 009 corresponds to the area number for MPS and 001 corresponds to the area version which shall match the issue number of this document.

6.4.2 The latest version of the MPS specification is directly available from the address:

https://beta.sanaregistry.org/r/moschemas/ServiceDefMPD.xml

[2] This Red book contains references to the beta SANA registry and will be changed to the formal SANA registry on publication as a Blue Book

ANNEX A

PROTOCOL IMPLEMENTATION CONFORMANCE STATEMENT (PICS) PROFORMA

(NORMATIVE)

A1 INTRODUCTION

A1.1 OVERVIEW

This annex provides the Protocol Implementation Conformance Statement (PICS) Requirements List (RL) for an implementation of the Mission Operations MPDS standard. The PICS for an implementation is generated by completing the RL in accordance with the instructions below. An implementation claiming conformance must satisfy the mandatory requirements referenced in the RL.

The MO MPDS standard includes optional elements, as outlined in 2.7. These comprise:

- Optional services and service capability sets as summarized in 2.7. A compliant deployment can support any combination of MO MPD services. If a service is supported, then it must support any mandatory service capability sets; other capability sets are optional.

An implementation's completed RL is called the PICS. The PICS states which protocol features have been implemented. The following entities can use the PICS:

- the protocol implementer, as a checklist to reduce the risk of failure to conform to the standard through oversight;
- the supplier and acquirer or potential acquirer of the implementation, as a detailed indication of the capabilities of the implementation, stated relative to the common basis for understanding provided by the standard PICS proforma;
- the user or potential user of the implementation, as a basis for initially checking the possibility of interworking with another implementation (while interworking can never be guaranteed, failure to interwork can often be predicted from incompatible PICSes);
- a protocol tester, as the basis for selecting appropriate tests against which to assess the claim for conformance of the implementation.

A1.2 NOTATION

A1.2.1 Status Column Symbols

The following are used in the RL to indicate the status of features:

Symbol	Meaning
M	Mandatory
C	Conditional
O	Optional

The Conditional status is used when a feature is mandatory only if a higher-level Optional feature has been selected. In the context of this standard, it is used for a mandatory capability set within an optional service; the capability set is only mandatory if the service itself is selected.

A1.2.2 Support Column Symbols

The support of every item as claimed by the implementer is stated by entering the appropriate answer (Y, N, or N/A) in the support column.

Symbol	Meaning
Y	Yes, supported by the implementation
N	No, not supported by the implementation
N/A	Not applicable

A2 GENERAL INFORMATION

A2.1 IDENTIFICATION OF PICS

Ref	Question	Response
1	Date of Statement (DD/MM/YYYY)	
2	CCSDS document number containing the PICS	
3	Date of CCSDS document containing the PICS	

A2.2 IDENTIFICATION OF IMPLEMENTATION UNDER TEST (IUT)

Ref	Question	Response
1	Implementation name	
2	Implementation version	
3	Machine name	
4	Machine version	
5	Operating System name	

6	Operating System version	
7	Special Configuration	
8	Other Information	

A2.3 USER IDENTIFICATION

Supplier	
Contact Point for Queries	
Implementation name(s) and Versions	
Other Information Necessary for full identification, e.g., name(s) and version(s) for machines and/or operating systems; System Name(s)	

A2.4 INSTRUCTIONS FOR COMPLETING THE RL

An implementer shows the extent of compliance to the protocol by completing the RL; the resulting completed RL is called a PICS.

A3 MPS SERVICES PICS

The MPS RL has an entry for each service and service capability set.

There are 3 separate MPD services defined. Each service is shown as a top level item, with subsidiary items for each capability set, using the format *s.c* in the Item column, where *s* is the service number, and *c* is the capability set number. All services are optional, but if implemented, some capability sets are mandatory. As each MPD service has a single capability set, this is mandatory if the service is implemented. Compliance with a service implies compliance with the service operations and message structures defined in 3, together with applicable high-level, functional, and structural requirements defined therein. For capability sets, the service operations it comprises are listed in the Protocol Feature column.

While a compliant deployment shall support the full structure of messages exchanged at the service interface for supported services and capability sets, it is not required to support data structures that are not used within the supported services and capability sets.

Item	Protocol Feature	Reference	Status	Support
1	Product Retrieval Service	3.2	O	
1.1	listProducts getProduct getProductFile	3.2	C	
2	Order Management Service	3.3	O	

2.1	listStandingOrders submitStandingOrder cancelStandingOrder	3.3	C	
3	Product Delivery Service	3.4	O	
3.1	notifyProductDelivery deliverProducts	3.4	C	

ANNEX B

SECURITY, SANA, AND PATENT CONSIDERATIONS

(INFORMATIVE)

B1 SECURITY CONSIDERATIONS

B1.1 SYSTEM SECURITY REQUIREMENTS

Security requirements are specific to the deployed mission system and can vary significantly between different mission systems. The MPD services or file formats support a limited subset of the interactions supported by a typical mission system, and as such must be capable of deployment in the context of a mission or organization specific security architecture that supports multiple services.

The mission security architecture shall address the following:

- Protection of the communications link between MPD service consumer and provider to ensure data integrity and confidentiality. This may or may not include the encryption of service messages, depending on mission specific requirements.
- Control of access to specific MPD services, service operations, and service data through the management of access rights associated with registered service users.
- Authentication to ensure only genuine registered service users have access to MPD services and to ascertain their level of access rights.

For the MPD services, the security considerations of this specification are the same as those of the MAL in reference [2]. Specifically, authentication and authorization of a participating consumer or provider is provided by the MAL access control concept and is covered in subsections 3.6, 5.2, and 5.3 of the Reference Model (reference [1]).

