Open Geospatial Consortium

Date: 2012-01-09

Reference number of this OGC® project document: **06-141r6**

http://www.opengis.net/doc/oseo

Version: 1.0

Category: OpenGIS® Implementation Standard

Editors: Daniele Marchionni, Stefania Pappagallo

Ordering Services Framework for Earth Observation Products Interface Standard

Warning

This document is an OGC Member approved international standard. This document is available on a royalty free, non-discriminatory basis. Recipients of this document are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

Document type:	OpenGIS® Standard
Document subtype:	Interface Standard
Document stage:	Approved
Document language:	English

License Agreement

Contents

Figures

Tables

i. Preface

This OGC® standard specifies the interfaces, bindings, requirements, conformance classes, and a framework for implementing extensions that enable complete workflows for ordering of Earth Observation (EO) data products. This standard provides the interfaces for supporting the following EO Product ordering scenarios:

- Ordering products from EO Catalogues
- Subscribing to automatic delivery of EO products
- Bulk EO Product orders
- Ordering of future products

The EO products orders can be delivered in several different ways:

- On media via mail
- On file via different online protocols (e.g. ftp, sftp, ftps, etc.)
- Online via online data access protocols (e.g. WCS, WMS, etc.)

The ordered items can be customized in detail, one by one or altogether, via the processing options and scene selection options. These options are dynamically discovered and set from the clients by calling appropriate Ordering Services operations. This specification includes a comprehensive list of processing (20 product order options listed in Table 7-6 and 3 subscription options listed in Table 7-7) and scene selection options (14 scene options listed in Table 7-9) derived on the basis of inputs from several Satellite Agencies and Operators:

- ESA
- EUMETSAT
- CNES
- DLR
- CSA
- SPOT Image

In cases in which these already identified options are not sufficient for the specific mission, they can be extended following the SWE Common 2.0 framework.

Due to the number of supported ordering scenarios, covering different and also alternative needs, a number of Requirements Classes have been defined collecting the specific requirements a conformant implementation has to comply with. In parallel a number of Conformance Classes have been defined regrouping all tests a server has to pass for claiming the compliance with the corresponding Requirement Class. A server can comply with some Requirement Classes, it is not required to implement all classes.

Procurement Agencies and implementers shall be aware that:

- Not all scenarios (Requirement Classes) shall be implemented, but only the Core plus the requirements that are necessary for implementing a specific use case (see §2). However a server has to specify the supported Conformance Classes as evidence of the provided functionality.

- If order options are supported, then the implementation has to use a sub-set of the already identified options unless they are not fitting with their needs; in that case an application profile listing the new options, to be modeled with SWE Common 2.0, shall be defined and implemented.

- The standard has “extension” points that allow for the addition of XML elements in the EO Product order payload. These elements are not necessary for implementing the basic functionality specified in thisstandard, but can be used for accommodating implementation specific needs and then allowing the definition of “extensions”.

This document expands on the work presented in “Best Practices for Earth Observation Products” OGC-05-057r4, separating the order services from the catalogue services which have been presented in 06-079 and in 06-131. The final goal of the work was agreement on a coherent set of interfaces for ordering of EO products to support access to data from heterogeneous systems dealing with derived data products from satellite based measurements of the earth’s surface and environment.

ii. Document terms and definitions

This document uses the standard terms defined in Subclause 5.3 of [OGC 05-008], which is based on the ISO/IEC Directives, Part 2. Rules for the structure and drafting of International Standards. In particular, the word “shall” (not “must”) is the verb form used to indicate a requirement to be strictly followed to conform to this standard.

iii. Submitting organisations

The following organisations will submit the original document or its revisions to the OGC^{TM} Ordering SWG:

- **ESA – European Space Agency**
- **Telespazio**
- **con terra GmbH**
- **Eumetsat**

The editors would like to acknowledge that this work is the result of collaboration and review of many organizations and would like to thank for the comments and contributions from:

- **CNES**
- **SPOT Image**
- **DLR**
- **MDA**
- **VITO**

- **Spacebel s.a.**

Note: this does not imply a complete endorsement from these organizations..

iv. Document contributor contact points

All questions regarding this document should be directed to the editor or the contributors:

Name	Organisation	Email
Daniele Marchionni	TELESPAZIO	daniele.marchionni<at>telespazio.com
Stefania Pappagallo	TELESPAZIO	stefania.pappagallo<at>telespazio.com
Uwe Voges	con terra	u.voges<at>conterra.de
Michael Schick	EUMETSAT	Michael.Schick<at>eumetsat.int

v. Revision history

See Annex F

vi. Changes to the OGC® Abstract Specification

The OGC® Abstract Specification does not require changes to accommodate the technical contents of this document.

vii. Future work

None.

Foreword

This version cancels and replaces the second draft version edition (OGC OGC-141r5), of which has been technically revised.

This document includes six annexes; Annexes A (Abstract Test Suite) and B (XML Schema Definition) are normative, and other Annexes are informative.

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

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the specification set forth in this document, and to provide supporting documentation.

Introduction

The service proposed in this standard is intended to support the ordering of Earth Observation (EO) data products either from previously identified data set collections via a typical catalogue interaction or from future acquisitions specified via a Programming service [OR9] [OR10].

The intent of this standard is to describe an interface that can be supported by many data providers (satellite operators, data distributors, etc.), most of whom have existing (and relatively complex) facilities for the management of these data.

OGC® Ordering Services for Earth Observation Products Interface Standard (OSEO)

1 Scope

This standard describes the interfaces, bindings and encodings required to order Earth Observation (EO) products products in a heterogenous, distributed environment.

2 Compliance

This standard defines the interfaces and the requirements for implementing a server supporting the ordering of Earth Observation Products. Then this specification is targeted to the following Standardization Target:

Order Server

Earth Observation Products can be ordered in different ways through the preparation and submission of different types of orders:

- **Product Order**: order listing the different EO products by providing their identifier and options;
- **Tasking Order**: order specifying the acquisition of future products by providing the identifieir of a Tasking Request issued through a companion SPS server;
- **Subscription Order**: order specifying the periodical delivery of loosely defined products.

Therefore different Requirement Classes have been defined specifying the requirements for implementing the ordering process for each of the identified order types. Moreover additional classes have been defined for defining optional functions that might be implemented by Order Server having extended functionality.

The complete list of **Requirements Classes** is listed below:

- **Core**, which specifies the mimimum behaviour that all order servers shall implement.
  - **ProductOrder**, which specifies the basic requirements an Order Server allowing the ordering of precisely identified Earth Observation Products shall implement.
    - **SceneSelection**, which specifies the additional requirements an Order Server has to comply for supporting scene selection.
  - **SubscriptionOrder**, which specifies the basic requirements an Order Server allowing the sub-scription to Earth Observation Products shall implement.
  - **TaskingOrder**, which specifies the basic requirement an Order Server allowing submission of tasking requests shall implement.
  - **OnlineDataAccess**: it specifies the requirements an Order Server shall implement for supporting orders with on-line access (e.g. ftp, http, etc.).
  - **OnlineDataDelivery**: it specifies the requirements an Order Server shall implement for deliverying the ordered products at the on-line address specified by the client.
  - **MediaDelivery**: it specifies the requirements an Order Server shall implement for supporting orders on media.
  - **Cancellation**: it specifies the requirements a server has to comply with for supporting cancellation of already submitted orders.
    - **Async Cancellation**: it specifies the requirements a server has to comply with for supporting asynchronous cancellation of submitted orders.

- **AsyncSubmit**: it specifies the requirements a server has to comply with for supporting asynchronous notification on the order status.
- **Quotation**: it specifies the requirements for supporting order quotation.
  - **QuotationSync**, which specifies the requirements for a server providing order quotation in real time;
  - **QuotationMonitoring**, which specifies the requirements for a server providing order quotation by monitoring;
  - **QuotationAsync**, which specifies the requirements for a server providing order quotation by asynchronous notification;
  - **QuotationOffLine**, which specifies the requirements for a server providing order quotation by mail / e-mail;
- **Notification**: it specifies the requirements for supporting notification about the order status via WS-notification protocol.

Every OSEO implementation is expected to comply with at least one of **ProductOrder, SubscriptionOrder, TaskingOrder** and optionally with one of the other classes.

A server which complies only with Core is just a skeleton accepting and returning valid messages, it does not provide any other functionality.

The following diagram shows the relation ships between the defined Requirement Classes.

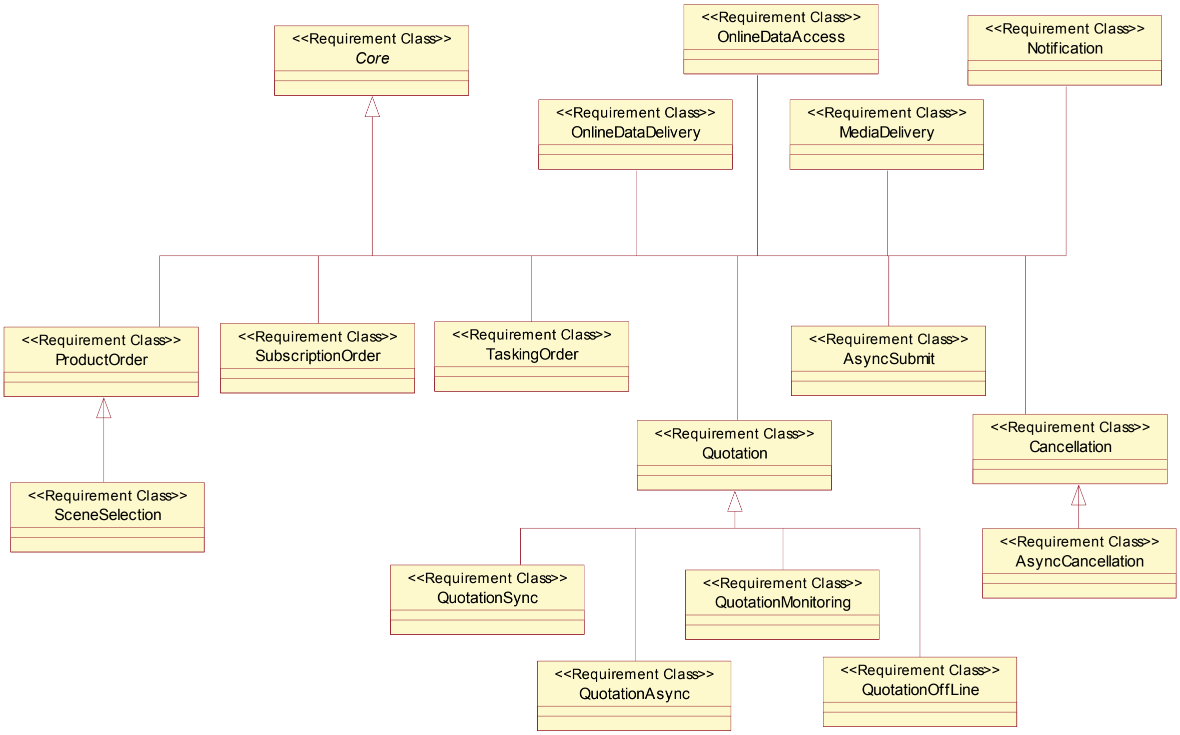


Figure 2-1: OSEO Requirement Classes.

The inheritance relationship between the different classes represents the inheritance of all requirements from the super class. E.g.: ProductOrder class defines its specific requirements and includes also the requirements defined in the Core class, hence an Order Server claiming the compliance with ProductOrder class has to comply with ProductOrder class requirements plus Core class requirements.

The following table reports:

- The Requirement Class name
- the URI
- the dependency with other requirements classes.

Requirement Class	Requirement Class URI	Dependency
AsyncCancellation	http://www.opengis.net/spec/OSEO/1.0/req/AsyncCancellation	Cancellation
AsyncSubmit	http://www.opengis.net/spec/OSEO/1.0/req/AsyncSubmit	Core
Cancellation	http://www.opengis.net/spec/OSEO/1.0/req/Cancellation	Core
Core	http://www.opengis.net/spec/OSEO/1.0/req/Core	
MediaDelivery	http://www.opengis.net/spec/OSEO/1.0/req/MediaDelivery	Core
Notification	http://www.opengis.net/spec/OSEO/1.0/req/Notification	Core
OnlineDataAccess	http://www.opengis.net/spec/OSEO/1.0/req/OnlineDataAccess	Core
OnlineDataDelivery	http://www.opengis.net/spec/OSEO/1.0/req/OnlineDataDelivery	Core
ProductOrder	http://www.opengis.net/spec/OSEO/1.0/req/ProductOrder	Core
Quotation	http://www.opengis.net/spec/OSEO/1.0/req/Quotation	Core
QuotationAsync	http://www.opengis.net/spec/OSEO/1.0/req/QuotationAsync	Quotation
QuotationMonitoring	http://www.opengis.net/spec/OSEO/1.0/req/QuotationMonitoring	Quotation
QuotationOffLine	http://www.opengis.net/spec/OSEO/1.0/req/QuotationOffLine	Quotation
QuotationSync	http://www.opengis.net/spec/OSEO/1.0/req/QuotationSync	Quotation
SceneSelection	http://www.opengis.net/spec/OSEO/1.0/req/SceneSelection	ProductOrder
SubscriptionOrder	http://www.opengis.net/spec/OSEO/1.0/req/SubscriptionOrder	Core
TaskingOrder	http://www.opengis.net/spec/OSEO/1.0/req/TaskingOrder	Core

Table 2-1: Requirements Classes vs. Standardization Targets.

The root path of all Requirements and conformance test URIs defined in this document is http://www.opengis.net/spec/OSEO/1.0/

Compliance with this standard shall be checked using all the relevant tests specified in Annex A (normative).

3 Normative references

This document references several external standards and specifications as dependencies:

[NR1] W3C Recommendation January 1999, Namespaces In XML, http://www.w3.org/TR/2000/REC-xml-names.

[NR2] W3C Recommendation 6 October 2000, Extensible Markup Language (XML) 1.0 (Second Edition), http://www.w3.org/TR/REC-xml

[NR3] W3C Recommendation 2 May 2001: XML Schema Part 0: Primer, http://www.w3.org/TR/2001/REC-xmlschema-0-20010502/

[NR4] W3C Recommendation 2 May 2001: XML Schema Part 1: Structures, http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/

[NR5] W3C Recommendation 2 May 2001: XML Schema Part 2: Datatypes, http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/

[NR6] W3C Recommendation (24 June 2003): SOAP Version 1.2 Part 1: Messaging Framework, http://www.w3.org/TR/SOAP/

[NR7] WSDL, Web Services Description Language (WSDL) 1.1. Available [online]: http://www.w3.org/TR/wsdl

[NR9] OWS Common Implementation Specification, May 2005 OGC 05-008c1

[NR10] W3C Recommendation (9 May 2006): Web Services Addressing 1.0 – Core, http://www.w3.org/TR/2006/REC-ws-addr-core-20060509

[NR12] OpenGIS® Web Services Common Specification OGC 06-121r9

[NR13] Unified Modeling Language (UML) Version 1.3, The Object Management Group (OMG): http://www.omg.org/cgi-bin/doc?formal/00-03-01

[NR14] The Extensible Markup Language (XML), World Wide Web Consortium, http://www.w3.org/TR/1998/REC-xml-19980210

[NR15] W3C Recommendation (24 June 2003): SOAP Version 1.2 Part 1, Messaging Framework, http://www.w3.org/TR/2003/REC-soap12-part1-20030624/

[NR16] WSDL, Web Services Description Language (WSDL) 1.1, http://www.w3.org/TR/2001/NOTE-wsdl-20010315

[NR17] W3C Recommendation (9 May 2006): Web Services Addressing 1.0 – Core, http://www.w3.org/TR/2006/REC-ws-addr-core-20060509

[NR18] Template for OWS Implementation Standards (10 september 2007) - 05-009r3

[NR19] Policy Directives for Writing and Publishing OGC Standard: TC Decisions (15 june 2009) - OGC 06-135r7

[NR20] The specification Model – A Standard for Modular specifications (19 october 2009) - OGC 08-131

[NR21] OGC® OWS-6 SWE Event Architecture Engineering Report - OGC 09-032

[NR22] Web Services Base Notification 1.3 (WS-BaseNotification) - OASIS Standard, 1 October 2006

[NR23] Web Services Topics 1.3 (WS-Topics) - OASIS Standard, 1 October 2006

[NR24] Name type specification – definitions – OGC 09-048.

3.1 Other References

[OR2] OpenGIS Catalogue Services – Best Practices for EO Products OGC-05-057r4

[OR3] OGC™ Catalogue Services Standard 2.0 - Extension Package for ebRIM Application Profile: Earth Observation Products - OGC 06-131r6 V1.0.0

[OR4] OGC™ CSW-ebRIM Registry Service – Part 2 : Basic extension package, version 1.0.0, 2008/02/29 -OGC 07-144r2

[OR5] OGC™ CSW-ebRIM Registry Service - Part 1: ebRIM profile of CSW, version 1.0.0, 2008/02/29. OGC 07-110r2

[OR8] OpenGIS® User Management Interfaces For Earth Observation Services - OGC 07-118r4 V0.0.6

[OR9] OGC® Sensor Planning Service Implementation Standard – OGC 09-000 V2.0.0

[OR10] OpenGIS® SWE Service Model Implementation Standard – OGC 09-001 V1.0.0

[OR11] OGC® SWE Common Data Model Encoding Standard – OGC 08-094 V2.0.0

In addition to this document, this standard includes several normative XML Schema Document files as specified in Annex B.

4 Terms and definitions

For the purposes of this document, the following terms and definitions apply:

asynchronous operation
client server interaction in which the client initiates the process calling the operation, but the completion result is not received at the time of the operation call, but later on by explicit notification from the server to the client.

client
software component that can invoke an **operation** from a **server.**

data level
stratum within a set of layered levels in which data is recorded that conforms to definitions of types found at the application model level [ISO 19101].

dataset series (dataset collection[1])
collection of datasets sharing the same product specification [ISO 19113, ISO 19114, ISO 19115]. In this context, a collection metadata record in the catalogue describes a collection of EO Products, typically a dataset collection corresponds to datasets (i.e. products) generated by a single sensor in a specific mode on a particular EO satellite.

EO Product
Data product, tipically stored on computer file, generated by sensors carried by Earth Observation Satellites.

geographic dataset
dataset with a spatial aspect [ISO 19115].

geographic information
information concerning phenomena implicitly or explicitly associated with a location relative to the Earth [ISO 19128 draft].

[1] Due to historical reasons we´ll mainly use the term ‘dataset collection’ in this document although the term ‘dataset series’ is used in the ISO/TC211 Terminology Maintenance Group.

georesource
geographic information of a specific type (e.g. geographic dataset, geographic application, geographic service).

identifier
a character string that may be composed of numbers and characters that is exchanged between the client and the server with respect to a specific identity of a resource.

interface
named set of operations that characterise the behaviour of an entity [ISO 19119].

metadata dataset (metadataset)
metadata describing a specific dataset [ISO 19101].

metadata entity
group of metadata elements and other metadata entities describing the same aspect of data.

NOTE 1 A metadata entity may contain one or more metadata entities.

NOTE 2 A metadata entity is equivalent to a class in UML terminology [ISO 19115].

metadata schema
conceptual schema describing metadata.

NOTE ISO 19115 describes a standard for a metadata schema. [ISO 19101]

metadata section
subset of metadata that defines a collection of related metadata entities and elements [ISO 19115].

notification
notice given to the client with the status of the request or the esit of request.

operation
specification of a transformation or query that an object may be called to execute [ISO 19119].

parameter
variable whose name and value are included in an operation **request** or **response.**

profile
set of one or more base standards and - where applicable - the identification of chosen clauses, classes, subsets, options and parameters of those base standards that are necessary for accomplishing a particular function [ISO 19101, ISO 19106]

product order
Ordering request, issued via this specification, asking the processing and delivery of precisely identified EO Products.

programming service
Service allowing the client to request the acquisition of EO Products on specified area / time intervals.

qualified name
name that is prefixed with its naming context.

EXAMPLE The qualified name for the road no attribute in class Road defined in the Roadmap schema is RoadMap.Road.road_no. [ISO 19118].

request
invocation of an **operation** by a **client.**

response
result of an **operation,** returned from a **server** to a **client.**

schema
formal description of a model [ISO 19101, ISO 19103, ISO 19109, ISO 19118]

server
service instance
a particular instance of a **service** [ISO 19119]

service
distinct part of the functionality that is provided by an entity through interfaces [ISO 19119]

capability which a service provider entity makes available to a service user entity at the interface between those entities [ISO 19104 terms repository]

service interface
shared boundary between an automated system or human being and another automated system or human being [ISO 19101]

service metadata
metadata describing the **operations** and **geographic information** available at a **server** [ISO 19128 draft]

synchronous operation
operation of a Web Service, in which the complete result of the service call is returned in the synchronous answer.

state
condition that persists for a period

NOTE The value of a particular feature attribute describes a condition of the feature [ISO 19108].

subscription

ordering request, issued via this specification, allowing the client to adhere to the periodical delivery of products processed and delivered according to predefined rules.

tasking request

request for future acquisition segments for generate products to order.

transfer protocol

common set of rules for defining interactions between distributed systems [ISO 19118]

version

version of an Implementation Specification (document) and XML Schemas to which the requested operation conforms

NOTE An OWS Implementation Specification version may specify XML Schemas against which an XML encoded operation request or response must conform and should be validated.

5 Conventions

5.1 Abbreviated terms

Some frequently used abbreviated terms:

ANX	Ascending Node Crossing
API	Application Program Interface
COTS	Commercial Off The Shelf
CQL	Common Query Language
CRS	Coordinate Reference System
CSW	Catalogue Service-Web
DCE	Distributed Computing Environment
DCP	Distributed Computing Platform
DDS	Data Dissemination Service
DNX	Descending Node Crossing
EO	Earth Observation
HMA	Heterogeneous Missions Accessibility
HTTP	Hyper Text Transport Protocol
ISO	International Organisation for Standardisation
OGC	Open GIS Consortium
OSEO	Ordering Services for Earth Observation
SOAP	Simple Object Access Protocol
SQL	Structured Query Language
SWG	Standard Working Group
UML	Unified Modeling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
URN	Uniform Resource Name
UTF-8	Unicode Transformation Format-8
WSDL	Web Service Definition Language
W3C	World Wide Web Consortium
XML	eXtensible Markup Language

5.2 UML notation

5.2.1 Introduction

Some diagrams that appear in this standard are presented using the Unified Modeling Language (UML) static structure diagram, as described in Subclause 5.2 of [OGC 05-008].

The UML uses several kinds of models for system description. For the scope of this document the following diagrams are considered:

- **Class Diagrams**
  Class diagrams show the static structure of the model, in particular, the things that exist (such as classes and types), their internal structure, and their relationships to other things. Class diagrams do not show temporal information, although they may contain occurrences of things that have or describe temporal behaviour.
- **Sequence Diagrams**
  A sequence diagram shows an interaction arranged in time sequence. In particular, it shows the objects participating in the interaction by their "lifelines" and the messages that they exchange arranged in time sequence. It does not show the associations among the objects.

5.2.2 UML Class Diagrams

A class diagram is a picture providing generic descriptions of possible systems. Class diagrams and object diagrams are alternate representations of object models. Class diagrams contain classes and object diagrams contain objects, but it is possible to mix classes and objects when dealing with various kinds of metadata, so the separation is not rigid. Class diagrams contain icons representing classes, interfaces, and their relationships. In particular, class diagrams contain:

- **Logical Packages**
  Packages purpose is to partition the logical model of a system. They are clusters of highly related classes that are themselves cohesive, but are loosely coupled with other such clusters. You can use packages to group classes, interfaces, and other packages.
- **Classes**
  A class captures the common structure and common behaviour of a set of objects. A class is an abstraction of real-world items. When these items exist in the real world, they are instances of the class, and referred to as objects.
- **Interfaces**
  An interface specifies the externally visible operations of a class and/or component, and has no implementation of its own. An interface typically specifies only a limited part of the behaviour of a class or component.
- **Parameterised Classes**
  A parameterised class is a template for creating any number of instantiated classes that follow its format. It declares formal parameters. You can use other classes, types, and constant expressions as parameters. You cannot use the parameterised class itself as a parameter. You must instantiate a parameterised class before you can create its objects. In its simplest form, you can use parameterised classes to build container classes.
- **Instantiated Classes**
  An instantiated class is a class formed from a parameterised class by supplying actual values for parameters. It is created by supplying actual values for the formal parameters of the parameterised class. This instantiation process forms a concrete class in the family of the parameterised class. The instantiated class should be put at the client end of an instantiate relationship (accessible through the Create Entry on the Tools menu) that points to the corresponding parameterised class.
- **Association Relationships**
  An association represents a semantic connection between two classes, or between a class and an interface. Associations are bi-directional; they are the most general relationship and also the most semantically weak.
- **Aggregate Relationship**

The aggregate relationship is used for showing a whole and part relationship between two classes.
The class at the client end of the aggregate relationship is sometimes called the aggregate class. An instance of the aggregate class is an aggregate object. The class at the supplier end of the aggregate relationship is the part whose instances are contained or owned by the aggregate object. The aggregate relationship is used for showing that the aggregate object is physically constructed from other objects or that it logically contains another object. The aggregate object has ownership of its parts.

- **Generalize/Inherits Relationships**
  A generalize relationship between classes shows that the subclass shares the structure or behaviour defined in one or more super-classes. Use a generalize relationship to show an "is- a" relationship between classes.
- **Instantiates Relationships**
  An instantiates relationship represents the act of substituting actual values for the parameters of a parameterised class or parameterised class utility to create a specialized version of the more general item. In most cases, you will also draw a uses relationship between the instantiated class and another concrete class that is used as an actual parameter.
- **Dependency Relationships**
  Draw a dependency relationship between two classes, or between a class and an interface, to show that the client class depends on the supplier class/interface to provide certain services, such as:
  - The client class accesses a value (constant or variable) defined in the supplier class/interface.
  - Operations of the client class invoke operations of the supplier class/interface.
  - Operations of the client class have signatures whose return class or arguments are instances of the supplier class/interface.

The next picture shows the items just explained.

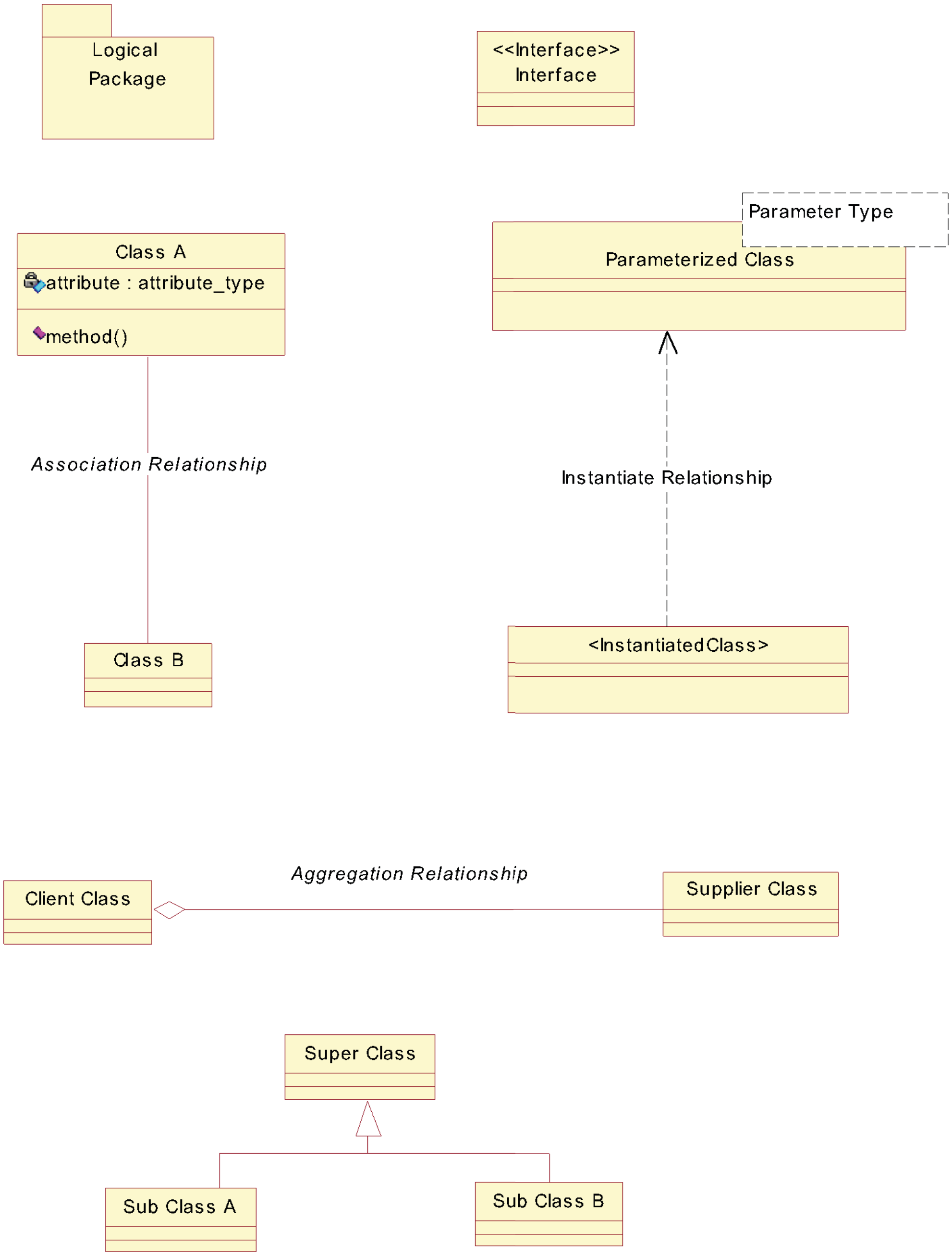


Figure 5-1 - UML Class Diagram notations.

5.2.3 UML Sequence Diagrams

Sequence diagrams are a representation of an interaction between objects. A sequence diagram traces the execution of an interaction in time.

The picture below illustrates a sequence diagram.

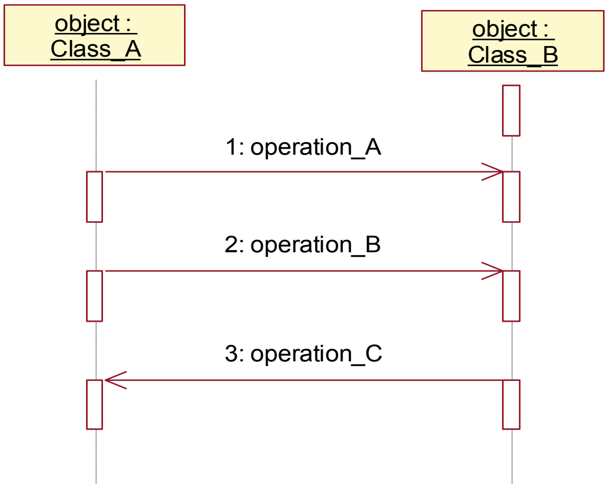


Figure 5-2 - UML Sequence Diagrams Notations.

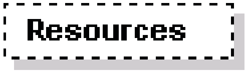
Each interaction between objects is the activation of an operation of an object, which includes input and output parameters.

5.3 *XML notation*

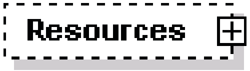
Most diagrams that appear in this specification are presented using an XML schema notation defined by the XMLSpy tool and described in this subclause.

Hereafter the symbols defined in the XML schema notation are described:

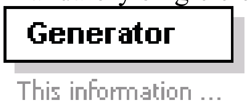
- Optional single element without child elements



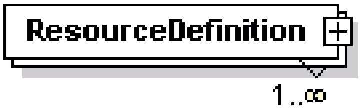
- Optional single element with child elements



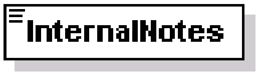
- Mandatory single element.



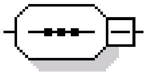
- Mandatory multiple element containing child elements. This element must occur at least once (Minimum Occurrence = 1) and may occur as often as desired (Maximum Occurrence = unbounded).



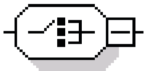
- Mandatory single element with containing simple content (e.g. text) or mixed complex content (e.g. text with xhtml markup).



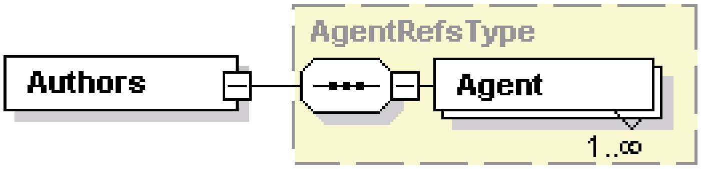
- A sequence of elements. The elements must appear exactly in the sequence in which they appear in the schema diagram.



- A choice of elements. Only a single element from those in the choice may appear at this position.



- Types. If an element refers to a complex global type, the type is shown with a border and yellow background.



- Complex Type. The following figure illustrates the use of a complex type for defining an XML element

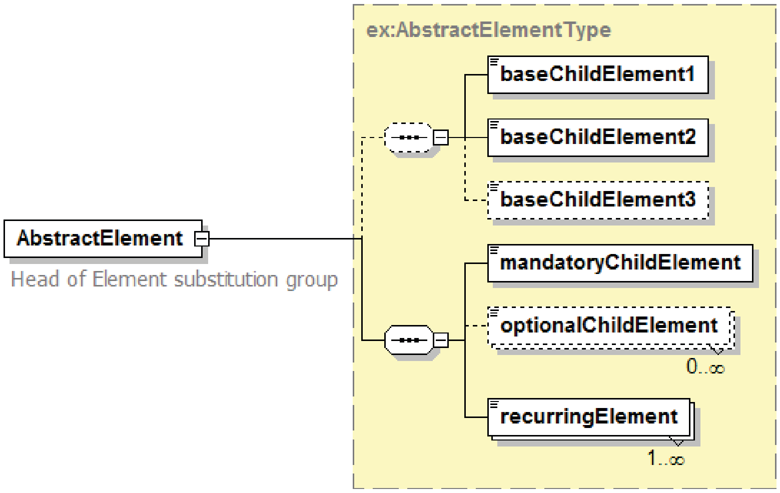


Figure 5-3 – XML Diagrams Notations.

5.4 *Used parts of other documents*

This document uses significant parts of documents:

- OGC® Sensor Planning Service Implementation Standard – OGC 09-000 V2.0.0
- OpenGIS® SWE Service Model Implementation Standard – OGC 09-001 V2.0.0

To reduce the need to refer to that document, this document copies some of those parts with small modifications. To indicate those parts to readers of this document, it is used the reference [OR9] [OR10].

5.5 *Platform-neutral and platform-specific standards*

As specified in Clause 10 of OGC Abstract Specification Topic 12 "OpenGIS Service Architecture" (which contains ISO 19119), this document includes both Distributed Computing Platform-neutral and platform-specific standards. This document first specifies each operation request and response in platform-neutral fashion. This is done using a table for each data structure, which lists and defines the parameters and other data structures contained.

The specified platform-neutral data could be encoded in many alternative ways, each appropriate to one or more specific DCPs. This document now specifies encoding appropriate for use of HTTP transfer of operations requests (using XML encoding). However, the same operation requests and responses (and other data) could be encoded for other specific computing platforms.

5.6 Data dictionary tables

The XML data dictionary used to describe the parameters within this document is specified herein in a series of tables. The contents of the columns in these tables are described in Table 5-1.

Column title	Column contents
Names (left column)	Name of the XML element to describe. The name uses the XML encoding capitalization specified in Subclause 11.6.2 of [OGC 05-008]. The name capitalization rules used are specified in Subclause 11.6.2 of [OGC 05-008]. Some names in the tables may appear to contain spaces, but no names contain spaces.
Definition (second column)	Specifies the definition of this element.
Data type and value (third column)	Normally contains two items: The mandatory first item is often the data type used for this parameter, using data types appropriate in the model, in which this parameter is a named attribute of a XML entity. Alternately, the first item can identify the data structure (or class) referenced by this association, and references a separate table used to specify the contents of that class (or data structure). The optional second item in the third column of each table should indicate the source of values for this parameter, the alternative values, or other value information, unless the values are quite clear from other listed information.
Multiplicity and use (fourth column)	Normally contains two items: The item specifies the multiplicity and optionality of this parameter in this data structure, either “One (mandatory)”, “One or more (mandatory)”, “Zero or one (optional)”, or “Zero or more (optional)”. If it needs to describe the multiplicity of an XML choise element, it needs to use the word “mandatory/choise” and “optional/choise”. The choice element infact provides an XML representation for describing a single selection from a set of element types where each selection item can be defined ‘mandatory’.
Product Ordering (fifth column)	Optional column. If this column is checked, it specifies that the parameter is meaningful for Product Orders.
Subscription (sixth column)	Optional column. If this column is checked, it specifies that the parameter is meaningful for Subscription Orders.

Table 5-1 - Contents of data dictionary tables

When the data type used for a parameter, in the third column of such a table, is an other complex type, all the values must be specified and listed, together with the meaning of each value. When this information is extensive, these values and meanings are specified in a separated table and a cross reference to it is put in the table.

6 Ordering Service for Earth Observation Products (OSEO) overview

This section focuses on the purpose, scope and policies of ordering services that comply with this specification. It documents special requirements and describes the context of use.

The Ordering Service described in this specification has the objective of supporting the following types of orders of Earth Observation products:

- **Order from catalogues of EO products.**

  This service allows the preparation and the submission of an order including products identified via a search in a catalogue of EO products ([OR3]).

- **Order of products derived from a programming request.**

  This service allows the submission of an order including products which can be derived from a set of future acquisition segments specified via an external Programming Service ([OR9] [OR10]).

- **Subscription to EO products.**

  This service allows users the periodical reception of products of interest on the areas of interest or to issue orders for bulk of products.

For these purposes, the Ordering Service for Earth Observation Products specifies the following operations:

- **GetCapabilities** allows a client to request and receive service metadata (or Capabilities) document that describes the abilities of the specific server implementation (see §8).
- **GetOptions** allows clients to retrieve the options for issuing an order:
  - in case of product ordering it returns the options for ordering a specific type of product;
  - in case of subscriptions it returns the possible parameters to set for specifying the scope of the subscription (e.g.: area of interest, expiration date, etc.);
  - in case of order from tasking request it returns the processing and delivery options for ordering the future products coming from the tasking request.

  see §9.
- **GetQuotation** allows the client to get a quotation of either the order or the tasking request or the subscription going to be issued (see §10). This operation, in order to support the wider set of clients and service providers, supports several interaction models:
  - **Synchronous** quotation;
  - **Asynchronous via Notification**: the client has to implement a call-back operation (**GetQuotationResponse** §11) which is called by the server when the quotation is available.
  - **Asynchronous via Monitoring**: after the first activation, in which the client specifies all order parameters, the client has to call it again (referencing the first call) until the server is able to return the quotation.
  - **Off-line**: the quotation is not provided by on-line interaction but via mail / e-mail.
- **Submit** allows either submitting an order of products (from EO catalogue or from a programming request) or subscribing to a subscription (see §12). This operation is asynchronous and generally the client has to implement a call-back operation (**SubmitResponse** §13) for receiving the result of the operation. However, it is also allowed to ask no notification and then no call back operation is to be implemented from the client.

- **GetStatus** allows to retrieve either the status of submitted orders or the status of subscribed subscriptions (see §14).
- **DescribeResultAccess** allows accessing the products ordered with on-line delivery (see §15).
- **Cancel** allows either to ask the cancellation of an already submitted order or to unsubscribe a subscription. This operation is asynchronous and then the client has to implement a call-back operation (**CancelResponse** §17) for receiving the result of the operation (see §16).

6.1 Essential Use-cases

6.1.1 Ordering from catalogue of EO Products

The following figure shows the typical usage of the Ordering Service operations to order products from EO Catalogues:

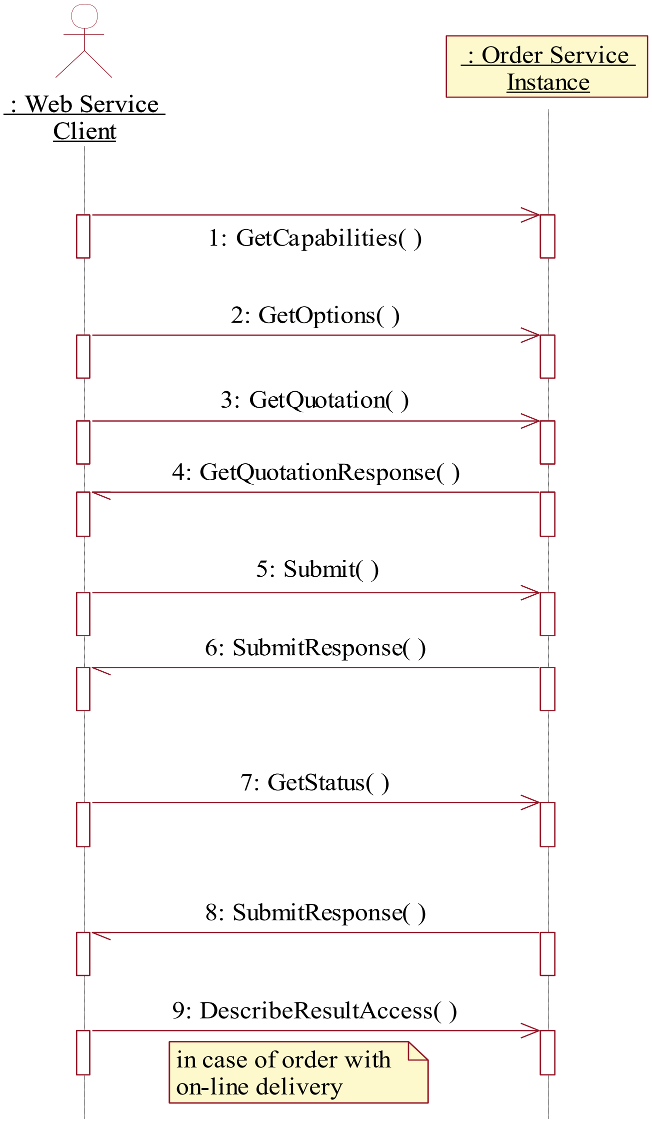


Figure 6-1 - Sequence of steps generally performed for ordering products from EO Catalogue.

In the scenario the following entities are specified:

- Web Service Client, which represents the user submitting requests to the Order Service;
- Order Service Instance: it is the server implementing the order service.

The typical scenario is:

- The list of products to be ordered has been prepared on client side by querying EO Catalogues (through the [OR-3] protocol).
- The client gets the list of supported operations from the Ordering Service instance (**GetCapabilities**).

- The list of ordering options is retrieved for each product to be ordered (**GetOptions**).
- Then the order is prepared on the client side choosing the requested options for each of the products to order.
- The quotation of the just prepared order can be asked by calling the **GetQuotation** operation. The quotation can be received either synchronously or via asynchronous notification depending on client and server capabilities.
- In case the quotation is accepted, the order can be submitted to the Order Service (**Submit**). The Order Service returns back an acknowledgement and starts the execution of the order.
- After the order has been submitted the following events are possible:
  - The client asks the status of the order to verify the progress of the order (**GetStatus**);
  - The client is notified of possible status updates (**SubmitResponse**);
  - The client can ask the cancellation of the order (**Cancel**).
- When the order processing is completed, the ordered products are either delivered to the user or can be retrieved on-line by calling DescribeResultAccess operation depending on the selected delivery method.

6.1.2 Order of Future Products derived from tasking requests

For future products the scenario is very similar to the previous one: instead of getting the catalogue identifier of the products to order, the client has to identify the future acquisition segments needed for generate the products to order. This step is performed accessing an external programming service ([OR9] [OR10]). These future acquisition segments are identified within the products order via the identifier of the corresponding tasking request returned by the Programming service.

Then the submission of this type of order can be summarized with the following steps:

- Identification of the necessary future acquisition segments and their request to the external Programming Service ([OR9] [OR10]). As a result the programming request identifiers of all needed acquisitions are available on client side.
- Get order options for the products that can be ordered from the identified acquisitions: call GetOptions specifying as input parameters the programming identifiers.
- The order is prepared on client side choosing the options necessary for getting the needed products from the identified acquisitions.
- The quotation of the just prepared order can be asked by calling the **GetQuotation** operation. The quotation can be received either synchronously or via asynchronous notification depending on client and server capabilities.
- In case the quotation is accepted, the order can be submitted to the Order Service (**Submit**). The Order Service returns back an acknowledgement and start the execution of the order.
- After the order has been submitted the following events are possible:
  - The client asks the status of the order to verify the progress of the order (**GetStatus**);
  - The client is notified of possible status updates (**SubmitResponse**).
  - The client can ask the cancellation of the order (**Cancel**)
  - When the order processing is completed, the ordered products are either delivered to the user or can be retrieved on-line by calling **DescribeResultAccess** operation depending on the selected delivery method.

6.1.3 Subscribe to EO Products

This section explains the steps to follow for subscribing to published EO products subscriptions.

