

Draft Recommendation for
Space Data System Standards

|  |
| --- |
| Mission OperationsMission Product Distribution Services |

Draft Recommended Standard

CCSDS 5xx.1-W-0

WHITE Book

December 2023

AUTHORITY

|  |
| --- |
|  |
|  | Issue: | White Book, Draft A |  |
|  | Date: | 22/12/2023 |  |
|  | 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 address below.

This document is published and maintained by:

CCSDS Secretariat

Space Communications and Navigation Office, 7L70

Space Operations Mission Directorate

NASA Headquarters

Washington, DC 20546-0001, USA

FOREWORD

[Foreword text specific to this document goes here. The text below is boilerplate.]

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. CCSDS shall not be held responsible for identifying any or all such patent rights.

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 *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 addressed to the CCSDS Secretariat at the address indicated on page i.

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

Member Agencies

* Agenzia Spaziale Italiana (ASI)/Italy.
* 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 Federal Science Policy Office (BFSPO)/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.
* Chinese 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.
* 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.
* Indian Space Research Organization (ISRO)/India.
* Institute of Space Research (IKI)/Russian Federation.
* KFKI Research Institute for Particle & Nuclear Physics (KFKI)/Hungary.
* Korea Aerospace Research Institute (KARI)/Korea.
* Ministry of Communications (MOC)/Israel.
* 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.
* 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 ‘White Book’ status indicates that its contents are not stable, and several iterations resulting in substantial technical changes are likely to occur before it is considered to be sufficiently mature to be released for review by the CCSDS Agencies.

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 and Issue** | **Date** | **Status** |
| CCSDS xxx.1-W-0 | Mission Operations Mission Product Distribution Services, Proposed Draft Recommended Standard, Issue 0 Draft A | October 2023 | Initial Draft |
| CCSDS xxx.1-W-0 | Mission Operations Mission Product Distribution Services, Proposed Draft Recommended Standard, Issue 0 Draft B | November 2023 | Outcome of Fall WG Meeting, with edits and comments. |
| CCSDS xxx.1-W-0 | Mission Operations Mission Product Distribution Services, Proposed Draft Recommended Standard, Issue 0 Draft C | December 2023 | Revised draft after actioning WG comments. |

CONTENTS

Section Page

[1. Introduction 1-1](#_Toc156918201)

[1.1 General 1-1](#_Toc156918202)

[1.2 Purpose and Scope 1-2](#_Toc156918203)

[1.3 Applicability 1-2](#_Toc156918204)

[1.4 Rationale 1-2](#_Toc156918205)

[1.5 Definitions 1-4](#_Toc156918206)

[1.6 Nomenclature 1-6](#_Toc156918207)

[1.6.1 Normative Text 1-6](#_Toc156918208)

[1.6.2 Informative Text 1-6](#_Toc156918209)

[1.7 Conventions 1-7](#_Toc156918210)

[1.7.1 Diagrams 1-7](#_Toc156918211)

[1.7.2 Tables 1-7](#_Toc156918212)

[1.8 References 1-8](#_Toc156918213)

[2. Overview 2-1](#_Toc156918214)

[2.1 General 2-1](#_Toc156918215)

[2.2 Mission Product Distribution Use Cases 2-1](#_Toc156918216)

[2.2.1 Use Case 1: Retrieval (Pull) of Distinct Products 2-2](#_Toc156918217)

[2.2.2 Use Case 2: Recurrent Retrieval (Pull) of Distinct Products 2-3](#_Toc156918218)

[2.2.3 Use Case 3: Delivery (Push) of Products Subscribed to by Standing Order 2-3](#_Toc156918219)

[2.3 Mission Product Distribution Concept 2-5](#_Toc156918220)

[2.4 MPDS Information Model Overview 2-6](#_Toc156918221)

[2.5 MPDS Services Overview 2-8](#_Toc156918222)

[2.6 Relationship To Mission Operations Services 2-9](#_Toc156918223)

[2.7 Optional Elements of the Standard 2-11](#_Toc156918224)

[3. MPDS Service Specifications 3-12](#_Toc156918225)

[3.1 General 3-12](#_Toc156918226)

[3.2 Service: Product Retrieval 3-13](#_Toc156918227)