Security of the communications link carrying the MPS services is delegated to the implementation of the underlying Transport Layer.

B1.2 POTENTIAL THREATS

The mission data **products** generated by mission systems may include information about the status of the mission and/or data acquired by the mission. This data may be sensitive and therefore restricted to those authorized to view it. MPD services could be used by unauthorized users to view confidential data.

- MPD services could be used to view sensitive information about the status of the mission by those with an intent to sabotage the mission.

- MPD services could be used to access confidential mission **products** where data has been acquired for a specific user, or a product generated for that user, where the data itself, or even the fact that it exists is confidential. For example, Earth observation data may relate to a specific imaging request, for which the target is confidential for security or commercial reasons.

- MPD services could be used to cancel a **standing order** denying access to **products** by an authorized user.

B1.3 ACCESS CONTROL

The MPD services are closely tied to the Access Control aspect of the MAL where obtained authentication identifiers are used in the MAL message header to authenticate and authorize messages via Access Control.

Registered users are assigned roles (access rights) that may limit their access to MPD services. The set of access control roles is specific to the service deployment. An implementation of the MPD service provider can than restrict access to services, service capability sets, and individual service operations based on the assigned roles. Similarly roles can be used to restrict access to a subset of service data, either by data class or **domain**.

Which access control roles are supported is specific to the mission deployment and depends on the access control requirements for the mission. Typical MPD roles include access to:

- Individual MPD Services: as each service has a single capability set, the need to restrict access at a lower level than service is unlikely;

- A specific subset of **products**: not all users may have access to all defined products, but only a restricted set appropriate to their needs;

- Data class or **domain** of contained information (for example to restrict access to a specific spacecraft, subsystem, or payload in terms of available **products**);

- Information pertaining to other users: access for users may be restricted to their own *standing orders* or *products*.

It is the responsibility of the implementation of the MPD service provider to enforce access control based on the assigned user roles.

B1.4 DATA INTEGRITY

As stated previously, the confidentiality and integrity of MPD service messages is delegated to the implementation of the underlying transport layer.

This is dependent on the technology used for the implementation of the transport layer and the corresponding MAL technology binding. It may include the encryption of the service messages.

B1.5 AUTHENTICATION

Authentication for the MPD Services, as for all MO Services, may be supported through the MO Common Login Service (reference [D2]).

The Login service allows a service user to provide authentication information to the system. It takes the user's credentials and uses a deployment-specific mechanism to authenticate the user; the result of this is used by the MAL during access control.

The Login service and the access control provided by the MAL are fully dependent on a deployment-specific security architecture (for example, the authentication protocol Kerberos). Both layers (Common and MAL) provide access to, and use of, this security service; they do not implement it themselves.

B1.6 CONFIDENTIALITY

For some missions, there may be commercial or security considerations that result in a need for confidentiality of mission **products**.

Where this is the case, the MPD service provider may be required to implement an access control filter on the return of information to users. In particular:

– Access to a specific subset of **product types** (based on a specific access control role).

– Visibility of **products** may be restricted through the **domain** associated with the product (based on a specific access control role).

– Visibility of **products** may be restricted to those associated with the current user, through the product's 'source' reference.

It is mission and deployment specific how this is implemented, but it is expected that this would make use of special access control roles assigned to users.

B1.7 AUDITING

Storage of and access to service history is not directly supported by the MPD Service specification. In order to provide an audit trail for MPD services, a record of the following transactions should be maintained by the service provider:

– Delivery of a product to a specific user, whether via the Product Retrieval or Product Delivery service. This should include a reference to the **product**, the user to whom the product was delivered, the delivery method, and URI if applicable.

– Creation/deletion of **standing orders**.

B1.8 AVAILABILITY

Availability requirements are mission specific. The required availability of an MPD service provider implementation will impact its design, both in terms of the physical deployment architecture and its software implementation.

B2 SANA CONSIDERATIONS

The recommendations of this document request SANA populate the registry specified in reference [2] with the schema and XML detailed in section 6 of this document.

As stated in reference [2], the registration rule for change to this registry requires an engineering review by a designated expert. The expert shall be assigned by the M&C WG Chair, or in absence, MOIMS Area Director.

Specifically, this applies to the following registries:

http://sanaregistry.org/r/moschemas/ for entries relating to the MPD Service Specifications.

B3 PATENT CONSIDERATIONS

The recommendations of this document have no patent issues.

ANNEX C

DEFINITION OF ACRONYMS

(INFORMATIVE)

Acronym	Definition
APID	application process identifier
M&C	monitoring & control
MAL	Message Abstraction Layer
MO	mission operations
MOIMS	Mission Operations and Information Management Services
MPD	Mission Product Distribution (Area name)
MPDS	Mission Product Distribution Services
PI	principal investigator
PICS	protocol implementation conformance statement
SANA	Space Assigned Numbers Authority
SFP	short form part
TM	telemetry
UML	Unified Modelling Language
URI	Uniform Resource Identifier
XML	eXtensible Markup Language

ANNEX D

INFORMATIVE REFERENCES

(INFORMATIVE)

[D1] *Mission Operations Services Concept*. Issue 3. Report Concerning Space Data System Standards (Green Book), CCSDS 520.0-G-3. Washington, D.C.: CCSDS, December 2010.

[D2] *Mission Operations—Common Services*. Issue 1. Recommendation for Space Data System Standards (Blue Book), CCSDS 522.0-B-1. Washington, D.C.: CCSDS, May 2020.

[D3] *Space Packet Protocol*. Issue 2. Recommendation for Space Data System Standards (Blue Book), CCSDS 133.0-B-2. Washington, D.C.: CCSDS, June 2020.