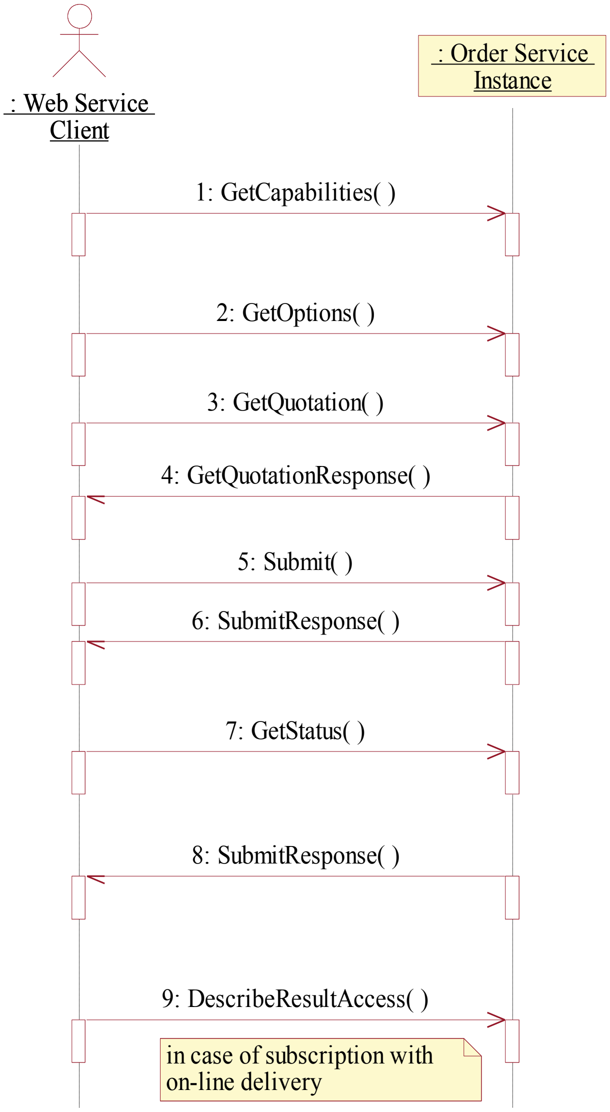


Figure 6-2 - Sequence of steps generally performed for subscribing to EO products.

In the scenario the following entities are specified:

- Web Service Client, which represents the user submitting requests to the Order Service;
- Order Service Instance: it is the server providing the order service;

The typical scenario is:

- The client gets the list of supported operations from the server (**GetCapabilities**).
- The list of subscriptions to subscribe has been prepared on client side querying a catalogue storing the advertised subscriptions, and then the next step is to retrieve the list of possible subscription options (**GetOptions**). Possible examples options are:
  - region of interest, which allows to receive only the available products overlapping this area;

    - type of area coverage, specifying how the products to be returned are spatially related to the specified area (e.g. overlap, inclusion, etc.)
    - expiration date, which specifies the limit date & time of validity of the subscription;
    - repetition, which specifies the number of time the base observation period is repeated.
    - the number of products per observation.
- The subscription order is prepared on the client side setting the available subscription options.
- The quotation of the just prepared order can be asked by calling the **GetQuotation** operation. The quotation can be received either synchronously or via asynchronous notification depending on client and server capabilities.
- In case the quotation is accepted, the subscriptions are subscribed to the Order Service by calling Submit operation. The Order Service returns back an acknowledge confirming the activation of the subscription.
- After the order has been submitted the following events are possible:
  - The client asks the status of his / her subscriptions (**GetStatus**).
  - The progress of the subscriptions are directly notified to the client (**SubmitResponse**)
  - The client is allowed to unsubscribe the subscription (**Cancel**).
- When some products are ready, they can be retrieved by **DescribeResultAccess** operation if ordered with on-line delivery.

7 Shared aspects

7.1 Introduction

This clause specifies aspects of the Ordering Service behaviour that are shared by several operations defining the interfaces provided by its components.

7.2 Information models

As previously anticipated, this standard supports different type of orders:

- **Product Orders**
  - **Order from EO Catalogue** ([OR3]): the products to be ordered are identified from catalogue performing a search. The retrieved identifiers are used for building the items within the order.
  - **Future product orders via Programming service** ([OR9] [OR10])

    For building a future product order two steps are necessary:

    - A programming request has to be defined in order to specify the needed acquisition segments. This step is performed via the SPS instance linked to the Ordering service which support future products ordering.

      The tasking request deals only with the parameters needed for the acquisition e.g.: start & stop time / orbit or area to cover, polarization, incidence angles, sensor mode, etc. At this stage the level of product, the format, the delivery options are not specified.
    - A product order has to be specified, referencing the tasking request, for setting all parameters needed to deliver products to the user e.g.: required product type, product format, media, delivery address, accounting and billing information.
- **Subscription**

  For subscribing a subscription it is sufficient specifying the appropriate collection identifier, the area and the temporal extension of the subscription.

In the following sub-sections the data structures for modelling these three type of orders are described.

7.3 Information model for EO product ordering

The information models section deals with the information item managed through the Order Service operations, which are:

- Order Options (§7.3.2)
- Order Specification (§7.3.6)
- Order Item (§7.3.8)
- Order Quotation (§7.3.9)
- Order Monitoring (§7.3.12)
- Order Item Monitoring (§7.3.13)

7.3.1 XML schema approach

This specification deals mainly with 3 different types of requests: product orders, future product orders, and subscription orders. These requests share several common parameters and have some differences, and then to model them in a XML schema the following options have been envisaged:

- Put all parameters together making optional the specific ones;

- Put all parameters together defining choice elements for the specific ones;
- Define a hierarchy with an abstract type grouping the common parameters and use type substitution.

The first approach is the simplest one, but it does not allow strict schema checks, because all the specific parameters are optional and cannot be verified whether the parameters needed to a specific type of order are provided or not.

The second approach implies the usage of lot of choices in the schema and if a new request has to be managed, all these choices have to be updated.

The third approach allows schema checks for the different type of requests and is more extensible than the previous one, because it is sufficient to add the new request to the substitution group and all places where the substitution group was used don't have to be updated.

In the first revisions of this specification the third approach was the choosen, but after having recognized problems for handling substitution groups with commonly available XML data binding tools, then a combination of first and third approach has been finally adopted.

7.3.2 Order Options

Order options specify all possible valid combinations of options for ordering products of a specified EO Product collection or for subscribing to a subscription.

7.3.2.1 CommonOrderOptionsType

The following figure represents the common order options.

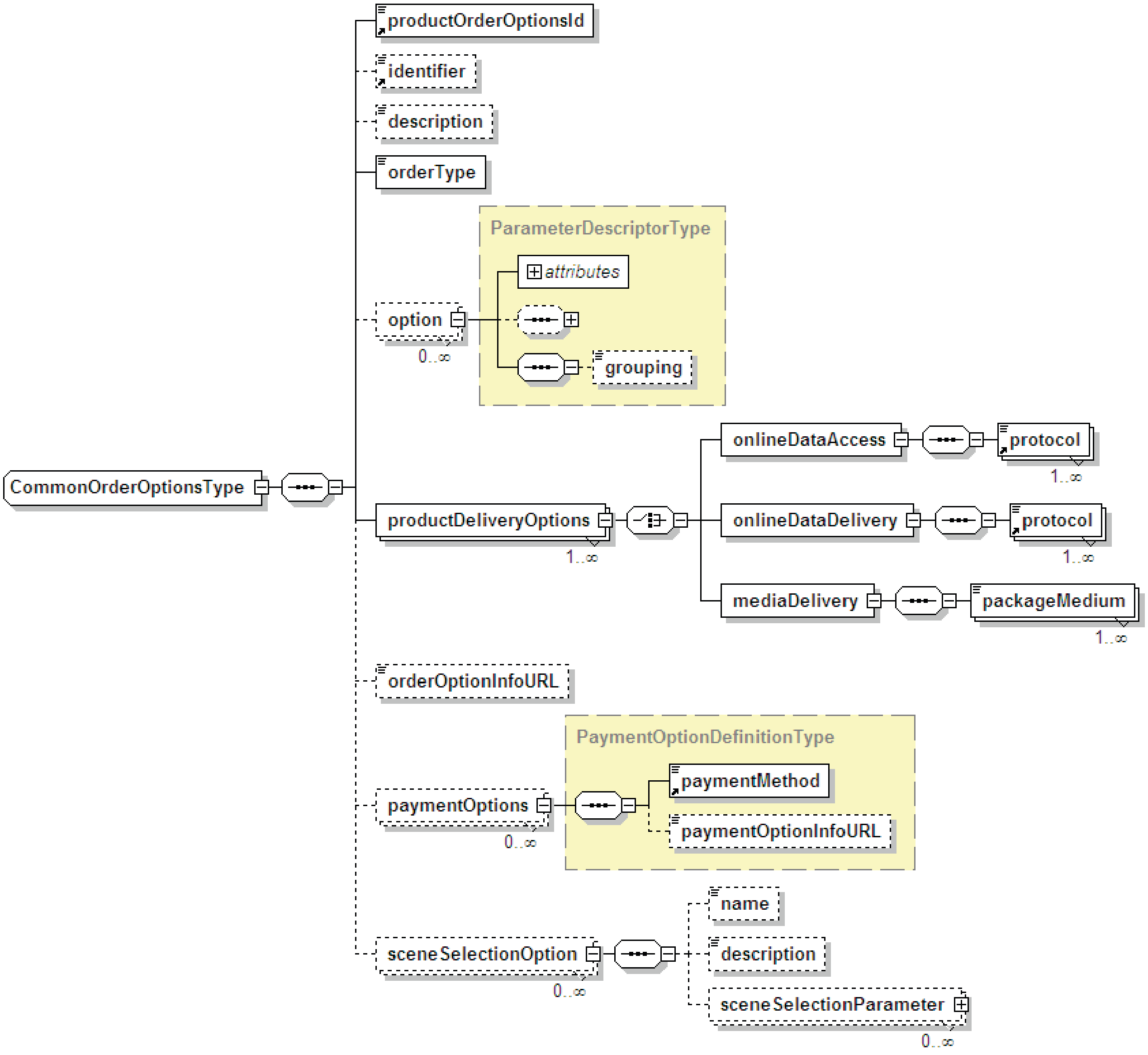


Figure 7-1 - CommonOrderOptionsType diagram.

CommonOrderOptionsType contains the common information about the different order options available for the order. The following table describes the CommonOrderOptionType using all information within the XML schema.

Names	Definition	Data type and value	Multiplicity and use	Product Ordering	Subscription
productOrderOptionsId	Identifier of the specific order option group.	Type: Not empty string (max 40 char) Example: "on-line retrieval"	One (mandatory)	X	X
identifier	Product identifier element. This field: – if present it means that the option is applicable only to this product; – if not present it means that the option is applicable to the whole collection.	Type Value: string Permitted Values: Not empty string	One (optional)	X	
description	Free text description of the order option group.	Type: string (max 255 char)	One (optional)	X	X
orderType	This tag specifies whether the order option is related to a subscription order or to a product order.	Type: String Permitted Values: □ PRODUCT_ORDER, □ SUBSCRIPTION_ORDER □ TASKING_ORDER	One (mandatory)	X	X
option	Ordering Options	Type: ParameterDescriptorType (see §7.3.3) This element reports the list of order options available for the specified collection / order item. See Table 7-6 for the list of pre-defined order options.	Zero or more (optional)	X	X
productDeliveryOptions	Delivery Options for the order item.	Type: productDeliveryOptions	One or more (mandatory)		
onlineDataAccess	Delivery option in case of On-line Access			X	X
protocol		Type: ProtocolType (§7.3.2.2)	One or more (mandatory)		
onlineDataDelivery	Delivery option in case of On-line Delivery			X	X

Names	Definition	Data type and value	Multiplicity and use	Product Ordering	Subscription
protocol		Type: ProtocolType (§7.3.2.2)	One or more (mandatory)		
mediaDelivery	Delivery option in case of Media Delivery			X	X
packageMedium	Definition of the medium to use for deliverying ordered products.	Type: PackageMedium (§7.3.2.3)	One or more (mandatory)	X	X
orderOptionInfoURL	Pointer to external information about the current order option.	Type: xs:anyURI	One (optional)	X	X
paymentOptions	Payment options	Type: PaymentOptionsDefinitionType	Zero or more (optional)	X	X
paymentMethod	Payment methods.	Type: String (max 40 char) Example: quota, invoice, prepay (to be indicated for free products), deposit account, credit card, credit card previously supplied	One (mandatory)	X	X
paymentOptionInfoURL	Pointer to external information about the current payment option.	Type: xs:anyURI	One (optional)	X	X
sceneSelectionOption	Set of scene selection option groups.		Zero or more (optional)	X	
name	Name of the scene selection options group. It identifies the group of scene selection options.	Type: string (max 60 chars)	One (optional)	X	
description	Description of the scene selection option group	Type: string (max 4000 chars)	One (optional)	X	
sceneSelectionParameter	Scene selection option parameters description.	Type: SceneSelectionDescriptorType (see §7.3.5)	Zero or more (optional)	X	

Table 7-1 - CommonOrderOptionsType description.

7.3.2.2 ProtocolType

Type Name	Definition	Data type and value
ProtocolType	This type lists the possible protocols for getting the item delivered either via Online Data Access or Online DataDelivery mechanisms.	Type: String Permitted Values: – ftp, File Transfer Protocol – sftp, SSH File Transfer Protocol) – ftps, FTP over SSL – P2P, peer to peer - Torrent – wcs, OGC Web Coverage Service – wms, OGC Web Map Service – e-mail – dds ESA satellite-based Earth Observation Data Dissemination System (DDS). It allows an authorised user to deliver a product file resident on the user ftp server to a DDS recipient station via the ESA satellites through Digital Video Broadcasting (DVB) standard. – http, Hyper Text Transfer Protocol – https

Table 7-2: ProtocolType definition.

7.3.2.3 PackageMedium

Names	Definition	Data type and value
packageMedium	Definition of the medium to use for deliverying ordered products.	Type: Not empty string (max 40 chars) Permitted Values: – NTP IBM's New Technology Prototype 10 GB tape – 1995 – DAT Digital Audio Tape, tape format defined by SONY in early ’90 – Exabyte

Names	Definition	Data type and value
		Data8 magnetic tape format from Exabyte company - 1987 – CD-ROM Compact Disc - Read-Only Memory – DLT Digital Linear Tape (DLT) (previously called CompacTape), magnetic tape data storage technology developed by Digital Equipment Corporation (DEC) - 1984 – D1 SMPTE digital VTR video standard, also a Sony and Bosch - BTS product. Tape format suitable for professional digital video format, introduced in 1986 – DVD Digital Versatile Disc or Digital Video Disc, optical disc storage media format – 1995 – BD Blu-ray Disc (BD), optical disc storage medium designed to supersede the standard DVD format. – LTO Linear Tape-Open (or LTO), magnetic tape data storage technology – 1990 – LTO2 – LTO4

Table 7-3: Package Medium definition.

7.3.3 ParameterDescriptorType

The ParameterDescriptorType ([OR9] [OR10]) defines the input a client has to provide for setting ordering options. One instance of this type represents one order option to be set for the specified order item.

The ParameterDescriptorType is an extension of a swe:AbstratDataComponentPropertyType (see §7.2.3 and 8.2.1 in [OR11]). It is an "*AbstractDataComponent*" complex type so an abstract base class for all components that in this context are order options. Each option can be:

- optional or mandatory, if the parameter can or shall be provided by the client.
- updatable or not, "updateable" is defined in the SPS specification, but in this context has to be set always false because update operation is not supported in ordering service.

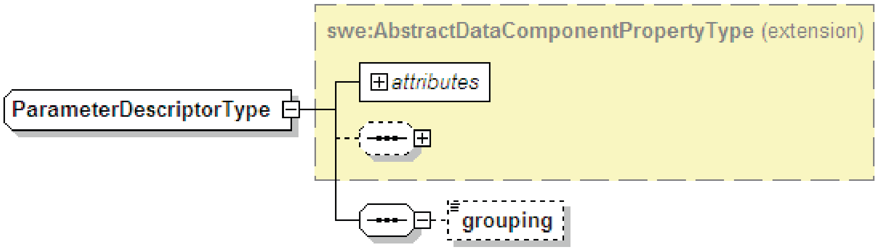


Figure 7-2 - ParameterDescriptorType diagram.

Using swe entities each option can be identified by an entity (simple or complex) as shown in Figure 7-3 and Table 7-5.

Names	Definition	Data type and value	Multiplicity and use	Product Ordering	Subscription
swe:AbstractDataComponentProperty	Swe entity for order/scene option definition	Type swe:AbstractDataComponentPropertyType (see Table 7-5)	One (mandatory)	X	X
grouping	Additional identifier for grouping correlated order options.	Type: string (max 40)	One (optional)	X	X

Table 7-4 - ParameterDescriptorType description.

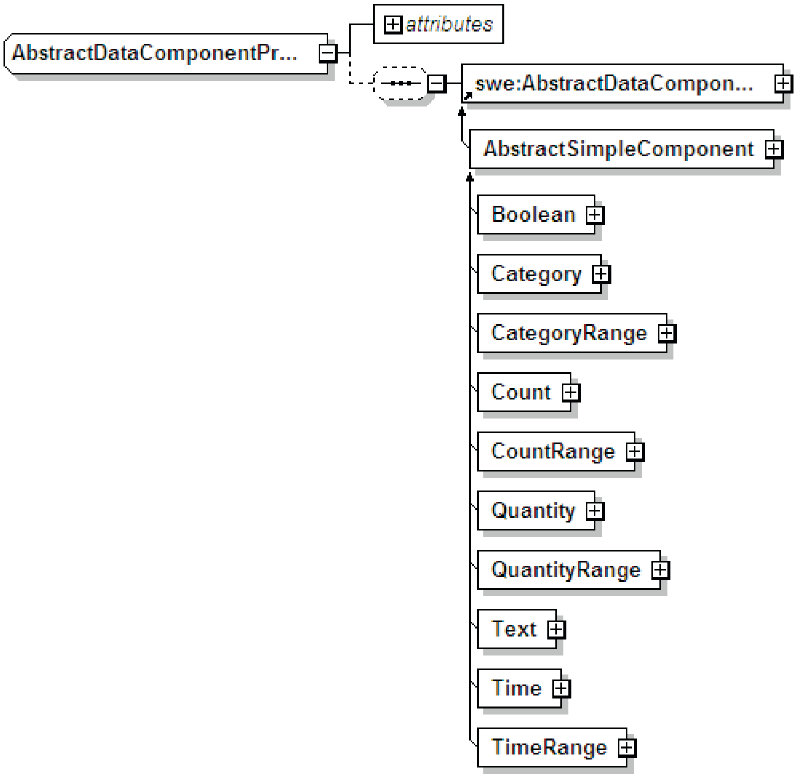


Figure 7-3 – swe:AbstractDataComponentDataType diagram.

Names	Definition	Data type and value	Multiplicity and use
Boolean	Swe scalar type. To specify a scalar data component with a boolean representation.	Type swe:BooleanType (see §8.2.2 [OR11])	One (optional)
Category	Swe basic type. To specify a scalar data component with a categorical representation	Type swe:CategoryType (see §8.2.4 [OR11])	One (optional)
CategoryRange	Swe Range Type. To specify a scalar data component with a discrete numerical representation	Type swe:CategoryRangeType (see §8.2.8 [OR11])	One (optional)
Count	Swe basic type. To specify a scalar data component with a countable representation	Type swe:CountType (see §8.2.5 [OR11])	One (optional)
CountRange	Swe Range Type. To specify a range extending the Count	Type swe:CountRangeType (see §8.2.9 [OR11])	One (optional)
Quantity	Swe basic type. To specify a scalar data component with a continuos numerical representation	Type swe:QuantityType (see §8.2.6 [OR11])	One (optional)

Names	Definition	Data type and value	Multiplicity and use
QuantityRange	Swe Range Type. To specify a range extending the Quantity	Type swe:QuantityRangeType (see §8.2.10 [OR11])	One (optional)
Text	Swe basic type. To specify a scalar data component with a textual representation	Type swe:TextType (see §8.2.3 [OR11])	One (optional)
Time	Swe basic type. To specify a scalar data component with a data-time representation	Type swe:TimeType (see §8.2.6 [OR11])	One (optional)
TimeRange	Swe Range Type. To specify a range extending the Time	Type swe:TimeRangeType (see §8.2.11 [OR11])	One (optional)
DataArray	Swe Complex Type. To define a array with some items	Type swe:DataArrayType (see §8.4.1[OR11])	One (optional)
DataChoice	Swe Complex Type. To specify a choise between more components.	Type swe:DataChoiseType (see §8.3.2 [OR11])	One (optional)
DataRecord	Swe Complex Type To define a record with some fields	Type swe:DataRecordType (see §8.3.1 [OR11])	One (optional)
Matrix	Swe Complex Type. To define multi-dimensions quantity.	Type swe:MatrixType (see §8.4.2 [OR11])	One (optional)
Vector	Swe Complex Type. To define multi-dimensions quantity	Type swe:VectorType (see §8.3.3 [OR11])	One (optional)

Table 7-5 – swe:AbstractDataComponentDescriptorType description.

7.3.4 Pre-defined list of ordering options

This section reports the list of already identified order options prepared through a survey on ESA, CNES, EUMETSAT, DLR and CSA ground segments.

Even if the list of order options is completely dynamic, as specified in the previous paragraphs, for the sake of interoperability it is strongly recommended to re-use the following list as much as possible.

For each order option the following table reports:

- the name;
- the type i.e. the SWE Common element used for representing it
- the constraint model: i.e. the SWE Common elements is used for defining restrictions on the possible values.
- the description;
- an example XML fragment.

Name (swe:Field or swe:elementType)	Type	Constraints Model	Description	Example
BitsOutput	swe:DataRecord / swe:field/ swe:Count	swe:Count/ swe:constraint/ swe:AllowedValues / swe:value	Bits per output format Possible values: – 8 – 16	<option> <swe:DataRecord> <swe:field name="BitsOutput"> <swe:Count updatable= " false" optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/BitsOuputs"> <swe:constraint> <swe:AllowedValues> <swe:value>8</swe:value> <swe:value>16</swe:value> </swe:AllowedValues> </swe:constraint> </swe:Count> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>
Compression	swe:DataRecord / swe:field/ swe:Category	swe:Category/ swe:constraint/ swe:AllowedTokens /swe:value	Type of compression applied to the delivered products. E.g.: none, zip, gzip, bzip2, pkzip, comp	<option> <swe:DataRecord> <swe:field name="Compression"> <swe:Category updatable= " false" optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/Compression"> <swe:constraint> <swe:AllowedTokens> <swe:value>zip</swe:value> <swe:value>gzip</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Count> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>

Name (swe:Field or swe:elementType)	Type	Constraints Model	Description	Example
Consolidation	swe:DataRecord / swe:field/ swe:Category	swe:Category/ swe:constraint/ swe:AllowedTokens /swe:value	Allow to define if the processing is consolidated or not. Possible allowed values: – unconsolidated – consolidated	<option> <swe:DataRecord> <swe:field name="Consolidation"> <swe:Category updatable= " false" optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/Consolidation"> <swe:constraint> <swe:AllowedTokens> <swe:value>unconsolidated</swe:value> <swe:value>consolidated</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>
DopplerGrid	swe:DataRecord / swe:field/ swe:Boolean		Allow to define if it is included a Doppler grid.	<option> <swe:DataRecord> <swe:field name="DopplerGrid"> <swe:Boolean updatable= ”false” optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/DopplerGrid"/> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>

Name (swe:Field or swe:elementType)	Type	Constraints Model	Description	Example
EarthModel	swe:DataRecord / swe:field/ swe:Category	swe:Category/ swe:constraint/ swe:AllowedTokens /swe:value	For geo-coded products. Possible allowed values: - http://www.opengis.net/def/crs/OGC/1.3/CRS84	<option> <swe:DataRecord> <swe:field name="EarthModel"> <swe:Category updatable= ”false” optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/EarthModel"> <swe:constraint> <swe:AllowedTokens> <swe:value>http://www.opengis.net/def/crs/OGC/1.3/CRS84</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>
ElevationCorrection	swe:DataRecord / swe:field/ swe:Category	swe:Category/ swe:constraint/ swe:AllowedTokens /swe:value	For geo-coded products. Possible allowed values: – none – mean terrain height – DEM	<option> <swe:DataRecord> <swe:field name="ElevationCorrection"> <swe:Category updatable= ”false” optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/ElevationCorrection"> <swe:constraint> <swe:AllowedTokens> <swe:value>none</swe:value> <swe:value>mean terrain height</swe:value> <swe:value>DEM</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>
Format	swe:DataRecord / swe:field/ swe:Category	swe:Category/ swe:constraint/ swe:AllowedTokens /swe:value	Format file to use for delivery data.Between the possible values there is also the JPEG2000 format, in this	<option> <swe:DataRecord> <swe:field name="Format"> <swe:Category updatable= ”false” optional="true" definition= "http://www.opengis.net/def/order/OGC-

Name (swe:Field or swe:elementType)	Type	Constraints Model	Description	Example
			case it has not to use the compression. Some examples of allowed values: – ENVISAT – jpeg2000LL – jpeg2000LS – jpeg2000VL – CEOS – ESA – JAXA CEOS – TIFF/1byte – TIFF/2bytes – GEOTIFF/1byte – GEOTIFF/2bytes – DIMAP – GEOTIFF – BUFR – GRIB – HDF5 – JPEG – NetCDF – NITF – PNG – PNGBW – TIFF – TIFFBW – ZIP	EO/0/Format"> <swe:constraint> <swe:AllowedTokens> <swe:value>ENVISAT</swe:value> <swe:value>ESA</swe:value> <swe:value>JPEG</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>

Name **(swe:Field or swe:elementType)**	**Type**	**Constraints Model**	**Description**	**Example**
GeoCorrection	swe:DataRecord / swe:field/ swe:Category	swe:Category/ swe:constraint/ swe:AllowedTokens /swe:value	Geometric Correction. Possible allowed values: – geo-coded – geo-referenced.	<option> <swe:DataRecord> <swe:field name="GeoCorrection"> <swe:Category updatable= ”false” optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/GeoCorrection"> <swe:constraint> <swe:AllowedTokens> <swe:value>geo-coded</swe:value> <swe:value>geo-referenced</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>
MapDirection	swe:DataRecord / swe:field/ swe:Category	swe:Category/ swe:constraint/ swe:AllowedTokens /swe:value	Allow to define the direction for the map. Possible allowed values: – map north – true north – satellite	<option> <swe:DataRecord> <swe:field name="MapDirection"> <swe:Category updatable= ”false” optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/MapDirection"> <swe:constraint> <swe:AllowedTokens> <swe:value>map north</swe:value> <swe:value>satellite</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>

Name (swe:Field or swe:elementType)	Type	Constraints Model	Description	Example
MapProjection	swe:DataRecord / swe:field/ swe:Category	swe:Category/ swe:constraint/ swe:AllowedTokens /swe:value	Allow to define the projection for the map. Possible allowed values: – UTM – PS – ARC – TPL	<option> <swe:DataRecord> <swe:field name="MapProjection"> <swe:Category updatable= ”false” optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/MapProjection"> <swe:constraint> <swe:AllowedTokens> <swe:value>ARC</swe:value> <swe:value>TPL</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>
PixelSpacing	swe:DataRecord / swe:field/ swe:Quantity	swe:Quantity/ swe:constraint/ swe:AllowedValues /swe:value	Allow to define the distance between pixels. Possible allowed values: – 6.25 – 12.5	<option> <swe:DataRecord> <swe:field name="PixelSpacing"> <swe:Category updatable= ”false” optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/PixelSpacing"> <swe:constraint> <swe:AllowedValues> <swe:value>6.25</swe:value> <swe:value>12.5</swe:value> </swe:AllowedValues> </swe:constraint> </swe:Category> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>

Name (swe:Field or swe:elementType)	Type	Constraints Model	Description	Example
Polarisation	swe:DataRecord / swe:field/ swe:Category	swe:Category/ swe:constraint/ swe:AllowedTokens /swe:value	Allow to choose the polarisation. Possible allowed values: – HH/VV – HH/HV – VV/VH – default – H/H – V/V	<option> <swe:DataRecord> <swe:field name="Polarisation"> <swe:Category updatable= ”false” optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/Polarisation"> <swe:constraint> <swe:AllowedTokens> <swe:value>default</swe:value> <swe:value>VV/VH</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>
ProcessingLevel	swe:DataRecord / swe:field/ swe:Category	swe:Category/ swe:constraint/ swe:AllowedTokens /swe:value	Level associated to the products. E.g: for ESA ALOS AVNIR-2 Possible allowed values: – 0 – 1A – 1B1 – 1B2	<option> <swe:DataRecord> <swe:field name="ProcessingLevel"> <swe:Category updatable= ”false” optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/ProcessingLevel"> <swe:constraint> <swe:AllowedTokens> <swe:value>1A</swe:value> <swe:value>1B1</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>

Name (swe:Field or swe:elementType)	Type	Constraints Model	Description	Example
ProcessingMode	swe:DataRecord / swe:field/ swe:Category	swe:Category/ swe:constraint/ swe:AllowedTokens /swe:value	Possible values are: – standard – short latency	<option> <swe:DataRecord> <swe:field name="ProcessingMode"> <swe:Category updatable= ”false” optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/ProcessingMode"> <swe:constraint> <swe:AllowedTokens> <swe:value>standard</swe:value> <swe:value>short latency</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>
ProductType	swe:DataRecord / swe:field/ swe:Category	swe:Category/ swe:constraint/ swe:AllowedTokens /swe:value	Allow to define the type of product in relation to the mission and the sensor. Possible allowed values (some values from ENVISAT mission): – ASA_WV__0P – ASA_WVI_1P – ASA_WVS_1P – ASA_WVW_2P	<option> <swe:DataRecord> <swe:field name="ProductType"> <swe:Category updatable= ”false” optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/ProductType"> <swe:constraint> <swe:AllowedTokens> <swe:value>ASA_WV__0P</swe:value> <swe:value>ASA_WVI_1P</swe:value> <swe:value>ASA_WVS_1P</swe:value> <swe:value>ASA_WVW_2P</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>

Name (swe:Field or swe:elementType)	Type	Constraints Model	Description	Example
QualityOfService	swe:DataRecord/ swe:field/ swe:Category	swe:Category/ swe:constraint/ swe:AllowedTokens /swe:value	Allow to define the quality of the service. Possible allowed values: – RUSH – STANDARD – FASTEST	<option> <swe:DataRecord> <swe:field name="QualityOfService"> <swe:Category updatable= ”false” optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/QualityOfService"> <swe:constraint> <swe:AllowedTokens> <swe:value>STANDARD</swe:value> <swe:value>RUSH</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>
RadioCorrection	swe:DataRecord/ swe:field/ swe:Category	swe:Category/ swe:constraint/ swe:AllowedTokens /swe:value	Allow to modify the values in order to account for noise (the intervening atmosphere, the sun-sensor geometry, ecc..). In some cases, it needs to correct the data, for consistency between bands (for some multispectral techniques) or between image dates (temporal data) and sensors. Possible allowed values: – pre-flight – gain_life_time	<option> <swe:DataRecord> <swe:field name="RadioCorrection"> <swe:Category updatable= ”false” optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/RadioCorrection"> <swe:constraint> <swe:AllowedTokens> <swe:value>pre-flight</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>

Name (swe:Field or swe:elementType)	Type	Constraints Model	Description	Example
Resampling	swe:DataRecord / swe:field/ swe:Category	swe:Category/ swe:constraint/ swe:AllowedTokens /swe:value	To allow the definition of the sampling of the original data. Possible allowed values: – cubic convolution – nearest neighbour – bi-linear	<option> <swe:DataRecord> <swe:field name="Resampling"> <swe:Category updatable= ”false” optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/Resampling"> <swe:constraint> <swe:AllowedTokens> <swe:value>bi-linear</swe:value> <swe:value>cubic convolution</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>
SpectralBandColorComposition	swe:DataArray with swe:DataRecord / swe:field/ swe:Category name="RedBand" swe:field/ swe:Category name="GreenBand" swe:field/ swe:Category name="BlueBand"	For each field swe:Category/ swe:constraint/ swe:AllowedTokens /swe:value		<option> <swe:DataArray> <swe:elementCount> <swe:Count><swe:value>1</swe:value></swe:Count> </swe:elementCount> <swe:elementType name="SpectralBandColorComposition" > <swe:DataRecord updatable="false" optional="true" definition="http://www.opengis.net/def/order/OGC-EO/0/SpectralBandColorComposition"> <gml:description>A user can pick out of 12 bands 3 bands. A color can be assigned to a band (R,G,B). This is to allow a user to form a composite colored image</gml:description> <gml:name>spectralBandColorComposition</gml:name> <swe:field name="RedBand"> <swe:Category definition="http://www.opengis.net/def/order/OGC-EO/0/RedBand"> <swe:constraint> <swe:AllowedTokens> <swe:value>band1</swe:value> <swe:value>band2</swe:value> <swe:value>band3</swe:value> <swe:value>band4</swe:value> <swe:value>band5</swe:value>

Name (swe:Field or swe:elementType)	Type	Constraints Model	Description	Example
				<swe:value>band6</swe:value> <swe:value>band7</swe:value> <swe:value>band8</swe:value> <swe:value>band9</swe:value> <swe:value>band10</swe:value> <swe:value>band11</swe:value> <swe:value>band12</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> <swe:field name="GreenBand"> <swe:Category definition="http://www.opengis.net/def/order/OGC-EO/0/GreenBand"> <swe:constraint> <swe:AllowedTokens> <swe:value>band1</swe:value> <swe:value>band2</swe:value> <swe:value>band3</swe:value> <swe:value>band4</swe:value> <swe:value>band5</swe:value> <swe:value>band6</swe:value> <swe:value>band7</swe:value> <swe:value>band8</swe:value> <swe:value>band9</swe:value> <swe:value>band10</swe:value> <swe:value>band11</swe:value> <swe:value>band12</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> <swe:field name="BlueBand"> <swe:Category definition="http://www.opengis.net/def/order/OGC-EO/0/BlueBand"> <swe:constraint> <swe:AllowedTokens> <swe:value>band1</swe:value> <swe:value>band2</swe:value> <swe:value>band3</swe:value>

Name (swe:Field or swe:elementType)	Type	Constraints Model	Description	Example
				<swe:value>band4</swe:value> <swe:value>band5</swe:value> <swe:value>band6</swe:value> <swe:value>band7</swe:value> <swe:value>band8</swe:value> <swe:value>band9</swe:value> <swe:value>band10</swe:value> <swe:value>band11</swe:value> <swe:value>band12</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> </swe:DataRecord> </swe:elementType> </swe:DataArray> <grouping>Processing Option</grouping> </option>
SpectralBandComposition	swe:DataRecord / swe:Category	swe:Category/ swe:constraint/ swe:AllowedTokens /swe:value	Product Formats which can be provided in a multi spectral band selection shall be mapped to this order option. Between allowed values there are also: SEVEN_BANDS SEVEN_BANDS_PLUS_PAN	<option> <swe:DataRecord> <swe:field name="SpectralBandComposition"> <swe:Category updatable= ”false” optional="true" definition= "http://www.opengis.net/def/order/OGC-EO/0/SpectralBandComposition"> <swe:constraint> <swe:AllowedTokens> <swe:value> Band#1:VIS_0.6 </swe:value> <swe:value> Band#2:VIS_0.8</swe:value> <swe:value> Band#3: nIR_1.6</swe:value> <swe:value> Band#4: IR_3.9</swe:value> <swe:value> Band#5: WV_6.2 </swe:value> <swe:value> Band#6: WV_7.3 </swe:value> <swe:value> Band#7: IR_8.7 </swe:value> <swe:value> Band#8: IR_9.7</swe:value> <swe:value> Band#9: IR_10.8 </swe:value> <swe:value> Band#10: IR_12.0 </swe:value> <swe:value> Band#11: IR_13.4</swe:value> <swe:value> Band#12: HRV </swe:value> </swe:AllowedTokens> </swe:constraint>

Name (swe:Field or swe:elementType)	Type	Constraints Model	Description	Example
				</swe:Category> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>

Table 7-6 – Pre-defined list of Order Options.

7.3.4.1 Pre-defined list of options for Subscription orders

This section specifies the options for selecting a time window and the frequency of successive observations (for subscription orders).

Parameter Name	Type	Constraints Model	Description	Example
DateRange	swe:DataRecord/ swe:field/ swe:TimeRange		Allow to define the UTC time of starting and finishing period.	<option> <swe:DataRecord> <swe:field name="DateRange"> <swe:TimeRange updatable= "false" optional="false" definition= "http://www.opengis.net/def/dataType:ISO-19108:2002:TM_Period"> <swe:uom code="s"/> <swe:value> 2008-01-01T02:27:08.80Z 2008-01-31T02:27:08.80Z </swe:value> </swe:TimeRange> </swe:field> </swe:DataRecord> <grouping>Processing Option</grouping> </option>
FrequencyObservation	swe:DataArray with swe:DataRecord/ swe:field/ swe:Count name="numberOfObservations swe:field/ swe:Count name="observationGap"		Allow to specify how many times the revisiting has to be performed And the number of of days between two successive observations.	<option> <swe:DataArray> <swe:elementCount> <swe:Count><swe:value>1</swe:value></swe:Count> </swe:elementCount> <swe:elementType name="FrequencyObservation" > <swe:DataRecord definition= "http://www.opengis.net/def/order/OGC-EO/0/FrequencyObservation"> <swe:field name="numberOfObservations">

Parameter Name	Type	Constraints Model	Description	Example
				<swe:Count> <swe:value>20</swe:value> </swe:Count> </swe:field> <swe:field name="observationGap"> <swe:Count> <swe:value>2</swe:value> </swe:Count> </swe:field> </swe:DataRecord> </swe:elementType> </swe:DataArray> <grouping>Processing Option</grouping> </option>
RegionOfInterest	swe:DataArray with swe:Vector swe:field/ swe:Quentity name=”latitude” swe:field/ swe:Quantity name=”longitude”		This parameter allows users restricting the scope of the subscription to a specified area.	<option> <swe:DataArray definition="http://www.opengis.net/def/order/OGC-EO/0/RegionOfInterest" optional="false" updatable="false"> <gml:description>User area of interest on which the subscription shall be limited.</gml:description> <gml:name>RegionOfInterest</gml:name> <swe:elementType name="point"> <swe:Vector referenceFrame="http://www.opengis.net/def/crs/EPSG/7.1/4326"> <swe:coordinate name="latitude"> <swe:Quantity definition="http://sweet.jpl.nasa.gov/2.0/spaceCoordinates.owl#Latitude" axisID="Lat"> <swe:uom code="deg" /> </swe:Quantity> </swe:coordinate> <swe:coordinate name="longitude"> <swe:Quantity definition="http://sweet.jpl.nasa.gov/2.0/spaceCoordinates.owl#Longitude" axisID="Long"> <swe:uom code="deg" /> </swe:Quantity> </swe:coordinate> </swe:Vector> </swe:elementType> <swe:encoding><swe:XMLEncoding /> </swe:encoding> <swe:values>

Parameter Name	Type	Constraints Model	Description	Example
				<point><latitude>48.0</latitude><longitude>18.0</longitude></point> <point><latitude>48.0</latitude><longitude>47.0</longitude></point> <point><latitude>21.0</latitude><longitude>47.0</longitude></point> <point><latitude>21.0</latitude><longitude>18.0</longitude></point> <point><latitude>48.0</latitude><longitude>18.0</longitude></point> </swe:values> </swe:DataArray> <grouping>Processing Option</grouping> </option>

Table 7-7 – Pre-defined list of options for Subscription orders.

7.3.5 SceneSelectionDescriptorType

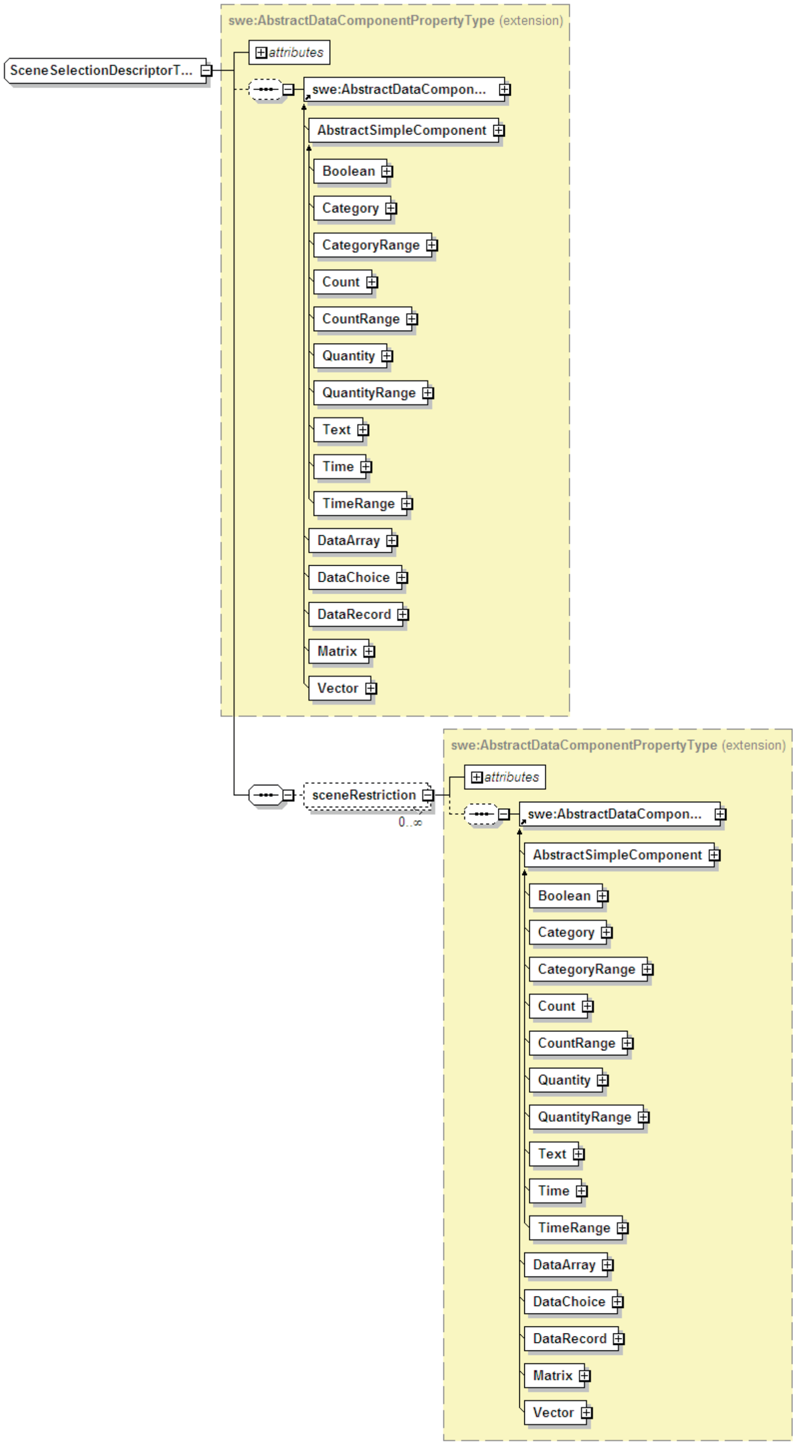


Figure 7-4 - SceneSelectionDescriptorType diagram.

For SceneSelectionDescriptorType definition see also ParameterDescriptorType (§7.3.3). The following table describes the SceneSelectionDescriptorType.

Scene selections are applicable only to Products Orders.

Names	Definition	Data type and values	Multiplicity and use
swe:AbstractDataComponentProperty	Swe entity for order/scene option definition.	Type swe:AbstractDataComponentPropertyType (see Table 7-5)	One (mandatory)
sceneRestriction	Restrictions defined during the scene definition	Type swe:AbstractDataComponentPropertyType (see Table 7-5)	Zero or more (optional)

Table 7-8 - SceneSelectionDescriptorType description.

7.3.6 Pre-defined list of Scene Selection Options

The following table reports the list of scene selection options identified on the basis of the contribution from the following Space Agencies:

- ESA
- EUMETSAT
- CNES
- DLR
- CSA

Even if the list of scene selection options is completely dynamic, as specified in the previous paragraphs, for the sake of interoperability it is strongly recommended to re-use the following list as much as possible.

For each scene selection option the following table reports:

- the name;
- the type i.e. the SWE Common element used for representing it
- the constraint model: i.e. the SWE Common elements is used for defining restrictions on the possible values.
- the description;
- an example XML fragment.