[3.2.1 Summary 3-13](#_Toc156918228)

[3.2.2 MPDS Data Items 3-14](#_Toc156918229)

[3.2.3 High-Level Requirements 3-15](#_Toc156918230)

[3.2.4 Functional Requirements 3-15](#_Toc156918231)

[3.2.5 Operation: listProducts 3-15](#_Toc156918232)

[3.2.6 Operation: getProduct 3-16](#_Toc156918233)

[3.2.7 Operation: getProductFile 3-17](#_Toc156918234)

[3.3 Service: Order Management 3-19](#_Toc156918235)

[3.3.1 Summary 3-19](#_Toc156918236)

[3.3.2 MPDS Data Items 3-20](#_Toc156918237)

[3.3.3 High-Level Requirements 3-20](#_Toc156918238)

[3.3.4 Functional Requirements 3-20](#_Toc156918239)

[3.3.5 Operation: listStandingOrders 3-21](#_Toc156918240)

[3.3.6 Operation: submitStandingOrder 3-22](#_Toc156918241)

[3.3.7 Operation: cancelStandingOrder 3-23](#_Toc156918242)

[3.4 Service: Product Delivery 3-24](#_Toc156918243)

[3.4.1 Summary 3-24](#_Toc156918244)

[3.4.2 MPDS Data Items 3-26](#_Toc156918245)

[3.4.3 High-Level Requirements 3-26](#_Toc156918246)

[3.4.4 Functional Requirements 3-26](#_Toc156918247)

[3.4.5 Operation: notifyProductDelivery 3-27](#_Toc156918248)

[3.4.6 Operation: deliverProducts 3-28](#_Toc156918249)

[4. MPDS Information Model 4-29](#_Toc156918250)

[4.1 General 4-29](#_Toc156918251)

[4.2 MPDS Data Items 4-30](#_Toc156918252)

[4.2.1 General 4-30](#_Toc156918253)

[4.2.2 Mission Products 4-32](#_Toc156918254)

[4.2.3 Standing Orders 4-36](#_Toc156918255)

[4.3 MPDS Data Types 4-38](#_Toc156918256)

[4.3.1 MPDS Base Data Types 4-38](#_Toc156918257)

[4.3.2 Additional MPDS Data Types 4-40](#_Toc156918258)

[4.3.3 MPDS Parameters 4-41](#_Toc156918259)

[5. Error Codes 5-1](#_Toc156918260)

[6. MPDS XML Specification 6-1](#_Toc156918261)

[6.1 Overview 6-1](#_Toc156918262)

[6.2 XML Schema Definition (XSD) for MO Services 6-1](#_Toc156918263)

[6.3 MAL XML 6-1](#_Toc156918264)

[6.4 MPDS XML 6-1](#_Toc156918265)

[ANNEX A Protocol Implementation Conformance Statement (PICS) Proforma (Normative) 1](#_Toc156918614)

[ANNEX B Security, SANA and Patent Considerations (Informative) 1](#_Toc156918615)

[ANNEX C Definition of Acronyms (Informative) 1](#_Toc156918616)

[ANNEX D Informative References (Informative) 1](#_Toc156918617)

**Figures**

[Figure 2‑1: Flowchart of Service Operations for Use Cases 1 and 2 2-3](#_Toc156918623)

[Figure 2‑2: Flowchart of Service Operations for Use Case 3 2-4](#_Toc156918624)

[Figure 2‑3: Mission Product Distribution Concept 2-5](#_Toc156918625)

[Figure 2‑4: MPDS Information Model Overview 2-6](#_Toc156918626)

[Figure 2‑5: MPDS Services Overview 2-8](#_Toc156918627)

[Figure 2‑6: MO MPDS Services Generic Protocol Stack 2-9](#_Toc156918628)

[Figure 3‑1: Product Retrieval Operations 3-14](#_Toc156918629)

[Figure 3‑2: Order Management Operations 3-19](#_Toc156918630)

[Figure 3‑3: Product Delivery Operations 3-25](#_Toc156918631)

[Figure 4‑1: MPDS Data Items 4-30](#_Toc156918632)

[Figure 4‑2: Mission Product Objects 4-32](#_Toc156918633)