Name (swe:field or swe:elementType)	Type	Constraints Model	Description	Example
AcrossTrackSize	swe:DataRecord/ swe:field/ swe:Quantity (Kilometers)	swe:Quantity/ swe:constraint/ swe:AllowedValues/ swe:interval or swe:min or swe:max or swe:value	It is the scene across track size in Km.	<sceneSelectionParameter> <swe:DataRecord> <swe:field name="AcrossTrackSize"> <swe:Quantity updatable= ”false” optional="false" definition= "http://www.opengis.net/def/scene/OGC-EO/0/AcrossTrackSize"> <swe:uom code="km" /> <swe:constraint> <swe:AllowedValues> <swe:interval>1.0 10.0</swe:interval> </swe:AllowedValues> </swe:constraint> </swe:Quantity> </swe:field> </swe:DataRecord> </sceneSelectionParameter>
AlongTrackSize	swe:DataRecord/ swe:field/ swe:Quantity (Kilometers)	swe:Quantity/ swe:constraint/ swe:AllowedValues/ swe:interval or swe:min or swe:max or list of swe:value	It is the scene along track size in Km.	<sceneSelectionParameter> <swe:DataRecord> <swe:field name="AlongTrackSize"> <swe:Quantity updatable= ”false” optional="false" definition= "http://www.opengis.net/def/scene/OGC-EO/0/AlongTrackSize"> <swe:uom code="km" /> <swe:constraint> <swe:AllowedValues> <swe:interval>1.0 10.0</swe:interval> </swe:AllowedValues> </swe:constraint> </swe:Quantity> </swe:field> </swe:DataRecord> </sceneSelectionParameter>
ColumnsExtraction	Swe:DataRecord/ swe:field/ swe:CountRange	swe:CountRange/ swe:constraint/ swe:AllowedValues/ swe:interval or swe:min	AlbumExtract - First and Last column of the parent product to be extracted.	<sceneSelectionParameter> <swe:DataRecord> <swe:field name="ColumnsExtraction"> <swe:CountRange optional="false" updatable="false" definition="http://www.opengis.net/def/scene/OGC-EO/0/ColumnsExtraction">

Name (swe:field or swe:elementType)	Type	Constraints Model	Description	Example
		or swe:max or list of swe:value SubSatelliteLongitudePosition	The valid min/max values can be obtained using getOptions for a collection. This is also only valid for certain collections. Additionally for allowing the conversion between rows / columns and coordinates the SubSatelliteLongitudePosition is used.	<gml:description>First and last column on the parent product for scene extraction</gml:description> <gml:name>columnsExtraction</gml:name> <swe:constraint> <swe:AllowedValues> <swe:interval>1 100</swe:interval> </swe:AllowedValues> </swe:constraint> </swe:CountRange> </swe:field> </swe:DataRecord> <sceneRestriction> <swe:DataArray> <swe:elementCount><swe:Count><swe:value>1</swe:value></swe:Count></swe:elementCount> <swe:elementType name="SubSatellitePosition"> <swe:DataRecord definition="http://www.opengis.net/def/scene/OGC-EO/0/SubSatellitePosition" optional="false" updatable="false"> <gml:name>SubSatellitePosition</gml:name> <swe:field name="SubSatelliteLongitudePosition"> <swe:Quantity definition="http://www.opengis.net/def/scene/OGC-EO/0/SubSatellitePosition"> <swe:uom code="deg"></swe:uom> </swe:Quantity> </swe:field> <swe:field name="Satellite"> <swe:Category definition="http://www.opengis.net/def/scene/OGC-EO/0/Satellite"> </swe:Category> </swe:field> </swe:DataRecord> </swe:elementType> <swe:encoding> <swe:XMLEncoding></swe:XMLEncoding> </swe:encoding> <swe:values> <SubSatellitePosition> <SubSatelliteLongitudePosition>57.0</SubSatelliteLongitudePosition> <Satellite>MSG</Satellite>

Name (swe:field or swe:elementType)	Type	Constraints Model	Description	Example
				</SubSatellitePosition> </swe:values> </swe:DataArray> </sceneRestriction> </sceneSelectionParameter>
Frame	swe:DataRecord/ swe:field/ swe:Count	WRSInformation	The frame number specifies an acquisition in a well known position along the orbit. Each orbit is split in a predefined number of acquisitions with predefined distance between adjacent frames, predefined length and predefined overlap between adjacent frames. The position shall be calculated using the WRSInformation. The frame number is alternative to scene centre, sceneStartStopTime	See the annex GetOptionResponse_FullParameterSet.xml file

Name (swe:field or swe:elementType)	Type	Constraints Model	Description	Example
LookAngleP	swe:DataArray with swe:DataRecord/ swe:field/ swe:Quantity name="lookAngle " swe:field/ swe:Quantity name="FOV "	The limitation of the possible allowed pairs is implemented via the SceneRestriction	This parameter specifies all the allowed pairs of lookAngles and FOV – Field Of View	<sceneSelectionParameter> <swe:DataArray> <swe:elementCount> <swe:Count><swe:value>1</swe:value></swe:Count></swe:elementCount> <swe:elementType name="LookAngleP"> <swe:DataRecord optional="false" updatable="false" definition="http://www.opengis.net/def/scene/OGC-EO/0/LookAngleP"> <swe:field name="lookAngle"> <swe:Quantity definition="http://www.opengis.net/def/scene/OGC-EO/0/lookAngle"><swe:uom code="deg"></swe:uom></swe:Quantity> </swe:field> <swe:field name="FOV"> <swe:Quantity definition="http://www.opengis.net/def/scene/OGC-EO/0/FOV"><swe:uom code="deg"></swe:uom></swe:Quantity> </swe:field> </swe:DataRecord> </swe:elementType> </swe:DataArray> </sceneSelectionParameter>
Orbit	swe:DataRecord/ swe:field/ swe:Count	swe:Count/ swe:constraint/ swe:AllowedValues/ swe:interval or swe:min or swe:max or list of swe:value	Orbit number of the selected scene. The bounds can be returned by GetOptions calling this operation by product identifier.	<sceneSelectionParameter> <swe:DataRecord> <swe:field name="Orbit"> <swe:Count updatable= "false" optional="false" definition= "http://www.opengis.net/def/scene/OGC-EO/0/orbit"> <swe:constraint> <swe:AllowedValues> <swe:interval>1000 1000</swe:interval> </swe:AllowedValues> </swe:constraint> </swe:Count> </swe:field> </swe:DataRecord> </sceneSelectionParameter>
RowsExtraction	Swe:DataRecord/	swe:CountRange/	AlbumExtract - First	<sceneSelectionParameter>

Name (swe:field or swe:elementType)	Type	Constraints Model	Description	Example
	swe:field/ swe:CountRange	swe:constraint/ swe:AllowedValues/ swe:interval or swe:min or swe:max or list of swe:value SubSatelliteLongitudePosition	and Last row of the parent product to be extracted. The valid min/max values can be obtained using getOptions for a collection. This is also only valid for certain collections. Additionally for allowing the conversion between rows / columns and coordinates the SubSatelliteLongitudePosition is used.	<swe:DataRecord> <swe:field name="RowsExtraction"> <swe:CountRange updatable= ”false” optional="false" definition= " http://www.opengis.net/def/scene/OGC-EO/0/RowsExtraction"> <swe:constraint> <swe:AllowedValues> <swe:interval>1 100</swe:interval> </swe:AllowedValues> </swe:constraint> </swe:CountRange> </swe:field> </swe:DataRecord> <sceneRestriction> <swe:DataArray> <swe:elementCount> <swe:Count><swe:value>1</swe:value></swe:Count> </swe:elementCount> <swe:elementType name="SubSatellitePosition"> <swe:DataRecord definition="http://www.opengis.net/def/scene/OGC-EO/0/SubSatellitePosition" optional="false" updatable="false"> <gml:name>SubSatellitePosition</gml:name> <swe:field name="SubSatelliteLongitudePosition"> <swe:Quantity> <swe:uom code="deg"></swe:uom> </swe:Quantity> </swe:field> <swe:field name="Satellite"> <swe:Category definition="http://www.opengis.net/def/scene/OGC-EO/0/Satellite"></swe:Category> </swe:field> </swe:DataRecord> </swe:elementType> <swe:encoding> <swe:XMLEncoding></swe:XMLEncoding> </swe:encoding> <swe:values> <SubSatellitePosition> <SubSatelliteLongitudePosition>57.0</SubSatelliteLongitudePosition

Name **(swe:field or swe:elementType)**	**Type**	**Constraints Model**	**Description**	**Example**
				> <Satellite>MSG</Satellite> </SubSatellitePosition> </swe:values> </swe:DataArray> </sceneRestriction> </sceneSelectionParameter>
SAT	swe:DataRecord/ swe:field/ swe:Count	swe:Count/ swe:constraint/ swe:AllowedValues/ swe:interval or swe:min or swe:max or list of swe:value	SAT - Shift Along Track	<sceneSelectionParameter> <swe:DataRecord> <swe:field name="SAT"> <swe:Count optional="false" updatable="false" definition="http://www.opengis.net/def/scene/OGC-EO/0/SAT"> <swe:constraint><swe:AllowedValues> <swe:interval>1 10</swe:interval> </swe:AllowedValues> </swe:constraint> </swe:Count> </swe:field></swe:DataRecord> </sceneSelectionParameter>

Name (swe:field or swe:elementType)	Type	Constraints Model	Description	Example
SceneBoundingPolygon	swe:DataArray of swe:Vector swe:coordinate/ latitude swe:coordinate/ longitude	None	It is the scene polygon defined within the footprint of the parent product	<sceneSelectionParameter> <swe:DataArray definition="http://www.opengis.net/def/scene/OGC-EO/0/SceneBoundingPolygon" optional="false" updatable="false"> <gml:description>Scene Center coordinates</gml:description> <gml:name>SceneBoundingPolygon</gml:name> <swe:elementType name="point"> <swe:Vector referenceFrame=" http://www.opengis.net/def/crs/EPSG/7.1/4326"> <gml:name>Location Point</gml:name> <swe:coordinate name="latitude"> <swe:Quantity definition="http://sweet.jpl.nasa.gov/2.0/spaceCoordinates.owl#Latitude" axisID="Lat"> <swe:uom code="deg"/> </swe:Quantity> </swe:coordinate> <swe:coordinate name="longitude"> <swe:Quantity definition="http://sweet.jpl.nasa.gov/2.0/spaceCoordinates.owl#Longitude" axisID="Long"> <swe:uom code="deg"/> </swe:Quantity></swe:coordinate></swe:Vector></swe:elementType></swe:DataArray> </sceneSelectionParameter>

Name (swe:field or swe:elementType)	Type	Constraints Model	Description	Example
SceneCentreCoordinates	swe:Vector swe:coordinate/ latitude swe:coordinate/ longitude	None WRSInformation SceneWidth	It is the centre of the selected scene. It shall be within the footprint of the parent product. It is usually provided together other parameters (e.g. alongTrack, sceneStartStopTime). The parameter can be provided without constraints or with WRSInformation constraint (i.e. the scene centre shall comply with those of frames defined in the WRS) or with SceneWidth constraint.	<sceneSelectionParameter> <swe:DataRecord definition="http://www.opengis.net/def/scene/OGC-EO/0/SceneCentreCoordinates"> <swe:field name="SceneCentreCoordinates"> <swe:DataRecord> <swe:field name="latitude"> <swe:Quantity definition="http://sweet.jpl.nasa.gov/2.0/spaceCoordinates.owl#Latitude" axisID="Lat"> <swe:identifier>Latitude</swe:identifier> <swe:uom code="deg"/> </swe:Quantity> </swe:field> <swe:field name="longitude"> <swe:Quantity definition="http://sweet.jpl.nasa.gov/2.0/spaceCoordinates.owl#Longitude" axisID="Long"> <swe:identifier>Longitude</swe:identifier> <swe:uom code="deg"/> </swe:Quantity> </swe:field> </swe:DataRecord> </swe:field> </swe:DataRecord> </sceneSelectionParameter> Additional Example in GetOptionResponse_FullParameterSet.xml file

Name (swe:field or swe:elementType)	Type	Constraints Model	Description	Example
SceneBoundingRectangle	swe:DataArray of swe:Vector swe:coordinate/ latitude swe:coordinate/ longitude	None	Scene rectangle defined within the footprint of the parent product.	<sceneSelectionParameter> <swe:DataArray definition="http://www.opengis.net/def/scene/OGC-EO/0/SceneBoundingPolygon" optional="false" updatable="false"> <gml:description>Scene Center coordinates</gml:description> <gml:name>SceneBoundingPolygon</gml:name> <swe:elementType name="point"> <swe:Vector referenceFrame="http://www.opengis.net/def/crs/EPSG/7.1/4326"> <gml:name>Location Point</gml:name> <swe:coordinate name="latitude"> <swe:Quantity definition="http://sweet.jpl.nasa.gov/2.0/spaceCoordinates.owl#Latitude" axisID="Lat"> <swe:uom code="deg"/> </swe:Quantity> </swe:coordinate> <swe:coordinate name="longitude"> <swe:Quantity definition="http://sweet.jpl.nasa.gov/2.0/spaceCoordinates.owl#Longitude" axisID="Long"> <swe:uom code="deg"/> </swe:Quantity> </swe:coordinate> </swe:Vector> </swe:elementType> </swe:DataArray> </sceneSelectionParameter>
SceneStartStopTime	swe:DataRecord/ swe:field/ swe:TimeRange	Simple: swe:timeRange/swe:constraint/swe:AllowedTimes/swe:interval (2 swe:intervals) or swe:min or swe:max or list of swe:value	Temporal Selection using start/stop time. The valid start/stop times have to be obtained using getOptions with productId, since the products cover only a certain time range.	<sceneSelectionParameter> <swe:DataRecord> <swe:field name="SceneStartStopTime"> <swe:TimeRange optional="false" updatable="false" definition="http://www.opengis.net/def/scene/OGC-EO/0/SceneStartStopTime"> <swe:uom code="s"/> <swe:constraint> <swe:AllowedTimes> <swe:interval>before 2010-01-01T12:00:00.00Z</swe:interval> <swe:interval>after 2010-01-01T11:55:00.00Z</swe:interval> </swe:AllowedTimes> </swe:constraint>

Name (swe:field or swe:elementType)	Type	Constraints Model	Description	Example
		Complex: WRSInformation SceneWidth FloatingPassLength FloatingSceneLength	This is only available for certain collections. Depending on the mission, different constraints are possible: the simple one on time ranges or the complex ones: WRSInformation: the times shall comply with the values of frames defined in the WRS SceneWidth: the across track size can be limited FloatingPassLength: the time interval has also a minim length and step.	</swe:TimeRange> </swe:field> </swe:DataRecord> </sceneSelectionParameter> See also the annex GetOptionResponse_FullParameterSet.xml file

Name (swe:field or swe:elementType)	Type	Constraints Model	Description	Example
SceneType	swe:DataRecord/ swe:field/ swe:Category	swe:Count/ swe:constraint/ swe:AllowedTokens/ swe:value	This parameter specify the type of the selected scene e.g.: Floating Scene, Floating Pass, Floating Mini Scene, etc.	<sceneSelectionParameter> <swe:DataRecord> <swe:field name="SceneType"> <swe:Category optional="false" updatable="false" definition="http://www.opengis.net/def/scene/OGC-EO/0/SceneType" > <swe:constraint> <swe:AllowedTokens> <swe:value>FloatingScene</swe:value> <swe:value>FloatingPass</swe:value> <swe:value>FloatingMiniScene</swe:value> </swe:AllowedTokens> </swe:constraint> </swe:Category> </swe:field> </swe:DataRecord> </sceneSelectionParameter>
Track	swe:DataRecord/ swe:field/ swe:Count	swe:Count/ swe:constraint/ swe:AllowedValues/ swe:interval or swe:min or swe:max or list of swe:value	Track number of the selected scene. The bounds can be returned by GetOptions calling this operation by product identifier.	<sceneSelectionParameter> <swe:DataRecord> <swe:field name="Track"> <swe:Count optional="false" updatable="false" definition="http://www.opengis.net/def/scene/OGC-EO/0/Track"> <swe:constraint> <swe:AllowedValues> <swe:interval>100 100</swe:interval> </swe:AllowedValues> </swe:constraint> </swe:Count> </swe:field> </swe:DataRecord> </sceneSelectionParameter>

Table 7-9 - Scene Selection Parameter Definition.

The following table reports the identified list of constraints to be modelled with the sceneSelectionParameter/sceneRestriction element:

Name (swe:elementType)	Type	Description	Example
FloatingPassLength	swe:DataArray with swe:DataRecord/ swe:field/ swe:Quantity name="Length " swe:field/ swe:Category name="Step "	For floating pass scenes, the length shall comply with a min max range and the values shall comply with a defined step.	<sceneRestriction><swe:DataArray> <swe:elementCount><swe:Count><swe:value>1</swe:value></swe:Count></swe:elementCount> <swe:elementType name="FloatingPassLenght"> <swe:DataRecord definition="http://www.opengis.net/def/scene/OGC-EO/0/FloatingPassLength"> <swe:field name="Length"> <swe:QuantityRange> <swe:uom code="km"></swe:uom> </swe:QuantityRange> </swe:field> <swe:field name="Step"> <swe:Quantity> <swe:uom code="km"></swe:uom> </swe:Quantity> </swe:field> </swe:DataRecord> </swe:elementType> <swe:encoding> <swe:XMLEncoding></swe:XMLEncoding> </swe:encoding> <swe:values> <FloatingPassLenght><Lenght> <min>15.08</min><max>120</max> </Lenght> <Step>0.02</Step> </FloatingPassLenght> </swe:values></swe:DataArray></sceneRestriction>

Name (swe:elementType)	Type	Description	Example
FloatingSceneLength	swe:DataArray with swe:elementType/ swe:Quantity name="SubSatelliteLongitudePosition"	For floating scene, the length shall have a fixed value	<sceneRestriction> <swe:DataArray> <swe:elementCount><swe:Count><swe:value>1</swe:value></swe:Count> </swe:elementCount> <swe:elementType name="FloatingSceneLength"> <swe:Quantity optional="false" updatable="false" definition="http://www.opengis.net/def/scene/OGC-EO/0/FloatingSceneLength"> <gml:name>FloatingSceneLength</gml:name> > <swe:uom code="km"></swe:uom> </swe:Quantity> </swe:elementType> <swe:encoding> <swe:XMLEncoding></swe:XMLEncoding> </swe:encoding> <swe:values> <FloatingSceneLength>240.0</FloatingSceneLength> </swe:values> </swe:DataArray> </sceneRestriction>

Name (swe:elementType)	Type	Description	Example
SceneWidth	swe:DataArray with swe:elementType/ swe:Quantity name="SceneWidth"	This parameter specifies the allowed width in Kilometres.	<sceneRestriction> <swe:DataArray> <swe:elementCount><swe:Count><swe:value>1</swe:value></swe:Count></swe:elementCount> <swe:elementType name="SceneWidth"> <swe:Quantity definition="http://www.opengis.net/def/scene/OGC-EO/0/SceneWidth" updatable="false" optional="false"> <swe:uom code="km"/> </swe:Quantity> </swe:elementType> <swe:encoding> <swe:XMLEncoding></swe:XMLEncoding> </swe:encoding> <swe:values> <SceneWidth>450</SceneWidth> </swe:values> </swe:DataArray> </sceneRestriction>

Name (swe:elementType)	Type	Description	Example
SubSatellitePosition	swe:DataArray with swe:DataRecord/ swe:field/ swe:Quantity name="SubSatelliteLongitudePosition " swe:field/ swe:Category name="Satellite "	Parameter allowing the conversion between the album extract (rows and columns extraction) and the coordinates.	<sceneRestriction> <swe:DataArray> <swe:elementCount> <swe:Count><swe:value>1</swe:value> </swe:Count> </swe:elementCount> <swe:elementType name="SubSatellitePosition"> <swe:DataRecord definition="http://www.opengis.net/def/scene/OGC-EO/0/SubSatellitePosition" optional="false" updatable="false"> <swe:field name="SubSatelliteLongitudePosition"> <swe:Quantity> <swe:uom code="deg"></swe:uom> </swe:Quantity> </swe:field> <swe:field name="Satellite"> <swe:Category> </swe:Category> </swe:field> </swe:DataRecord> </swe:elementType> <swe:encoding> <swe:XMLEncoding></swe:XMLEncoding> </swe:encoding> <swe:values> <SubSatellitePosition> <SubSatelliteLongitudePosition>57.0 </SubSatelliteLongitudePosition> <Satellite>MSG</Satellite> </SubSatellitePosition> </swe:values> </swe:DataArray> </sceneRestriction>

Name (swe:elementType)	Type	Description	Example
WRSInformation	swe:DataArray of swe:DataRecord (one for each Phase) Each swe:DataRecord is composed of swe:field/swe:Count (name="FromOrbit") swe:field/swe:Count (name="ToOrbit") swe:field/swe:Quantity (name="TimePerFrame") swe:field/swe:Quantity (name="TimeBetweenFrameCenter") swe:field/swe:Count (name="FrameNumberStep") swe:field/swe:Count (name="FirstFrameNumber") swe:field/swe:Quantity (name="FirstFrameCenterTime") swe:field/swe:Count (name="LastFrameNumber")	This element describes the parameters for calculating the frame parameters within a specific satellite phase.	See the annex GetOptionResponse_FullParameterSet.xml file

Table 7-10 - Scene Selection Restriction Definition.

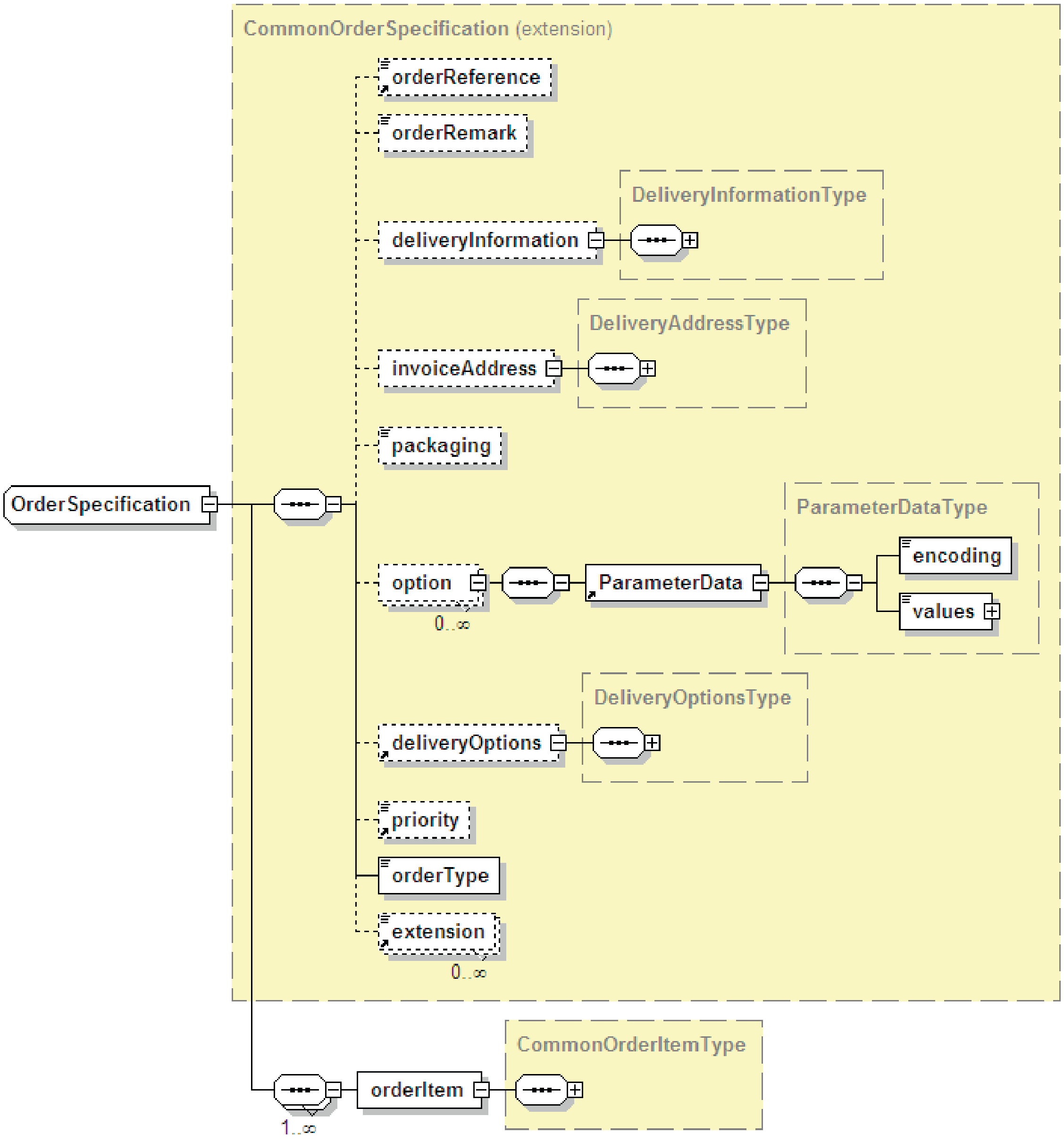


Figure 7-6 - OrderSpecification diagram.

The following table describes the OrderSpecification entity.

Names	Definition	Data type and values	Multiplicity and use	Product Ordering	Subscription
OrderSpecification	Specific parameters for submitting orders.	Inherited from CommonOrderSpecificatio n. (see Table 7-11)			
orderItem	Order Item element.	Type: CommonOrderItemType (see §7.3.8)	One or more (mandatory)	X	X

Table 7-12 - OrderSpecification description.

7.3.7.1 DeliveryInformationType

The following figure gives a graphical representation of DeliveryInformationType:

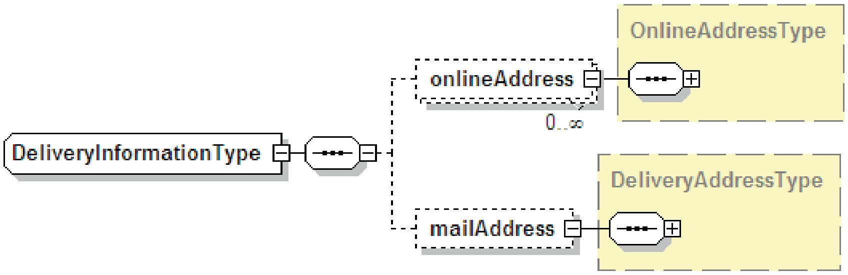


Figure 7-7 - DeliveryInformationType diagram.

The following table describes the DeliveryInformationType using all information within XML schema

Names	Definition	Data type and value	Multiplicity and use
onlineAddress	The on-line address for delivering the products	Type: OnlineAddressType (see §7.3.7.2)	One (optional)
mailAddress	Mail element.	Type: DeliveryAddressType (see §7.3.7.3).	One (optional)

Table 7-13 - DeliveryInformationType description.

7.3.7.2 OnlineAddressType

The following figure gives a graphical representation of FTPAddressType:

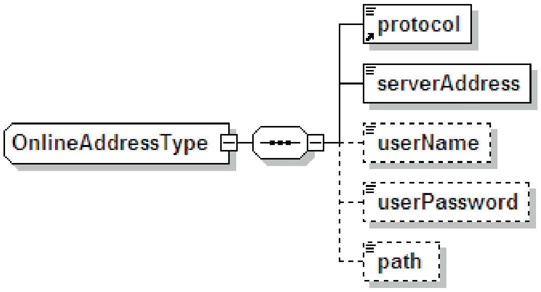


Figure 7-8 - OnlineAddressType diagram.

The following table describes the FTPAddressType using all information within XML schema

Names	Definition	Data type and value	Multiplicity and use
protocol	The protocol for on-line access/delivery of products	Type: ProtocolType (§7.3.2.2)	One (mandatory)
serverAddress	Address of the user FTP server the ordering service has to put the data.	Type: Not empty string (max 255)	One (mandatory)
userName	Identifier of the user for accessing the FTP server.	Type: Not empty string (max 8 chars)	One (mandatory)
userPassword	User password for accessing the FTP server.	Type: Not empty string (max 20)	One (mandatory)
path	Directory file path name where the ordering service has to put the products.	Type: Not empty string (max 1024 chars)	One (mandatory)

Table 7-14 - FTPAddressType description.

7.3.7.3 DeliveryAddressType

The following figure gives a graphical representation of DeliveryAddressType:

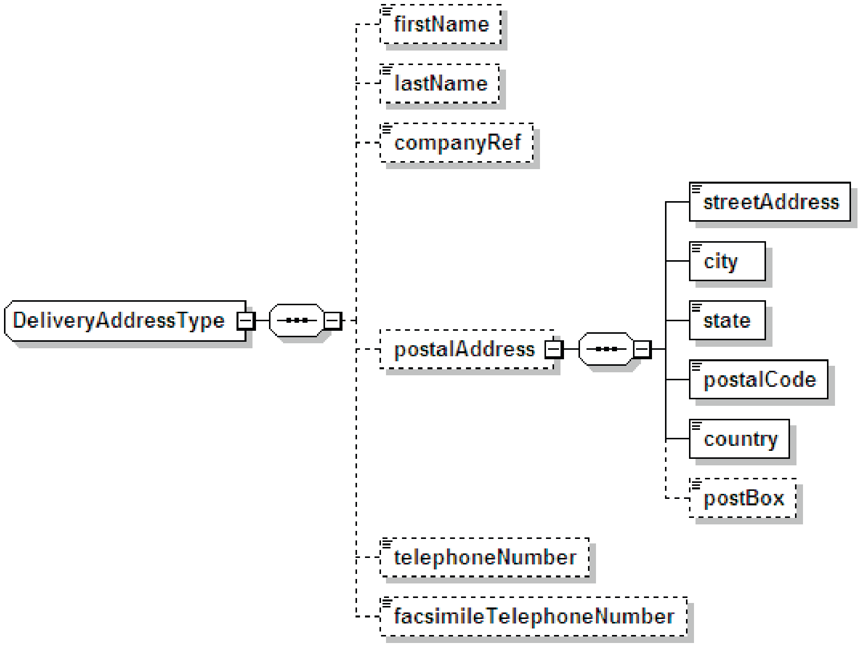


Figure 7-9 - DeliveryAddressType diagram.

The following table describes the DeliveryAddressType using all information within XML schema

Names	Definition	Data type and value	Multiplicity and use
firstName	FirstName to identify the receiving person.	Type: Not empty string (max 40 chars)	One (optional)
lastName	LastName to identify the receiving person.	Type: Not empty string (max 40 chars)	One (optional)
companyRef	Identification of the receiving entity.	Type: Not empty string (max 40 chars)	One (optional)
postalAddress	Postal Address of the user.		One (optional)
streetAddress	Street Address element.	Type: String (max 40 chars)	One (mandatory)
city	City element.	Type: String (max 40 chars)	One (mandatory)
state	State element.	Type: String (max 40 chars)	One (mandatory)
postalCode	Postal Code element.	Type: String (max 12 chars)	One (mandatory)
country	Country element. Compliant with ISO3166 (http://www.iso.org/iso/en/prods-services/iso3166ma/02iso-3166-code-lists/listen1.html)	Type: String (max 40 chars)	One (mandatory)

Names	Definition	Data type and value	Multiplicity and use
postBox	Postal Box element. Only number part, only digits allowed	Type: String (max 12 chars)	One (optional)
telephoneNumber	Telephone number of the receiving person.	Type: Not empty string (max 18 chars) matching the following regular expression: “\+?[0-9\(\)-\s]+” (An optional “+” sign followed by a series of (at least one) digit, “(“, “)”, “-“ and blank chars)	One (optional)
facsimileTelephoneNumber	FAX number of the receiving person.	Type: Not empty string (max 18 chars) matching the following regular expression: “\+?[0-9\(\)-\s]+” (An optional “+” sign followed by a series of (at least one) digit, “(“, “)”, “-“ and blank chars)	One (optional)

Table 7-15 - DeliveryAddressType description.

7.3.7.4 DeliveryOptionsType

The following figure gives a graphical representation of DeliveryOptionsType.

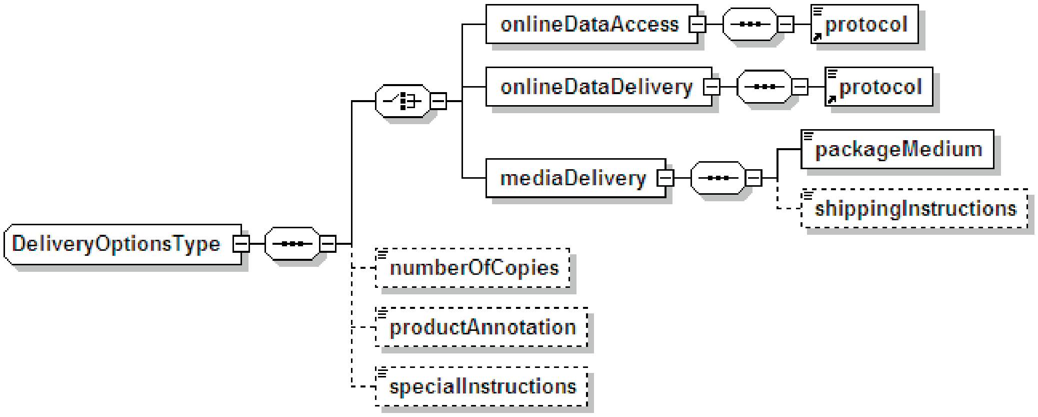


Figure 7-10 - DeliveryOptionsType diagram.

The following table describes the DeliveryAddressType.

Names	Definition	Data type and value	Multiplicity and use
onlineDataAccess	This element defines the selected Online Data Access		One (choice)

Names	Definition	Data type and value	Multiplicity and use
	options. It is alternative to onlineDataDelivery and to mediaDelivery.		
protocol		Type: ProtocolType (§7.3.2.2)	
onlineDataDelivery	This element defines the selected Online Data Delivery options. It is alternative to onlineDataAccess and to mediaDelivery.		One (choice)
protocol		Type: ProtocolType (§7.3.2.2)	
mediaDelivery	This element defines the selected Media Delivery options. It is alternative to onlineDataDelivery and to onlineDataAccess.		One (choice)
packageMedium	Definition of the medium to use for deliverying ordered products.	Type: PackageMedium (§7.3.2.3)	One (optional)
shippingInstructions	Instruction for delivery with ship	Type: string Permitted Values: – As each product is ready – Once all products are ready – other	One (optional)
numberOfCopies	Specifies how many copies of the same order item have to be delivered.	Type: integer	One (optional)
productAnnotation	an optional free-text field for delivered products	Type: string (max 4000 chars)	One (optional)
specialInstructions	Special instructions for delivery	Type: string (max 4000 chars)	One (optional)

Table 7-16 - DeliveryOptionsType description.

7.3.8 Order Item

This section defines all parameters a client has to specify for one item within an order (products/subscription).

7.3.8.1 CommonOrderItemType

The following figure gives a graphical representation of CommonOrderItemType.

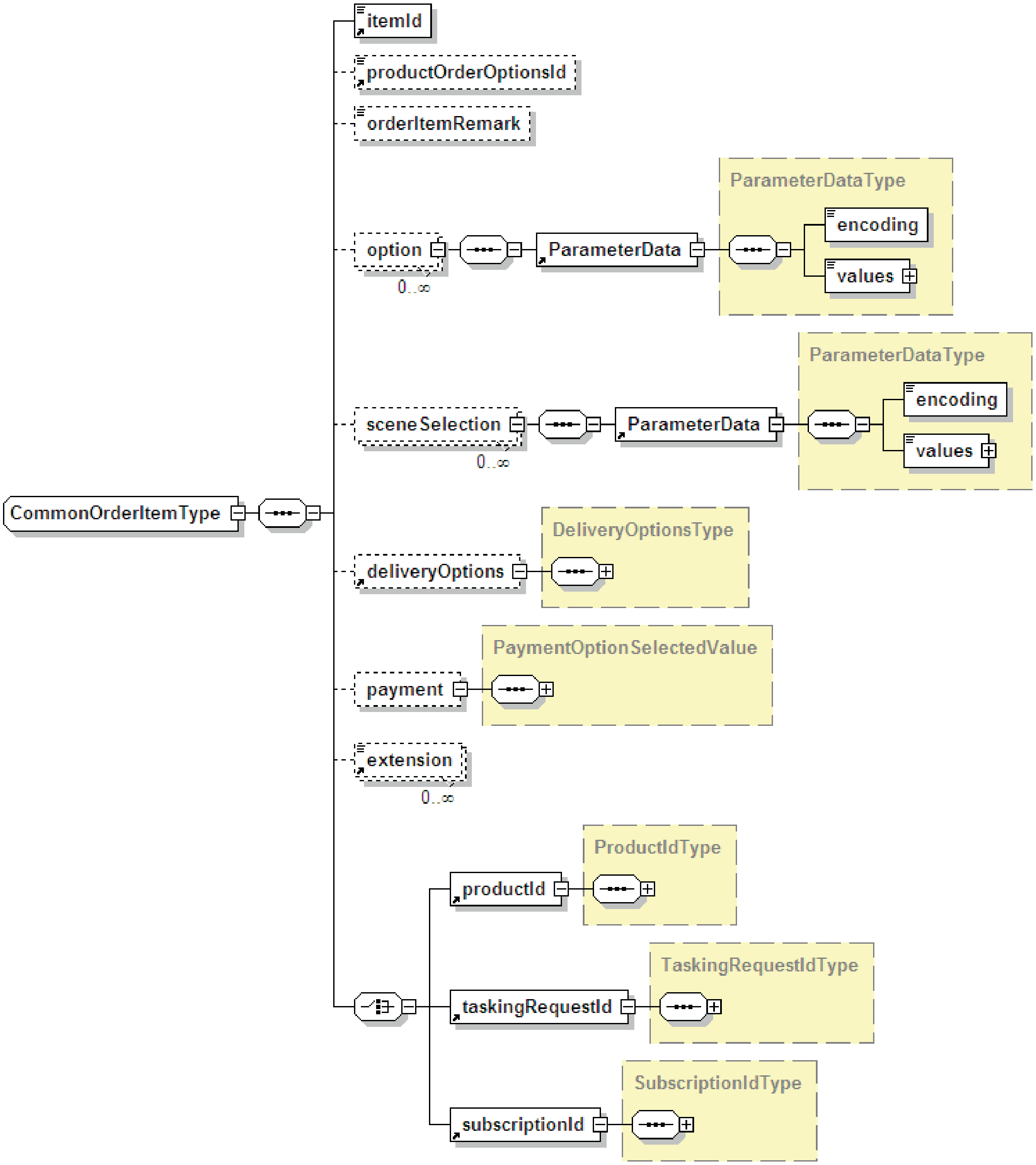


Figure 7-11 - CommonOrderItemType diagram.

The following table describes the CommonOrderItemType, which regroups all common parameters for one order item.

Names	Definition	Data type and values	Multiplicity and use	Product Ordering	Subscription
orderReference	User defined name assigned to that order.	Type: Not empty string (max 30 chars)	One (optional)	X	X
orderRemark	Textual remark on the order.	Type: Not empty string (max 4000 chars)	One (optional)	X	X
deliveryInformation	Delivery Information element.	Type: DeliveryInformationType (see §7.3.7.1)	One (optional)	X	X
invoiceAddress	Invoice Address element.	Type: DeliveryAddressType (see §7.3.7.3)	One (optional)	X	X
packaging	This element allows packing all ordered items in the same file.	Type: string Allowed values: zip, tar, tgz, bzip, bzip2, gzip, rar, 7z	One (optional)	X	X
option	This element represents the order options common to all order items. By default these options are applied to all order items, unless they are re-defined in the order item itself.	Type: sps:ParameterDataPropertyType (see §7.3.9)	Zero or more (optional)	X	X
deliveryOptions	This element represents the delivery options common to all order items within the order. By default these options are applied to all order items, unless they are re-defined in the order item itself.	Type: DeliveryOptionsType (see §7.3.7.4)	One (optional)	X	X
priority	Priority of the order	Type: string Allowed Values: STANDARD, FAST_TRACK	One (optional)		
orderType	This tag specifies whether the order option is related to a subscription order or to a product order.	Type: string Allowed Values: PRODUCT_ORDER, SUBSCRIPTION_ORDER TASKING_ORDER	One (mandatory)	X	X
extension	This element allows to specify additional implementation specific parameters.	Type: anyType	Zero or more (optional)	X	X

7.3.7 Order Specification

This section defines all parameters a client has to specify for submitting an order (products/tasking request / subscription).

For the definition of orders the following hierarchy of complex types has been declared:

- CommonOrderSpecification, which is the root of the hierarchy and includes all parameters common to order specification and order monitoring;
- OrderSpecification, inherited from CommonOrderSpecification and add the parameters specific for submitting orders;

These types are described in the following figures and tables:

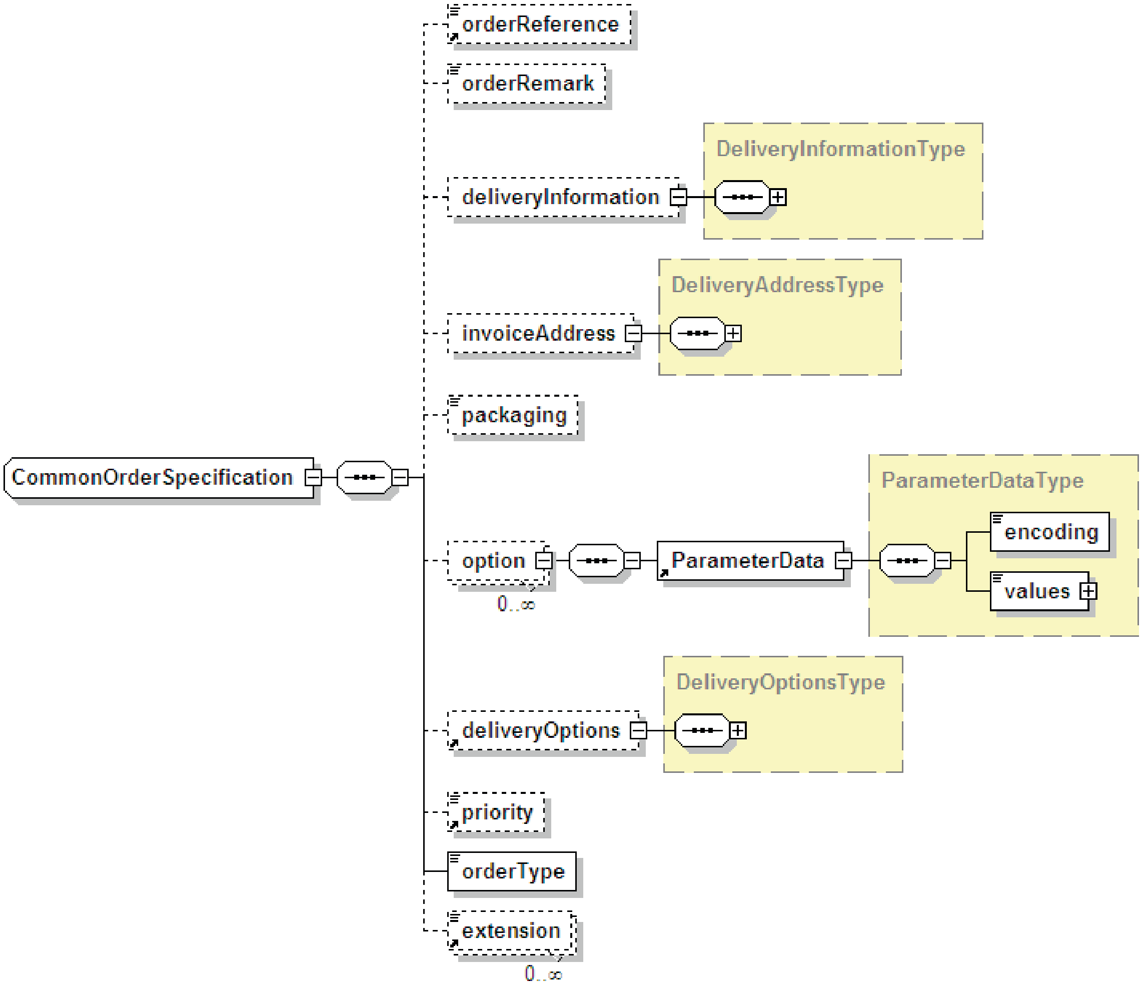


Figure 7-5 - CommonOrderSpecification diagram.

The following table describes the CommonOrderSpecification using all information within the XML schema.

Names	Definition	Data type and values	Multiplicity and use	Product Ordering	Subscription
itemId	Unique identifier of the item within the order.	Type: non empty string (max 80 char)	One (mandatory)	X	X
productOrderOptionsId	Identifier of chosen order option group	Type: non empty string (max 40 char)	One (optional)	X	X
orderItemRemark	Textual remark on the order item put by the issuer.	Type: Not empty string (max 4000 chars).	One (optional)	X	X
option	It specifies the options to be applied to the ordered item. The list of values specified in this element shall comply with the one returned by GetOptionsResponse. To be noted that order options can be specified also at order level (§7.3.6), but the definition at order item level overrides the global one.	Type: sps:ParameterDataPropertyType (see §7.3.9)	Zero or more (optional)	X	X
sceneSelection	This field has two purposes: - It specifies the selection of the scene from the product that is to be delivered. Then it is used from the processing chain for extracting the scene from the parent product. - It is the footprint of the product that has to be delivered. Then this value is used along the ordering chain for e.g.: rending on map, checking spatial restrictions, etc. This field has to be **always** provided unless the product has not a spatial coverage.	Type: SceneSelection (see §7.3.5)	Zero or more (optional)	X	
deliveryOptions	This element specifies the delivery options selected for the order item. To be noted that delivery options can be specified also at order level (§7.3.6), but the definition at order item level overrides the	Type: DeliveryOptionsType (see §7.3.7.4)	One (optional)	X	X

Names	Definition	Data type and values	Multiplicity and use	Product Ordering	Subscription
payment	Selected payment information by user. If specified, it overrides the possible payment method and payment info stored in the user profile.	Type: PaymentOptionSelectedValue	One (optional)	X	X
paymentMethod	Selected payment method: quota, credit card, etc.	Type: : non empty string (max 40 char) Examples: quota, invoice, prepay (to be indicated for free products), deposit account, credit card, credit card previously supplied	One (mandatory)	X	X
orderAccount	In case of payment by quota, this field specifies the account under which the user is authorised to order from the specific provider e.g. a project or service name. This field is alternative of 'creditCardInfo'.	Type: non empty string (max 20 chars)	One (optional/choise)	X	X
creditCardInfo	In case of payment by credit card it specifies the credit card information. This element should be managed in more secure way. This field is alternative of 'orderAccount'.	Type: String (max 255)	One (optional/choise)	X	X
extension	This element allows to specify additional implementation specific parameters.	Type: anyType	Zero or more (optional)	X	X
productId	This field is set for ordering items from Catalogue. Identifier of the target product on which the order item is based. This field is alternative to 'taskingRequestId' and 'subscriptionId'.	Type: ProductIdType (see Table 7-23)	One (mandatory/ choise)	X	

Names	Definition	Data type and values	Multiplicity and use	Product Ordering	Subscription
taskingRequestId	This element is set for ordering products coming from a tasking request issued to SPS instance linked to the Ordering Service. This field is alternative to 'ProductId' and 'subscriptionId'.	Type: TaskingRequestIDType (see Table 7-24)	One (mandatory/ choise)	X	
subscriptionId	It identifies the target subscription on which the order item is based. This field is alternative to 'ProductId' and 'taskingRequestId'	Type: SubscriptionIdType (see Table 7-25)	One (mandatory/ choise)		X

Table 7-17 - CommonOrderItemType description.