[Figure 4‑3: Mission Product Service Structures 4-34](#_Toc156918634)

[Figure 4‑4: Standing Order Data Structures 4-36](#_Toc156918635)

[Figure 4‑5: MO Objects 4-39](#_Toc156918636)

[Figure 4‑6: MPDS Additional Data Types 4-40](#_Toc156918637)

[Figure 4‑7: MPDS Parameters 4-41](#_Toc156918638)

**Tables**

[Table 2-1: Optional Service Capabilities 2-11](#_Toc156918639)

# Introduction

## 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 decribing 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 MPDS 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 MPDS 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 MPDS services in conformance with the CCSDS MO service framework.

The MPDS 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 MPDS 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.

## Purpose and Scope

This recommended standard defines, in an abstract manner, the MPDS services in terms of:

1. an information model that describes the structure of MPDS data, including products, standing orders and supporting information objects that are referenced by the MPDS services;
2. 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:

1. individual implementations or products;
2. the implementation of entities or interfaces within real systems;
3. the methods or technologies required for communications;
4. how required MPDS service configuration data is made available to deployed MPDS functions;
5. 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.

## 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.

## 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 §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.

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

## Definitions

The following table contains the definition of terms used in the document. Acronyms are to be found in .

| Term | Definition |
| --- | --- |
| 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. |
| 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. |
| 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. Thismay correspond to, but is not restricted to:* 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 productcomprises a set of metadata decribing 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. |

## Nomenclature

### Normative Text

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

1. the words ‘shall’ and ‘must’ imply a binding and verifiable specification;
2. the word ‘should’ implies an optional, but desirable, specification;
3. the word ‘may’ implies an optional specification;
4. 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.

Specific requirements are identified in the text by a prefix of the following structure:

**Req\_<section#>.<requirement\_type>.<requirement#>**

for example, **Req\_.H.99**

where the section number corresponds to the section of this document that contains the requirement and the requirement number is a serial number assigned to each requirement in this document. The requirement\_type may be one of:

 H High-level Requirement

 F Functional Requirement

 E Encoding Requirement

### 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.

## Conventions

### Diagrams

Unified Modelling Language (UML) 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.

### 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 [2][[1]](#footnote-2).

## 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, CCSDS Recommended Practice (Magenta Book), CCSDS 520.1-M-1, Washington, D.C.: CCSDS, July 2010
2. *Mission Operations Message Abstraction Layer,* Issue 2, CCSDS Recommended Standard (Blue Book), CCSDS 521.0-B-2, Washington, D.C.: CCSDS, March 2013
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 – Informative references are listed in ANNEX D.

# Overview

## General

This chapter introduces the concepts behind the Mission Product Distribution Services [MPDS] 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

## Mission Product Distribution Use Cases

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

In the context of this specification, we do not distinguish whether the products requested or delivered through the MPDS 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.

While the ultimate end-user of the MPDS services may be, for example, a Principal Investigator (PI) interested in retrieving science data from their instrument, the actual user of the MPDS services is the implementer of the mission product distribution system that provides the user interface to that PI, or the engineer writing software or scripts for the automated retrieval of products.

In inter-operability scenarios, the list of types of product, subject to exchange, are typically predefined and captured in out-of-band agreements (ICDs). Event files, consolidated 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 MPDS 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 sub-sections, together with typical examples of how they relate to real-world scenarios.

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

### 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 TM Packets. A product type exists for a daily TM packet extract, for which the 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.



Figure ‑: 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).

### 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 TM Packets 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.

### 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.



Figure ‑: 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.

## Mission Product Distribution Concept



Figure ‑: Mission Product Distribution Concept

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

The Product Client is the consumer of the MPDS 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 MPDS 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 product 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.

## MPDS Information Model Overview

 

Figure ‑: MPDS Information Model Overview

The concept for the MPDS 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.

## MPDS Services Overview



Figure ‑: 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 MPDS 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.

## 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 ‑: 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:

1. 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);
2. 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;
3. 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 MPDS 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.

The MAL technology binding used in a specific deployment ensures on-the-wire interoperability at communications protocol level. 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 MPDS 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 MPDS 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 [D2] that can be used in conjunction with any MO application level service. These include a Directory, Login and Configuration services.