7.3.9 ParameterData

The ParameterData is used to provide the value for a specific order / scene selection option. The ParameterData element is therefore rather simple in its definition, it is identified by its name attribute that can be used for the XML encoding step.

In case of order option with nested definition, the value has to be set by defining appropriately the nested swe components element.

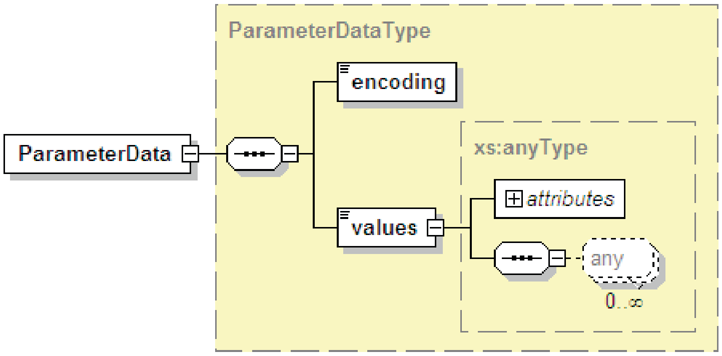


Figure 7-12 - ParameterData element diagram.

See also [OR9] [OR10].
The following table describes the sps:ParameterData using all information within the XML schema.

Names	Definition	Data type and values	Multiplicity and use	Product Ordering	Subscription
encoding	Value for identify the kind of encoding of the values assigned to options. Type of encoding foreseen are binaryEncoding, TextEncoding or XMLEncoding	Type: SWEEncoding Allowed values: □ XMLEncoding □ TextEncoding	One (mandatory)	X	X
values	Values to assign to the options. The representation of the values depends by the kind of encoding. (see §8.5 [OR11])	Type: anyType	One (mandatory)	X	X

Table 7-18 -ParameterData element description.

7.3.9.1 OGC HTTP URI Name Specification

As specified in the OGC 06-135r9, all new OGC identifiers issued for persistent public OGC resources shall be http URIs, instead of URNs; additionally new standards and new major versions of existing standards SHALL use http URIs for persistent public OGC resources to replace OGC URN identifiers defined in previous standards and versions, unless the OGC-NA approves an exception.

Then the identifiers within this document have been prepared according to the following ABNF rule (OGC 09-048r3):

URI = "http://www.opengis.net/" OGCResource "/" ResourceSpecificPath

OGCResource = "def"

ResourceSpecificPath = definition-type "/" authority "/" version "/" code

definition-type = segment-nz-nc ; a token from the register of OGC definition types (http://www.opengis.net/register/ogc-na/def-type).

authority = segment-nz-nc ; a token from the register of OGC authorities (http://www.opengis.net/register/ogc-na/authority).

version = segment-nz-nc / "0" ; use 0 for un-versioned names

code = segment-nz-nc *("/" segment-nz-nc)

segment-nz-nc = 1*pchar-nc

"version" is a required field. For un-versioned definitions the version field shall be "0".

The actual code may be composed of a sequence of fields delimited by "/".

7.3.9.2 Examples

This clause presents some examples relate to the application of OGC rules for URN definition.

7.3.9.2.1 swe:Category, single value

```
<swe:Category updatable=”false” optional=”false”
definition=”http://www.opengis.net/def/order/OGC-EO/0/QualityOfService”>
      <gml:name>Quality of the Service</gml:name>
      <swe:value>STANDARD</swe:value>
</swe:Category>
```

7.3.9.2.2 swe:Count, single value

```
<swe:Count updatable=”false” optional=”false”
definition=”http://www.opengis.net/def/scene/OGC-EO/0/Orbit”>
      <gml:name>Absolute orbit number</gml:name>
      <swe:value>100</swe:value>
</swe:Count>
```

7.3.9.3 Default values

All swe data components defined in [OR11] can be used either as data descriptors or data containers. To use it as data descriptors the *values* attributes are not set. To use it as a data container, the attribute *values* is set. Given values indicate default values. The Ordering Service can set default values for each parameter, but providing default values is optional. The client can either accept or overwrite this default value.

7.3.9.3.1 Example without default values

```
<swe:Quantity updatable=”false” optional=”false”
definition=”http://www.opengis.net/def/scene/OGC-EO/0/Longitude”>
      <gml:name>Longitude of the point</gml:name>
      <swe:uom code="deg"/>
</swe:Category>
```

7.3.9.3.2 Example with default values

```
<swe:Quantity updatable=”false” optional=”false”
definition=”http://www.opengis.net/def/scene/OGC-EO/0/Longitude”>
      <gml:name>Longitude of the point</gml:name>
      <swe:uom code="deg"/>
       <swe:value>45.0</swe:value>
</swe:Category>
```

7.3.10 Order Quotation

This section describes the information provided into the order quotation.

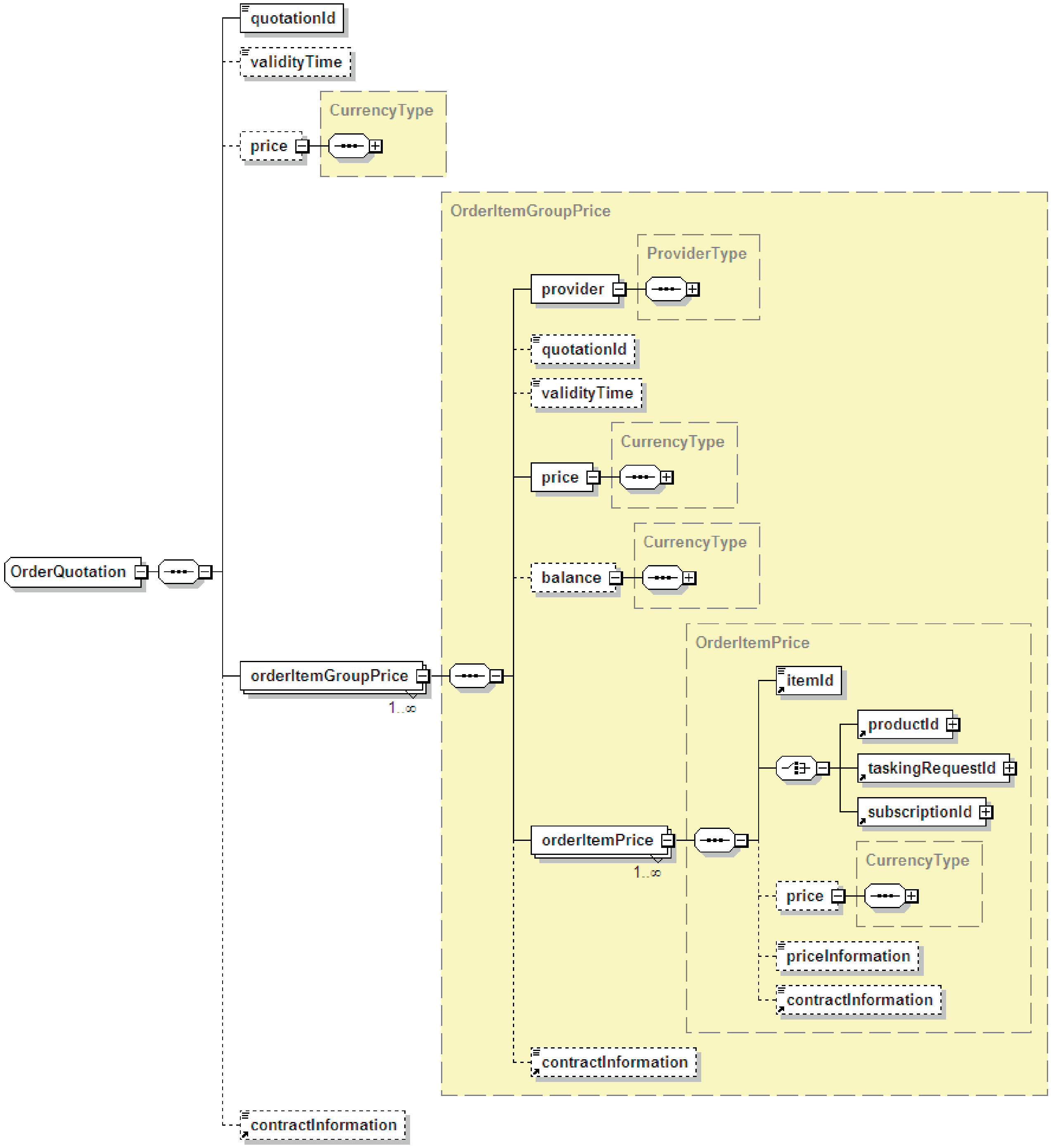


Figure 7-13 - OrderQuotation diagram.

The following table describes the OrderQuotation using all information within the XML schema.

Names	Definition	Data type and values	Multiplicity and use	Product Ordering	Subscription
quotationId	Identifier of the whole order quotation. URI conventions are followed for encoding this identifier.	Type: anyURI	One (mandatory)	X	X
validityTime	Time until the quotation of the whole order is valid. It is the minimum of the validity time of all orderItem groups.	Type: xs:dateTime	One (optional)	X	X
price	Total price of the order. Is not provided when the different orderItem groups have different payment methods (e.g. quota and credit card (USD))	Type: CurrencyType (see Table 7-22)	One (optional)	X	X
orderItemGroupPrice	Price information of order items groups. The ordered items are grouped because: – All items managed by the same provider have the same quotation rules; – Discounts can be applied when several items are ordered together; – Different provider can support different payment methods.	Type: OrderItemGroupPrice (see Table 7-20)	One or more (mandatory)	X	X
contractInformation	Textual description of rights and conditions applied to the whole order.	Type: string (max 1024 chars)	One (optional)	X	X

Table 7-19 - OrderQuotation description.

The following table describes the OrderItemGroupPrice using all information within the XML schema. The OrderItemGroupPrice provides all information relate to a group of ordered items.

Names	Definition	Data type and values	Multiplicity and use	Product Ordering	Subscription
provider	Provider which accepted the request for quotation for this group of items	Type: provider	One (mandatory)	X	X
serviceName	Service Name	Type: String (max 40 chars)	One (mandatory)	X	X
organization	Provider’s organization name.	Type: String (max 40 chars)	One (mandatory)	X	X
quotationId	Identifier of the order group quotation. This identifier is optional because can be returned when the quotation of the ordered items group is performed by an organization different from the one providing the Ordering service itself. URI conventions are followed for encoding this identifier.	Type: anyURI	One (optional)	X	X
validityTime	Time until the quotation of the order item group is valid.	Type: xs:dateTime	One (optional)	X	X
price	Price of the orderItemGroup.	Type: CurrencyType (see Table 7-22)	One (mandatory)	X	X
balance	In case of providers supporting payment by quota, this field returns the balance of the quota considering the price of this orderItemGroup.	Type: CurrencyType (see Table 7-22)	One (optional)	X	X
orderItemPrice	Price of the each item of the group.	Type: OrderItemPrice. (see Table 7-21)	One or more (mandatory)	X	X
contractInformation	Textual description of rights and conditions applied to the ordered items.	Type: String (max 1024 chars)	One (optional)	X	X

Table 7-20 - OrderItemGroupPrice description.

The following table describes the OrderItemPrice using all information within the XML schema. The OrderItemPrice provides all information relate to a single ordered item.

Names	Definition	Data type and values	Multiplicity and use	Product Ordering	Subscription
itemId	Identifier of the order item within the order. It is the same identifier specified in the order.	Type: string (max 80 chars)	One (mandatory)	X	X
productId	Used for ordering items from Catalogue. It identifies the target product on which the order item is based. This field is alternative to 'TaskingRequestId' and 'subscriptionId'	Type: ProductIdType (see Table 7-23)	One (mandatory/ choise)	X	
taskingRequestId	This element is set for ordering products coming from a tasking request issued to SPS instance linked to the Ordering Service. This field is alternative to 'productId' and 'subscriptionId'	Type: TaskingRequestIdType (see Table 7-24)	One (mandatory/choise)	X	
subscriptionId	It identifies the target subscription on which the order item is based. This field is alternative to 'TaskingRequestId' and 'productId'	Type: SubscriptionIdType (see Table 7-25)	One (mandatory/choise)		X
price	List price of the order item. Optional because the actual price is the one at group level.	Type: CurrencyType (see Table 7-22)	One (optional)	X	X
priceInformation	This element provide textual description in cases when the price for the item is not specified (e.g. the quotation for some items of the order is not supported / available off-line)	Type: xs:string (max 255)	One (optional)	X	X

Names	Definition	Data type and values	Multiplicity and use	Product Ordering	Subscription
contractInformation	Textual description of rights and conditions applied to the ordered item.	Type: string (max 1024 chars)	One (optional)	X	X

Table 7-21 - OrderItemPrice description.

The following table describes the CurrencyType using all information within the XML schema.

Names	Definition	Data type and values	Multiplicity and use	Product Ordering	Subscription
value	value	Type: double	One (mandatory)	X	X
currency	Currency including ISO 4217 (e.g.: EUR, USD (US Dollar), CAD (Canada Dollar), AUD (Australia Dollar), GBP (United Kingdom Pounds), etc.) and also special values not conflicting with the ISO ones for representing quota.	Type: string (max 10 chars)	One (mandatory)	X	X

Table 7-22 - CurrencyType description.

7.3.11 Order Item Identifier

This paragraph reports the definition of the identifier of the three possible items that can be ordered:

- Product
- Subscription
- Tasking Request

The following table describes the ProductType using all information within the XML schema.

Names	Definition	Data type and values	Multiplicity and use	Product Ordering	Subscription
identifier	Identifier for product element. Depending on the selected options, this item is either: – the ordered product – the parent product from which the product required by the client has to be prepared by applying the selected **options**.	Type: Not empty string Syntax: The identifier format is not mandated by the specification, but is expected to be a persistent identifier.	One (mandatory)	X	
collectionId	Search space for the product requested. It is referenced as parentIdentifier in the [OR3].	Type: Not empty string (max 255 chars) Syntax: it shall be a valid URN compliant with OGC 06-131, which report the definition of the parentIdentifier, that is equivalent to the collectionId in this specification.	One (optional)	X	

Table 7-23 - ProductType description.

The following table describes the TaskingRequestIdType using all information within the XML schema

Names	Definition	Data type and values	Multiplicity and use	Product Ordering	Subscription
ID	Identifier for the feasibility study / tasking request.	Type: anyURI	One (mandatory)	X	

Table 7-24 - TaskingRequestIdType description

The following table describes the SubscriptionIdType using all information within the XML schema

Names	Definition	Data type and values	Multiplicity and use	Product Ordering	Subscription
collectionId	Search space for the product requested. It is referenced as parentIdentifier in the [OR3].	Type: Not empty string (max 255 chars) Syntax: it shall be a valid URN compliant with OGC 06-131, which report the definition of the parentIdentifier, that is equivalent to the collectionId in this specification.	One (mandatory)		X

Table 7-25 - SubscriptionIdType description

7.3.12 Order Monitoring Specification

This section defines all parameters returned to the client when getting the status of submitted orders.

For the definition of orders the following hierarchy of complex types has been declared:

- CommonOrderSpecification, which is the root of the hierarchy and includes all parameters common to order specification and order monitoring;
- CommonOrderMonitorSpecification, which includes all order monitoring parameters;

The first one has been already defined in previous sections, the second one is defined below.

7.3.12.1CommonOrderMonitorSpecification

This section describes the information provided into the order monitor.

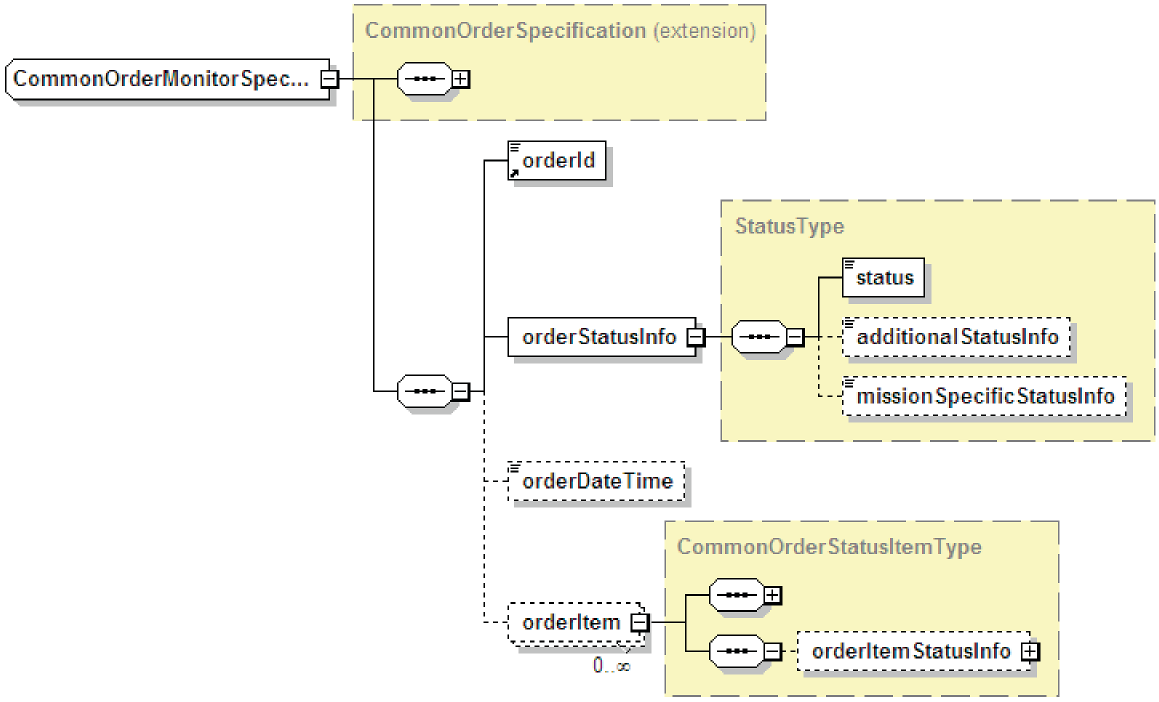

CommonOrderSpecification (extension)
CommonOrderMonitorSpec...
orderId
StatusType
status
orderStatusInfo
additionalStatusInfo
missionSpecificStatusInfo
orderDateTime
CommonOrderStatusItemType
orderItem
0..∞
orderItemStatusInfo

Figure 7-14 - CommonOrderMonitorSpecification diagram.

The CommonOrderMonitorSpecification specifies the common parameters returned into an order monitor XML document. CommonOrderMonitorSpecification is Inherited from CommonOrderSpecification (see Table 7-11).

The following table describes the CommonOrderMonitorSpecification using all information within the XML schema.

Names	Definition	Data type and value	Multiplicity and use	Product Ordering	Subscription	Presentation
CommonOrderMonitorSpecification	It specifies the common parameters returned into an order monitor XML document.	Inherited from CommonOrderSpecification (see Table 7-11)				The CommonOrderSpecification attributes are returned regardless the presentation
orderId	Unique identifier of the order for the order server.	Type: xs:anyURI	One (mandatory)	X	X	Brief & full
orderStatusInfo	Contains the Order status information.	Type: StatusType	One (mandatory)	X	X	Brief & full
status	Status of a product Order.	Type**:** String Current List of Valid Values: – Submitted (INTERMEDIATE status) – Accepted (INTERMEDIATE status) – InProduction (INTERMEDIATE status) – Suspended (INTERMEDIATE status) – Cancelled (previously accepted item cancelled at request of the customer). FINAL status. – Completed (made available to the user as per defined DeliveryMethod). FINAL	One (mandatory)	X	X	Brief & full

Names	Definition	Data type and value	Multiplicity and use	Product Ordering	Subscription	Presentation
		occurred during the processing of the order. FINAL status. – Terminated, when a subscription has been completed. FINAL status. – Downloaded, in case of on-line order items, it specifies whether the item has been downloaded from the user.				
additionalStatusInfo	Description associated with the Order status.	Type**:** Not empty string (max 4000 chars)	One (optional)	X	X	Brief & full
missionSpecificStatusInfo	Additional text description where mission specific information can be put.	Type: String (max 4000 chars)	One (optional)	X	X	Brief & full
orderDateTime	Date and Time of the order submission/update	Type: xs:dateTime	One (optional)	X	X	Brief & full
orderItem	This element reports the status information of the order item. This information is reported only in case of FULL presentation.	Type: CommonOrderStatusItemType.	One or More (optional)	X	X	full

Table 7-26 - CommonOrderMonitorSpecification description.

7.3.13 Order Item Monitoring Specification

This section specifies the status information returned for product and subscription order items.

For the definition of order item monitoring info, the following complex types have been defined:

- CommonOrderItemType, which regroups all attributes common to order submission and order monitoring;
- CommonOrderStatusItemType, which regroups the order status attributes;

The first one has been already defined in previous sections, the second one is defined below.

7.3.13.1CommonOrderStatusItemType

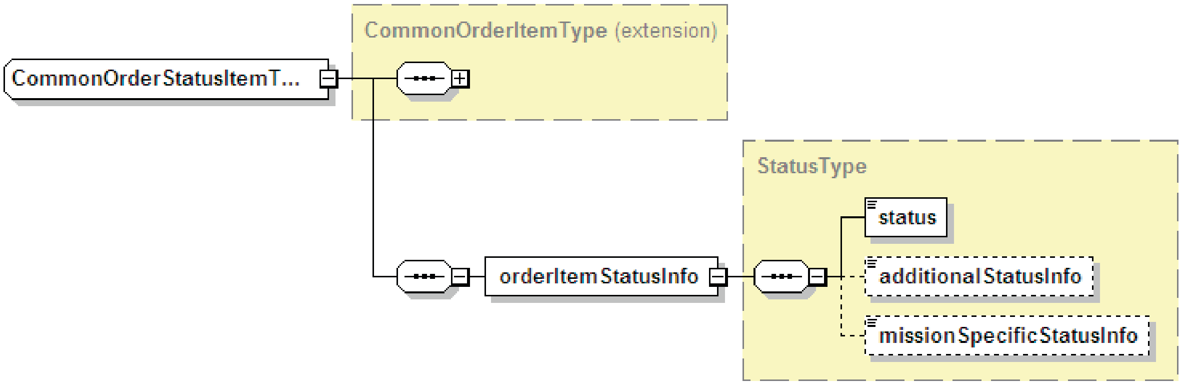

CommonOrderItemType (extension)
CommonOrderStatusItemT...
StatusType
status
orderItemStatusInfo
additionalStatusInfo
missionSpecificStatusInfo

Figure 7-15 - CommonOrderStatusItemType diagram.

The CommonOrderStatusItemType regroups the common status attributes and is inherited from CommonOrderItemType (see Table 7-17). The following table describes the CommonOrderStatusItemType using all information within the XML schema.

Names	Definition	Data type and value	Multiplicity and use	Product Ordering	Subscription
CommonOrderStatusItemType	It regroups the common status attributes.	Inherited from CommonOrderItemType (see Table 7-17)		X	X
orderItemStatusInfo	Status information at item level.	Type: StatusType	One (mandatory)	X	X
status	Status of order item.	Type: String Current List of Valid Values: – Submitted (INTERMEDIATE status) – Accepted (INTERMEDIATE status) – InProduction (INTERMEDIATE status) – Suspended (INTERMEDIATE status) – Cancelled (previously accepted item cancelled at request of the customer). FINAL status. – Completed (made available to the user as per defined DeliveryMethod). FINAL status. – Failed, when an error occurred during the processing of the order item. FINAL status. – Terminated, when a subscription has been completed. FINAL status. – Downloaded, in case of on-	One (mandatory)	X	X

Names	Definition	Data type and value	Multiplicity and use	Product Ordering	Subscription
		downloaded from the user			
additionalStatusInfo	Description associated with the Order status.	Type: Not empty string (max 4000 chars)	One (optional)	X	This free text field can be used for specifying subscription specific info e.g. delivered items, items going to be delivered, etc...
missionSpecificStatusInfo	Additional text description where mission specific information can be put.	Type: String (max 4000 chars)	One (optional)	X	X

Table 7-27 - CommonOrderStatusItemType description.

7.3.14 Extensibility Model

Even if this specification is quite flexible (e.g. the order and scene selection options are dynamic) an additional flexibility mechanism has been included: in fact the following element:

```
<xs:element name="extension" type="xs:anySimpleType"/>
```

With multiplicit 0 ..N has been included in:

- **OrderSpecification**, for setting additional parameter – implementation specific – at order level, which are not already covered by order options;
- **CommonOrderItemType**, for setting additional parameter – implementation specific – at order item level, which are not already covered by order options;
- **OrderSearchCriteriaType**, for setting additional - implementation specific – filtering criteria;

These elements allows to experiment additional functionalities – maintaining the backward compatibility – that can be included in future revisions of this specification.

Additionally they open pave the way for the definition and formalization of extensions for this specification (see SPS and SPS EO specifications).

7.4 Operation encoding

All operations must support the embedding of requests and responses in SOAP messages. Only SOAP messaging (via HTTP/POST) with document/literal style has to be used.

Messages must conform to SOAP 1.2 (http://www.w3.org/TR/SOAP/). The message payload will be in the body of the SOAP envelope.

For SOAP transfer, each XML-encoded operation request shall be encapsulated in the body of a SOAP envelope. In other words, the SOAP-Body shall be used for transmitting the request.
The optional SOAP-Header shall be used for optional elements in order to invoke the service. For instance, these elements could be identity tokens, licenses or other elements that are not necessarily required by the implementation specification.

Similarly, each XML-encoded operation response shall be encapsulated in the body of a SOAP envelope. Again, the optional header should be used for elements not directly related to the operation response, e.g. licenses or information targeted at an SOAP receiver in the SOAP message path.

If an error is detected while processing an operation request encoded in a SOAP envelope, the Ordering Service shall generate a SOAP response message where the content of the Body element is a Fault element containing an ExceptionReport element within the SOAP-Body.

7.4.1 Examples Operation encoding

7.4.1.1 Example of Request (GetStatus - Request)

XML Request encoding

```
<?xml version="1.0" encoding="UTF-8"?>
<GetStatus …>
…
</GetStatus>
```

SOAP Request encoding

```
<soap:Envelope xmlns:soap=”http://www.w3.org/2003/05/soap-envelope”
xmlns:ows=”http://www.opengis.net/ows/2.0”>
<soap:Header>
             …
      </soap:Header>

      <soap:Body>
       <GetStatus …> …
        </GetStatus>
      </soap:Body>
</soap:Envelope>
```

7.4.1.2 Example of Response (GetStatus – Response with success)

XML Response encoding

```
<?xml version="1.0" encoding="UTF-8"?>
<GetStatusResponse …>
             <status>success</status>
              … all info about the status of the order …
</GetStatusResponse>
```

SOAP Response encoding

```
<soap:Envelope xmlns:soap=”http://www.w3.org/2003/05/soap-envelope”
xmlns:ows=”http://www.opengis.net/ows/2.0” >
       <soap:Header>
             …
      </soap:Header>

      <soap:Body>
        <GetStatusResponse …>
             <status>success</status>
              … all info about the status of the order …
       </GetStatusResponse>
      </soap:Body>
</soap:Envelope>
```

7.4.1.3 Example of Response (GetStatus – Response with partial success)

XML Response encoding

```
<?xml version="1.0" encoding="UTF-8"?>
<GetStatusResponse …>
      <status>partial</status>
      <errorMessage>Reason of the partial result</errorMessage>
       … partial info about the status of the order …
</GetStatusResponse>
```

SOAP Response encoding

```
<soap:Envelope xmlns:soap=”http://www.w3.org/2003/05/soap-envelope”
xmlns:ows=”http://www.opengis.net/ows/2.0”>
       <soap:Header>
             …
      </soap:Header>

      <soap:Body>
        <GetStatusResponse …>
             <status>partial</status>
             <errorMessage>Reason of the partial result</errorMessage>
              … partial info about the status of the order …
         </GetStatusResponse>
      </soap:Body>
</soap:Envelope>
```

7.4.1.4 Example of Response (GetStatus – Response failed → SOAP FAULT)

XML Response encoding

```
<?xml version="1.0" encoding="UTF-8"?>
<soap:Fault>
 <soap:Code><soap:Value>…</soap:Value></soap:Code>
 <soap:Reason>
   <soap:Text>…<soap:Text>
 </soap:Reason>
 <soap:Detail>
    <ows:ExceptionReport>
         <Exception exceptionCode="…" locator="…">
          <ExceptionText>…</ExceptionText>
        </Exception>
    </ows:ExceptionReport>
 </soap:Detail>
</soap:Fault>
```

The ‘soap:Code’ element shall have the Value:

- “soap:Server” indicating that this is a Server exception
- “soap:Client” indicating that this is a Client exception.

The ‘soap:Reason’ element shall have the ‘soap:Text’ value:

- “Server exception was encountered”, indicating that this is a Server exception
- “Client exception was encountered”, indicating that this is a Client exception

The ‘ows:ExceptionReport’ element shall have the:

- ‘Exception’ entity with the ‘exceptionCode’ and ‘locator’ values that depend by operation.
- ‘ExceptionText’ entity with the value that describes the reason of the exception.

SOAP Response encoding

```
<soap:Envelope xmlns:soap=http://www.w3.org/2003/05/soap-envelope
xmlns:ows=”http://www.opengis.net/ows/2.0”>
       <soap:Header>
             …
      </soap:Header>

      <soap:Body>
       <soap:Fault>
          <soap:Code><soap:Value>…</soap:Value></soap:Code>
          <soap:Reason>
            <soap:Text>…<soap:Text>
          </soap:Reason>
          <soap:Detail>
              <ows:ExceptionReport>
                <Exception exceptionCode="…" locator="…">
                 <ExceptionText>…</ExceptionText>
                </Exception>
              </ows:ExceptionReport>
          </soap:Detail>
        </soap:Fault>
      </soap:Body>
</soap:Envelope>
```

7.4.2 List of Operations

The following table summarises the Ordering operations and their encoding methods that are applied in this document.

Ordering Operation	Request encoding	Sync / Async	Comment
GetCapabilities	**XML/SOAP**	**Synchronous request**	
GetOptions	**XML/SOAP**	**Synchronous request**	
GetQuotation	**XML/SOAP**	**Synchronous / Asynchronous request**	This operation can be used in different ways depending on the client and server ability: – Synchronous, when the server is able to provide a real time response; – Asynchronous via Notification, when the client is able to work as a server for getting the asynchronous notification – Asynchronous via Monitoring: the client has to ask the server until it returns the quotation. In case of Asynchronous via Notification usage, after the reception of this request, the order service calls the GetQuotationResponse operation provided by the client for sending the quotation.
GetQuotationResponse	**XML/SOAP**	**Call-back for asynchronous request**	This operation is called by the order service for sending the quotation in case of Asynchronous via Notification usage.
Submit	**XML/SOAP**	**Asynchronous request**	This operation is asynchronous and then, after the reception of this request, the order service calls (optionally) the SubmitResponse operation provided by the client.
SubmitResponse	**XML/SOAP**	**Call-back for asynchronous request**	This operation is called by the order service after the reception of Submit operations.
GetStatus	**XML/SOAP**	**Synchronous request**	
Cancel	**XML/SOAP**	**Asynchronous request**	This operation is asynchronous and then, after the reception of this request, the order service calls (optionally) the CancelResponse operation provided by the client.
CancelResponse	**XML/SOAP**	**Call-back for asynchronous request**	This operation is called by the order service after the reception of Cancel operations.
DescribeResultAccess	**XML/SOAP**	**Synchronous request**	

Table 7-28 - Operation request encoding

7.4.3 Asynchronous requests and replies

Regarding the asynchronous requests and replies the WS-addressing ([NR10]) SOAP header extensions have been used:

– In the SOAP header of GetQuotation, Submit and Cancel operations request messages the following tags have to be included (**wsa** is the namespace of WS-addressing definitions http://www.w3.org/2005/08/addressing):

```
<wsa:ReplyTo>
<wsa:Address> order service URI of the client </wsa:Address>
</wsa:ReplyTo>
<wsa:MessageID> unique identifier of the request </wsa:MessageID>
```

In case the reply address is set to:

```
http://www.w3.org/2005/08/addressing/anonymous
```

it means that the client does not have to be notified. It is useful for standard clients not having server capabilities. In this case the client has to ask the status of asynchronous requests calling dedicated operations (e.g. GetStatus for checking the status Submit and Cancel operations).

– In the SOAP header of GetQuotationResponse / SubmitResponse / CancelResponse operations request messages the following tags have to be included (**wsa** is the namespace of WS-addressing definitions):

```
<wsa:RelatesTo RelationshipType="wsa:Response">
Identifier of the GetQuotation, Submit or Cancel previously submitted requests
</wsa:RelatesTo>
```

The following sequence diagram explains the usage scenario of Submit, Cancel and GetQuotation operations:

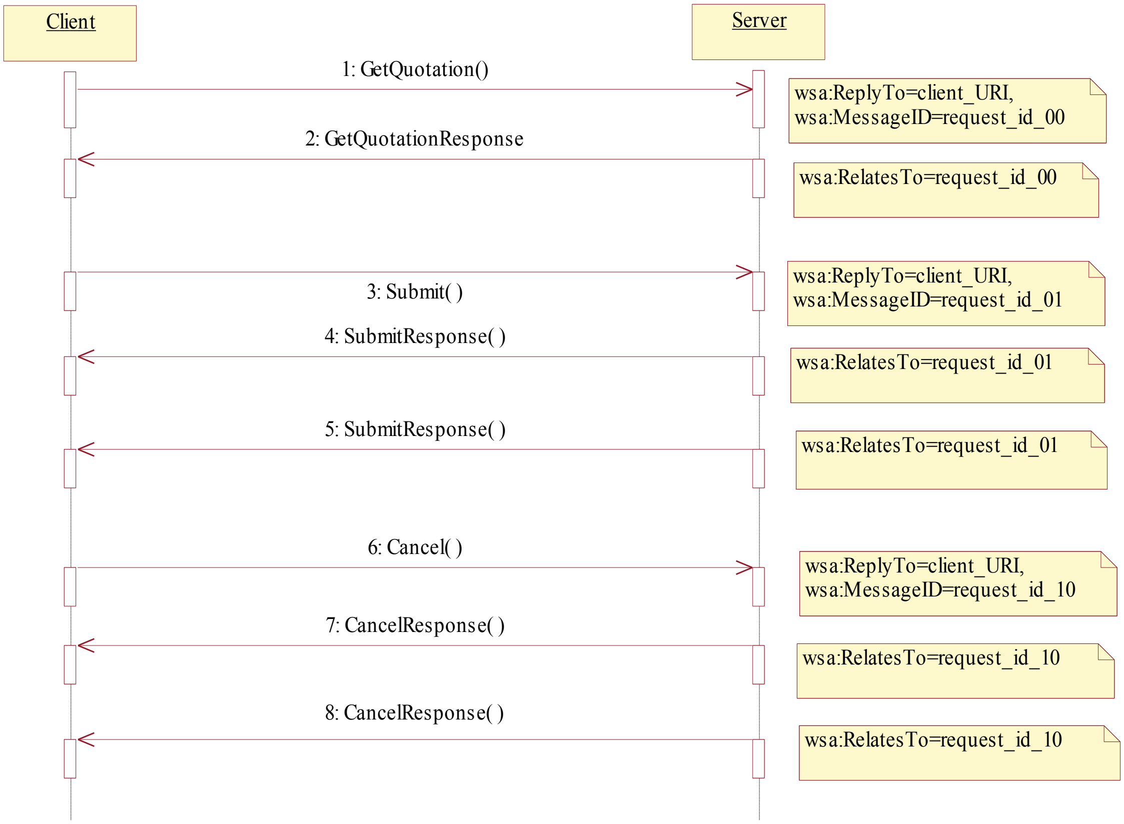


Figure 7-16 - Asynchronous requests scenario.

The previous figure explains the sequence of requests and responses for asynchronous operations (Cancel, Submit, GetQuotation). The asynchronous interaction is performed in 2 steps:

Step1. The client sends a SOAP request to the server and receives an acknoweledge from it.

- Submit:
  - Client request: Submit;
  - Server response: SubmitAck.
- Cancel:
  - Client request: Cancel;
  - Server response: CancelAck.
- GetQuotation:
  - Client request: GetQuotation;
  - Server response: GetQuotationAck.

Step2. The server sends the SOAP notification to the client (i.e. it calls an operation implemented by the client) and receives an acknoweledge from it.

- Submit:
  - o Server Notification (i.e. it issues a SubmitResponse request) and Client Response (SubmitResponseAck).
- Cancel:
  - o Server Notification (i.e. it issues a CancelResponse request) and Client Response (CancelResponseAck).
- GetQuotation:
  - o Server Notification (i.e. it issues a GetQuotationResponse request) and Client Response (GetQuotationResponseAck).

This mechanism is not the only one supported by this specification for notifying the clients. In fact it is also supported the WS-notification standard (See clause 7.5).

7.4.4 statusNotification element

Submit and **Cancel** operations are able to work in synchronous and asynchronous mode depending on the element:

statusNotification

This element is in charge of specifying whether the client has to be notified or not and, if yes, the amount of notifications:

- no notifications
- all status changes to be notified
- just the completion of the order submission to be notified

Names	Definition	Data type and value
statusNotification	This element specifies how many status notifications are sent back to the client.	Type: String Permitted Values: – None (no status notification sent back) – All (all status changes are notified) – Final (only the completion of the order is notified)

Table 7-29 - statusNotification description.

Considering that the asynchronous interactions are managed with the WS-addressing, then the statusNotification element and the WS-addressing have to be set accordingly:

- Synchronous answer:
  - o statusNotification = **None**
  - o wsa:ReplyTo = http://www.w3.org/2005/08/addressing/anonymous
- Asynchronous answer:
  - o statusNotification = **All** or **Final**
  - o wsa:ReplyTo = client notification address

other combinations raise the following SOAP Fault (§12.4.4, §16.4.4):

InvalidNotificationValue

7.4.5 User Identity Information

The format and protocol of user information, which are needed for authenticating and authorizing ordering service requests, is not covered by this ICD, but it is objective of [OR8] document.

7.5 Publish and Subscribe - usage of WS-Notification

This specification allows the definition of events which may be published to interested client via a publish / subscribe interface. These events represent events related to the status of orders or quotations. Using WS-Notification, the Publish/Subscribe functionality is enabled. OASIS WS-BaseNotification defines the interfaces required for basic publish / subscribe. WS-BaseNotification, WS-Topics and WS-BrokeredNotification provide a broad set of functionality to support more sophisticated use cases. A tutorial on WS-Notification is provided in [NR21] Annex A.

To better understand the notification behaviour it is useful to define the following roles:

- **NotificationConsumer:** it is an endpoint, represented by a WS-Addressing endpoint reference, designated to receive Notifications produced by a NotificationProducer as a result of a subscription.
- **NotificationProducer:** a Web service that implements the message exchanges associated with the NotificationProducer interface. It is capable of producing Notifications for those NotificationConsumers for which Subscriptions have been registered, based on situations that occur and on the parameters supplied with the requests from which the Subscriptions were created.
- **NotificationBroker:** it is a special case for NotificationProducer when the NotificationMessages were produced by a separate Publisher entity (not by the NotificationProducer).
- **Subscription**: for the simpler case, it represents the relationship between a NotificationConsumer and a NotificationProducer, including any filtering parameters such as Topic and various other optional filter expressions, along with any relevant policies and context information.
- **Subscriber**: it is any entity that sends a SubscribeRequest message to a NotificationProducer. Suscriber and NotificationConsumer may be the same entity.
- **Topic**: is the mean used to categorize Notifications and their related Notification schemas. Topics are used as part of the matching process that determines which (if any) subscribing NotificationConsumers should receive a Notification.
- **Topic Set:** The collection of Topics supported by a NotificationProducer .

A Web service that wishes to obtain information (which we call NotificationConsumers or Client) is registered dynamically with a Web service that is capable of distributing information (which we call NotificationProducer or Server). As part of this registration process, the NotificationConsumers provides some indication of the nature of the information that they wish to receive (a Subscription).

The NotificationProducer disseminates information by sending messages to the NotificationConsumers that are registered to receive the information. It is possible that more than one NotificationConsumer is registered to consume the same information.

WS-Notification specifies a standard set of message exchanges that define the roles of NotificationProducer and NotificationConsumer. The WS-Notification interfaces have been defined in a way which allows the implementation to be delegated to a middleware provider. This specification allows different middleware providers to interoperate. The following diagram explains the usage scenario of Publish/Subscribe functionality:

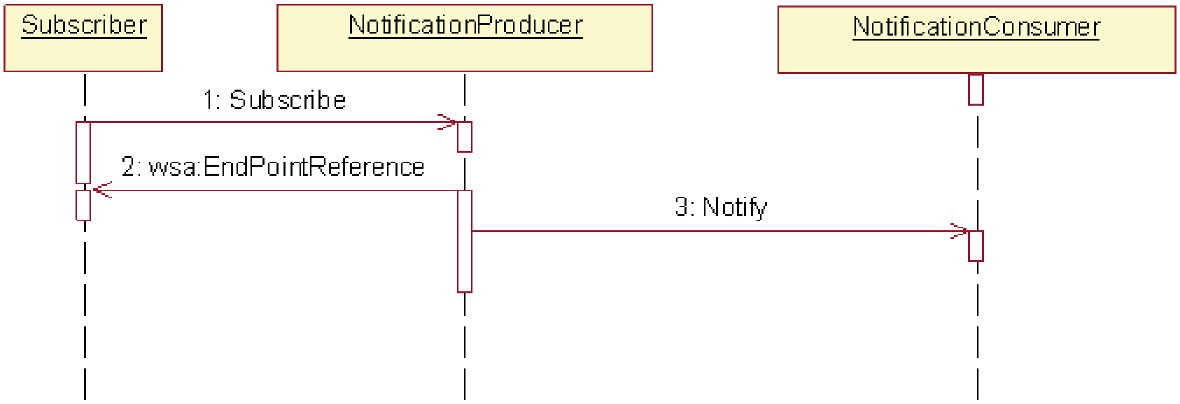


Figure 7-17 - Sequence diagram for Publish/Subscribe functionality

In the diagram the Subscriber and the NotificationConsumer are not the same entity but this is what happens often.
The Subscriber sends a Subscribe request message to the NotificationProducer, indicating:

- the address of the NotificationConsumer,
- the kinds of notification for the Subscription,
- and other related Subscription information.

In response to this message, the NotificationProducer creates a Subscription resource and returns an EndpointReference [WS-Addressing] to this Subscription.
At some later time, the NotificationProducer issues a Notification that matches the Subscription. The NotificationProducer uses the Notify message to deliver this to the NotificationConsumer.

The following diagram explains the usage scenario when the NotificationProducer has been substituted by NotificationBroker.

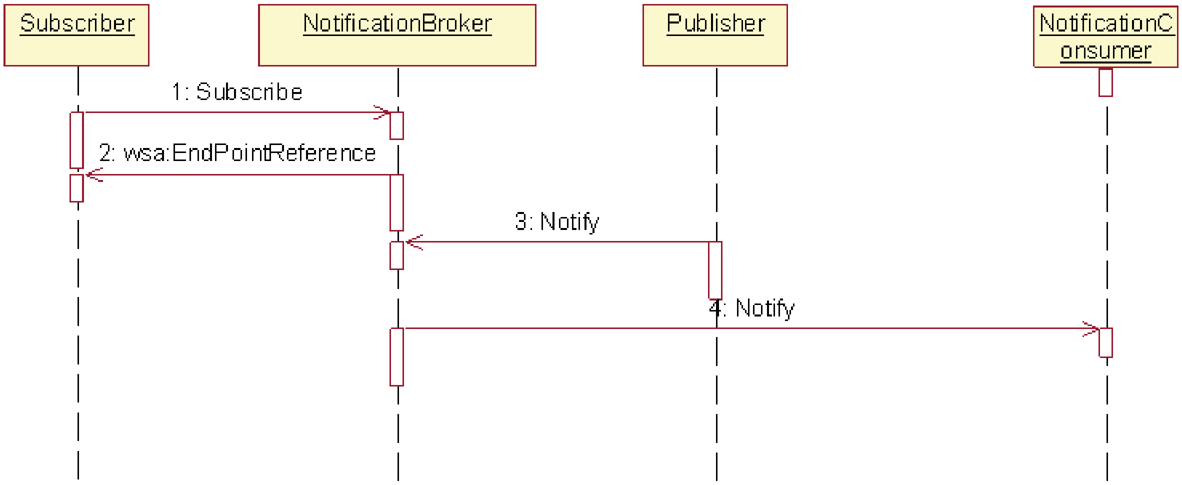


Figure 7-18 - Sequence diagram for Publish/Subscribe functionality (with Broker)

In the brokered case, the sequence of message exchanges between Subscriber and the NotificationBroker is the same as the sequence of message exchanges between Subscriber and NotificationProducer in the non-brokered case.

Instead of interacting directly with the ultimate NotificationConsumers, Publishers interact with the NotificationBroker using a sequence of message exchanges supported by the NotificationBroker. The Publisher publishes a NotificationMessage to the NotificationBroker, using the Notify message. At some point subsequent to this publication, the NotificationBroker delivers the NotificationMessages to any NotificationConsumer identified by Subscriptions which match the publication.

7.5.1 Topics and Events

The Topics mechanism provides a convenient means by which Subscribers can reason about Notifications of interest. Topics appear in several places within the WS-Notification system.

The NotificationProducer uses these sets of Topics as part of the matching process: a Notification is delivered to a NotificationConsumer if the set of Topics associated with the Subscription has a non-empty intersection with the set of Topics associated with the Notification. It is important to understand the distinction between a “Topic Namespace” and the “Topic Set” supported by a NotificationProducer.

7.5.2 Topic Namespace

A **Topic Namespace** is just an abstract set of Topic definitions.

The OASIS WS-Topics standard defines the TopicNamespace type as a mean to group and describe topics that belong to a specific namespace. An example of topic namespace is defined through the following example.

```
<wstop:TopicNamespace name="Topic-Namespace"
targetNamespace="http://www.opengis.net/oseo/1.0" final="true">
  <wstop:Topic name="OrderEvent">
    <wstop:Topic name="OrderFailure"/>
    <wstop:Topic name="OrderCompletion"/>
    <wstop:Topic name="OrderDeletion"/>
  </wstop:Topic>
 </wstop:TopicNamespace>
```

Within the example the Topic Namespace defines 4 Topics – the root Topics and their children.

A Topic in the Topic Namespace can be marked using the ‘final’ attribute true or false.

- If ‘final=true”, then no further child Topics can be added dynamically to that Topic.
- If Topic is not marked with the ‘final’ attribute with value=”true”, then a NotificationProducer could add further child Topics to that Topic within its Topic Set, and permit Subscriptions to such child Topics. By default Topic has been marked with final=”false”.

If a NotificationProducer wishes to use only a subset of the Topics previously defined (e.g. two of these Topics) it will define a TopicSet. The following is an example of TopicSet document.

```
<wstop:TopicSet xmlns:wstop="http://docs.oasis-open.org/wsn/t-1"
xmlns:tns="http://example.org/topics/example1"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://docs.oasis-open.org/wsn/t-1
http://docs.oasis-open.org/wsn/t-1.xsd">
   <tns:OrderEvent wstop:topic="true">
            <OrderCompletion wstop:topic="true"/>
   </tns:OrderEvent>
</wstop:TopicSet>
```

7.5.2.1 EventCode

The EventCode type is a list of codes signifying events that happen and they have to be identified by a GML dictionary like this EventCode.xml:

```
<?xml version="1.0" encoding="UTF-8"?>
<gml:Dictionary                   xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.opengis.net/gml/3.2
http://schemas.opengis.net/gml/3.2.1/gml.xsd" gml:id="EventCode">
    <gml:identifier
codeSpace="http://www.opengis.net/oseo/1.0">SWES_Event_CodeList</gml:identif
ier>
    <gml:name>EventCode</gml:name>
    <gml:dictionaryEntry>
        <gml:Definition gml:id="ORDERING_COMPLETED">
            <gml:description>The ordering is completed</gml:description>
            <gml:identifier
codeSpace="http://www.opengis.net/oseo/1.0">ORDERING_COMPLETED</gml:identifi
er>
            <gml:name>ORDERING_COMPLETED</gml:name>
        </gml:Definition>
    </gml:dictionaryEntry>
    <gml:dictionaryEntry>
        <gml:Definition gml:id="ORDERING_DELETED">
            <gml:description>The ordering has been deleted</gml:description>
            <gml:identifier
codeSpace="http://www.opengis.net/oseo/1.0">ORDERING_DELETED</gml:identifier
>
            <gml:name>ORDERING_DELETED</gml:name>
        </gml:Definition>
    </gml:dictionaryEntry>
     <gml:dictionaryEntry>
        <gml:Definition gml:id="ORDERING_FAILED">
            <gml:description>The ordering is failed</gml:description>
            <gml:identifier
codeSpace="http://www.opengis.net/oseo/1.0">ORDERING_FAILED</gml:identifier>
            <gml:name>ORDERING_FAILED</gml:name>
        </gml:Definition>
    </gml:dictionaryEntry>
</gml:Dictionary>
```

The following table presents the association between events and topics.

Event Name	Definition	Topic
Ordering_Completed “ORDERING_COMPLETED”	Order has been completed	OrderEvent/OrderCompletion
Ordering_Deleted “ORDERING_DELETED”	Order has been deleted	OrderEvent/OrderDeletion
Ordering_Failed “ORDERING_FAILED”	Failure of order	OrderEvent/OrderFailure

Table 7-30 – Events and Topics association

When matching process between an event and a Topic on which there is a subscribe gives a positive result, the notify will send to the NotificationConsumer.

8 GetCapabilities operation

8.1 Introduction

The GetCapabilities operation allows clients to retrieve service metadata from a server. The response to a GetCapabilities request shall be an XML document containing service metadata about the server, including specific information about an Order Service.
This section specifies the XML document that an Order Service server must return to describe its capabilities.

8.2 Requirements

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 1	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetCapabilities	Core	The Order Server shall implement the GetCapabilities operation.
Req 2	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetCapabilities/service	Core	The GetCapabilities request shall contain the **service** element set with “OS”.
Req 3	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetCapabilities/req	Core	The GetCapabilities request shall consist of an XML instance document as validated by the entity GetCapabilities in the oseo.xsd XML Schema.
Req 4	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetCapabilities/resp	Core	The response to a successful GetCapabilities request shall consist of an XML instance document as validated by the entity Capabilities in the oseo.xsd XML Schema.
Req 5	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetCapabilities/metadata	Core	The response to a successful GetCapabilities request shall contain all information about the operations supported by the Order Server.
Req 6	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetCapabilities/order_status	Core	The Capabilities document returned by an Order Server shall have: □ Capabilities/Contents/GetStatusCapabilities/@orderSearch = true □ Capabilities/Contents/GetStatusCapabilities/@orderRetrieve = true □ Capabilities/Contents/GetStatusCapabilities/@full = true
Req 7	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetCapabilities/encoding	Core	A compliant Order Server shall support the following SWE encoding: XMLEncoding Then the returned Capabilities document shall have: □ Capabilities/Contents/ContentsType supportedEncoding = XMLEncoding
Req 8	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetCapabilities/exception	Core	When a Order Server encounters an error while performing a GetCapabilities operation, it shall return an ows:ExceptionReport according to the clause 8 of [NR9].

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 9	http://www.opengis.net/spec/OSEO/1.0/req/ProductOrder/GetCapabilities/product_order	ProductOrder	The Order Server shall return a Capabilities document including: □ Either Capabilities/Contents/ProductOrders/@supported = true □ Or at least one element Capabilities/Contents/SupportedCollection/ProductOrders/@supported = true
Req 10	http://www.opengis.net/spec/OSEO/1.0/req/SubscriptionOrder/GetCapabilities/subscription	SubscriptionOrder	The Order Server shall return a Capabilities document including: □ Either Capabilities/Contents/SubscriptionOrders/@supported = true □ Or at least one element Capabilities/Contents/SupportedCollection/SubscriptionOrders/@supported = true
Req 11	http://www.opengis.net/spec/OSEO/1.0/req/TaskingOrder/GetCapabilities/tasking	TaskingOrder	The Order Server shall return a Capabilities document including: Capabilities/Contents/ProgrammingOrders/@supported = true and Capabilities/Contents/ProgrammingOrders/@SPS_URL set.
Req 12	http://www.opengis.net/spec/OSEO/1.0/req/AsyncSubmit/GetCapabilities/async_submit	AsyncSubmit	The Order Server shall return a Capabilities document including: Capabilities/Contents/SubmitCapabilities/@asynchronous = true
Req 13	http://www.opengis.net/spec/OSEO/1.0/req/Quotation/GetCapabilities/quotation	Quotation	The Order Server shall return a Capabilities document including: □ Capabilities/Contents/GetQuotationCapabilities/@supported = true and at least one of: □ Capabilities/Contents/GetQuotationCapabilities/@synchronous □ Capabilities/Contents/GetQuotationCapabilities/@asynchronous □ Capabilities/Contents/GetQuotationCapabilities/@monitoring □ Capabilities/Contents/GetQuotationCapabilities/@off-line set to true.
Req 14	http://www.opengis.net/spec/OSEO/1.0/req/QuotationSync/GetCapabilities/quotation	QuotationSync	The Order Server shall return a Capabilities document including: □ Capabilities/Contents/GetQuotationCapabilities/@supported = true □ Capabilities/Contents/GetQuotationCapabilities/@synchronous = true

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 15	http://www.opengis.net/spec/OSEO/1.0/req/QuotationMonitoring/GetCapabilities/quotation	QuotationMonitoring	The Order Server shall return a Capabilities document including: □ Capabilities/Contents/GetQuotationCapabilities/@supported = true □ Capabilities/Contents/GetQuotationCapabilities/@monitoring = true
Req 16	http://www.opengis.net/spec/OSEO/1.0/req/OnlineDataAccess/GetCapabilities/oda	OnlineDataAccess	The Order Server shall return a Capabilities document including: □ Either Capabilities/Contents/DescribeResultAccessCapabilities/@supported = true □ Or at least one element Capabilities/Contents/SupportedCollection/DescribeResultAccessCapabilities/@supported = true
Req 17	http://www.opengis.net/spec/OSEO/1.0/req/Cancellation/GetCapabilities/cancel	Cancellation	The Order Server shall return a Capabilities document including: □ Either Capabilities/Contents/CancelCapabilities/@supported = true □ Or at least one element Capabilities/Contents/SupportedCollection/CancelCapabilities/@supported = true
Req 18	http://www.opengis.net/spec/OSEO/1.0/req/Notification/GetCapabilities/notification	Notification	The Order Server shall return a Capabilities document including: □ Capabilities/Notifications non empty.

Table 8-1: GetCapabilities requirements.

8.3 GetCapabilities operation request

The following figure describes the GetCapabilities operation using an XML diagram.

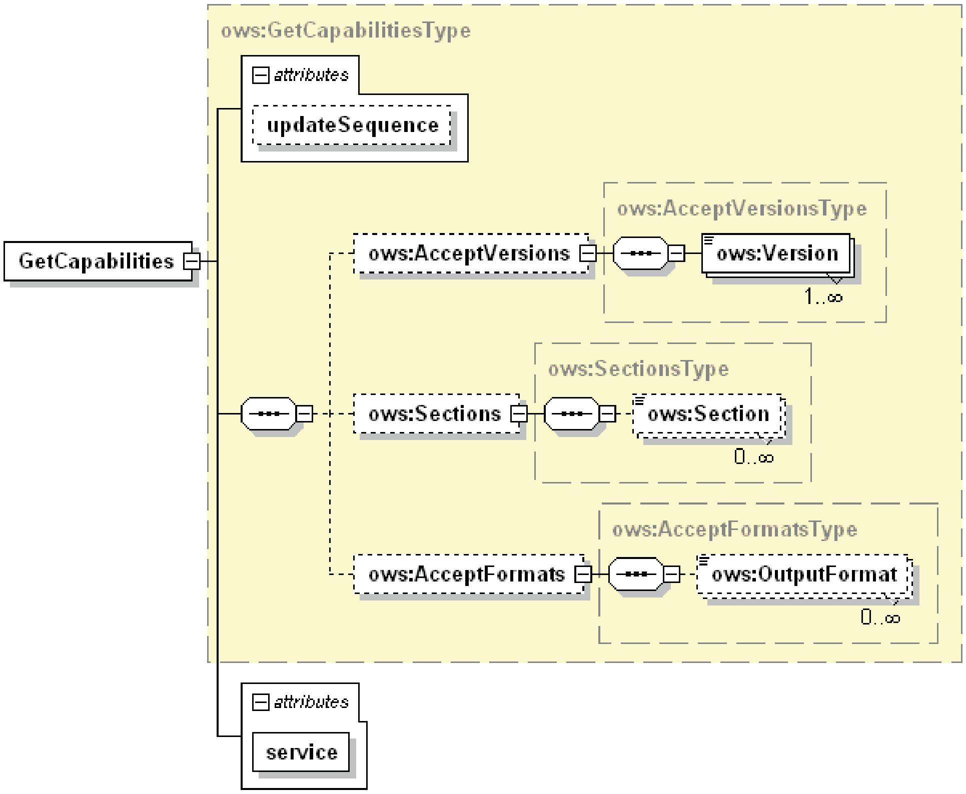


Figure 8-1 - GetCapabilities request diagram.

The GetCapabilities operation request shall be as specified in Subclauses 7.2 and 7.3 of [NR9]. The value of the “service” parameter shall be “OS”. The allowed set of service metadata (or Capabilities) XML document section names and meanings shall be as specified in Tables 3 and 7 of [NR9].

The “Multiplicity and use” column in Table 1 of [NR9] specifies the optionality of each listed parameter in the GetCapabilities operation request. The following table specifies the implementation of those parameters by Order Service clients and servers.

8.3.1 GetCapabilities request parameters

The following table describes the parameters for a GetCapability request.

Name	Multiplicity	Client implementation	Server implementation
service	One (mandatory)	The parameter shall be implemented by all clients, using the specified value	The parameter shall be implemented by all servers, checking that each parameter is received with specified value
AcceptVersions	Zero or one (optional)	Should be implemented by all software clients, using specified values	Shall be implemented by all servers, checking if parameter is received with specified value(s)
Sections	Zero or one (optional)	Each parameter may be implemented by each client If parameter not provided, shall expect default	Each parameter may be implemented by each server If parameter not implemented or not received, shall provide default
updateSequence	Zero or one (optional)		

Name	Multiplicity	Client implementation	Server implementation
AcceptFormats	Zero or one (optional)	response If parameter provided, shall allow default or specified response	response If parameter implemented and received, shall provide specified response

Table 8-2 - Parameters in GetCapabilities operation request

8.3.2 GetCapabilities request XML encoding

In the following an XML schema fragment for GetCapabilities request is provided.

```
<?xml version="1.0" encoding="UTF-8"?>
<!--  File                :     Oseo.xsd
      File Type     :     W3C Schema
      Abstract      :     OSEO Schema
      Uses          :
      Author        :     Daniele Marchionni
      -->
<xs:schema                                     xmlns="http://www.opengis.net/oseo/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:swes="http://www.opengis.net/swes/2.0"
xmlns:ows="http://www.opengis.net/ows/2.0"  xmlns:swe="http://www.opengis.net/swe/2.0"
xmlns:oseo="http://www.opengis.net/oseo/1.0"
targetNamespace="http://www.opengis.net/oseo/1.0"      elementFormDefault="qualified"
attributeFormDefault="unqualified">
      <xs:annotation>
            <xs:documentation>
Ordering Services For Earth Observation Products is an OGC Standard Copyright (c)
2010 Open Geospatial Consortium, Inc. All Rights Reserved.
To obtain additional rights of use, visit http://www.opengeospatial.org/legal/.
            </xs:documentation>
      </xs:annotation>
      <xs:import                            namespace="http://www.opengis.net/swe/2.0"
schemaLocation="http://schemas.opengis.net/sweCommon/2.0/simple_components.xsd"/>
      <xs:import                            namespace="http://www.opengis.net/ows/2.0"
schemaLocation="http://schemas.opengis.net/ows/2.0/owsGetCapabilities.xsd"/>
      <xs:import                           namespace="http://www.opengis.net/swes/2.0"
schemaLocation="http://schemas.opengis.net/swes/2.0/swes.xsd"/>
      <!-- ====================================================================== -->
      <!--       =======================       Operations       root       element
=============================-->
      <!-- ====================================================================== -->
      <xs:element name="GetCapabilities">
            <xs:annotation>
                  <xs:documentation>Request  to  a  Order  Service  to  perform  the
GetCapabilities  operation.  This  operation  allows  a  client  to  retrieve  service
metadata (capabilities XML) providing metadata for the specific Order server. In this
XML  encoding,  no  "request"  parameter  is  included,  since  the  element  name  specifies
the specific operation. </xs:documentation>
```

```
        </xs:annotation>
        <xs:complexType>
            <xs:complexContent>
                <xs:extension base="ows:GetCapabilitiesType">
                    <xs:attribute name="service" type="ows:ServiceType"
use="required" fixed="OS"/>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
    <xs:element name="Capabilities">
        <xs:annotation>
            <xs:documentation>XML encoded Order Service GetCapabilities
operation response. This document provides clients with service metadata about a
specific service instance. If the server does not implement the updateSequence
parameter, the server shall always return the complete Capabilities document, without
the updateSequence parameter. When the server implements the updateSequence parameter
and the GetCapabilities operation request included the updateSequence parameter with
the current value, the server shall return this element with only the "version" and
"updateSequence" attributes. Otherwise, all optional elements shall be included or
not depending on the actual value of the Sections parameter in the GetCapabilities
operation request. </xs:documentation>
        </xs:annotation>
        <xs:complexType>
            <xs:complexContent>
                <xs:extension base="ows:CapabilitiesBaseType">
                    <xs:sequence>
                        <xs:element name="Contents"
type="OrderingServiceContentsType" minOccurs="0"/>
                        <xs:element name="Notifications"
type="swes:NotificationProducerMetadataPropertyType" minOccurs="0"/>
                    </xs:sequence>
                </xs:extension>
            </xs:complexContent>
        </xs:complexType>
    </xs:element>
…
</xs:schema>
```

8.3.3 GetCapabilities request example

The following is an example of GetCapabilities request.

Note that the example includes only the GetCapabilities element without the SOAP envelope.

```
<GetCapabilities xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd"
xmlns:m0="http://www.opengis.net/ows/2.0" updateSequence="" service="OS" >
    <m0:AcceptVersions>
        <m0:Version>1.0.0</m0:Version>
    </m0:AcceptVersions>
</GetCapabilities>
```

8.4 GetCapabilities operation response

The following figure provides a graphical representation of the Capabilities XML document.

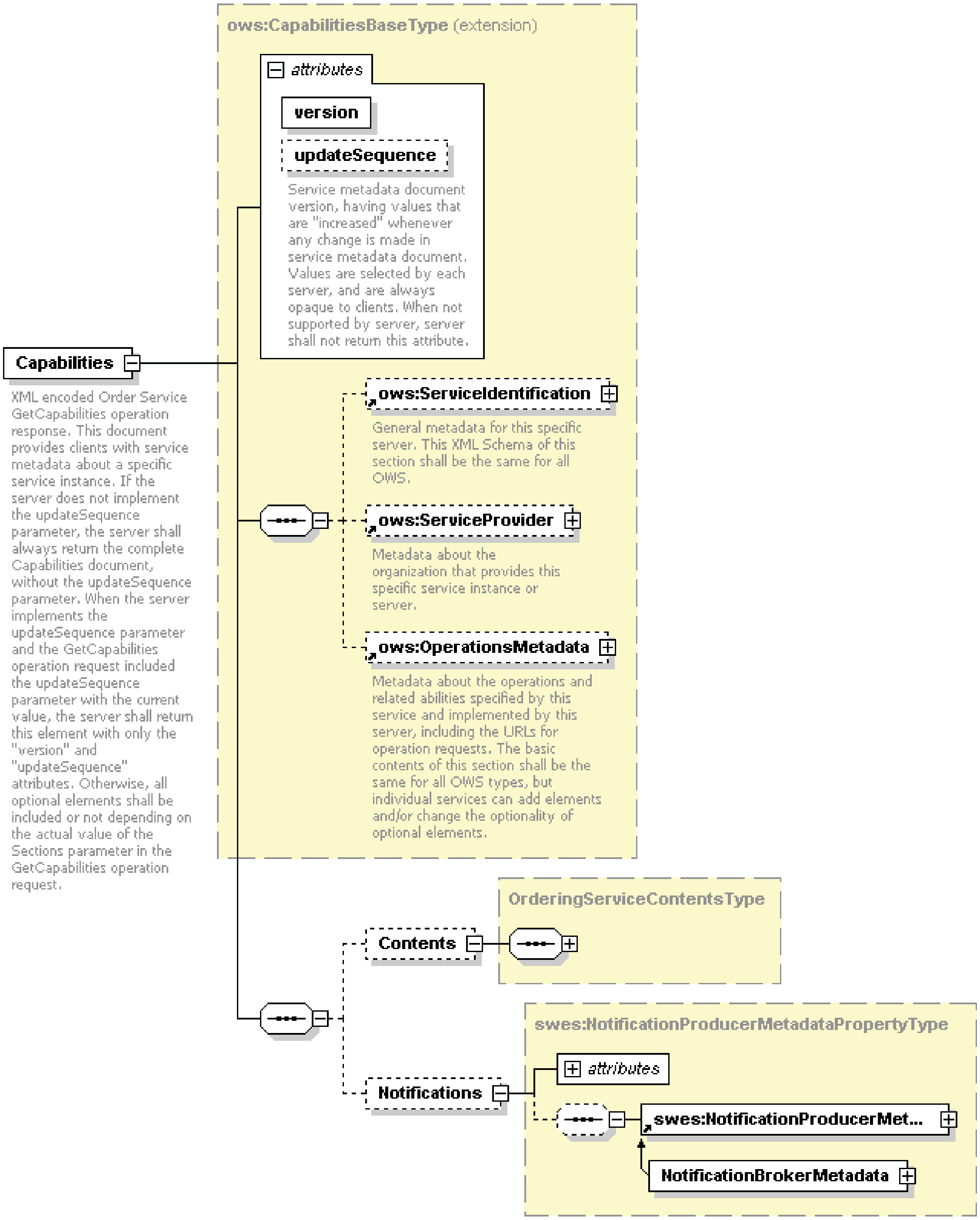

ows:CapabilitiesBaseType (extension)
attributes
version
updateSequence
Service metadata document version, having values that are "increased" whenever any change is made in service metadata document. Values are selected by each server, and are always opaque to clients. When not supported by server, server shall not return this attribute.
Capabilities
XML encoded Order Service GetCapabilities operation response. This document provides clients with service metadata about a specific service instance. If the server does not implement the updateSequence parameter, the server shall always return the complete Capabilities document, without the updateSequence parameter. When the server implements the updateSequence parameter and the GetCapabilities operation request included the updateSequence parameter with the current value, the server shall return this element with only the "version" and "updateSequence" attributes. Otherwise, all optional elements shall be included or not depending on the actual value of the Sections parameter in the GetCapabilities operation request.
ows:ServiceIdentification
General metadata for this specific server. This XML Schema of this section shall be the same for all OWS.
ows:ServiceProvider
Metadata about the organization that provides this specific service instance or server.
ows:OperationsMetadata
Metadata about the operations and related abilities specified by this service and implemented by this server, including the URLs for operation requests. The basic contents of this section shall be the same for all OWS types, but individual services can add elements and/or change the optionality of optional elements.
OrderingServiceContentsType
Contents
swes:NotificationProducerMetadataPropertyType
attributes
Notifications
swes:NotificationProducerMet...
NotificationBrokerMetadata

Figure 8-2 - Capabilities diagram.

The GetCapabilities response shall contain the Order Service sections specified in the following table. Depending on the values in the Sections parameter of the GetCapabilities operation request, any combination of these sections can be requested and shall be returned when requested.

Section name	Contents
ServiceIdentification	Metadata about this specific server. The schema of this section shall be the same as for all OWSs, as specified in Subclause 7.4.3 and owsServiceIdentification.xsd of [NR9].
ServiceProvider	Metadata about the organization operating this server. The schema of this section shall be the same for all OWSs, as specified in Subclause 7.4.4 and owsServiceProvider.xsd of [NR9].
OperationsMetadata	Metadata about the operations specified by this service and implemented by this server, including the URLs for operation requests. The basic contents and organization of this section shall be the same as for all OWSs, as specified in Subclause 7.4.5 and owsOperationsMetadata.xsd of [NR9].

Table 8-3 - Section name values and contents.

In addition to these sections, each service metadata document shall include the mandatory “version” and optional updateSequence parameters specified in Table 6 in Subclause 7.4.1 of [NR9].

8.4.1 OperationsMetadata section standard contents

For the Order Service, the OperationsMetadata section shall be the same as for all OGC Web Services, as specified in Subclause 7.4.5 and owsOperationsMetadata.xsd of [NR9]. The mandatory values of various (XML) attributes shall be as specified in Table 8-4. Similarly, the optional attribute values listed in Table 8-5 shall be included or not depending on whether that operation is implemented by that server. In these tables the “Attribute name” column uses dot-separator notation to identify parts of a parent item. The “Attribute value” column references an operation parameter, in this case an operation name, and the meaning of including that value is listed in the right column.

Attribute name	Attribute value	Meaning of attribute value
Operation.name	GetCapabilities	Implemented by the server
Operation.name	GetOptions	Implemented by the server
Operation.name	Submit	Implemented by the server
Operation.name	GetStatus	Implemented by the server

Table 8-4 - Required values of OperationsMetadata section attributes

Attribute name	Attribute value	Meaning of attribute value
Operation.name	GetQuotation	Implemented by the server
Operation.name	Cancel	Implemented by the server
Operation.name	DescribeResultAccess	Implemented by the server

Table 8-5 - Optional values of OperationsMetadata section.

8.4.2 Notifications section contents

To configure the WS-Notification interface, the Ordering Service needs to use the Notifications entity.

Names	Definition	Data type and value	Multiplicity and use
Notifications	metadata about the supported notification functionality	NotificationProducerMetadata (see Table 8-7)	Zero or one (optional) inclusion depends on the values in the Sections parameter of the GetCapabilities operation request

Table 8-6 - Notifications section.

Names	Definition	Data type and value	Multiplicity and use
producerEndpoint	endpoint of the web service implementing the NotificationProducer interface defined by WS-BaseNotification.	EndpointReference see WS-Addressing specification	One (mandatory)
supportedDialects	the filter dialects (used in WS-Notification Subscribe requests) supported by the service	FilterDialectMetadata see clause 8.2.2 in [OR10]	One (mandatory)
fixedTopicSet	indicates if the set of served topics is static throughout the lifetime of the service instance	Boolean true if the topic set is static, else false	One (mandatory)
servedTopics	collection of topics supported by the service may change if the topic set is not fixed	TopicSet See WS-Topics specification	One (mandatory)
usedTopicNamespace	definition of a topic namespace used in the topic set of the service	TopicNamespace see WS-Topics specification	Zero or more (optional) required if the topic set contains one or more topics

Table 8-7 - NotificationProducerMetadata section.

Names	Definition	Data type and value	Multiplicity and use

Names	Definition	Data type and value	Multiplicity and use
producerEndpoint	Same as Table 8-7	Same as Table 8-7	Same as Table 8-7
supportedDialects	Same as Table 8-7	Same as Table 8-7	Same as Table 8-7
fixedTopicSet	Same as Table 8-7	Same as Table 8-7	Same as Table 8-7
servedTopics	Same as Table 8-7	Same as Table 8-7	Same as Table 8-7
usedTopicNamespace	Same as Table 8-7	Same as Table 8-7	Same as Table 8-7
requiresRegistration	defines if a new publisher needs to be registered at the broker before it is allowed to send notifications	Boolean true if registration of new publishers is required, else false	One (mandatory)

Table 8-8 – NotificationBrokerMetadata section.

For more detailed information see §8 of [OR10].

8.4.3 Contents section contents

This section provides additional information about the order service.

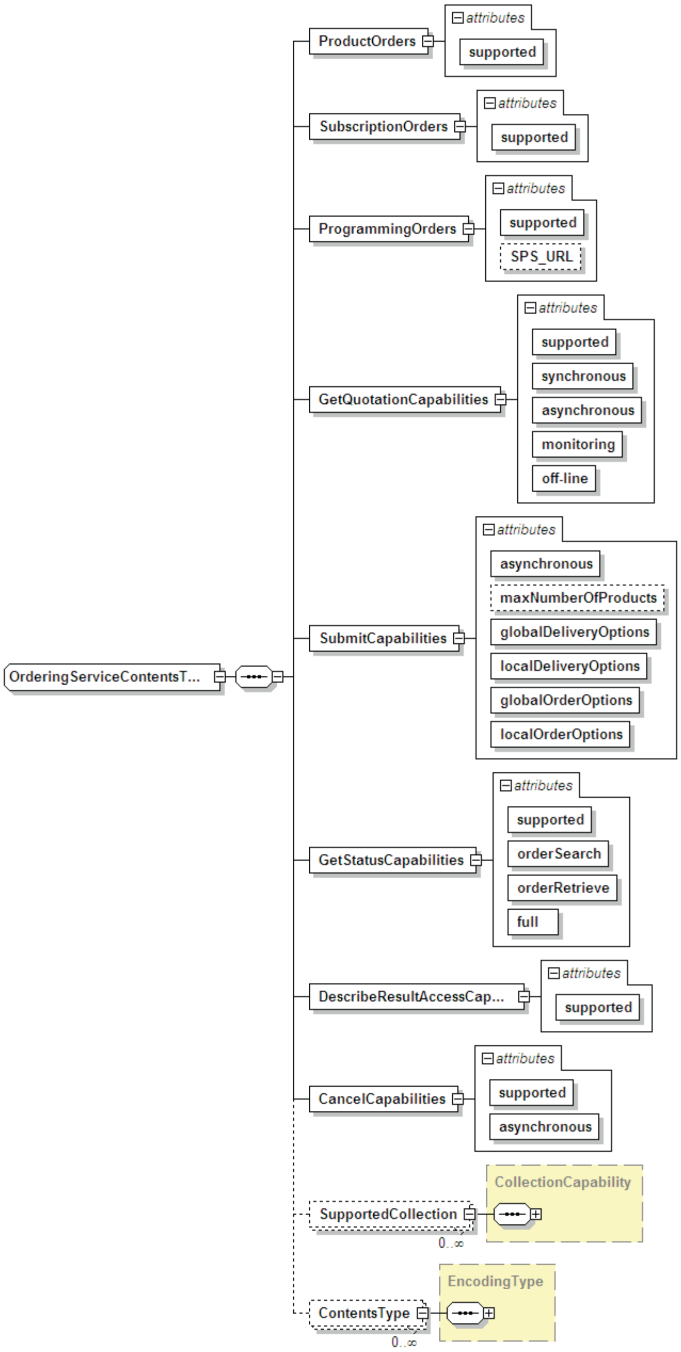


Figure 8-3 - OrderingServiceContentsType diagram.

Names	Definition	Data type and value	Multiplicity and use	Product Ordering	Subscription
ProductOrders	This element specifies whether product orders are supported by the Order Server. This is applicable to all collections supported by this server unless overridden in Table 8-10.	Type: ProductOrders	One (mandatory)	X	X
supported	Set to true whether the products orders are supported.	Type: xs:boolean	One (mandatory)	X	X
SubscriptionOrders	This element specifies whether subscription orders are supported by the Order Server. This is applicable to all collections supported by this server unless overridden in Table 8-10.	Type: SubscriptionOrders	One (mandatory)	X	X
supported	Set to true whether the subscription orders are supported.	Type: xs:boolean	One (mandatory)	X	X
ProgrammingOrdering	This element specifies whether the ordering service supports also future product ordering and, if it is the case, the URL of the SPS instance to use with it. This is applicable to all collections supported by this server unless overridden in Table 8-10.	Type: ProgrammingOrdering	One (mandatory)	X	
supported	Set to true when the Ordering Service supports future products ordering.	Type: xs:boolean	One (mandatory)	X	
SPS_URL	It is the SPS instance URL for submitting tasking requests for future products orders. This element shall be specified in case the “supported” flag is true.	Type: anyURI	One (optional)	X	

Names	Definition	Data type and value	Multiplicity and use	Product Ordering	Subscription
GetQuotationCapabilities	This element specifies how the quotation is supported by the Ordering service. In fact, depending on client and service capabilities different options are considered in this specification: – Quotation can be supported or not; – Quotation can be provided either synchronously or asynchronously; – Quotation can be provided via operation interaction; – In case of asynchronous quotation, the client can get it either via asynchronous notification or asking the quotation again until the service actually returns it. The different attributes of GetQuotationCapabilities element specify which of these possibilities are supported by the service. To be noted that, if GetQuotation is supported, then Submit operation must support order submission via quotationId.	Type: GetQuotationCapabilities	One (mandatory)	X	X
supported	Set to true whether the quotation is supported.	Type: xs:boolean	One (mandatory)	X	X
synchronous	Set to true if the service is able to provide synchronous answer to GetQuotation.	Type: xs:boolean	One (mandatory)	X	X
asynchronous	Set to true if the service is able to provide an asynchronous answer to GetQuotation.	Type: xs:boolean	One (mandatory)	X	X

Names	Definition	Data type and value	Multiplicity and use	Product Ordering	Subscription
monitoring	This attribute is set to true when the client has to call GetQuotation several times until the server provides the quotation. Then in this case the quotation process works in this way: – at the first call of GetQuotation the client specifies all order parameters; – the server replies with the quotationId; – then the client calls GetQuotation specifying the quotationId received at the previous call; – if the quotation is available, then the server returns it, otherwise the quotationId is returned again. – This process continues until the server returns the quotation.	Type: xs:boolean	One (mandatory)	X	X
off-line	Set to true if the service is able to provide answer to quotation via mail / e-mail notification.	Type: xs:boolean	One (mandatory)	X	X
SubmitCapabilities	This element specifies how the order submit is supported: – Asynchronous (the synchronous behaviour is mandatory). – Max number of orderable products – Local / global options	Type: SubmitCapabilities	One (mandatory)	X	X
asynchronous	Set to true if the service is able to provide an asynchronous answer to Submit.	Type: xs:boolean	One (mandatory)	X	X
maxNumberOfProducts	It specifies the max number of products that can be ordered in Submit operation. If not specified, then no limitation	Type: xs:integer	One (optional)	X	
globalDeliveryOptions	Set to true if the service is able to accept orders with delivery options shared between all order items.	Type: xs:boolean	One (mandatory)	X	X
localDeliveryOptions	Set to true if the service is able to accept orders with delivery options specified for each order item.	Type: xs:boolean	One (mandatory)	X	X
globalOrderOptions	Set to true if the service is able to accept orders with order options shared between all order items.	Type: xs:boolean	One (mandatory)	X	X
localOrderOptions	Set to true if the service is able to accept orders with order options specified for each order item.	Type: xs:boolean	One (mandatory)	X	X
GetStatusCapabilities	This element specifies if and how the order monitoring is supported.	Type: GetStatusCapabilities	One (mandatory)	X	X

Names	Definition	Data type and value	Multiplicity and use	Product Ordering	Subscription
orderSearch	Set to true if the service is able to search between the already submitted orders applying a query filter.	Type: xs:boolean	One (mandatory)	X	X
orderRetrieve	Set to true if the service is able to retrieve orders providing the order identifier.	Type: xs:boolean	One (mandatory)	X	X
full	Set to true if the service is able to return full information of the retrieved order.	Type: xs:boolean	One (mandatory)	X	X
DescribeResultAccessCapabilities	This element specifies if the DescribeResultAccess operation is supported or not. This is applicable to all collections supported by this server unless overridden in Table 8-10.	Type: DescribeResultAccess Capabilities	One (mandatory)	X	X
supported	Set to true whether the DescribeResultAccess is supported.	Type: xs:boolean	One (mandatory)	X	X
CancelCapabilities	This element specifies if the Cancel operation is supported or not. This is applicable to all collections supported by this server unless overridden in Table 8-10.	Type: CancelCapabilities	One (mandatory)	X	X
supported	Set to true whether the Cancel is supported.	Type: xs:boolean	One (mandatory)	X	X
asynchronous	Set to true if the service is able to provide an asynchronous answer to Cancel.	Type: xs:boolean	One (mandatory)	X	X
SupportedCollection	This element specifies the list of collections supported by this Ordering service. This list includes collections for products ordering and for subscriptions.	Type: CollectionCapability	Zero or more (optional)	X	X
ContentsType	Information about supported encoding rules for options values	Type: EncodingType	Zero or more (optional)	X	X
supportedEncoding	String to define the encoding rules.	Type: gml:CodeType Allowed values: XMLEncoding,TextEncoding	One (mandatory)	X	X

Table 8-9 - Description of Contents section of Capabilities document

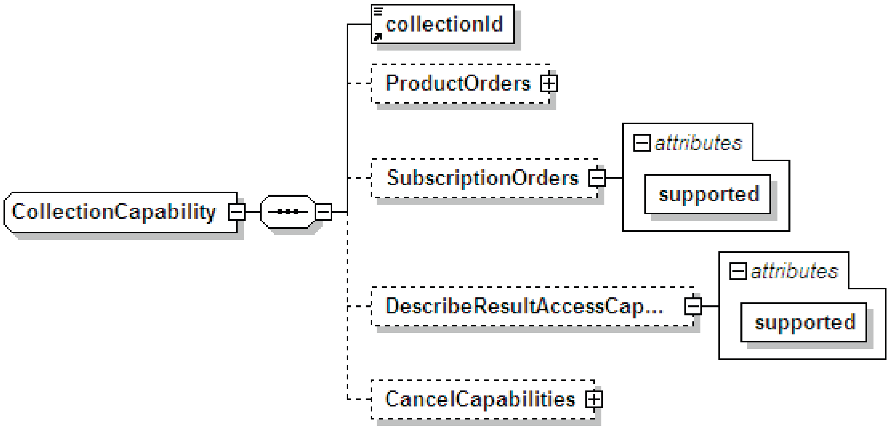


Figure 8-4 - CollectionCapability complex type diagram.

Names	Definition	Data type and value	Multiplicity and use	Product Ordering	Su
CollectionCapability	This element specifies the list of collections supported by this Ordering service. This list includes collections for products ordering and for subscriptions.	Type: CollectionCapability		X	
collectionId	Identifier for Collection Syntax: it shall be a valid URN compliant with OGC 06-131, which report the definition of the parentIdentifier, that is equivalent to the collectionId in this specification.	Type: non empty string (max 255 chars)	One (mandatory)	X	
ProductOrders	This element specifies whether product orders are supported **on this collection** by the Order Server. **This element overrides the same element defined at upper level.**	Type: ProductOrders	One (optional)	X	
supported	Set to true whether the products orders are supported.	Type: xs:boolean	One (mandatory)	X	

Names	Definition	Data type and value	Multiplicity and use	Product Ordering	Subscription
SubscriptionOrders	This element specifies whether subscription orders are supported **on this collection** by the Order Server. **This element overrides the same element defined at upper level.**	Type: SubscriptionOrders	One (optional)	X	X
supported	Set to true whether the subscription orders are supported.	Type: xs:boolean	One (mandatory)	X	X
DescribeResultAccessCapabilities	This element specifies if the DescribeResultAccess operation is supported or not **on this collection**. **This element overrides the same element defined at upper level.**	Type: DescribeResultAccessCapabilities	One (optional)	X	X
supported	Set to true whether the DescribeResultAccess is supported.	Type: xs:boolean	One (mandatory)	X	X
CancelCapabilities	This element specifies if the Cancel operation is supported or not **on this collection**. **This element overrides the same element defined at upper level.**	Type: CancelCapabilities	One (optional)	X	X
supported	Set to true whether the cancellation is supported.	Type: xs:boolean	One (mandatory)	X	X
asynchronous	Set to true if the service is able to provide an asynchronous answer to Cancel.	Type: xs:boolean	One (mandatory)	X	X

Table 8-10 - CollectionCapability complex type description.

8.4.4 GetCapabilities response XML encoding

In the following an XML schema fragment for this operation encoded in XML is reported:

```
<xs:schema xmlns="http://www.opengis.net/oseo/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:swes="http://www.opengis.net/swes/2.0"
xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:sps="http://www.opengis.net/sps/2.0"
targetNamespace="http://www.opengis.net/oseo/1.0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
```

```
      <xs:import namespace="http://www.opengis.net/swe/2.0"
schemaLocation="../swe/sweCommon/2.0.0/swe.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsGetCapabilities.xsd"/>
      <xs:import namespace="http://www.opengis.net/sps/2.0"
schemaLocation="Order_spsCommon.xsd"/>
      <xs:import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="../gml/3.2.1/base/gml.xsd"/>
<xs:import namespace="http://www.opengis.net/swes/2.0"
schemaLocation="../swes/2.0.0/swes.xsd"/>

      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsExceptionReport.xsd"/>
…
             <xs:element name="Capabilities">
             <xs:annotation>
                    <xs:documentation>XML encoded Order Service GetCapabilities
operation response. This document provides clients with service metadata about a
specific service instance. If the server does not implement the updateSequence
parameter, the server shall always return the complete Capabilities document, without
the updateSequence parameter. When the server implements the updateSequence parameter
and the GetCapabilities operation request included the updateSequence parameter with
the current value, the server shall return this element with only the "version" and
"updateSequence" attributes. Otherwise, all optional elements shall be included or
not depending on the actual value of the Sections parameter in the GetCapabilities
operation request. </xs:documentation>
             </xs:annotation>
             <xs:complexType>
                    <xs:complexContent>
                           <xs:extension base="ows:CapabilitiesBaseType">
                    <xs:sequence>
<xs:element name="Notifications" type="swes:NotificationProducerMetadataPropertyType"
minOccurs="0"/>
                    </xs:sequence>
                     <xs:sequence>
      <xs:element name="Contents" type="OrderingServiceContentsType" minOccurs="0"/>
                    </xs:sequence>
                    </xs:extension>
                    </xs:complexContent>
             </xs:complexType>
      </xs:element>
….
</schema>
```

8.4.5 GetCapabilities response example

The following is an example of GetCapabilities response.

Note that the example includes only the Capabilities element without the SOAP envelope.

```
<Capabilities xmlns:ows="http://www.opengis.net/ows/2.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xlink="http://www.w3.org/1999/xlink" xmlns="http://www.opengis.net/oseo/1.0"
xmlns:swes="http://www.opengis.net/swes/1.0"
xmlns:gml="http://www.opengis.net/gml/3.2" xmlns:swe="http://www.opengis.net/swe/2.0"
xmlns:sps="http://www.opengis.net/sps/2.0" xmlns:wstop="http://docs.oasis-
open.org/wsn/t-1" xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd" version="1.0.0">
      <ows:ServiceProvider>
             <ows:ProviderName>ESA EECF</ows:ProviderName>
```

```
                <ows:ProviderSite/>
                <ows:ServiceContact>
                        <ows:IndividualName>John Smith</ows:IndividualName>
                        <ows:PositionName>EO Help Desk Operator</ows:PositionName>
                        <ows:ContactInfo>
                                <ows:Phone>
                                        <ows:Voice>+39 06 90 180 999</ows:Voice>
                                </ows:Phone>
                        </ows:ContactInfo>
                        <ows:Role codeSpace="http://www.xmlspy.com">String</ows:Role>
                </ows:ServiceContact>
        </ows:ServiceProvider>
        <ows:OperationsMetadata>
                <ows:Operation name="GetCapabilities">
                        <ows:DCP>
                                <ows:HTTP>
                                        <ows:Post xlink:href="http://earth.esa.int"/>
                                </ows:HTTP>
                        </ows:DCP>
                </ows:Operation>
                <ows:Operation name="GetOptions">
                        <ows:DCP>
                                <ows:HTTP>
                                        <ows:Post xlink:href="http://earth.esa.int"/>
                                </ows:HTTP>
                        </ows:DCP>
                </ows:Operation>
                <ows:Operation name="GetQuotation">
                        <ows:DCP>
                                <ows:HTTP>
                                        <ows:Post xlink:href="http://earth.esa.int"/>
                                </ows:HTTP>
                        </ows:DCP>
                </ows:Operation>
                <ows:Operation name="Submit">
                        <ows:DCP>
                                <ows:HTTP>
                                        <ows:Post xlink:href="http://earth.esa.int"/>
                                </ows:HTTP>
                        </ows:DCP>
                </ows:Operation>
                <ows:Operation name="GetStatus">
                        <ows:DCP>
                                <ows:HTTP>
                                        <ows:Post xlink:href="http://earth.esa.int"/>
                                </ows:HTTP>
                        </ows:DCP>
                </ows:Operation>
                <ows:Operation name="Cancel">
                        <ows:DCP>
                                <ows:HTTP>
                                        <ows:Post xlink:href="http://earth.esa.int"/>
                                </ows:HTTP>
                        </ows:DCP>
                </ows:Operation>
                <ows:Operation name="DescribeResultAccess">
                        <ows:DCP>
                                <ows:HTTP>
                                        <ows:Post xlink:href="http://earth.esa.int"/>
                                </ows:HTTP>
                        </ows:DCP>
                </ows:Operation>
```

```
      </ows:OperationsMetadata>
      <Contents>
             <ProductOrders supported="true"/>
             <SubscriptionOrders supported="true"/>
             <ProgrammingOrders supported="true"/>
             <GetQuotationCapabilities asynchronous="true" monitoring="false" off-
line="false" supported="true" synchronous="true"/>
             <SubmitCapabilities localDeliveryOptions="true" localOrderOptions="true"
globalOrderOptions="true" globalDeliveryOptions="true" asynchronous="true"/>
             <GetStatusCapabilities orderSearch="true" full="true"
orderRetrieve="true"/>
             <DescribeResultAccessCapabilities supported="true"/>
             <CancelCapabilities supported="false" asynchronous="false"/>
             <SupportedCollection>

      <collectionId>http://www.opengis.net/def/EOP/ESA/0/ESA.EECF.ENVISAT_ASA_IMx_xS<
/collectionId>
                   <DescribeResultAccessCapabilities supported="false"/>
             </SupportedCollection>
             <SupportedCollection>

      <collectionId>http://www.opengis.net/def/EOP/ESA/0/ESA.EECF.ENVISAT_ASA_IMx_xF<
/collectionId>
                   <DescribeResultAccessCapabilities supported="false"/>
             </SupportedCollection>           <ContentsType>
                   <supportedEncoding>XMLEncoding</supportedEncoding>
                   <supportedEncoding>TextEncoding</supportedEncoding>
             </ContentsType>
      </Contents>
      <Notifications>
      <swes:NotificationProducerMetadata>
             <swes:producerEndpoint xmlns:wstop="http://docs.oasis-open.org/wsn/t-1"
xmlns:wsa="http://www.w3.org/2005/08/addressing">
                   <wsa:EndpointReference>

      <wsa:Address>http://example.com/OrderingConsumer</wsa:Address>
                   </wsa:EndpointReference>
             </swes:producerEndpoint>
             <swes:supportedDialects>
                   <swes:FilterDialectMetadata>
                          <swes:topicExpressionDialect codeSpace="http://docs.oasis-
open.org/wsn/2004/06/TopicExpression/Simple"/>
                   </swes:FilterDialectMetadata>
             </swes:supportedDialects>
             <swes:fixedTopicSet>true</swes:fixedTopicSet>
      <swes:servedTopics >
      <wstop:TopicSet>
      <wstop:documentation>
          <wstop:TopicNamespace targetNamespace="http://www.opengis.net/oseo/1.0">
             <wstop:Topic name="OrderFailure" messageTypes="sps:StatusReport"/>
             <wstop:Topic name="OrderCompletion" messageTypes="sps:StatusReport"/>
           </wstop:TopicNamespace>
      </wstop:documentation>
      </wstop:TopicSet>
      </swes:servedTopics>
      </swes:NotificationProducerMetadata>
      </Notifications>
</Capabilities>
```

8.4.6 Exceptions

In the event that an Order Server encounters an error servicing a GetCapabilities, it shall return an exception within a SOAP Fault (in particular an ows:ExceptionReport according to the clause 8 of [NR9]).