## Optional Elements of the Standard

The MPDS services specification defines an MPDS information model and four services. Compliance of an individual system deployment to the MPDS service specification does not imply that the 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 MPDS 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 [D2]. An entry in the directory is made for each MPDS service supported by each MPDS service provided, which lists the supported capability sets for that service. For the MPDS services, each service comprises a single capability set.

The following table summarizes the MPDS 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 -: 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 |

# MPDS Service Specifications

## General

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

* **Product Retrieval Service**
* **Order Management Service**
* **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:

1. Summary: contains a table listing the service operations and grouping them into capability sets, and UML sequence diagrams illustrating the service operations.
2. MO Objects and other data items: identifies the set of MO objects and other data items (defined in chapter 4) that are operated on by the service or referenced by service operations.
3. High-Level Requirements: such as the set of configuration data required by the service.
4. Functional Requirements: specific requirements on the behaviour of the service provider and consumer.
5. Operations: specification of each service operation.

All services defined in this document are part of the MPDS Area, Area number 6.

## Service: Product Retrieval

### 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 |
| MPDS | ProductRetrieval | 6 | 1 | 1 |

| Interaction Pattern | Operation Identifier | Operation Number | Capability Set |
| --- | --- | --- | --- |
| REQUEST | listProducts | 1 | 1 |
| PROGRESS | getProduct | 2 |
| PROGRESS | getProductFile | 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.



Figure ‑: Product Retrieval Operations

### MPDS Data Items

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

### High-Level Requirements

**Req\_.H.** The following set of mission data product configuration data shall be available to both provider and consumers in a any deployment of the product retrieval service:

1. Product definitions

### Functional Requirements

**Req\_.F.** 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.

**Req\_.F.** 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.

**Req\_.F.** In response to a getProductFile operation, the service provider shall return each referenced product by file transfer to the specified URI 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.

### Operation: listProducts

#### 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 | NoYesYes | productFilter : (ProductFilter)creationDate : (TimeWindow)timeWindow : (TimeWindow) |
| OUT | RESPONSE | No | productList : (List<ProductSummary>) |

#### 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 | 0 | SecondaryErrorCodeEnum | A UInteger giving a secondary error code that details the reason for invalidity. |

### Operation: getProduct

#### 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 | getProduct |
| 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 |

#### 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 | 0 | SecondaryErrorCodeEnum | A UInteger giving a secondary error code that details the reason for invalidity. |

### Operation: getProductFile

#### 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 | getProductFile |
| Interaction Pattern | PROGRESS |

| Pattern Sequence | Message | Nullable | Body Signature |
| --- | --- | --- | --- |
| IN | PROGRESS | NoNo | productRefs : (List <MAL::ObjectRef<Product>>)deliverTo : (MAL::URI) |
| OUT | ACK | No | Empty |
| OUT | UPDATE | NoNoNo | productSummary : (ProductSummary)filename : (MAL::String)deliveredTo : (MAL::URI) |
| OUT | RESPONSE | No | Empty |

#### 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 | 0 | SecondaryErrorCodeEnum | A UInteger giving a secondary error code that details the reason for invalidity. |
| DELIVERY\_FAILED | 1 | MAL::String | ExtraInfo provides additional information on the reason for failure as a free format string. |

## Service: Order Management

### 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 |
| MPDS | OrderManagement | 6 | 2 | 1 |

| Interaction Pattern | Operation Identifier | Operation Number | Capability Set |
| --- | --- | --- | --- |
| REQUEST | listStandingOrders | 1 | 1 |
| REQUEST | submitStandingOrder | 2 |
| SUBMIT | cancelStandingOrder | 3 |



Figure ‑: 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.

### MPDS Data Items

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

### High-Level Requirements

**Req\_.H.** 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:

1. Product definitions (as MDProductType objects [§4.2.2] )

### Functional Requirements

**Req\_.F.** 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.

**Req\_.F.** 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.

**Req\_.F.** 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.

**Req\_.F.** 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.

**Req\_.F.** If the end of the specified validity window is reached, the service provider shall cease delivery of products associated with the standing order.