The SOAP Fault Message (see §7.3.14) shall have one of the following exceptionCode value:

“exceptionCode” value	**Meaning of code**	**“locator” value**	**“ExceptionText” value**
MissingParameterValue	Operation request does not include a parameter value, and this server did not declare a default value for that parameter	Name of missing parameter	“Missing value for Parameter”
InvalidParameterValue	Operation request contains an invalid parameter value	Name of parameter with invalid value	“Invalid value for Parameter”
NoApplicableCode	No other exceptionCode specified by this service and server applies to this exception	None, omit “locator” parameter	“Code not applicable”
VersionNegotiationFailed	List of version in “AcceptedVersions” parameter value in GetCapabilities request did not include any version supported by the server	None, omit “locator” parameter	“Version is not supported”
ServiceNotSupported	The Service name defined within the GetCapabilities request is not supported by the server.	Attribute “service”	“Service is not supported”

Table 8-11 – Exception codes for GetCapabilities operation.

9 GetOptions Operation

9.1 Introduction

This operation allows getting the possible options for preparing an EO products order or subscribing to EO products or submitting an order from a tasking request. Depending on the input parameters, this operation works in different ways:

- **Options by product identifiers**

  Basic use case: order options are queried on product identifier basis.

- **Options by collection identifier**

  Due to the fact that the collection identifiers can refer either a product collection or a subscription, 2 cases have been identified:

  - Product collection: In this case the operation returns options for ordering products from that collection. All products shall have the same options, otherwise an exception is thrown.
  - Subscription: Subscription options returned.

- **Options by tasking identifier**

  When the tasking identifier is specified (i.e.ID), which can refer to either a feasibility study or task submitted to SPS instance linked to this Ordering service, the operation returns the options available for ordering products starting from the acquisitions corresponding to the provided ID.

  The ID can refer different type of requests:

  - Precisely identifier acquisitions, i.e. acquisitions are identified by their key attributes e.g.: orbit and start and stop equator crossing times OR UTC start and stop times etc.
  - Coverage request, i.e. acquisitions are specified via the definition of the area to cover and the related time frame.
  - Standing request, similar to the previous one, but the coverage is performed several times with a defined periodicity.

Returned options are structured in array of groups, each identified by the **productOrderOptionsId** and representing a valid combination of options (e.g. one group can specify the options for getting a level 1 product, another group specifies options for getting another level 1 product, another for level 2 product, etc.). Each group includes mainly the following attributes:

- Delivery options
- Payment options
- Scene selection options (not applicable for subscriptions)
- Extensible list of options e.g.: processing level, product format. This list can be product specific and then the identifier of the product can be optionally specified.

In case the client is an application provided with a user interface, then this operation is basically called for building the form where the user has to insert the values for finalizing the order. In this case, it is suggested to show the items in the same order as they are specified in the GetOptionsResponse message, and if there are several values for the same item, show the first item first.

For example, when showing the delivery options, the client has to show the first instance of

productDeliveryOptions/deliveryMethod

with the first instance of

productDeliveryOptions/packageMedium

9.2 Requirements

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 19	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetOptions	Core	The Order Server shall implement the GetOption operation.
Req 20	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetOptions/req	Core	The GetOption request shall consist of an XML instance document as validated by the entity GetOption in the oseo.xsd XML Schema.
Req 21	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetOptions/resp	Core	The response to a successful GetOption request shall consist of an XML instance document as validated by the entity GetOptionResponse in the oseo.xsd XML Schema.
Req 22	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetOptions/exception	Core	When an Ordering Server encounters an error while performing a GetOptions operation, it shall return an ows: ExceptionReport according to the clause 8 of [NR9].
Req 23	http://www.opengis.net/spec/OSEO/1.0/req/ProductOrder/GetOptions/req/id	ProductOrder	The Order Server shall accept a GetOptions by product identifier i.e. shall accept a GetOptions request having one or more instances of: GetOptions/identifier
Req 24	http://www.opengis.net/spec/OSEO/1.0/req/ProductOrder/GetOptions/resp/id	ProductOrder	The successful answer to a GetOptions by product identifier shall include, for each asked product identifier, at least one instance of: GetOptionsResponse/orderOptions/option
Req 25	http://www.opengis.net/spec/OSEO/1.0/req/ProductOrder/GetOptions/req/coll_id	ProductOrder	The Order Server shall accept a GetOptions by collection identifier i.e. shall accept a GetOptions request having one instance of: GetOptions/collectionId
Req 26	http://www.opengis.net/spec/OSEO/1.0/req/ProductOrder/GetOptions/resp/coll_id	ProductOrder	The successful answer to a GetOptions by collection identifier shall include at least one instance of: GetOptionsResponse/orderOptions
Req 27	http://www.opengis.net/spec/OSEO/1.0/req/ProductOrder/GetOptions/resp/prod_order	ProductOrder	The successful answer to a GetOptions shall contain at least one element: GetOptionsResponse/orderOptions/orderType = set to: PRODUCT_ORDER
Req 28	http://www.opengis.net/spec/OSEO/1.0/req/SceneSelection/GetOptions/scene	SceneSelection	The Order Server shall return for at least one collection or product at least one non empty element: GetOptionsResponse/orderOptions/sceneSelectionOption
Req 29	http://www.opengis.net/spec/OSEO/1.0/req/SubscriptionOrder/GetOptions/req/coll_id	SubscriptionOrder	The Order Server shall accept a GetOptions by collection identifier i.e. shall accept a GetOptions request having one instance of: GetOptions/collectionId

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 30	http://www.opengis.net/spec/OSEO/1.0/req/SubscriptionOrder/GetOptions/resp/coll_id	SubscriptionOrder	The successful answer to a GetOptions by collection identifier shall include at least one instance of: GetOptionsResponse/orderOptions
Req 31	http://www.opengis.net/spec/OSEO/1.0/req/SubscriptionOrder/GetOptions/resp/subscription	SubscriptionOrder	The successful answer to a GetOptions shall contain at least one element: GetOptionsResponse/orderOptions/orderType = set to: SUBSCRIPTION_ORDER
Req 32	http://www.opengis.net/spec/OSEO/1.0/req/TaskingOrder/GetOptions/req/task_id	TaskingOrder	The Order Server shall accept a GetOptions by tasking identifier i.e. shall accept a GetOptions request having one instance of: GetOptions/taskingRequestId
Req 33	http://www.opengis.net/spec/OSEO/1.0/req/TaskingOrder/GetOptions/resp/coll_id	TaskingOrder	The successful answer to a GetOptions by tasking identifier shall include at least one instance of: GetOptionsResponse/orderOptions
Req 34	http://www.opengis.net/spec/OSEO/1.0/req/TaskingOrder/GetOptions/resp/prod_order	TaskingOrder	The successful answer to a GetOptions shall contain at least one element: GetOptionsResponse/orderOptions/orderType = set to: TASKING_ORDER
Req 35	http://www.opengis.net/spec/OSEO/1.0/req/Quotation/GetOptions/resp/prod_order	Quotation	The Order Server shall return for at least one collection or product at least one instance of: GetOptionsResponse/orderOptions/paymentOptions
Req 36	http://www.opengis.net/spec/OSEO/1.0/req/OnlineDataAccess/GetOptions/resp	OnlineDataAccess	The Order Server shall return for at least one collection or product at least one instance of: GetOptionsResponse/orderOptions/productDeliveryOptions/onlineDataAccess/protocol
Req 37	http://www.opengis.net/spec/OSEO/1.0/req/OnlineDataDelivery/GetOptions/resp	OnlineDataDelivery	The Order Server shall return for at least one collection or product at least one instance of: GetOptionsResponse/orderOptions/productDeliveryOptions/onlineDataDelivery/protocol
Req 38	http://www.opengis.net/spec/OSEO/1.0/req/MediaDelivery/GetOptions/resp	MediaDelivery	The Order Server shall return for at least one collection or product at least one instance of: GetOptionsResponse/orderOptions/productDeliveryOptions/mediaDelivery/packageMedium

Table 9-1: GetOptions requirements.

9.3 GetOptions operation request

The type of GetOptions is OrderOptionsRequestType. The following figure provides a graphical representation of this type.

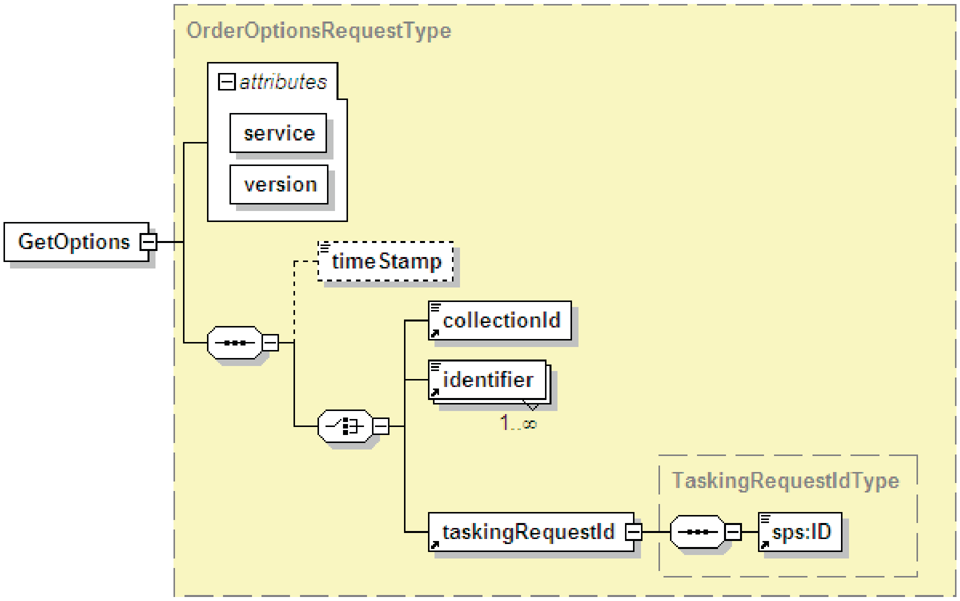


Figure 9-1 - GetOptions element diagram.

9.3.1 GetOptions request parameters

A request to perform GetOptions operation shall include the data structure specified in the previous figure and the following table.

Names	Definition	Data type and values	Multiplicity and use	Product Ordering	Subscriptio n
service	Service type identifier	Type: non-empty string Allowed values: OS	One (mandatory)	X	X
version	Specification version for operation	Type: non-empty string. Format: <x>.<y>.<z>	One (mandatory)	X	X
timeStamp	It is the time when the request has been issued.	Type: xs:DateTime	One (optional)	X	X

Names	Definition	Data type and values	Multiplicity and use	Product Ordering	Subscriptio n
collectionId	It is the identifier of the dataset collection to get ordering options. It is referenced as parentIdentifier in the [OR3]. This field is alternative to 'identifier' and 'taskingRequestId'	Type Value: non empty string (max 255 chars) Syntax: it shall be a valid URN compliant with OGC 06-131, which report the definition of the parentIdentifier, that is equivalent to the collectionId in this specification.	One (mandatory/ choice)	Identifier for the collection storing the EO product to order.	Identifier for the subscription.
identifier	Product identifier element. This element is alternative to the other ones. GetOptions returns the order options corresponding to all the identified products. If GetOptions is called without specifying the identifier but the options depend always on it, the operation returns SOAP Fault.	Type Value: string Permitted Values: Not empty string	One or more (mandatory/ choice)	X	
taskingRequest	This field is alternative to collectionId and identifier. This field has to be used in case of future product orders issued via both Ordering ICD and Programming ICD [OR9] [OR10].	Type: TaskingRequestIdType (see Table 7-24)	One (mandatory/ choice)	X	

Table 9-2 - GetOptions element description.

9.3.2 GetOptions request XML encoding

A XML schema fragment for this operation encoded in XML:

```
<xs:schema xmlns="http://www.opengis.net/oseo/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:sps="http://www.opengis.net/sps/2.0"
targetNamespace="http://www.opengis.net/oseo/1.0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
	<xs:import namespace="http://www.opengis.net/swe/2.0"
schemaLocation="../swe/sweCommon/0.0.0/swe.xsd"/>
```

```
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/1.0.0/owsGetCapabilities.xsd"/>
      <xs:import namespace="http://www.opengis.net/sps/2.0"
schemaLocation="Order_spsCommon.xsd"/>
      <xs:import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="../gml/3.2.1/base/gml.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsExceptionReport.xsd"/>
…
<xs:element name="GetOptions" type="OrderOptionsRequestType"/>
…
<xs:complexType name="OrderOptionsRequestType">
            <xs:complexContent>
                   <xs:extension base="OrderRequestBaseType">
                         <xs:sequence>
                         <xs:element name="timeStamp" type="xs:dateTime"
minOccurs="0"/>
                         <xs:choice>
                         <xs:element ref="collectionId"/>
                         <xs:element ref="identifier" maxOccurs="unbounded"/>
                         <xs:element ref="taskingRequestId"/>
                         </xs:choice>
                         </xs:sequence>
                   </xs:extension>
            </xs:complexContent>
      </xs:complexType>
</schema>
```

9.3.3 GetOptions request example

The following is an example of GetOptions request.

Note that the example includes only the GetOptions element without the SOAP envelope.

```
<GetOptions xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd" service="OS" version="1.0.0">
      <collectionId>http://www.opengis.net/def/EOP/ESA/0/ESA.EECF.ENVISAT_ASA_IMx_xS<
/collectionId>
</GetOptions>
```

9.4 GetOptions operation response

The following figure provides a graphical representation of this element.

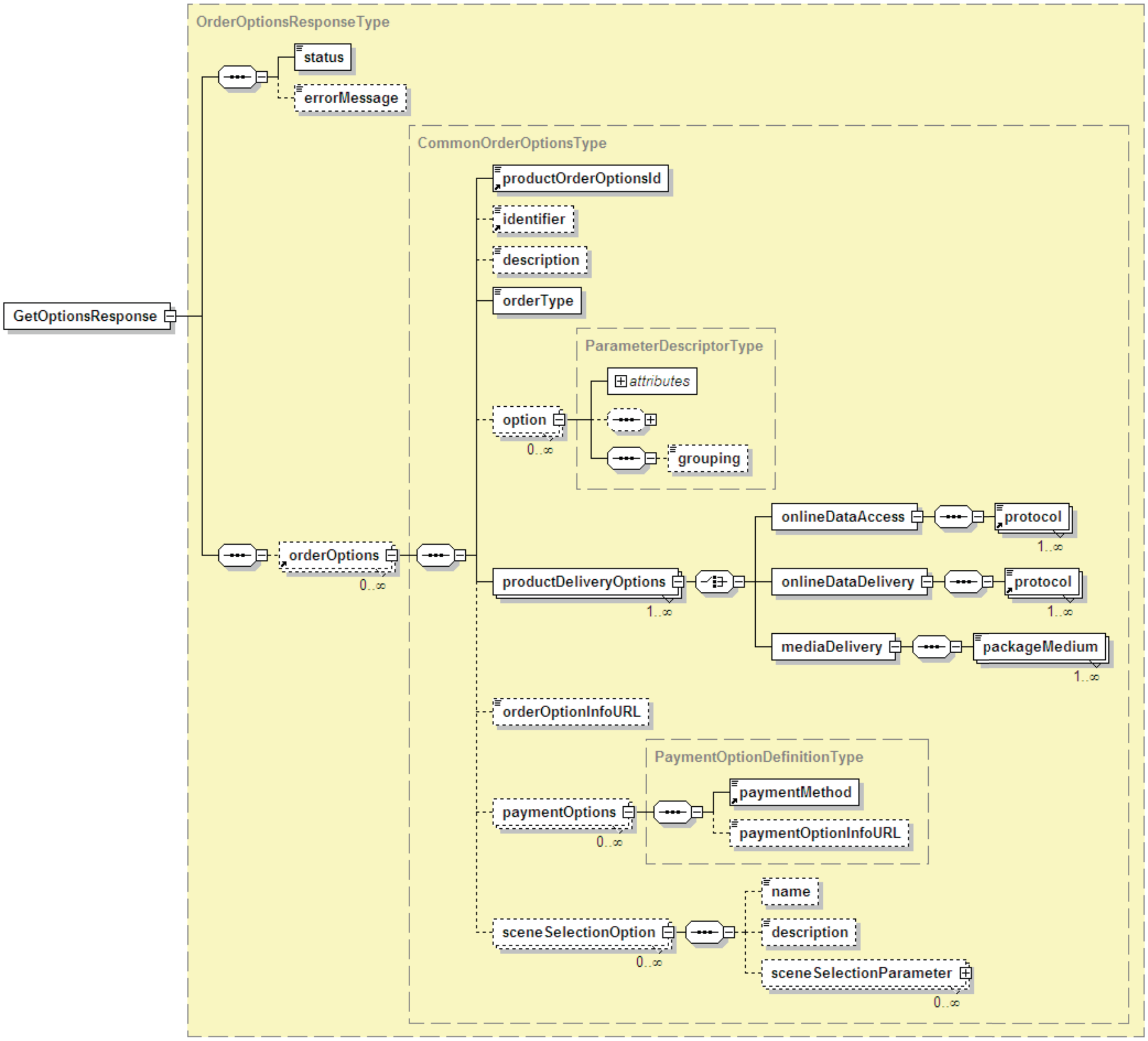


Figure 9-2 - GetOptionsResponse diagram

9.4.1 GetOptions response parameters

The following table reports the definition of the GetOptionsResponse element.

Names	Definition	Data type and values	Multiplicity and use
status	Completion result of the operation: – success: operation successfully executed; – partial: some error occurred during the processing of the request which lead to an incomplete response. In case of server error, which prevent also the provision of an incomplete answer, then SOAP fault is generated according (see §9.4.4)	Type**:** String Permitted Values**:** success, partial	One (mandatory)
errorMessage	Message to explain the reasons of the partial result.	Type**:** Not empty string (max 255 chars).	One (optional)
orderOptions	Order Options Set (at least one istance)	Type:CommonOrderOptionsType (see §7.3.2.1)	Zero or more (optional)

Table 9-3 – GetOptionsResponse description.

9.4.2 GetOptions response XML encoding

A XML schema fragment for this operation encoded in XML:

```
<xs:schema xmlns="http://www.opengis.net/oseo/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:sps="http://www.opengis.net/sps/2.0"
targetNamespace="http://www.opengis.net/oseo/1.0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
      <xs:import namespace="http://www.opengis.net/swe/2.0"
schemaLocation="../swe/sweCommon/2.0.0/swe.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsGetCapabilities.xsd"/>
      <xs:import namespace="http://www.opengis.net/sps/2.0"
schemaLocation="Order_spsCommon.xsd"/>
      <xs:import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="../gml/3.2.1/base/gml.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsExceptionReport.xsd"/>
…
      <xs:element name="GetOptionsResponse" type="OrderOptionsResponseType"/>
…
<xs:complexType name="OrderOptionsResponseType">
             <xs:complexContent>
                    <xs:extension base="OrderResponseBaseType">
                           <xs:sequence>
                                  <xs:element ref="orderOptions" minOccurs="0"
maxOccurs="unbounded">
                                         </xs:annotation>
                                  </xs:element>
                           </xs:sequence>
                    </xs:extension>
             </xs:complexContent>
      </xs:complexType>
<xs:complexType name="OrderResponseBaseType">
```

```
             <xs:annotation>
                   <xs:documentation>Base type for all Ordering Service operation
responses.</xs:documentation>
             </xs:annotation>
             <xs:sequence>
                   <xs:element name="status" type="OrderResponseStatusType"
nillable="false"/>
                   <xs:element name="errorMessage" minOccurs="0">
                          <xs:annotation>
                                <xs:documentation>This field is set when status
element is different from success. It provides some information about the occurred
problem.</xs:documentation>
                          </xs:annotation>
                          <xs:simpleType>
                                <xs:restriction base="xs:string">
                                       <xs:maxLength value="255"/>
                                </xs:restriction>
                          </xs:simpleType>
                   </xs:element>
             </xs:sequence>
</xs:complexType>
….
</schema>
```

9.4.3 GetOptions response example

The following is an example of GetOptions response.

Note that the example includes only the GetOptionsResponse element without the SOAP envelope.

```
<GetOptionsResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0" xmlns:sps="http://www.opengis.net/sps/2.0"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd">
      <status>success</status>
      <orderOptions>
             <productOrderOptionsId>Level 1,Product PRI
(ASA_IMP)</productOrderOptionsId>
             <description>Order Options for producing ASAR IMP 1P
product</description>
             <orderType>PRODUCT_ORDER</orderType>
             <option>
                   <swe:DataRecord>
                   <swe:field name="processingLevel">
                          <swe:Category updatable="false" optional="true"
definition="http://www.opengis.net/def/property/OGC-EO/0/options/processingLevel">
                                <gml:description>Product Level</gml:description>
                                <gml:name>processing Level</gml:name>
                                <swe:constraint>
                                       <swe:AllowedTokens>
                                              <swe:value>1B</swe:value>
                                       </swe:AllowedTokens>
                                </swe:constraint>
                          </swe:Category>
                          </swe:field>
                   </swe:DataRecord>
                   <grouping>Processing Option</grouping>
             </option>
             <option>
                   <swe:DataRecord>
                          <swe:field name="productType">
```

```
                                 <swe:Category updatable="false" optional="true" 
definition="http://www.opengis.net/def/property/OGC-EO/0/options/productType">
                                        <gml:description>Product 
type.</gml:description>
                                        <gml:name>Product Type</gml:name>
                                        <swe:constraint>
                                              <swe:AllowedTokens>

      <swe:value>ASA_IMP_1P</swe:value>
                                              </swe:AllowedTokens>
                                        </swe:constraint>
                                 </swe:Category>
                          </swe:field>
                    </swe:DataRecord>
                    <grouping>Processing Option</grouping>
             </option>
             <option>
                    <swe:DataRecord>
                          <swe:field name="qualityOfService">
                          <swe:Category updatable="false" optional="true" 
definition="http://www.opengis.net/def/property/OGC-EO/0/options/qualityOfService">
                          <gml:description>Quality of the Service</gml:description>
                                 <gml:name>qualityOfService</gml:name>
                                 <swe:constraint>
                                        <swe:AllowedTokens>
                                              <swe:value>STANDARD</swe:value>
                                              <swe:value>NRT</swe:value>
                                              <swe:value>RUSH</swe:value>
                                        </swe:AllowedTokens>
                                 </swe:constraint>
                          </swe:Category>
                          </swe:field>
                    </swe:DataRecord>
             <grouping>Processing Option</grouping>
             </option>
             <!-- ==================================================================
-->
             <!-- Example of complex option (suggested by Eumetsat AR#24) -->
             <!-- ==================================================================
-->
             <option>
             <swe:DataArray>
                    <swe:elementCount><swe:Count></swe:Count></swe:elementCount>
                    <swe:elementType name=”spectralBandColorComposition”>
                    <swe:DataRecord updatable="false" optional="true" 
definition="http://www.opengis.net/def/property/OGC-
EO/0/options/spectralBandColorComposition">
                          <gml:description>A user can pick out of 12 bands 3 bands. A 
color can be assigned to a band (R,G,B). This is to allow a user to form a composite 
colored image</gml:description>
                          <gml:name>spectralBandColorComposition</gml:name>
                          <swe:field name="RedBand">
                                 <swe:Category>
                                        <swe:constraint>
                                              <swe:AllowedTokens>
                                                     <swe:value>band1</swe:value>
                                                     <swe:value>band2</swe:value>
                                                     <swe:value>band3</swe:value>
                                                     <swe:value>band4</swe:value>
                                                     <swe:value>band5</swe:value>
                                                     <swe:value>band6</swe:value>
                                                     <swe:value>band7</swe:value>
                                                     <swe:value>band8</swe:value>
                                                     <swe:value>band9</swe:value>
                                                     <swe:value>band10</swe:value>
                                                     <swe:value>band11</swe:value>
                                                     <swe:value>band12</swe:value>
                                               </swe:AllowedTokens>
                                        </swe:constraint>
                                 </swe:Category>
                           </swe:field>
                           <swe:field name="GreenBand">
                                 <swe:Category>
                                        <swe:constraint>
                                               <swe:AllowedTokens>
                                                     <swe:value>band1</swe:value>
                                                     <swe:value>band2</swe:value>
                                                     <swe:value>band3</swe:value>
                                                     <swe:value>band4</swe:value>
                                                     <swe:value>band5</swe:value>
                                                     <swe:value>band6</swe:value>
                                                     <swe:value>band7</swe:value>
                                                     <swe:value>band8</swe:value>
                                                     <swe:value>band9</swe:value>
                                                     <swe:value>band10</swe:value>
                                                     <swe:value>band11</swe:value>
                                                     <swe:value>band12</swe:value>
                                               </swe:AllowedTokens>
                                        </swe:constraint>
                                 </swe:Category>
                           </swe:field>
                           <swe:field name="BlueBand">
                                 <swe:Category>
                                        <swe:constraint>
                                               <swe:AllowedTokens>
                                                     <swe:value>band1</swe:value>
                                                     <swe:value>band2</swe:value>
                                                     <swe:value>band3</swe:value>
                                                     <swe:value>band4</swe:value>
                                                     <swe:value>band5</swe:value>
                                                     <swe:value>band6</swe:value>
                                                     <swe:value>band7</swe:value>
                                                     <swe:value>band8</swe:value>
                                                     <swe:value>band9</swe:value>
                                                     <swe:value>band10</swe:value>
                                                     <swe:value>band11</swe:value>
                                                     <swe:value>band12</swe:value>
                                               </swe:AllowedTokens>
                                        </swe:constraint>
                                 </swe:Category>
                           </swe:field>
                    </swe:DataRecord>
                   </swe:elementType>
                   </swe:DataArray>
                    <grouping>Processing Option</grouping>
             </option>
             <productDeliveryOptions>
                    <mediaDelivery>
                           <packageMedium>CD-ROM</packageMedium>
                    </mediaDelivery>
             </productDeliveryOptions>
<orderOptionInfoURL>http://www.provider.com/envisat/orderoptions.html</orderOptionInf
oURL>
             <paymentOptions>
```

```
                  <paymentMethod>quota</paymentMethod>
            </paymentOptions>
            <sceneSelectionOption>
                  <name>Envisat ASAR Image Floating Scene (Time and coord)</name>
                  <description>Scene Selection Options for Envisat ASAR Image
Floating Scenes (by times and coordinates)</description>
                  <sceneSelectionParameter>
                  <swe:DataRecord>
                        <swe:field name="sceneType">
                              <swe:Category optional="false" updatable="false"
definition="http://www.opengis.net/def/property/OGC-EO/0/options/sceneType">
                              <gml:description>First and last column on the parent
product for scene extraction</gml:description>
                              <gml:name>Type of the scene</gml:name>
                              <swe:constraint>
                              <swe:AllowedTokens>
                                    <swe:value>FloatingScene</swe:value>
                                    </swe:AllowedTokens>
                              </swe:constraint>
                              </swe:Category>
                        </swe:field>
                  </swe:DataRecord>
                  </sceneSelectionParameter>
      <!-- sceneCentreCoordinates Parameter - with WRSInformation constraint -->
      <sceneSelectionParameter>
      <swe:Vector referenceFrame="http://www.opengis.net/def/crs/EPSG/7.1/4326">
      <gml:name>sceneCentreCoordinates</gml:name>
            <swe:coordinate name="Latitude">
            <swe:Quantity definition="http://www.opengis.net/def/property/OGC-
EO/0/options/Latitude" axisID="Lat">
                  <gml:name>Latitude</gml:name>
                  <swe:uom code="deg"/>
            </swe:Quantity>
            </swe:coordinate>
            <swe:coordinate name="Longitude">
            <swe:Quantity definition="http://www.opengis.net/def/property/OGC-
EO/0/options/Longitude" axisID="Long">
                  <gml:name>Longitude</gml:name>
                  <swe:uom code="deg"/>
            </swe:Quantity>
            </swe:coordinate>
      </swe:Vector>
      <sceneRestriction>
      <swe:DataArray optional="false" updatable="false"
definition="http://www.opengis.net/def/property/OGC-EO/0/options/WRSInformation">
            <gml:name>WRSInformation</gml:name>
            <swe:elementCount>
            <swe:Count>
                  <swe:value>2</swe:value>
            </swe:Count>
            </swe:elementCount>
            <swe:elementType name="Phase">
            <swe:DataRecord>
                  <swe:field name="FromOrbit">
                              <swe:Count></swe:Count>
                  </swe:field>
                  <swe:field name="ToOrbit">
                        <swe:Count></swe:Count>
                  </swe:field>
                  <swe:field name="TimePerFrame">
                        <swe:Quantity>
                              <swe:uom code="s"/>
```

```
                                </swe:Quantity>
                        </swe:field>
                        <swe:field name="TimeBetweenFrameCenter">
                                <swe:Quantity>
                                        <swe:uom code="s"/>
                                </swe:Quantity>
                        </swe:field>
                        <swe:field name="FrameNumberStep">
                                <swe:Count>
                                </swe:Count>
                        </swe:field>
                        <swe:field name="FirstFrameNumber">
                                <swe:Count>
                                </swe:Count>
                        </swe:field>
                        <swe:field name="FirstFrameCenterTime">
                                <swe:Quantity>
                                        <swe:uom code="s"/>
                                </swe:Quantity>
                        </swe:field>
                        <swe:field name="LastFrameNumber">
                                <swe:Count>
                                </swe:Count>
                        </swe:field>
                </swe:DataRecord>
                </swe:elementType>
                <swe:encoding>
                        <swe:TextEncoding blockSeparator="&#10;" tokenSeparator=","/>
                        </swe:encoding>
                <swe:values>
                        126,2104,15069.767,837.209,1,0,0.0000,7199
                        2354,3713,15069.767,837.209,1,0,0.0000,7199
                </swe:values>
        </swe:DataArray>
        </sceneRestriction>
</sceneSelectionParameter>
        </orderOptions>
```

9.4.4 Exceptions

In the event that an Order Server encounters an error servicing a GetOptions, it shall return an exception within a SOAP Fault (in particular an ows:ExceptionReport according to the clause 8 of [NR9]).

The SOAP Fault Message (see §7.3.14) shall have one of the following exceptionCode value:

“exceptionCode” value	**Meaning of code**	**“locator” value**	**“ExceptionText” value**
MissingParameterValue	Operation request does not include a parameter value, and this server did not declare a default value for that parameter	Name of missing parameter	“Missing value for Parameter”
InvalidParameterValue	Operation request contains an invalid parameter value	Name of parameter with invalid value	“Invalid value for Parameter”

"exceptionCode" value	**Meaning of code**	**"locator" value**	**"ExceptionText" value**
NoApplicableCode	No other exceptionCode specified by this service and server applies to this exception	None, omit "locator" parameter	"Code not applicable"
AuthenticationFailed	Invalid or missing identity information	"identity_token"	"Invalid or missing identity information"
AuthorizationFailed	The client is not authorized to call the operation or it is not authorized to provide the specified parameters.	Possible values: □ "GetOptions" □ "collectionId" □ "identifier" □ "taskingRequestId"	Text describing the item not authorized. E.g.: □ "The client is not authorized to call the operation."
UnsupportedCollection	Operation request contains a collection id which is neither a product collection nor a subscription.	"collectionId" parameter	"Subscription not supported"
UnsupportedProduct	Operation request contains a product identifier, but either the server does not support product ordering or the product is not in the catalogue.	"Identifier" parameter	"Ordering not supported"
UnsupportedTaskingRequest	Operation request contains a tasking identifier, but either the server does not support future product ordering or the task ID is not in the companion SPS Server.	"ID" parameter	"Programming not supported"
InvalidIdentifierValue	Operation request contains a type of identifier incompatible with to the type of the operation (e.g. a tasking identifier for a subscription)	Name of the identifier for each type of request	"Identifier not valid for this request"

Table 9-4 – Exception Codes for GetOptions operation.

10 GetQuotation Operation

10.1 Introduction

This operation allows getting the quotation of the order specified as input parameter.

Order quotation can be performed in different ways depending on client and server capabilities:

- **Synchronous quotation**

  The server is able to provide the quotation in real time and then it is returned in the acknowledge message of GetQuotation operation.

  Synchronous quotation is performed when:

  - Capabilities/Content/GetQuotationCapabilities/synchronous is set to true;

- **Asynchronous quotation with notification**

  The server supports asynchronous quotation and the client is able to work as a server for getting the notification carrying on the quotation.

  The asynchronous quotation with notification is performed:

  - Capabilities/Content/GetQuotationCapabilities/asynchronous is set to true;
  - <wsa:ReplyTo> of GetQuotation message is set with the address where the client is listening to the notification and <wsa:MessageID> shall include a unique identifier of the request.

  Then the quotation process is composed of the following interactions:

  - The client calls GetQuotation of the server specifying its address in <wsa:ReplyTo> element;
  - The server will call GetQuotationResponse operation of the client for sending the produced quotation.

- **Asynchronous quotation with monitoring**

  It is a variation of the previous one: the server support asynchronous quotation, but the client cannot work as a server. In this case the client will receive an id at the first call and then it has to call again GetQuotation providing the previously received id until the server will return the quotation in the acknowledge message of the operation.

  The mechanism is performed when:

  - Capabilities/Content/GetQuotationCapabilities/monitoring is set to true;
  - <wsa:ReplyTo> of GetQuotation message is set with the anonymous address:

    http://www.w3.org/2005/08/addressing/anonymous

- **Off-line quotation**

  In this case the quotation is sent to the user via fax / mail to the address specified in the **invoiceAddress** element of the order specification.

  Off line quotation is performed when:

  - Capabilities/Content/GetQuotationCapabilities/off-line is set to true;
  - <wsa:ReplyTo> is set with the anonymous address.
  - No other mechanisms are possible.

To be noted that, if the order server supports GetQuotation, then Submit must support order submission via quotation id.

10.2 Requirements

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 39	http://www.opengis.net/spec/OSEO/1.0/req/Quotation/GetQuotation	Quotation	The Order Server shall implement the GetQuotation operation.
Req 40	http://www.opengis.net/spec/OSEO/1.0/req/Quotation/GetQuotation/req	Quotation	The GetQuotation request shall consist of an XML instance document validated by the entity GetQuotation in the oseo.xsd XML Schema.
Req 41	http://www.opengis.net/spec/OSEO/1.0/req/Quotation/GetQuotation/req/order	Quotation	The Order Server shall accept GetQuotation request reporting the order specification (any supported order type): GetQuotation/orderSpecification
Req 42	http://www.opengis.net/spec/OSEO/1.0/req/Quotation/GetQuotation/resp	Quotation	The Order Server shall return, in answer to a succesfull GetQuotation request, a GetQuotationAck element that complies with the oseo.xsd XML Schema.
Req 43	http://www.opengis.net/spec/OSEO/1.0/req/Quotation/GetQuotation/exception	Quotation	When the Order Server encounters an error while performing a GetQuotation operation, it shall return an ows: ExceptionReport according to the clause 8 of [NR9].
Req 44	http://www.opengis.net/spec/OSEO/1.0/req/QuotationSync/GetQuotation	QuotationSync	The Order Server shall return, in answer to a succesfull GetQuotation request, a GetQuotationAck response message having: GetQuotationAck/quotation set with the order quotation.
Req 45	http://www.opengis.net/spec/OSEO/1.0/req/QuotationMonitoring/GetQuotation/req_initial	QuotationMonitoring	The Order Server shall accept a GetQuotation request including: □ the order specification: GetQuotation/orderSpecification □ Empty reply address: soapenv:Envelope/soapenv:Header/wsa:ReplyTo = http://www.w3.org/2005/08/addressing/anonymous. The Order Server shall return the quotation id of the received order: GetQuotationAck/quotationId
Req 46	http://www.opengis.net/spec/OSEO/1.0/req/QuotationMonitoring/GetQuotation/req_next	QuotationMonitoring	The Order Server shall accept a GetQuotation request including: □ the identifier of a previously asked quotation: GetQuotation/quotationId (the quotationId returned by the initial GetQuotation call). □ Empty reply address: soapenv:Envelope/soapenv:Header/wsa:ReplyTo = http://www.w3.org/2005/08/addressing/anonymous

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 47	http://www.opengis.net/spec/OSEO/1.0/req/QuotationMonitoring/GetQuotation/resp	QuotationMonitoring	The Order Server shall reply to a GetQuotation request, including the quotation identifier, returning either: □ the quotationId GetQuotationAck/quotationId if the quotation is not ready yet □ or the quotation GetQuotationAck/quotation when the quotation is available. A compliant server shall make the quotation available.
Req 48	http://www.opengis.net/spec/OSEO/1.0/req/QuotationAsync/GetQuotation/req	QuotationAsync	The Order Server shall accept a GetQuotation request including: □ The reply address: soapenv:Envelope/soapenv:Header/wsa:ReplyTo □ The order (of any supported order type): GetQuotation/orderSpecification
Req 49	http://www.opengis.net/spec/OSEO/1.0/req/QuotationAsync/GetQuotation/req_ack	QuotationAsync	The Order Server shall reply to a GetQuotation request returning the quotationId: GetQuotationAck/quotationId
Req 50	http://www.opengis.net/spec/OSEO/1.0/req/QuotationAsync/GetQuotation/resp	QuotationAsync	The Order Server shall send the asynchronous response to a successful GetQuotation request including an XML instance document validated by the entity GetQuotationResponse in the oseo.xsd XML Schema.
Req 51	http://www.opengis.net/spec/OSEO/1.0/req/QuotationAsync/GetQuotation/resp/quotation	QuotationAsync	The Order Server shall send the asynchronous response to a successful GetQuotation request including the order quotation: GetQuotationResponse/quotation
Req 52	http://www.opengis.net/spec/OSEO/1.0/req/QuotationAsync/GetQuotation/resp_ack	QuotationAsync	The Order Server shall accept, in acknowledge to a GetQuotationResponse message, an XML instance document validated by the entity GetQuotationResponseAck in the oseo.xsd XML Schema.
Req 53	http://www.opengis.net/spec/OSEO/1.0/req/QuotationOffLine/GetQuotation/req	QuotationOffLine	The Order Server shall accept a GetQuotation request including: □ the order to be quoted (of any supported order type): GetQuotation/orderSpecification □ Empty reply address: soapenv:Envelope/soapenv:Header/wsa:ReplyTo = http://www.w3.org/2005/08/addressing/anonymous
Req 54	http://www.opengis.net/spec/OSEO/1.0/req/QuotationOffLine/GetQuotation/req_ack	QuotationOffLine	The Order Server shall reply to a GetQuotation request returning: □ the quotationId GetQuotationAck/quotationId

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 55	http://www.opengis.net/spec/OSEO/1.0/req/QuotationOffLine/GetQuotation/quotation	QuotationOffLine	The Order Server shall provide the order quotation via mail / e-mail to the address specified by the user.

Table 10-1: GetQuotation requirements.

10.3 GetQuotation operation request

The type of GetQuotation is GetQuotationRequestType. The following figure provides a graphical representation of this type.

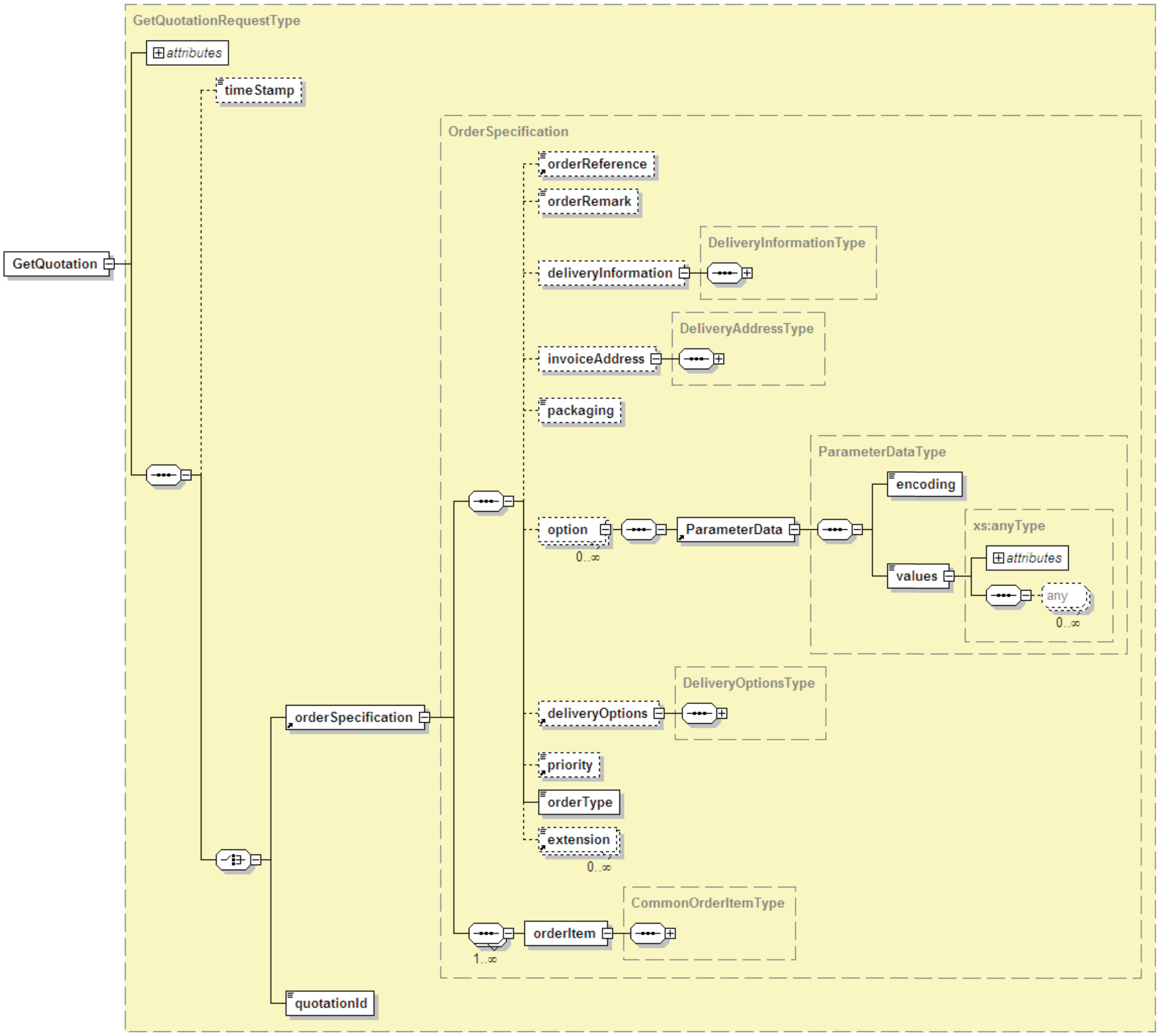


Figure 10-1 - GetQuotation diagram.

10.3.1 GetQuotation request parameters

A request to perform GetQuotation operation shall include the use the data structure specified in the previous figure and the following table.

Names	Definition	Data type and values	Multiplicity and use
service	Identifier for the service type	Type: non-empty string Allowed values: OS	One (mandatory)
version	Specification version for operation	Type: non-empty string. Format: <x>.<y>.<z>	One (mandatory)
timeStamp	It is the time when the request has been issued.	Type: xs:DateTime	One (optional)
orderSpecification	Order Specification	Type: OrderSpecification (see Table 7-12)	One (mandatory)
quotationId	This field is alternative to the previous one. It is a quotation identifier returned by a previous call to GetQuotation operation. In this case GetQuotation will return either the quotation of the order or again the same identifier. This behaviour is supported when Capabilities/Content/GetQuotationCapabilities/monitoring is true.	Type:QuotationIdType	One (mandatory)

Table 10-2 - GetQuotation description

10.3.2 GetQuotation request XML encoding

A XML schema fragment for this operation encoded in XML:

```
<xs:schema xmlns="http://www.opengis.net/oseo/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:sps="http://www.opengis.net/sps/2.0"
targetNamespace="http://www.opengis.net/oseo/1.0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
      <xs:import namespace="http://www.opengis.net/swe/2.0"
schemaLocation="../swe/sweCommon/2.0.0/swe.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsGetCapabilities.xsd"/>
      <xs:import namespace="http://www.opengis.net/sps/2.0"
schemaLocation="Order_spsCommon.xsd"/>
      <xs:import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="../gml/3.2.1/base/gml.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsExceptionReport.xsd"/>
…
<xs:element name="GetQuotation" type="GetQuotationRequestType">
             <xs:annotation>
```

```
                   <xs:documentation>GetQuotation operation - request
message</xs:documentation>
             </xs:annotation>
      </xs:element>
<xs:complexType name="GetQuotationRequestType">
             <xs:complexContent>
                   <xs:extension base="OrderRequestBaseType">
                   <xs:sequence>
                   <xs:element name="timeStamp" type="xs:dateTime" minOccurs="0"/>
                   <xs:choice>
                   <xs:element ref="orderSpecification">
                          <xs:annotation>
                                 <xs:documentation>This choice is set for getting the
quotation of an order.</xs:documentation>
                          </xs:annotation>
                   </xs:element>
                   <xs:element name="quotationId" type="QuotationIdType">
                          <xs:annotation>
                                 <xs:documentation>This choice is set when quotation
monitoring is supported and a quotation request has been already
submitted.</xs:documentation>
                          </xs:annotation>
                   </xs:element>
                   </xs:choice>
                   </xs:sequence>
                   </xs:extension>
             </xs:complexContent>
      </xs:complexType>
….
</schema>
```

10.3.3 GetQuotation request example

The following is an example of GetQuotation request.