**Req\_.F.** 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.

### Operation: listStandingOrders

#### 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 | YesYes | user : (MAL::Identifier)domain : (List <MAL::Identifier>) |
| OUT | RESPONSE | No | standingOrders : (List <StandingOrder>) |

#### 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 | 3 | MAL::Identifier | ExtraInfo contains the specified user or domain ID that does not exist. |

### Operation: submitStandingOrder

#### 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) |

#### 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 | 0 | SecondaryErrorCodeEnum | A UInteger giving a secondary error code that details the reason for invalidity. |
| ORDER\_FAILED | 1 | 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”. |

### Operation: cancelStandingOrder

#### 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.

|  |  |
| --- | --- |
| Operation Identifier | cancelStandingOrder |
| Interaction Pattern | SUBMIT |

| Pattern Sequence | Message | Nullable | Body Signature |
| --- | --- | --- | --- |
| IN | SUBMIT | No | orderRef : (MAL::Identifier) |

#### 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 | 3 | MAL::Identifier | ExtraInfo contains the specified order ID that does not exist. |

## Service: Product Delivery

### 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 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 |
| MPDS | 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.



Figure ‑: Product Delivery Operations

### MPDS Data Items

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

### High-Level Requirements

**Req\_.H.** 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:

1. Product definitions (as MDProductType objects [§4.2.2] )

Note: the service consumer only needs to be aware of its own User identity.

### Functional Requirements

**Req\_.F.** 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.

**Req\_.F.** 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.

### Operation: notifyProductDelivery

#### 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 | NoNoNo | productSummary : (ProductSummary)filename : (MAL::String)deliveredTo : (MAL::URI) |

#### 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 | 0 | SecondaryErrorCodeEnum | A UInteger giving a secondary error code that details the reason for invalidity. |
| DELIVERY\_FAILED | 1 | MAL::String | ExtraInfo provides additional information on the reason for failure as a free format string. |
| UNKNOWN | 3 | MAL::Identifier | ExtraInfo contains the specified user or order ID that does not exist. |

### Operation: deliverProducts

#### 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 : (MDProduct) |

#### 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 | 0 | SecondaryErrorCodeEnum | A UInteger giving a secondary error code that details the reason for invalidity. |
| UNKNOWN | 3 | MAL::Identifier | ExtraInfo contains the specified user or order ID that does not exist. |

# MPDS Information Model

## General

This chapter 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 section 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.

The structure of MPDS MO objects is fully defined within this document, together with their relationships.

While the information model itself is not normative, the data formats and services derived from it and specified in this document are normative. It describes both the data actively exchanged by MPDS Services and that required as common configuration data by service providers and users.

The information model diagrams contained in this section are expressed in UML and are informative, while the associated tables that define MPDS data structures are consistent with the MO framework and normative. The diagrams and tables follow the conventions introduced in §1.7.

The chapter is organized into the following main sections:

1. This Introduction
2. MPDS Data Items
3. 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 MPDS 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

## MPDS Data Items

### General

MPDS data items are the principal elements of the MPDS information model that are represented as MO objects. These have been introduced in §4.1 of this chapter and shown in **Error! Reference source not found.** 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.



Figure ‑: 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 MPDS 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:

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

### Mission Products

#### Mission Product Objects



Figure ‑: 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.

Note that 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.

Note that 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 MDProductType, 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 MPDS 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. |

#### Mission Product Service Structures

 

Figure ‑: 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. |
| 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. Note 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.

Note 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> | No | Reference to the ProductType definition |
| domain | List <MAL::Identifier> | No | 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. |

### Standing Orders

 

Figure ‑: 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 | No | Specifies the filter criteria for the standing order, including the product type and optional filters on product source and metadata parameters. |
| validityPeriod | TimeWindow | No | Period of time over which the standing order is required to be active. |
| 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. |

## MPDS Data Types

### 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 ‑: 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 and 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.

### Additional MPDS Data Types



Figure ‑: 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 |  |

| Attribute | Type | Nullable | Description |
| --- | --- | --- | --- |
| start | MAL::Time | No | Start time of the time window. |
| end | MAL::Time | No | End time of the time window. |

### MPDS Parameters



Figure ‑: 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 and optionally units and a free text description.