Note that the example includes only the GetQuotation element without the SOAP envelope.

```
<GetQuotation xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0" xmlns:sps="http://www.opengis.net/sps/2.0"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd" service="OS" version="1.0.0">
      <orderSpecification>
             <orderReference>example_0001</orderReference>
             <orderRemark>example</orderRemark>
             <deliveryInformation>
                   <mailAddress>
                          <recipient>Mr. John Smith</recipient>
                          <companyRef>DTMT</companyRef>
                          <postalAddress>
                                 <streetAddress>Esrin Esa No. 1</streetAddress>
                                 <city>Frascati</city>
                                 <state>-</state>
                                 <postalCode>00100</postalCode>
                                 <country>IT</country>
                                 <postBox>-</postBox>
                          </postalAddress>
                          <telephoneNumber>00390694180999</telephoneNumber>
                   </mailAddress>
             </deliveryInformation>
             <orderType>PRODUCT_ORDER</orderType>
             <orderItem>
                   <itemId>item_0001</itemId>
```

```
                    <productOrderOptionsId>Level 1,Product SLC
(ASA_IMS)</productOrderOptionsId>
                    <orderItemRemark>First product</orderItemRemark>
                    <option>
                           <sps:ParameterData>
                                  <sps:encoding>
                                         <swe:XMLEncoding
defaultNamespace="http://www.opengis.net/oseo/1.0/dataset"/>
                                  </sps:encoding>
                                  <sps:values
xmlns:ns="http://www.opengis.net/oseo/1.0/dataset">
                                         <ns:processingLevel>1B</ns:processingLevel>
                                  </sps:values>
                           </sps:ParameterData>
                    </option>
                    <sceneSelection>
                           <sps:ParameterData>
                                  <sps:encoding>
                                         <swe:XMLEncoding
defaultNamespace="http://www.opengis.net/oseo/1.0/dataset"/>
                                  </sps:encoding>
                           <sps:values
xmlns:ns="http://www.opengis.net/oseo/1.0/dataset">
                             <ns:sceneType>FloatingScene</ns:sceneType>
                             <ns:sceneStartStopTime>
                                <ns:min>2010-01-01T11:55:00.00Z</ns:min>
                                <ns:max>2010-01-01T12:00:00.00Z
                              </ns:sceneStartStopTime>
                           </sps:values>
                           </sps:ParameterData>
                    </sceneSelection>
                    <deliveryOptions>
                    <mediaDelivery>
                           <packageMedium>CD-ROM</packageMedium>
                    </mediaDelivery>
                    </deliveryOptions>
                    <payment>
                           <paymentMethod>quota</paymentMethod>
                           <orderAccount>project_10000</orderAccount>
                    </payment>
                    <productId>

      <identifier>http://www.opengis.net/def/EOP/ESA/0/ESA.EECF.ENVISAT_ASA_IMx_xS/EN
1-05121921114570-3322.XI</identifier>

      <collectionId>http://www.opengis.net/def/EOP/ESA/0/ESA.EECF.ENVISAT_ASA_IMx_xS
</collectionId>
                    </productId>
             </orderItem>
      </orderSpecification>
</GetQuotation>
```

10.4 GetQuotation operation response

GetQuotation can be a synchronous or asynchronous operation. The synchronous output message is defined by the XML GetQuotationAck entity. The type of GetQuotationAck is GetQuotationAckType. The following figure provides a graphical representation of this element.

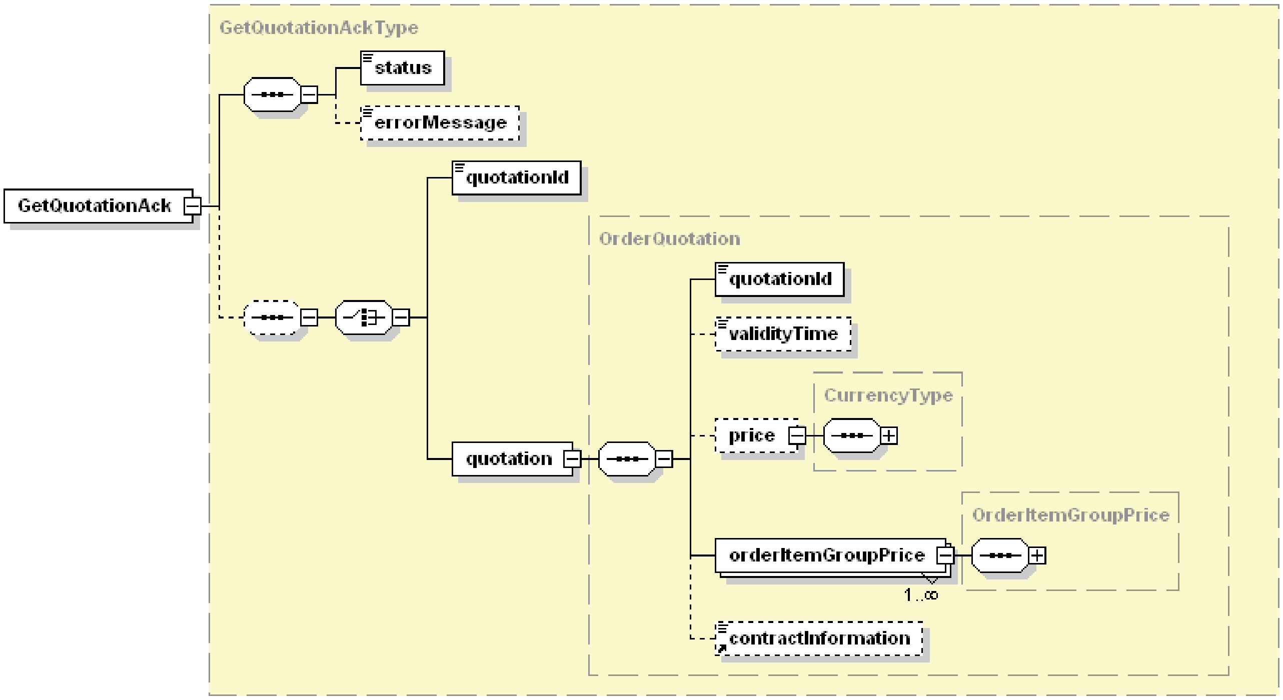


Figure 10-2 - GetQuotationAck element diagram.

10.4.1 GetQuotation operation response parameters

The GetQuotationAck defines the acknowledge to GetQuotation.

The following tables presents the parameters of GetQuotationAck element.

Names	Definition	Data type and values	Multiplicity and use
status	Completion result of the operation: – success: operation successfully executed; – partial: non blocking errors occurred during processing of the request. If the Server error occurred the request will abort and it will be generated a SOAP-Fault.	Type: String Permitted Values: success, partial	One (mandatory)
errorMessage	Message to explain the partial result.	Type**:** Not empty string (max 255 chars).	One (optional)
quotationId	Identifier of the requested quotation. This field is set in case of asynchronous quotation. This field is alternative to 'quotation'	Type: QuotationIdType	One (mandatory/choise)

Names	Definition	Data type and values	Multiplicity and use
quotation	This field is alternative to the previous one. It specifies the quotation of the order. It is set only in case of synchronous / asynchronous with Monitoring quotation requests. This field is alternative to 'quotationId'	Type: OrderQuotation. (see §7.3.10)	One (mandatory/choise)

Table 10-3 - GetQuotationAck description

10.4.2 GetQuotation response XML encoding.

A XML schema fragment for this operation encoded in XML :

```
<xs:schema xmlns="http://www.opengis.net/oseo/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:sps="http://www.opengis.net/sps/2.0"
targetNamespace="http://www.opengis.net/oseo/1.0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
      <xs:import namespace="http://www.opengis.net/swe/2.0"
schemaLocation="../swe/sweCommon/2.0.0/swe.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsGetCapabilities.xsd"/>
      <xs:import namespace="http://www.opengis.net/sps/2.0"
schemaLocation="Order_spsCommon.xsd"/>
      <xs:import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="../gml/3.2.1/base/gml.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsExceptionReport.xsd"/>

…
      <xs:element name="GetQuotationAck" type="GetQuotationAckType">
      <xs:annotation>
             <xs:documentation>GetQuotation operation - response
message</xs:documentation>
      </xs:annotation>
      </xs:element>
       <xs:complexType name="GetQuotationAckType">
      <xs:complexContent>
             <xs:extension base="OrderResponseBaseType">
             <xs:sequence minOccurs="0">
             <xs:choice>
             <xs:element name="quotationId" type="QuotationIdType">
             <xs:annotation>
                    <xs:documentation>This choice is set in case of non sync
quotations.</xs:documentation>
             </xs:annotation>
             </xs:element>
             <xs:element name="quotation" type="OrderQuotation">
             <xs:annotation>
                    <xs:documentation>This choice is set in case of synchronous
quotations or as answer to quotation monitoring.</xs:documentation>
             </xs:annotation>
             </xs:element>
             </xs:choice>
             </xs:sequence>
             </xs:extension>
      </xs:complexContent>
</xs:complexType>
….
</schema>
```

10.4.3 GetQuotation response example.

The following is an example of GetQuotation (synchronous) response.

Note that the example includes only the GetQuotationAck element without the SOAP envelope.

```
<GetQuotationAck xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:sps="http://www.opengis.net/sps/2.0" xmlns:swe="http://www.opengis.net/swe/2.0"
xmlns:gml="http://www.opengis.net/gml/3.2"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd">
      <status>success</status>
      <quotation>
             <quotationId>quotationId_0001</quotationId>
             <validityTime>2007-05-01T00:00:00.00Z</validityTime>
             <price>
                    <value>15</value>
                    <currency>QUOTA</currency>
             </price>
             <orderItemGroupPrice>
                    <provider>
                           <serviceName>ESA_ORDER_SERVICE</serviceName>
                           <organization>ESA</organization>
                    </provider>
                    <price>
                           <value>15</value>
                           <currency>QUOTA</currency>
                    </price>
                    <balance>
                           <value>20</value>
                           <currency>QUOTA</currency>
                    </balance>
                    <orderItemPrice>
                           <itemId>item_0001</itemId>
                           <productId>

      <identifier>http://www.opengis.net/def/EOP/ESA/0/ESA.EECF.ENVISAT_ASA_IMx/EN1-
05121921114570-3322.XI</identifier>

      <collectionId>http://www.opengis.net/def/EOP/ESA/0/ESA.EECF.ENVISAT_ASA_IMx</co
llectionId>
                           </productId>
                    </orderItemPrice>
             </orderItemGroupPrice>
      </quotation>
</GetQuotationAck>
```

10.4.4 Exceptions

In the event that an Order Server encounters an error servicing a GetQuotation, it shall return an exception within a SOAP Fault (in particular an ows:ExceptionReport according to the clause 8 of [NR9]).

The SOAP Fault Message (see §7.3.14) shall have one of the following exceptionCode value:

“exceptionCode” value	**Meaning of code**	**“locator” value**	**“ExceptionText” value**
MissingParameterValue	Operation request does not include a parameter value, and this server did not declare a default value for that parameter	Name of missing parameter	“Missing value for Parameter”
InvalidParameterValue	Operation request contains an invalid parameter value	Name of parameter with invalid value	“Invalid value for Parameter”
NoApplicableCode	No other exceptionCode specified by this service and server applies to this exception	None, omit “locator” parameter	“Code not applicable”
InvalidNotificationValue	In asynchronous mode, the operation request has the status notification active (with “Full” or “Final” value) but an address information to which the server has to notify the ordering status that is empty or invalid.	“ws-address” parameter	“Invalid value for notification”
AuthenticationFailed	Invalid or missing identity information	“identity_token”	“Invalid or missing identity information”
AuthorizationFailed	The client is not authorized to call the operation or it is not authorized to provide the specified parameters.	Possible values: □ “GetQuotation” □ “collectionId” □ “identifier” □ “taskingRequestId” □ etc..	Text describing the item not authorized. E.g.: □ “The client is not authorized to call the operation.” □ “the client is not authorized to order that product” □ Etc.
SubscriptionNotSupported	Operation request identifies a order type subscription but the server doesn’t support “SubscriptionOrders”	“orderType” parameter	“Subscription not supported”
ProductOrderingNotSupported	Operation request identifies a order type order product but the server doesn’t support “ProductOrders”	“orderType” parameter	“Ordering not supported”
FutureProductNotSupported	Operation request identifies a order type future products but the server doesn’t support “ProgrammingOrders”	“orderType” parameter	“Programming not supported”

Table 10-4 – Exception Codes for GetQuotation operation.

11 GetQuotationResponse operation

11.1 Introduction

GetQuotation can be an asynchronous operation, then if the server supports asynchronous notification, two operations are called:

- GetQuotation, from client to server, for submitting the quotation.
    - This operation is composed of two messages:
        - GetQuotation, it is the quotation submission request
        - GetQuotationAck, it is the acknowledge the server return in real time to the request.
- GetQuotationResponse, from server to client, for sending the quotation notification to the client. Then this is an operation to be implemented on the client.
    - This operation is composed of two messages:
        - GetQuotationResponse, it is the quotation notification sent to the client;
        - GetQuotationResponseAck, to return the successfull reception of the notification.

The asynchronous quotation is sent in the following case:

- The server supports asynchronous quotation (Capabilities/Content/GetQuotationCapabilities/asynchronous is set to true)
- The client specifies its notification address in <wsa:ReplyTo> tag and a valid identifier is specified in <wsa:MessageID> tag;

11.2 Requirements

See §10.2

11.3 GetQuotationResponse operation request

This operation has to be implemented by a client of Ordering service supporting asynchronous operations. This operation allows the Ordering Service to send the quotation to the client.

The following figure provides a graphical representation of the GetQuotationResponse element.

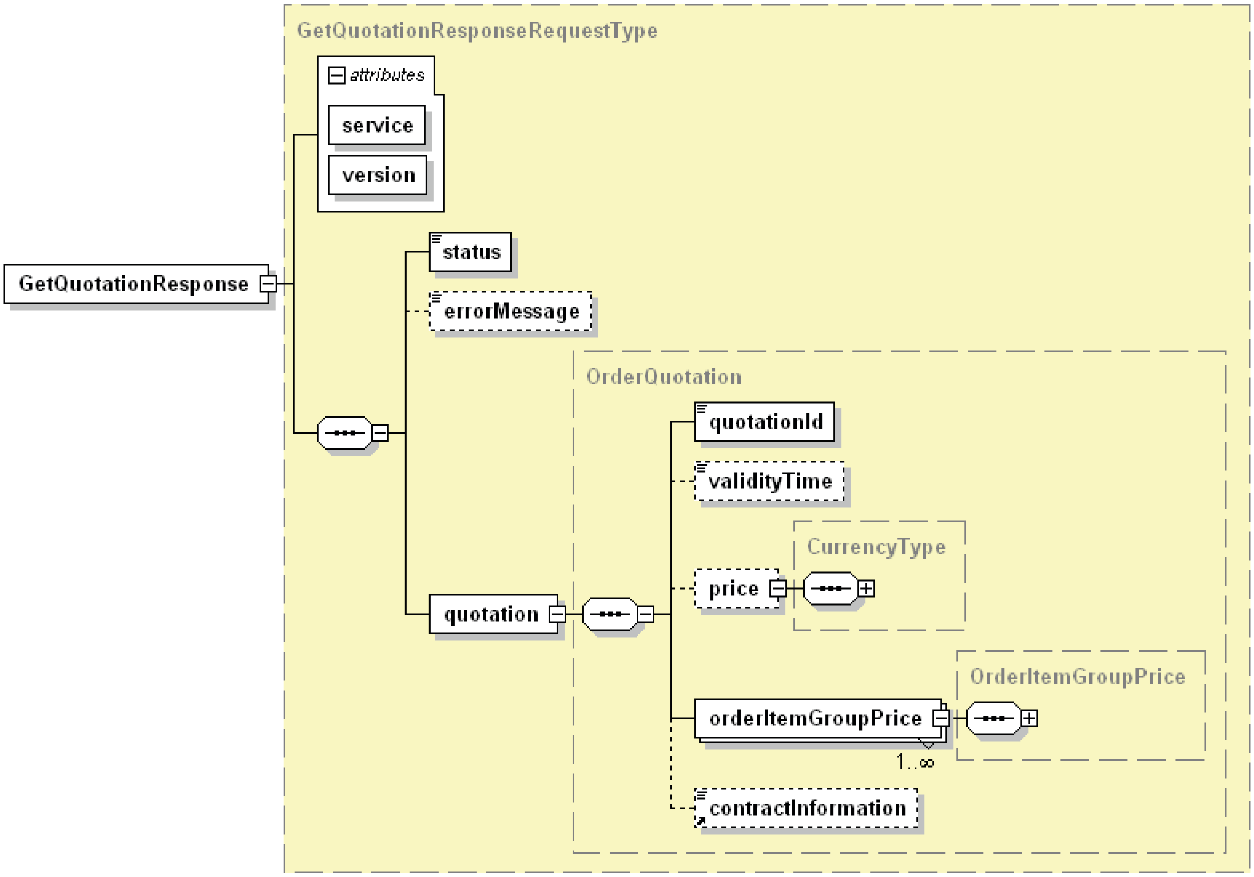


Figure 11-1 - GetQuotationResponse diagram.

11.3.1 GetQuotationResponse request parameters

The following table presents the parameters of the GetQuotationResponse.

Names	Definition	Data type and values	Multiplicity and use
service	Identifier of the service type	Type: non-empty string Allowed values: OS	One (mandatory)
version	Specification version for operation	Type: non-empty string. Format: <x>.<y>.<z>	One (mandatory)
status	Completion result of the quotation process: – success: operation successfully executed; – partial: some items have not been quoted. (e.g. if some servers don't support quotation) If the Server error occurred the request will abort and it will be generated a SOAP-Fault.	Type: String Permitted Values: success, partial	One (mandatory)
errorMessage	Message to explain the reasons for the partial result.	Type: Not empty string (max 255 chars).	One (optional)

Names	Definition	Data type and values	Multiplicity and use
quotation	It specifies the quotation of the order.	Type: OrderQuotation (see §7.3.9)	One (mandatory)

Table 11-1 - GetQuotationResponse description

11.3.2 GetQuotationResponse request XML encoding

A XML schema fragment for this operation encoded in XML

```
<xs:schema xmlns="http://www.opengis.net/oseo/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:sps="http://www.opengis.net/sps/2.0"
targetNamespace="http://www.opengis.net/oseo/1.0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
      <xs:import namespace="http://www.opengis.net/swe/2.0"
schemaLocation="../swe/sweCommon/2.0.0/swe.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsGetCapabilities.xsd"/>
      <xs:import namespace="http://www.opengis.net/sps/2.0"
schemaLocation="Order_spsCommon.xsd"/>
      <xs:import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="../gml/3.2.1/base/gml.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsExceptionReport.xsd"/>

…
<xs:element name="GetQuotationResponse" type="GetQuotationResponseRequestType">
             <xs:annotation>
                    <xs:documentation>Async reply to GetQuotation  - This message
carries on the quotation.</xs:documentation>
             </xs:annotation>
      </xs:element>
      <xs:element name="GetQuotationResponseAck" type="GetQuotationResponseAckType">
             <xs:annotation>
                    <xs:documentation>Response to acknowledge the receiption of
quotation.</xs:documentation>
             </xs:annotation>
      </xs:element>
<xs:complexType name="GetQuotationResponseRequestType">
             <xs:complexContent>
             <xs:extension base="OrderRequestBaseType">
                    <xs:sequence>
                    <xs:element name="status" type="OrderResponseStatusType"
nillable="false"/>
                    <xs:element name="errorMessage" minOccurs="0">
                    <xs:annotation>
                    <xs:documentation>This field is set when status element is
different from success. It provides some information about the occurred
problem.</xs:documentation>
                    </xs:annotation>
                    <xs:simpleType>
                    <xs:restriction base="xs:string">
                    <xs:maxLength value="255"/>
                    </xs:restriction>
                    </xs:simpleType>
                    </xs:element>
```

```
                     <xs:element name="quotation" type="OrderQuotation"/>
                     </xs:sequence>
                     </xs:extension>
              </xs:complexContent>
       </xs:complexType>
       <xs:complexType name="GetQuotationResponseAckType">
              <xs:complexContent>
                     <xs:extension base="OrderResponseBaseType"/>
              </xs:complexContent>
       </xs:complexType>
       <xs:complexType name="OrderQuotation">
              <xs:sequence>
              <xs:element name="quotationId" type="QuotationIdType"/>
              <xs:element name="validityTime" type="xs:dateTime" minOccurs="0"/>
              <xs:element name="price" type="CurrencyType" minOccurs="0">
              <xs:annotation>
              <xs:documentation>Price of the whole order; mandatory unless the 
provider uses quota concept or products are free of charge.</xs:documentation>
              </xs:annotation>
              </xs:element>
              <xs:element name="orderItemGroupPrice" type="OrderItemGroupPrice" 
maxOccurs="unbounded"/>
              <xs:element ref="contractInformation" minOccurs="0"/>
              </xs:sequence>
       </xs:complexType>
….
</schema>
```

11.3.3 GetQuotationResponse request XML example

The following is an example of GetQuotationResponse request.

Note that the example includes only the GetQuotationResponse element without the SOAP envelope.

```
<GetQuotationResponse service="OS" version="1.0.0" 
xmlns="http://www.opengis.net/oseo/1.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance" xsi:schemaLocation="http://www.opengis.net/oseo/1.0 
xsi:schemaLocation="http://www.opengis.net/oseo/1.0 
http://schemas.opengis.net/oseo/1.0/oseo.xsd">
       <status>success</status>
       <quotation>
              <quotationId>quotationId_0001</quotationId>
              <orderItemGroupPrice>
                     <provider>
                            <serviceName>ESA_ORDER_SERVICE</serviceName>
                            <organization>ESA</organization>
                     </provider>
                     <price>
                            <value>50</value>
                            <currency>QUOTA</currency>
                     </price>
                     <orderItemPrice>
                            <itemId>item_001</itemId>
                            <productId>
                                   <identifier>EN1-05121921114570-3322.XI</identifier>
                            </productId>
                     </orderItemPrice>
              </orderItemGroupPrice>
       </quotation>
</GetQuotationResponse>
```

11.4 GetQuotationResponse operation response

The following figure provides a graphical representation of this element.

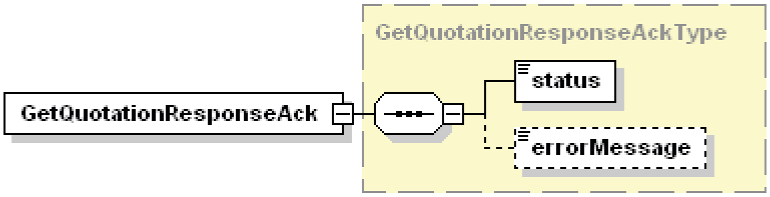


Figure 11-2 - GetQuotationResponseAck diagram

11.4.1 GetQuotationResponse response parameters

The following table describes the XML entities relate to it.

Names	Definition	Data type and values	Multiplicity and use
status	Completion result of the operation: – success: operation successfully executed; – partial: non blocking errors occurred during processing of the request. If the Server error occurred the request will abort and it will be generated a SOAP-Fault.	Type: String Permitted Values: success, partial	One (mandatory)
errorMessage	Message to explain the reasons of the partial result.	Type**:** Not empty string (max 255 chars).	One (optional)

Table 11-2 - GetQuotationResponseAck description

11.4.2 GetQuotationResponse response XML encoding

See §11.3.2

11.4.3 GetQuotationResponse response example.

The following is an example of GetQuotationResponse response (asynchronous).

Note that the example includes only the GetQuotationResponseAck element without the SOAP envelope.

```
<GetQuotationResponseAck xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0" xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:sps="http://www.opengis.net/sps/2.0" xmlns:swe="http://www.opengis.net/swe/2.0"
xmlns:gml="http://www.opengis.net/gml/3.2"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd">
<status>success</status>
</GetQuotationResponseAck>
```

11.4.4 Exceptions.

This operation has to be implemented by a client of Ordering service**.**

In the event that an error occurs servicing a GetQuotationResponse, it shall return an exception within a SOAP Fault (in particular an ows:ExceptionReport according to the clause 8 of [NR9]).

The SOAP Fault Message (see §7.3.14) shall have one of the following exceptionCode value:

“exceptionCode” value	**Meaning of code**	**“locator” value**	**“ExceptionText” value**
MissingParameterValue	Operation request does not include a parameter value, and this server did not declare a default value for that parameter	Name of missing parameter	“Missing value for Parameter”
InvalidParameterValue	Operation request contains an invalid parameter value	Name of parameter with invalid value	“Invalid value for Parameter”
NoApplicableCode	No other exceptionCode specified by this service and server applies to this exception	None, omit “locator” parameter	“Code not applicable”
AuthenticationFailed	Invalid or missing identity information	“identity_token”	“Invalid or missing identity information”
AuthorizationFailed	The client is not authorized to call the operation.	None, omit “locator” parameter	“The client is not authorized to call the operation.”

Table 11-3 – Exception codes for GetQuotationResponse operation.

12 Submit Operation

12.1 Introduction

This operation allows either to submit an EO products order or subscribe EO products or submit an order from a tasking request.

The order to submit can be specified in two different ways:

- **Via quotation identifier**

  When the server supports order quotation, then the order can be submitted specifying the quotation identifier received from the previously executed GetQuotation operation.

- **Via order specification**

  When the server does not support quotation or the user does not need it then the order can be submitted directly specifying all the order parameters.

Submit is an asynchronous operation because the acquisition, processing and delivery of EO products is in general a long lasting process.

However this specification allows the client to specify the amount of notification to receive:

- None: no asynchronous notification is sent to the client, then it is not required to implement the asynchronous protocol.

  This kind of notification is activated by:

  - setting **Submit/statusNotification** to **None**.

- Final: the client is notified when the whole order has been completed. This mechanism is possible only for clients working as server and implementing the SubmitResponse operation.

  This kind of notification is activated by:

  - setting **Submit/statusNotification** to **Final**;
  - setting <**wsa:ReplyTo**> of Submit message with the address where the client is listening to the notification and <wsa:MessageID> shall include a unique identifier of the request.

- Every change of the order status is notified to the client. This mechanism is possible only for clients working as server and implementing the SubmitResponse operation.

  This kind of notification is activated by:

  - setting **Submit/statusNotification** to **All**;
  - setting <wsa:ReplyTo> of Submit message with the address where the client is listening to the notification and <wsa:MessageID> shall include a unique identifier of the request.

12.2 Requirements

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 56	http://www.opengis.net/spec/OSEO/1.0/req/Core/Submit	Core	The Order Server shall implement the Submit operation.
Req 57	http://www.opengis.net/spec/OSEO/1.0/req/Core/Submit/order_spec	Core	The Order Server shall support order submission by order specification (any supported order type): Submit/orderSpecification

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 58	http://www.opengis.net/spec/OSEO/1.0/req/Core/Submit/no_notification	Core	The Order Server shall support orders with no notification: Submit/statusNotification = None
Req 59	http://www.opengis.net/spec/OSEO/1.0/req/Core/Submit/req_schema	Core	The Submit request shall consist of an XML instance document validated by the entity Submit in the oseo.xsd XML Schema.
Req 60	http://www.opengis.net/spec/OSEO/1.0/req/Core/Submit/resp_schema	Core	The message output generated by the Submit request shall consist of an XML instance document validated by the entity SubmitAck in the oseo.xsd XML Schema.
Req 61	http://www.opengis.net/spec/OSEO/1.0/req/Core/Submit/resp_id	Core	The response to a successful Submit request shall contain: □ Unique order identifier □ status information set to success.
Req 62	http://www.opengis.net/spec/OSEO/1.0/req/Core/Submit/exception	Core	When a Order Server encounters an error while performing a Submit operation, it shall return an ows: ExceptionReport according to the clause 8 of [NR9].
Req 63	http://www.opengis.net/spec/OSEO/1.0/req/AsyncSubmit/Submit/final	AsyncSubmit	The Order Server shall accept Submit request including: □ Status notification: Submit/statusNotification = Final □ The reply address: soapenv:Envelope/soapenv:Header/wsa:ReplyTo
Req 64	http://www.opengis.net/spec/OSEO/1.0/req/AsyncSubmit/Submit/all	AsyncSubmit	The Order Server shall accept Submit request including: □ Status notification: Submit/statusNotification = All □ The reply address: soapenv:Envelope/soapenv:Header/wsa:ReplyTo
Req 65	http://www.opengis.net/spec/OSEO/1.0/req/AsyncSubmit/SubmitResponse/req/schema	AsyncSubmit	The asynchronous notification (SubmitResponse) to a successful Submit request shall consist of an XML instance document as validated by the entity SubmitResponse in the oseo.xsd XML Schema.
Req 66	http://www.opengis.net/spec/OSEO/1.0/req/AsyncSubmit/SubmitResponse/req/final	AsyncSubmit	Ths asynchronous notification (SubmitResponse) to a Submit request with: Submit/statusNotification = Final shall be sent only once at order completion.
Req 67	http://www.opengis.net/spec/OSEO/1.0/req/AsyncSubmit/SubmitResponse/req/all	AsyncSubmit	Ths asynchronous notification (SubmitResponse) to a Submit request having: Submit/statusNotification = All shall be sent whenever the status of at least one item in the order is changed.

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 68	http://www.opengis.net/spec/OSEO/1.0/req/AsyncSubmit/SubmitResponse/resp	AsyncSubmit	The asynchronous notification (SubmitResponse) to a successful Submit request shall contain: □ The order date and time: SubmitResponse/timeStamp □ One status info for each ordered item: SubmitResponse/orderMonitorSpecification/orderItem
Req 69	http://www.opengis.net/spec/OSEO/1.0/req/AsyncSubmit/SubmitResponse/resp/ack_schema	AsyncSubmit	The message output generated by the response to a successful SubmitResponse request shall consist of an XML instance document validated by the entity SubmitResponseAck in the oseo.xsd XML Schema.
Req 70	http://www.opengis.net/spec/OSEO/1.0/req/ProductOrder/Submit/prod_order	ProductOrder	The Order Server shall accept Submit requests having: Submit/orderSpecification/orderType = ”PRODUCT_ORDER” And including at least one Earth Observation product: Submit/orderSpecification/orderItem/productId
Req 71	http://www.opengis.net/spec/OSEO/1.0/req/SceneSelection/Submit/scene	SceneSelection	The Order Server shall accept, for at least one collection or product, a Submit request including at least 1 scene selection option: Submit/orderSpecification/orderItem/sceneSelection
Req 72	http://www.opengis.net/spec/OSEO/1.0/req/SceneSelection/Submit/exception	SceneSelection	When a Order Server encounters an incorrect scene selection option it shall return an ows: ExceptionReport according to the clause 8 of [NR9].
Req 73	http://www.opengis.net/spec/OSEO/1.0/req/SubscriptionOrder/Submit/req	SubscriptionOrder	The Order Server shall accept Submit requests having: Submit/orderSpecification/orderType = “SUBSCRIPTION_ORDER” and including one Subscription to Earth Observation products: Submit/orderSpecification/orderItem/subscriptionId
Req 74	http://www.opengis.net/spec/OSEO/1.0/req/TaskingOrder/Submit/req	TaskingOrder	The Order Server shall accept Submit requests having: Submit/orderSpecification/orderType = “TASKING_ORDER” and including one Tasking Request: Submit/orderSpecification/orderItem/taskingRequestId
Req 75	http://www.opengis.net/spec/OSEO/1.0/req/Quotation/Submit/req	Quotation	The Order Server shall accept Submit requests having: □ The identifier of a previously asked quotation: Submit/quotationId

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 76	http://www.opengis.net/spec/OSEO/1.0/req/OnlineDataAccess/Submit/req	OnlineDataAccess	The Order Server shall accept for at least one collection / or product, Submit requests having online access delivery options i.e. it shall accept Submit requests including: Either Submit/orderSpecification/deliveryOptions/onlineDataAccess or Submit/orderSpecification/orderItem/deliveryOptions/onlineDataAccess
Req 77	http://www.opengis.net/spec/OSEO/1.0/req/OnlineDataDelivery/Submit/req	OnlineDataDelivery	The Order Server shall accept for at least one collection / or product, Submit requests having online data delivery options i.e. it shall accept Submit requests including: Either Submit/orderSpecification/deliveryOptions/onlineDataDelivery or Submit/orderSpecification/orderItem/deliveryOptions/onlineDataDelivery Additionally the delivery address for the specified protocol shall be provided: Submit/orderSpecification/deliveryInformation/onlineAddress
Req 78	http://www.opengis.net/spec/OSEO/1.0/req/MediaDelivery/Submit/req	MediaDelivery	The Order Server shall accept for at least one collection / or product, Submit requests having media delivery options i.e. it shall accept Submit requests including: Either Submit/orderSpecification/deliveryOptions/mediaDelivery or Submit/orderSpecification/orderItem/deliveryOptions/mediaDelivery Additionally the delivery address for the specified media shall be provided: Submit/orderSpecification/deliveryInformation/mailAddress

Table 12-1: Submit requirements.

12.3 Submit operation request

The type of Submit element is SubmitProductOrderRequestType. The following figure provides a graphical representation of this type.

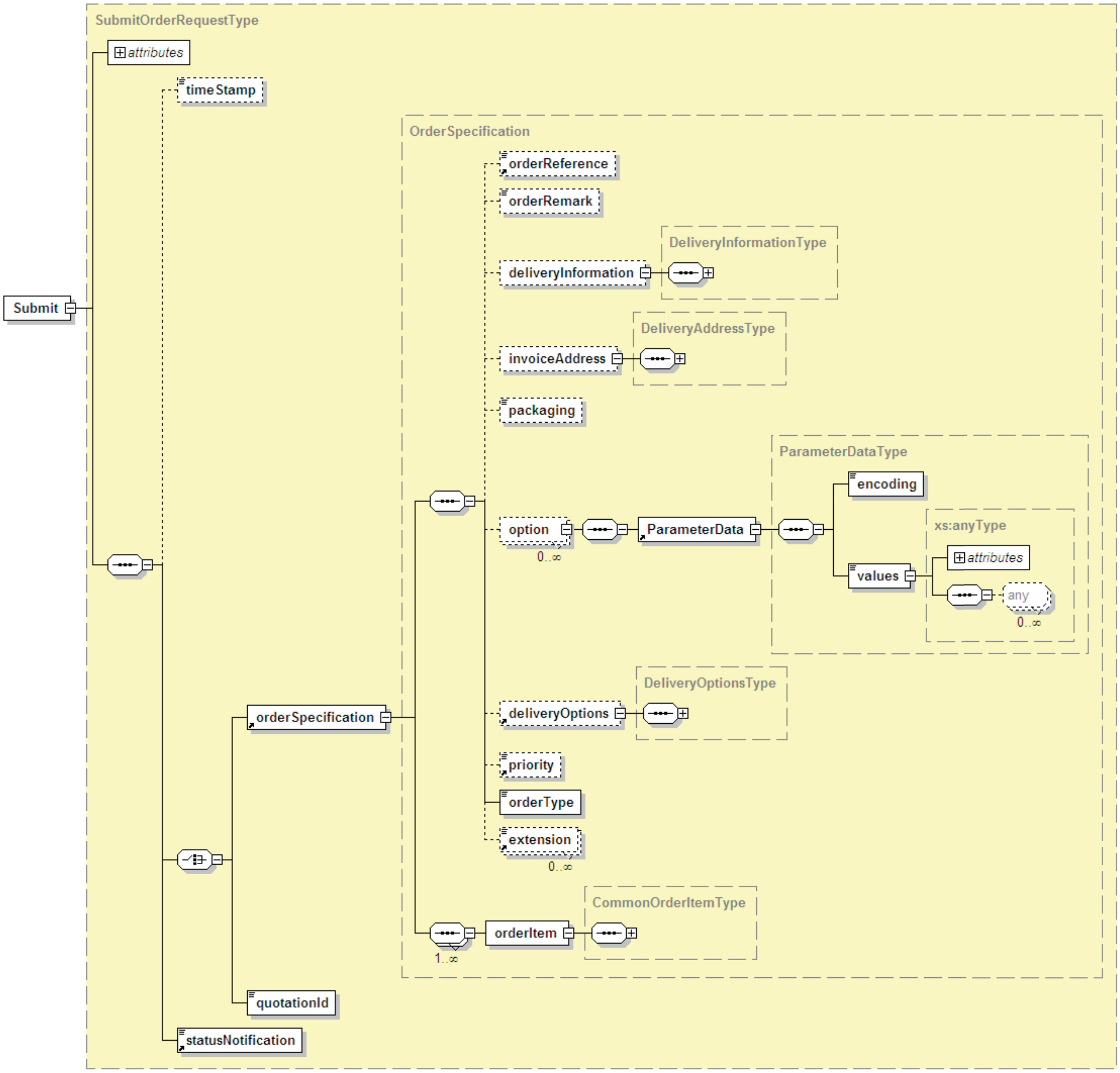


Figure 12-1 - Submit diagram.

12.3.1 Submit request parameters

The Submit operation contains all information to submit an order. A request to perform Submit operation shall include the data structure specified in the previous figure and the following table.

Names	Definition	Data type and values	Multiplicity and use
service	Service type identifier	Type: non-empty string Allowed values: OS	One (mandatory)

Names	Definition	Data type and values	Multiplicity and use
version	Specification version for operation	Type: non-empty string. Format: <x>.<y>.<z>	One (mandatory)
timeStamp	It is the time when the request has been issued.	Type: xs:DateTime	One (optional)
orderSpecification	Order Specification element. This field is alternative to 'quotationId'.	Type: OrderSpecification (see Table 7-12)	One (mandatory/choise)
quotationId	QuotationId returned by GetQuotation operation, This field is alternative to 'orderSpecification'.	Type: QuotationIdType.	One (mandatory/choise)
statusNotification	This element specifies how many status notifications are sent back to the client.	Type: String Allowed values:None, Final, All	One (mandatory)

Table 12-2 - Submit description

12.3.2 Submit request XML encoding

A XML schema fragment for this operation encoded in XML

```
<xs:schema xmlns="http://www.opengis.net/oseo/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:sps="http://www.opengis.net/sps/2.0"
targetNamespace="http://www.opengis.net/oseo/1.0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
      <xs:import namespace="http://www.opengis.net/swe/2.0"
schemaLocation="../swe/sweCommon/2.0.0/swe.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsGetCapabilities.xsd"/>
      <xs:import namespace="http://www.opengis.net/sps/2.0"
schemaLocation="Order_spsCommon.xsd"/>
      <xs:import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="../gml/3.2.1/base/gml.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsExceptionReport.xsd"/>
…
<xs:element name="Submit" type="SubmitOrderRequestType"/>
<xs:complexType name="SubmitOrderRequestType">
<xs:complexContent>
      <xs:extension base="OrderRequestBaseType">
      <xs:sequence>
      <xs:element name="timeStamp" type="xs:dateTime" minOccurs="0"/>
            <xs:choice>
            <xs:element ref="orderSpecification"/>
            <xs:element name="quotationId" type="QuotationIdType"/>
            </xs:choice>
            <xs:element ref="statusNotification">
            <xs:annotation>
```

```
             <xs:documentation>This element specifies how many status notifications
are sent back to the caller of the Submit operation.</xs:documentation>
             </xs:annotation>
             </xs:element>
      </xs:sequence>
      </xs:extension>
</xs:complexContent>
</xs:complexType>
…
</schema>
```

12.3.3 Submit request example

The following is an example of Submit request.

Note that the example includes only the Submit element without the SOAP envelope.

```
<Submit xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0" xmlns:sps="http://www.opengis.net/sps/2.0"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:gml="http://www.opengis.net/gml"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd" service="OS" version="1.0.0">
      <orderSpecification>
             <orderReference>example_0001</orderReference>
             <orderRemark>example</orderRemark>
             <deliveryInformation>
                    <mailAddress>
                           <recipient>Mr. John Smith</recipient>
                           <companyRef>DTMT</companyRef>
                           <postalAddress>
                                  <streetAddress>Esrin Esa No. 1</streetAddress>
                                  <city>Frascati</city>
                                  <state>-</state>
                                  <postalCode>00100</postalCode>
                                  <country>IT</country>
                                  <postBox>-</postBox>
                           </postalAddress>
                           <telephoneNumber>00390694180999</telephoneNumber>
                    </mailAddress>
             </deliveryInformation>
             <orderType>PRODUCT_ORDER</orderType>
             <orderItem>
                    <itemId>item_0001</itemId>
                    <productOrderOptionsId>Level 1,Product SLC
(ASA_IMS)</productOrderOptionsId>
                    <orderItemRemark>First product</orderItemRemark>
                    <option>
                           <sps:ParameterData>
                                  <sps:encoding>
                                         <swe:XMLEncoding
defaultNamespace="http://www.opengis.net/oseo/1.0/dataset"/>
                                  </sps:encoding>
                                  <sps:values
xmlns:ns="http://www.opengis.net/oseo/1.0/dataset">
                                  <ns:processingLevel>1B</ns:processingLevel>
                                  <ns:productType>ASA_IMS_1P</ns:productType>
                                  <ns:qualityOfService>STANDARD</ns:qualityOfService>
                                  </sps:values>
                           </sps:ParameterData>
                    </option>
                    <sceneSelection>
                           <sps:ParameterData>
```

```
                                 <sps:encoding>
                                 <swe:XMLEncoding
defaultNamespace="http://www.opengis.net/oseo/1.0/dataset"/>
                                 </sps:encoding>
                                 <sps:values xmlns:ns="urn:myorg:dataset">
                                 <ns:sceneType>FloatingScene</ns:sceneType>
                                 <ns:sceneStartStopTime>
                                    <ns:min>2010-01-01T11:55:00.00Z</ns:min>
                                    <ns:max>2010-01-01T12:00:00.00Z</ns:max>
                                  </ns:sceneStartStopTime>
                                 <ns:frame>7191</ns:frame>
                                 </sps:values>
                          </sps:ParameterData>
                    </sceneSelection>
                    <deliveryOptions>
                          <mediaDelivery>
                                 <packageMedium>CD-ROM</packageMedium>
                          </mediaDelivery>
                    </deliveryOptions>
                    <payment>
                          <paymentMethod>quota</paymentMethod>
                          <orderAccount>project_10000</orderAccount>
                    </payment>
                    <productId>

      <identifier>http://www.opengis.net/def/EOP/ESA/0/ESA.EECF.ENVISAT_ASA_IMx_xS/EN
1-05121921114570-3322.XI</identifier>
      <collectionId>http://www.opengis.net/def/EOP/ESA/0/ESA.EECF.ENVISAT_ASA_IMx_xS<
/collectionId>
                    </productId>
             </orderItem>
      </orderSpecification>
      <statusNotification>Final</statusNotification>
</Submit>
```

12.4 *Submit operation response*

Submit is an asynchronous operation where the client can define the frequency of notification to control the order status. When the request is sent, the server responses with a synchronous output message with the following mandatory information: the status of the request operation, and the orderId.

The synchronous output message is defined by the XML SubmitAck entity. The type of SubmitAck is SubmitProductOrderResponseType. The following figure provides a graphical representation of this type.

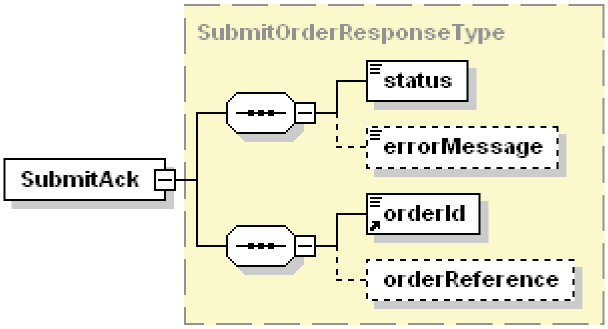


Figure 12-2 - SubmitAck diagram

12.4.1 Submit operation parameters

SubmitAck defines the acknowledgment to order submission. The following table presents the parameters of SubmitAck element.