Note 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 |  |

| 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 |  |

| Attribute | Type | Nullable | Description |
| --- | --- | --- | --- |
| minimum | MAL::Attribute | No | Minimum value of the value range (greater than or equal to). |
| maximum | MAL::Attribute | No | 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 |  |

| Attribute | Type | Nullable | Description |
| --- | --- | --- | --- |
| values | List <MAL::Attribute> | No | Set of allowed (or disallowed) values for the metadata parameter. |

StringPattern

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 |  |

| 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). |

# 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 | 0 | A field in the message contains an invalid value. If there are multiple errors, the first invalid field is reported. |
| DELIVERY\_FAILED | 1 | An attempt to deliver a product file to the nominated address failed. |
| ORDER\_FAILED | 2 | Creation of a new product order failed. |
| UNKNOWN | 3 | 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 |  |

| 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. |

# MPDS XML Specification

## Overview

The MO MAL specification [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 MPDS 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/>

## XML Schema Definition (XSD) for MO Services

The XML Schema Definition (XSD) that is used to validate the actual XML service specifications is located at:

<http://sanaregistry.org/r/moschemas/ServiceSchema-vBBB.xsd>

where the ‘BBB’ part is replaced with the issue number of the corresponding document.

## MAL XML

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

<http://sanaregistry.org/r/moschemas/areaAAA-vBBB-MAL.xml>

where the ‘AAA’ part is replaced with the area number (‘001’ for MAL) and the ‘BBB’ part is replaced with the area version which shall match the issue number of the corresponding document.

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

<http://sanaregistry.org/r/moschemas/ServiceDefMAL.xml>

## MPDS XML

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

<http://sanaregistry.org/r/moschemas/area006-v001-MPDS.xml>

where 006 corresponds to the area number for MPDS and 001 corresponds to the area version which shall match the issue number of this document.

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

<http://sanaregistry.org/r/moschemas/ServiceDefMPDS.xml>

1. Protocol Implementation Conformance
Statement (PICS) Proforma
(Normative)
	1. INTRODUCTION
		1. OVERVIEW

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

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

* Optional services and service capability sets as summarized in **Error! Reference source not found.**. A compliant deployment can support any combination of MO MPDS services. If a service is supported, then it must support any mandatory service capability sets; other capability sets are optional.

An implementation’s completed PRL 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..
	+ 1. Notation
			1. Status Column Symbols

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

|  |  |
| --- | --- |
| **Symbol** | **Meaning** |
| M | Mandatory |
| O | Optional |

* + - 1. 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 |

* 1. General Information
		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 |  |

* + 1. 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 |  |

* + 1. 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) |  |

* + 1. Instructions for Completing the PRL

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

* 1. MPS Services PICS

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

There are 3 separate MPDS 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 MPDS 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 |  M |  |
| 2 | Order Management Service | 3.3 | O |  |
| 2.1 |  listStandingOrders submitStandingOrder cancelStandingOrder | 3.3 |  M |  |
| 3 | Product Delivery Service | 3.4 | O |  |
| 3.1 |  notifyProductDelivery deliverProducts | 3.4 |  M |  |

1. Security, SANA and Patent Considerations
(Informative)
2. Definition of Acronyms
(Informative)

| Acronym | Definition |
| --- | --- |
| M&C | Monitoring & Control |
| MAL | Message Abstraction Layer |
| MO | Mission Operations |
| MOIMS | Mission Operations and Information Management Standards |
| MPDS | Mission Product Distribution Services |
| PI | Principal Investigator |
| PICS | Protocol Implementation Conformance Statement |
| SANA | Space Assigned Numbers Authority |
| SFN | Short Form Numbers |
| SFP | Short Form Part |
| UML | Unified Modelling Language |
| URI | Uniform Resource Identifier |
| XML | eXtensible Markup Language |

1. Informative References
(Informative)
2. *Mission Operations Services Concept*, Issue 3, CCSDS Informational Report (Green Book), CCSDS 520.0-G-3, Washington, D.C.: CCSDS, December 2010
3. *Mission Operations – Common Services*, Issue 1, CCSDS Recommended Standard (Blue Book), CCSDS 522.0-B-1, Washington, D.C.: CCSDS, May 2020
1. Minor differences in style/layout are to enable autogeneration of the XML specification from the tables. [↑](#footnote-ref-2)