Names	Definition	Data type and values	Multiplicity and use
status	Completion result of the operation: – success: operation successfully executed; – partial: non blocking errors occurred during processing of the request. If the Server error occurred the request will abort and it will be generated a SOAP-Fault.	Type: String Permitted Values: success, partial	One (mandatory)
errorMessage	Message to explain the reasons of partial result.	Type: Not empty string (max 255 chars).	One (optional)
orderId	Order identification number unique for this Provider.	Type: xs:anyURI	One (mandatory)
orderReference	Reference to the order (same message specified in the Submit request)	Type: String	One (optional)

Table 12-3 - SubmitAck description

12.4.2 Submit response XML encoding

A XML schema fragment for this operation encoded in XML

```
<xs:schema xmlns="http://www.opengis.net/oseo/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:sps="http://www.opengis.net/sps/2.0"
targetNamespace="http://www.opengis.net/oseo/1.0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
      <xs:import namespace="http://www.opengis.net/swe/2.0"
schemaLocation="../swe/sweCommon/2.0.0/swe.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsGetCapabilities.xsd"/>
      <xs:import namespace="http://www.opengis.net/sps/2.0"
schemaLocation="Order_spsCommon.xsd"/>
      <xs:import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="../gml/3.2.1/base/gml.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsExceptionReport.xsd"/>
…
<xs:element name="SubmitAck" type="SubmitOrderResponseType"/>
<xs:complexType name="SubmitOrderResponseType">
<xs:complexContent>
      <xs:extension base="OrderResponseBaseType">
            <xs:sequence>
                  <xs:element ref="orderId"/>
                  <xs:element name="orderReference" minOccurs="0"/>
            </xs:sequence>
      </xs:extension>
</xs:complexContent>
</xs:complexType>
…
</schema>
```

12.4.3 Submit response XML example

The following is an example of Submit Response (asynchronous).

Note that the example includes only the SubmitAck element without the SOAP envelope.

```
<SubmitAck xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd">
      <status>success</status>
      <orderId>urn:ESA:EECF:order_id_0001</orderId>
      <orderReference>example_0001</orderReference>
</SubmitAck>
```

12.4.4 Exceptions.

In the event that an Order Server encounters an error servicing a Submit, it shall return an exception within a SOAP Fault (in particular an ows:ExceptionReport according to the clause 8 of [NR9]).

The SOAP Fault Message (see §7.3.14) shall have one of the following exceptionCode value:

"exceptionCode" value	Meaning of code	"locator" value	"ExceptionText" value
MissingParameterValue	Operation request does not include a parameter value, and this server did not declare a default value for that parameter	Name of missing parameter	"Missing value for Parameter"
InvalidParameterValue	Operation request contains an invalid parameter value	Name of parameter with invalid value	"Invalid value for Parameter"
NoApplicableCode	No other exceptionCode specified by this service and server applies to this exception	None, omit "locator" parameter	"Code not applicable"
InvalidNotificationValue	In asynchronous mode, the operation request has the status notification active (with "All" or "Final" value) but an address information to which the server has to notify the ordering status that is empty or invalid.	"ws-address" parameter	"Invalid value for notification"
AuthenticationFailed	Invalid or missing identity information	"identity_token"	"Invalid or missing identity information"

“exceptionCode” value	Meaning of code	“locator” value	“ExceptionText” value
AuthorizationFailed	The client is not authorized to call the operation or it is not authorized to provide the specified parameters.	Possible values: □ “Submit” □ “collectionId” □ “identifier” □ “taskingRequestId” □ etc..	Text describing the item not authorized. E.g.: □ “The client is not authorized to call the operation.” □ “the client is not authorized to order that product” □ Etc.
SubscriptionNotSupported	Operation request identifies a order type subscription but the server doesn’t support “SubscriptionOrders”	“orderType” parameter	“Subscription not supported”
ProductOrderingNotSupported	Operation request identifies a order type order product but the server doesn’t support “ProductOrders”	“orderType” parameter	“Ordering not supported”
FutureProductNotSupported	Operation request identifies a order type future products but the server doesn’t support “ProgrammingOrders”	“orderType” parameter	“Programming not supported”

Table 12-4 – Exception info for Submit operation.

13 SubmitResponse operation

13.1 Introduction

Submit is an asynchronous operation, then if the server supports asynchronous notification, two operations are called:

- Submit, from client to server, for submitting the order.
  - This operation is composed of two messages:
    - Submit, it is the order submission request
    - SubmitAck, it is the acknowledge the server returns in real time to the request.
- SubmitResponse, from server to client, for sending the order notification to the client. Then this is a service to be implemented on the client.
  - This operation is composed of two messages:
    - SubmitResponse, sendig the order notification to the client;
    - SubmitResponseAck, just returning the successful receiption of the order notification.

This operation has to be implemented by the client (see §7.4.3).

13.2 Requirements

See §12.2.

13.3 SubmitResponse operation request

This operation has to be implemented by a client of Ordering service supporting asynchronous operations.

This operation allows the Ordering Service to send to the client notifications about the progress of submitted orders.

The following figure provides a graphical representation of this element.

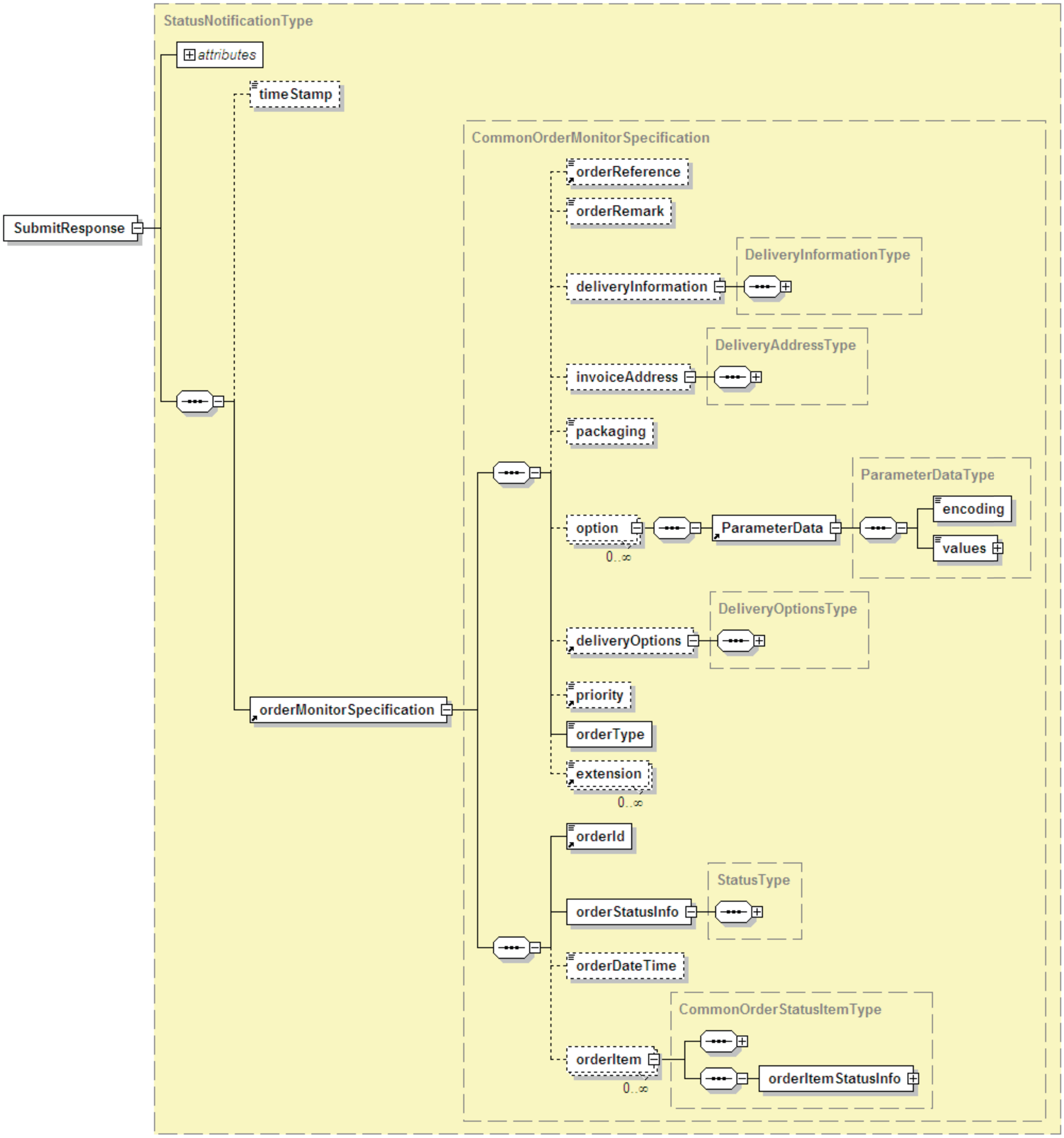


Figure 13-1 - SubmitResponse diagram.

13.3.1 SubmitResponse request parameters

The following table presents the parameters of the SubmitResponse.

Names	Definition	Data type and values	Multiplicity and use
service	Service type identifier	Type: non-empty string Allowed values: OS	One (mandatory)
version	Specification version for operation	Type: non-empty string. Format: <x>.<y>.<z>	One (mandatory)
timeStamp	It is the time when the request has been issued.	Type: xs:DateTime	One (optional)
orderMonitorSpecification	Order Monitor Specification element. The content of this element correspond to that returned by GetStatus with **full** presentation.	Type: CommonOrderMonitorSpecification (see §7.3.12.1)	One (mandatory)

Table 13-1 - SubmitResponse description.

13.3.2 SubmitResponse request XML encoding

A XML schema fragment for this operation encoded in XML

```
<xs:schema xmlns="http://www.opengis.net/oseo/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:sps="http://www.opengis.net/sps/2.0"
targetNamespace="http://www.opengis.net/oseo/1.0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
      <xs:import namespace="http://www.opengis.net/swe/2.0"
schemaLocation="../swe/sweCommon/2.0.0/swe.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsGetCapabilities.xsd"/>
      <xs:import namespace="http://www.opengis.net/sps/2.0"
schemaLocation="Order_spsCommon.xsd"/>
      <xs:import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="../gml/3.2.1/base/gml.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsExceptionReport.xsd"/>
…
<xs:element name="SubmitResponse" type="StatusNotificationType"/>
<xs:element name="SubmitResponseAck" type="StatusNotificationAckType">
      <xs:annotation>
      <xs:documentation>Response to acknowledge the receiption of order status
notification.</xs:documentation>
      </xs:annotation>
</xs:element>
       <xs:complexType name="StatusNotificationType">
      <xs:complexContent>
             <xs:extension base="OrderRequestBaseType">
             <xs:sequence>
             <xs:element name="timeStamp" type="xs:dateTime" minOccurs="0"/>
             <xs:element ref="orderMonitorSpecification"/>
             </xs:sequence>
             </xs:extension>
      </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="StatusNotificationAckType">
```

```
       <xs:complexContent>
              <xs:extension base="OrderResponseBaseType"/>
       </xs:complexContent>
</xs:complexType>
….
</schema>
```

13.3.3 SubmitResponse request example.

The following is an example of SubmitResponse request.

Note that the example includes only the SubmitResponse element without the SOAP envelope.

```
<SubmitResponse xmlns:sps="http://www.opengis.net/sps/2.0"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:gml="http://www.opengis.net/gml"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd" service="OS" version="1.0.0">
       <timeStamp>2006-11-18T21:08:00.000</timeStamp>
       <orderMonitorSpecification>
              <orderReference>example_0001</orderReference>
              <orderRemark>example</orderRemark>
              <deliveryInformation>
                     <mailAddress>
                            <recipient>Mr. John Smith</recipient>
                            <companyRef>DTMT</companyRef>
                            <postalAddress>
                                   <streetAddress>Esrin Esa No. 1</streetAddress>
                                   <city>Frascati</city>
                                   <state>-</state>
                                   <postalCode>00100</postalCode>
                                   <country>IT</country>
                                   <postBox>-</postBox>
                            </postalAddress>
                            <telephoneNumber>00390694180999</telephoneNumber>
                     </mailAddress>
              </deliveryInformation>
              <orderType>PRODUCT_ORDER</orderType>
              <orderId>123456-001</orderId>
              <orderStatusInfo>
                     <status>Completed</status>
                     <additionalStatusInfo>The order is
accomplished</additionalStatusInfo>
              </orderStatusInfo>
              <orderItem>
                     <itemId>item_0001</itemId>
                     <productOrderOptionsId>Level 1,Product SLC
(ASA_IMS)</productOrderOptionsId>
                     <orderItemRemark>First product</orderItemRemark>
                     <option>
                            <sps:ParameterData>
                                   <sps:encoding>
                                          <swe:XMLEncoding
defaultNamespace="http://www.opengis.net/oseo/1.0/dataset"/>
                                   </sps:encoding>
                                   <sps:values
xmlns:ns="http://www.opengis.net/oseo/1.0/dataset">
                                          <ns:processingLevel>1B</ns:processingLevel>
                                          <ns:productType>ASA_IMS_1P</ns:productType>

       <ns:qualityOfService>STANDARD</ns:qualityOfService>
```

```
                                   </sps:values>
                            </sps:ParameterData>
                     </option>
                     <sceneSelection>
                            <sps:ParameterData>
                                   <sps:encoding>
                                          <swe:XMLEncoding
defaultNamespace="http://www.opengis.net/oseo/1.0/dataset"/>
                                   </sps:encoding>
                                   <sps:values
xmlns:ns="http://www.opengis.net/oseo/1.0/dataset">
                                   <ns:sceneType>FloatingScene</ns:sceneType>
                                   <ns:sceneStartStopTime>
                                      <ns:min>2010-01-01T11:55:00.00Z</ns:min>
                                      <ns:max>2010-01-01T12:00:00.00Z</ns:max>
                                    </ns:sceneStartStopTime>
                                   <ns:frame>7191</ns:frame>
                                   </sps:values>
                            </sps:ParameterData>
                     </sceneSelection>
                     <deliveryOptions>
                            <mediaDelivery>
                                   <packageMedium>CD-ROM</packageMedium>
                            </mediaDelivery>
                     </deliveryOptions>
                     <payment>
                            <paymentMethod>quota</paymentMethod>
                            <orderAccount>project_10000</orderAccount>
                     </payment>
                     <productId>

       <identifier>http://www.opengis.net/def/EOP/ESA/0/ESA.EECF.ENVISAT_ASA_IMx_xS/EN
1-05121921114570-3322.XI</identifier>

       <collectionId>http://www.opengis.net/def/EOP/ESA/0/ESA.EECF.ENVISAT_ASA_IMx_xS<
/collectionId>
                     </productId>
              </orderItem>
       </orderMonitorSpecification>
</SubmitResponse>
```

13.4 *SubmitResponse operation response*

The following figure provides a graphical representation of this element.

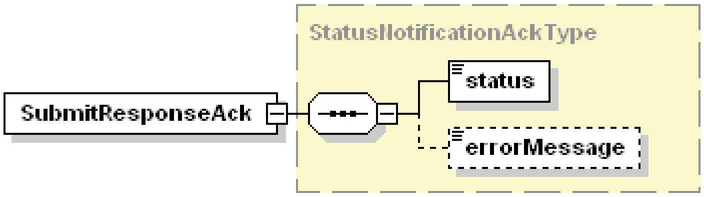


Figure 13-2 - SubmitResponseAck diagram.

13.4.1 SubmitResponse response parameters

The following table describes the SubmitResponseAck element.

Names	Definition	Data type and values	Multiplicity and use
status	Status of reception of SubmitResponse message. If a blocking error occurred, the request will abort and it will be generated a SOAP-Fault.	Type: String Permitted Values: success, partial	One (mandatory)
errorMessage	Message to explain a no blocking errors	Type: Not empty string (max 255 chars).	One (optional)

Table 13-2 - SubmitResponseAck description.

13.4.2 SubmitResponse response XML encoding

See §13.3.2

13.4.3 SubmitResponse response example

The following is an example of SubmitResponse response.

Note that the example includes only the SubmitResponseAck element without the SOAP envelope.

```
<SubmitResponseAck xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd">
      <status>success</status>
</SubmitResponseAck>
```

13.4.4 Exceptions

This operation has to be implemented by a Client of Ordering service**.**

In the event that an Order Server encounters an error servicing a SubmitResponse, it shall return an exception within a SOAP Fault (in particular an ows:ExceptionReport according to the clause 8 of [NR9]).

The SOAP Fault Message (see §7.3.14) shall have one of the following exceptionCode value:

"exceptionCode" value	Meaning of code	"locator" value	"ExceptionText" value
MissingParameterValue	Operation request does not include a parameter value, and this server did not declare a default value for that parameter	Name of missing parameter	"Missing value for Parameter"
InvalidParameterValue	Operation request contains an invalid parameter value	Name of parameter with invalid value	"Invalid value for Parameter"

“exceptionCode” value	Meaning of code	“locator” value	“ExceptionText” value
NoApplicableCode	No other exceptionCode specified by this service and server applies to this exception	None, omit “locator” parameter	“Code not applicable”
AuthenticationFailed	Invalid or missing identity information	“identity_token”	“Invalid or missing identity information”
AuthorizationFailed	The client is not authorized to call the operation.	None, omit “locator” parameter	Text describing the item not authorized.

Table 13-3 – Exception codes for SubmitResponse operation.

14 GetStatus Operation

14.1 Introduction

This operation is in charge of returning the status of submitted orders. It can be used in different ways:

- **Order search**

  In this way the operation returns all orders matching the filtering criteria:

  - Last update: all orders updated after the specified date are returned;
  - Order status: all orders having the specified status are returned;
  - Order Reference: the orders having that reference string are returned.

- **Order retrieve**

  Only the order matching the order identifier is returned.

The amount of returned information depends on the presentation:

- brief: only order level information are returned (no order items returned);
- full: whole information returned.

Both presentation values can be used for order search and order retrieve, but for the sake of efficiency the following usage is recommended:

- brief presentation to be used for order search;
- full presentation to be used for getting all details of some of the orders returned by the order search.

14.2 Requirements

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 79	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus	Core	The Order Server shall implement the GetStatus operation.
Req 80	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/req/schema	Core	The GetStatus request shall consist of an XML instance document as validated by the entity GetStatus in the oseo.xsd XML Schema.
Req 81	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/req/id	Core	The Order Server shall accept GetStatus request by order identifier: GetStatus/orderId
Req 82	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/req/search	Core	The Order Server shall accept order search requests: GetStatus/filteringCriteria
Req 83	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/req/search/lastUpdate	Core	The Order Server shall support the following order search criteria: □ GetStatus/filteringCriteria/lastUpdate □ GetStatus/filteringCriteria/lastUpdateEnd

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 84	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/req/search/orderStatus	Core	The Order Server shall support the following order search criteria: □ GetStatus/filteringCriteria/orderStatus (multiple conditions)
Req 85	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/req/search/orderReference	Core	The Order Server shall support the following order search criteria: □ GetStatus/filteringCriteria/orderReference
Req 86	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/req/presentation_brief	Core	The Order Server shall accept GetStatus requests with brief presentation: GetStatus/presentation = brief
Req 87	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/req/presentation_full	Core	The Order Server shall accept GetStatus requests with full presentation: GetStatus/presentation = full
Req 88	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/resp/schema	Core	The response to a successful GetStatus request shall consist of an XML instance document as validated by the entity GetStatusResponse in the oseo.xsd XML Schema.
Req 89	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/resp/id	Core	The response to a successful GetStatus by order identifier request shall return: □ All the order parameters specified in Submit operation. □ Plus the status of the specified order (regardless the order type).
Req 90	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/resp/id/exception	Core	If no order is found for a specified order identifier (GetStatus/orderId) the Order Server shall return a SOAP Fault.
Req 91	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/resp/search	Core	The response to a successful GetStatus by order search shall return the order monitoring info of all orders (regardless the order type) matching the search criteria. The order monitoring info includes: □ All the orders parameters specified in Submit operation. □ Plus the status of the orders (regardless the order type).
Req 92	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/resp/search/empty	Core	If no order is found when executing a GetStatus by order search then no orderMonitorSpecification element is returned in GetStatusResponse (no SOAP Fault shall be returned in this situation).
Req 93	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/resp/brief	Core	The response to a successful GetStatus request with BRIEF presentation shall return (regardless the order type): □ order identifier and the overall order monitoring information. □ No order items (GetStatusResponse/ orderMonitorSpecification orderItem).

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 94	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/resp/full	Core	The response to a successful GetStatus request with FULL presentation shall return (regardless the order type) the information returned in BRIEF plus: □ The status info all order items (GetStatusResponse/ orderMonitorSpecification orderItem)
Req 95	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/resp/date_time	Core	The response to a successful GetStatus request shall return (regardless the order type) the order date & time (GetStatusResponse/ orderMonitorSpecification/orderDateTime).
Req 96	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/resp/completed	Core	The Order Server shall return GetStatusResponse including: For all items ready: GetStatusResponse/orderMonitorSpecification/orderItem /orderItemStatusInfo/status = Completed
Req 97	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/resp/not_complete	Core	The Order Server shall return GetStatusResponse including: For all items not yet ready: GetStatusResponse/orderMonitorSpecification/orderItem /orderItemStatusInfo/status = Submitted or Accepted or InProduction
Req 98	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/resp/sort	Core	In case more GetStatusResponse/orderMonitorSpecification elements are returned, then they shall besorted by orderDateTime.
Req 99	http://www.opengis.net/spec/OSEO/1.0/req/Core/GetStatus/resp/exception	Core	When a Order Server encounters an error while performing a GetStatus operation, it shall return an ows: ExceptionReport according to the clause 8 of [NR9].
Req 100	http://www.opengis.net/spec/OSEO/1.0/req/ProductOrder/GetStatus/resp	ProductOrder	When a ProductOrder is returned in GetStatusResponse, the following elements shall be included: GetStatusResponse/orderMonitorSpecification/orderType = ”PRODUCT_ORDER” And including at least one Earth Observation product: GetStatusResponse/orderMonitorSpecification/orderItem/productId
Req 101	http://www.opengis.net/spec/OSEO/1.0/req/ProductOrder/GetStatus/resp	SceneSelection	When a ProductOrder with scene selection is returned in GetStatusResponse, at least one istance of: GetStatusResponse/orderMonitorSpecification/orderItem/sceneSelection shall be included.

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 102	http://www.opengis.net/spec/OSEO/1.0/req/SubscriptionOrder/GetStatus/resp	SubscriptionOrder	When a SubscriptionOrder is returned in GetStatusResponse, the following elements shall be included: GetStatusResponse/orderMonitorSpecification/orderType = ”SUBSCRIPTION_ORDER” and including one Subscription to Earth Observation products: GetStatusResponse/orderMonitorSpecification/orderItem/subscriptionId
Req 103	http://www.opengis.net/spec/OSEO/1.0/req/TaskingOrder/GetStatus/resp	TaskingOrder	When a TaskingOrder is returned in GetStatusResponse, the following elements shall be included: GetStatusResponse/orderMonitorSpecification/orderType = ”TASKING_ORDER” and including one Tasking Request: GetStatusResponse/orderMonitorSpecification/orderItem/taskingRequestId

Table 14-1 - GetStatus requirements.

14.3 GetStatus operation request

The type of GetStatus is GetStatusRequestType. The following figure provides a graphical representation of this type.

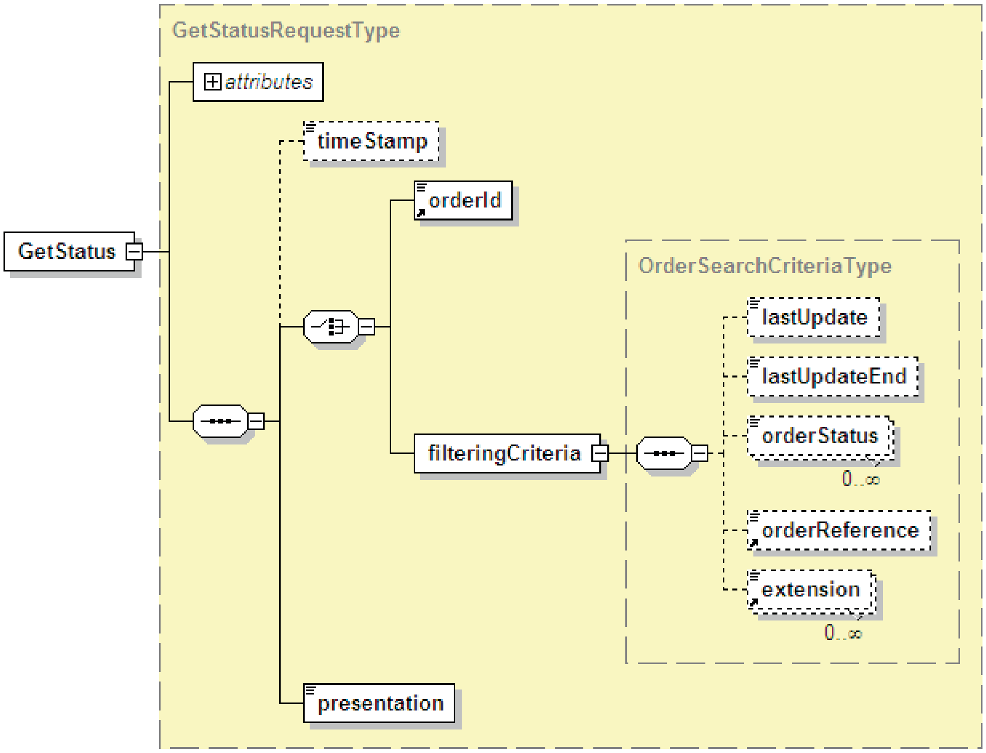


Figure 14-1 - GetStatus diagram.

14.3.1 GetStatus request parameters

GestStatus request uses an OrderMonitorRequest element. The following table presents the parameters that GetStatus uses within the request.

Names	Definition	Data type and values	Multiplicity and use
service	Service type identifier	Type: non-empty string Allowed values: OS	One (mandatory)
version	Specification version for operation	Type: non-empty string. Format: <x>.<y>.<z>	One (mandatory)
timeStamp	It is the time when the request has been issued.	Type: xs:DateTime	One (optional)
orderId	Identifier of the order to retrieve. This field is alternative to 'filteringCriteria'	Type: xs:anyURI	One (mandatory/choise)

Names	Definition	Data type and values	Multiplicity and use
filteringCriteria	Allows searching submitted orders. Supported search criteria are: – Last update: only orders which status has been changed after the specified date and time are returned – orderStatus: only orders having the specified statuses are returned. – orderReference: only the orders having the specified order reference are returned. This field is alternative to 'orderId'	Type: OrderSearchCriteriaType	One (mandatory/choise)
lastUpdate	It allows to retrieve the orders which lastUpdate is greater equal than it. If not present there is not such limitation.	Type: Type: date in ISO 8601 format (CCYY-MM-DD)	One (optional)
lastUpdateEnd	It allows to retrieve the orders which lastUpdate is less equal than it. If not present there is not such limitation.	Type: Type: date in ISO 8601 format (CCYY-MM-DD)	One (optional)
orderStatus	Status of an order. (see Table 7-27)	Type: String	Zero or more(optional)
orderReference	User defined string specified when the order has been submitted.	Type: String	One (optional)
extension	This element allows to specify additional implementation specific filtering parameters.	Type: anyType	One (optional)
presentation	This element specifies the amount of information to be returned by the GetStatus operation: – brief: only information about the whole order are returned. No order items are returned. – full: the whole order information are returned.	Type: enumerated string Permitted values: brief, full	One (mandatory)

Table 14-2 - GetStatus description

14.3.2 GetStatus request XML encoding

A XML schema fragment for this operation encoded in XML

```
<xs:schema xmlns="http://www.opengis.net/oseo/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:sps="http://www.opengis.net/sps/2.0"
targetNamespace="http://www.opengis.net/oseo/1.0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
	<xs:import namespace="http://www.opengis.net/swe/2.0"
schemaLocation="../swe/sweCommon/2.0.0/swe.xsd"/>
	<xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsGetCapabilities.xsd"/>
```

```
      <xs:import namespace="http://www.opengis.net/sps/2.0"
schemaLocation="Order_spsCommon.xsd"/>
      <xs:import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="../gml/3.2.1/base/gml.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsExceptionReport.xsd"/>
…
<xs:element name="GetStatus" type="GetStatusRequestType"/>
<xs:complexType name="GetStatusRequestType">
      <xs:complexContent>
             <xs:extension base="OrderRequestBaseType">
             <xs:sequence>
             <xs:element name="timeStamp" type="xs:dateTime" minOccurs="0"/>
             <xs:choice>
             <xs:element ref="orderId"/>
             <xs:element name="filteringCriteria" type="OrderSearchCriteriaType"/>
             </xs:choice>
             <xs:element name="presentation" type="PresentationType"/>
             </xs:sequence>
             </xs:extension>
      </xs:complexContent>
</xs:complexType>
….
</schema>
```

14.3.3 GetStatus request example

The following is an example of GetStatus request.

Note that the example includes only the GetStatus element without the SOAP envelope.

```
<GetStatus xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd" service="OS" version="1.0.0">
      <orderId>urn:ESA:EECF:order_id_0001</orderId>
      <presentation>full</presentation>
</GetStatus>
```

14.4 GetStatus operation response

The type of GetStatusResponse is GetStatusResponseType. The following figure provides a graphical representation of this type.

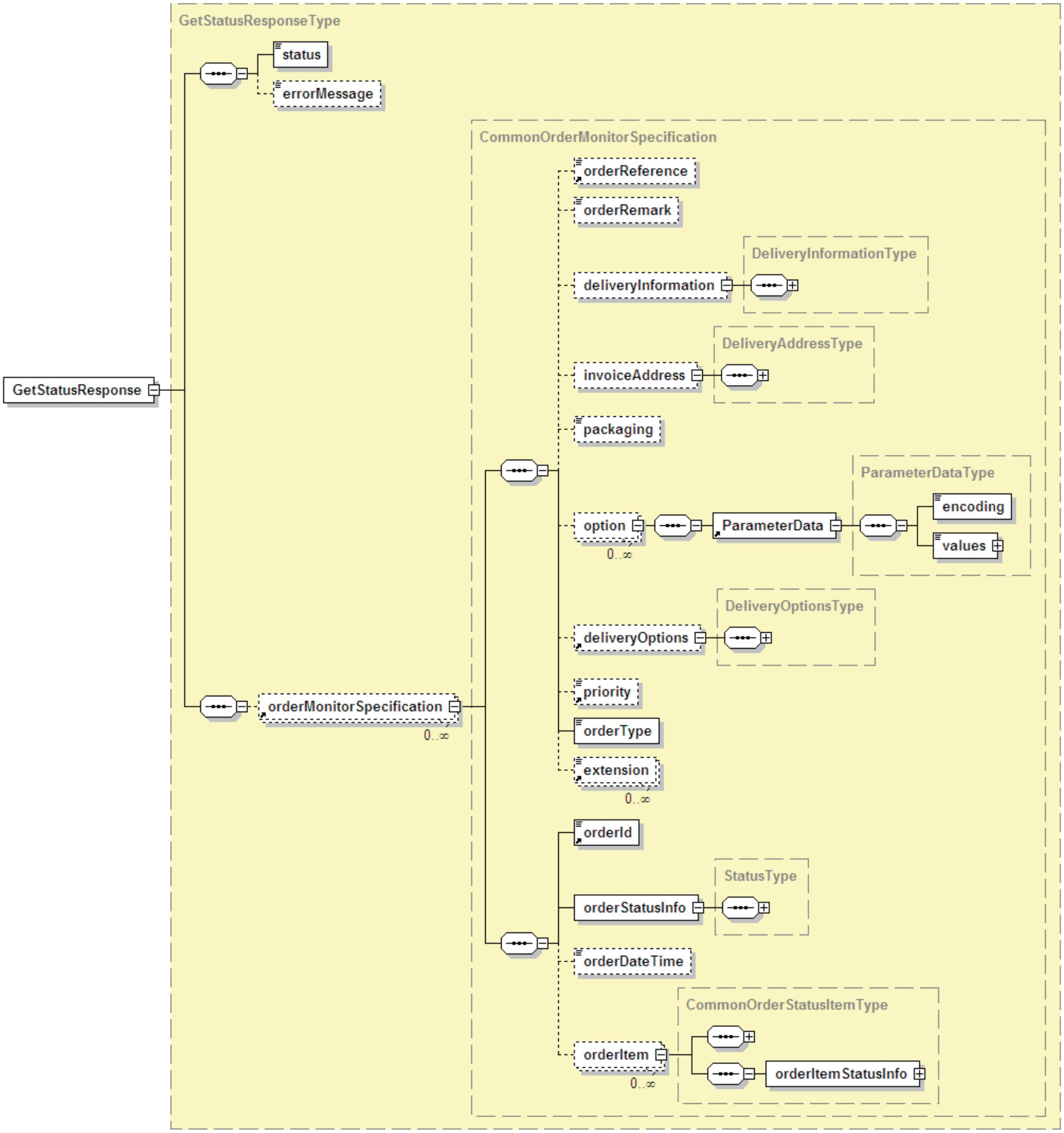


Figure 14-2 - GetStatusResponse element diagram.

14.4.1 GetStatus response parameters

The following parameters are used both for Product Ordering and for Subscription.GetStatusResponse defines OrderMonitorResponse element for submitted order.

Names	Definition	Data type and values	Multiplicity and use
status	Completion result of the operation: – success: operation successfully executed; – partial: some error occurred during the processing of the request which lead to an incomplete response. If the blocking error occurred the request will abort and it will be generated a SOAP-Fault.	Type: String Permitted Values: success, partial	One (mandatory)
errorMessage	Message to explain the reasons of the partial result.	Type: Not empty string (max 255 chars).	One (optional)
orderMonitorSpecification	Order Monitor Specification element. The amount of returned information depends on the presentation specified in the request message. If more istances are returned, then they have to be sorted by orderDateTime.	Type: CommonOrderMonitorSpecification (see §7.3.12.1)	Zero or More (optional)

Table 14-3 - GetStatusResponse description

14.4.2 GetStatus response XML encoding

A XML schema fragment for this operation encoded in XML

```
<xs:schema xmlns="http://www.opengis.net/oseo/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:sps="http://www.opengis.net/sps/2.0"
targetNamespace="http://www.opengis.net/oseo/1.0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
      <xs:import namespace="http://www.opengis.net/swe/2.0"
schemaLocation="../swe/sweCommon/2.0.0/swe.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsGetCapabilities.xsd"/>
      <xs:import namespace="http://www.opengis.net/sps/2.0"
schemaLocation="Order_spsCommon.xsd"/>
      <xs:import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="../gml/3.2.1/base/gml.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsExceptionReport.xsd"/>
…
<xs:element name="GetStatusResponse" type="GetStatusResponseType"/>
<xs:complexType name="GetStatusResponseType">
<xs:complexContent>
<xs:extension base="OrderResponseBaseType">
      <xs:sequence>
      <xs:element ref="orderMonitorSpecification" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
```

```
</xs:extension>
</xs:complexContent>
</xs:complexType>
….
</schema>
```

14.4.3 GetStatus response example

The following is an example of GetStatus response.

Note that the example includes only the GetStatusResponse element without the SOAP envelope.

```
<GetStatusResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0" xmlns:sps="http://www.opengis.net/sps/2.0"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:gml="http://www.opengis.net/gml"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd">
      <status>success</status>
      <orderMonitorSpecification>
            <orderReference>example_0001</orderReference>
            <orderRemark>example</orderRemark>
            <deliveryInformation>
                  <mailAddress>
                        <recipient>Mr. John Smith</recipient>
                        <companyRef>DTMT</companyRef>
                        <postalAddress>
                              <streetAddress>Esrin Esa No. 1</streetAddress>
                              <city>Frascati</city>
                              <state>-</state>
                              <postalCode>00100</postalCode>
                              <country>IT</country>
                              <postBox>-</postBox>
                        </postalAddress>
                        <telephoneNumber>00390694180999</telephoneNumber>
                  </mailAddress>
            </deliveryInformation>
            <orderType>PRODUCT_ORDER</orderType>
            <orderId>123456-001</orderId>
            <orderStatusInfo>
                  <status>Completed</status>
                  <additionalStatusInfo>The order is
accomplished</additionalStatusInfo>
            </orderStatusInfo>
            <orderItem>
                  <itemId>item_0001</itemId>
                  <productOrderOptionsId>Level 1,Product SLC
(ASA_IMS)</productOrderOptionsId>
                  <orderItemRemark>First product</orderItemRemark>
                  <option>
                        <sps:ParameterData>
                              <sps:encoding>
                                    <swe:XMLEncoding
defaultNamespace="http://www.opengis.net/oseo/1.0/dataset"/>
                              </sps:encoding>
                              <sps:values
xmlns:ns="http://www.opengis.net/oseo/1.0/dataset">
                                    <ns:processingLevel>1B</ns:processingLevel>
                                    <ns:productType>ASA_IMS_1P</ns:productType>

      <ns:qualityOfService>STANDARD</ns:qualityOfService>
                              </sps:values>
                        </sps:ParameterData>
                  </option>
                  <sceneSelection>
                        <sps:ParameterData>
                              <sps:encoding>
                                    <swe:XMLEncoding
defaultNamespace="http://www.opengis.net/oseo/1.0/dataset"/>
                              </sps:encoding>
                              <sps:values
xmlns:ns="http://www.opengis.net/oseo/1.0/dataset">
                              <ns:sceneType>FloatingScene</ns:sceneType>
                              <ns:sceneStartStopTime>
                                <ns:min>2010-01-01T11:55:00.00Z</ns:min>
                                <ns:max>2010-01-01T12:00:00.00Z</ns:max>
                               </ns:sceneStartStopTime>
                              <ns:frame>7191</ns:frame>
                              </sps:values>
                        </sps:ParameterData>
                  </sceneSelection>
                  <deliveryOptions>
                  <mediaDelivery>
                        <packageMedium>CD-ROM</packageMedium>
                  </mediaDelivery>
                  </deliveryOptions>
                  <payment>
                        <paymentMethod>quota</paymentMethod>
                        <orderAccount>project_10000</orderAccount>
                  </payment>
                  <productId>

      <identifier>http://www.opengis.net/def/EOP/ESA/0/ESA.EECF.ENVISAT_ASA_IMx_xS/EN
1-05121921114570-3322.XI</identifier>

      <collectionId>http://www.opengis.net/def/EOP/ESA/0/ESA.EECF.ENVISAT_ASA_IMx_xS<
/collectionId>
                  </productId>
                  <orderItemStatusInfo>
                        <status>Completed</status>
                  </orderItemStatusInfo>
            </orderItem>
      </orderMonitorSpecification>
</GetStatusResponse>
```

14.4.4 Exceptions

In the event that an Order Server encounters an error servicing a GetStatus, it shall return an exception within a SOAP Fault (in particular an ows:ExceptionReport according to the clause 8 of [NR9]).

The SOAP Fault Message (see §7.3.14) shall have one of the following exceptionCode value:

"exceptionCode" value	Meaning of code	"locator" value	"ExceptionText" value
MissingParameterValue	Operation request does not include a parameter value, and this server did not declare a default value for that parameter	Name of missing parameter	"Missing value for Parameter"
InvalidParameterValue	Operation request contains an invalid parameter value	Name of parameter with invalid value	"Invalid value for Parameter"

"exceptionCode" value	Meaning of code	"locator" value	"ExceptionText" value
NoApplicableCode	No other exceptionCode specified by this service and server applies to this exception	None, omit "locator" parameter	"Code not applicable"
AuthenticationFailed	Invalid or missing identity information	"identity_token"	"Invalid or missing identity information"
AuthorizationFailed	The client is not authorized to call the operation or it is not authorized to provide the specified parameters.	Possible values: □ "GetStatus" □ "orderId" □ "filteringCriteria"	Text describing the item not authorized. E.g.: □ "The client is not authorized to call the operation." □ "the client is not authorized to search for orders" □ Etc.
InvalidPresentationValue	Operation request contains a type of presentation that the service doesn't support. Allowed values are "brief" or "full".	"presentation" parameter	"Invalid value for presentation"
InvalidFilteringValue	Operation request contains a type of search that the service doesn't support.	"filteringCriteria" parameter	"Invalid value for filtering"
InvalidOrderIdentifier	Operation request contains an invalid order identifier.	"orderId" parameter	"Invalid value for order"
TooManyHits	Order search returning too many hits	None, omit "locator" parameter	"Too many hits"

Table 14-4 – Exception codes for GetStatus operation.

15 DescribeResultAccess operation

15.1 Introduction

This operation is in charge of returning the URL of products ordered specifying on-line delivery. It can be used in two different ways:

- **All ready items**

  In this way the operation returns the URLs of all items that are ready for download at the time the operation is called.

- **Last ready items**

  In this way the operation returns the URLs of all items that are ready for download since the last call.

15.2 Requirements

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 104	http://www.opengis.net/spec/OSEO/1.0/req/OnlineDataAccess/DescribeResultAccess	OnlineDataAccess	The Order Server shall implement the DescribeResultAccess operation.
Req 105	http://www.opengis.net/spec/OSEO/1.0/req/OnlineDataAccess/DescribeResultAccess/req/schema	OnlineDataAccess	The DescribeResultAccess request shall consist of an XML instance document validated by the entity DescribeResultAccess in the oseo.xsd XML Schema.
Req 106	http://www.opengis.net/spec/OSEO/1.0/req/OnlineDataAccess/DescribeResultAccess/req/allReady	OnlineDataAccess	The Order Server shall accept DescribeResultAccess request including: DescribeResultAccess/subFunction = AllReady
Req 107	http://www.opengis.net/spec/OSEO/1.0/req/OnlineDataAccess/DescribeResultAccess/req/nextReady	OnlineDataAccess	The Order Server shall accept DescribeResultAccess request including: DescribeResultAccess/subFunction = nextReady
Req 108	http://www.opengis.net/spec/OSEO/1.0/req/OnlineDataAccess/DescribeResultAccess/resp/schema	OnlineDataAccess	The response to a successful DescribeResultAccess request shall consist of an XML instance document as validated by the entity DescribeResultAccessResponse in the oseo.xsd XML Schema.

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 109	http://www.opengis.net/spec/OSEO/1.0/req/OnlineDataAccess/DescribeResultAccess/resp/allReady	OnlineDataAccess	The response to a successful DescribeResultAccess request with subFunction = AllReady shall return one instance of: DescribeResultAccessResponse/URLs for each ordered item which is ready for download. Comment: In general the ordered items are made available for download one by one when they have been processed by the Order Server. A call specifying “AllReady” returns all the items that are ready for download at the time of this call. The operation can return one or more or all ordered items depending on the status of the order.
Req 110	http://www.opengis.net/spec/OSEO/1.0/req/OnlineDataAccess/DescribeResultAccess/resp/nextReady	OnlineDataAccess	The response to a successful DescribeResultAccess request with subFunction = nextReady shall return one instance of: DescribeResultAccessResponse/URLs for each ordered item which is ready for download since the last call. Comment: In general the ordered items are made available for download one by one when they have been processed by the Order Server. A call specifying “nextReady” returns all the items that have been made available for download since the previous call to this operation. Then this type of call returns the “delta” URLs between different calls.
Req 111	http://www.opengis.net/spec/OSEO/1.0/req/OnlineDataAccess/DescribeResultAccess/exception	OnlineDataAccess	When a Order Server encounters an error while performing a DescribeResultAccessResponse operation, it shall return an ows: ExceptionReport according to the clause 8 of [NR9].
Req 112	http://www.opengis.net/spec/OSEO/1.0/req/OnlineDataAccess/DescribeResultAccess/resp/expiration	OnlineDataAccess	If the returned URLs (DescribeResultAccessResponse/URLs) are available on-line for a limited time window then the Order Server shall return DescribeResultAccessResponse/URLs/expirationDate specifying the date & time the item will no longer be available. If there is no limitation on time availability then the element shall be left empty.

Table 15-1: DescribeResultAccess requirements.

15.3 DescribeResultAccess operation request

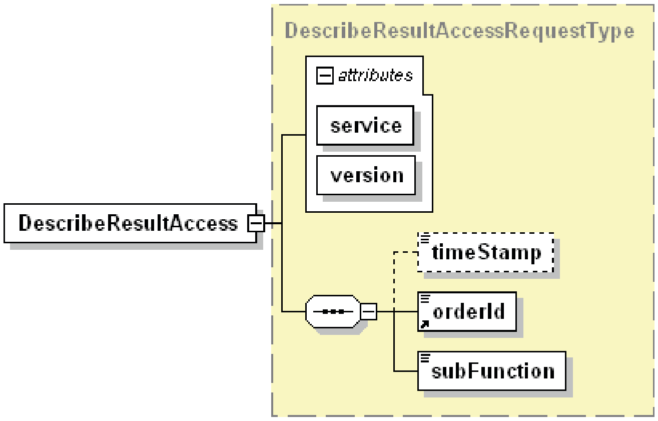


Figure 15-1 - DescribeResultAccess diagram.

15.3.1 DescribeResultAccess request parameters

The following table defines the information within the DescribeResultAccess request.

Names	Definition	Data type and values	Multiplicity and use
service	Service type identifier	Type: non-empty string Allowed values: OS	One (mandatory)
version	Specification version for operation	Type: non-empty string. Format: <x>.<y>.<z>	One (mandatory)
timeStamp	It is the time when the request has been issued.	Type: xs:DateTime	One (optional)
orderId	Order identification number unique for this Provider.	Type: xs: anyURI	One (mandatory)
subFunction	It is an enumerative string specifying the precise behaviour of the operation.	Type: String Permitted values: – allReady Flag indicating if all the currently completed items are to be retrieved. Calling DescribeResultAccess with this flag before at least one Item is in status "Completed" will return an empty list of URL(s). – nextReady Flag indicating if all the completed items since last call (or from the beginning of the processing anyway) are to be retrieved.	One (mandatory)

Table 15-2 - DescribeResultAccess description.

15.3.2 DescribeResultAccess request XML encoding

A XML schema fragment for this operation encoded in XML :

```
<xs:schema xmlns="http://www.opengis.net/oseo/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:sps="http://www.opengis.net/sps/2.0"
targetNamespace="http://www.opengis.net/oseo/1.0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
	<xs:import namespace="http://www.opengis.net/swe/2.0"
schemaLocation="../swe/sweCommon/2.0.0/swe.xsd"/>
	<xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsGetCapabilities.xsd"/>
	<xs:import namespace="http://www.opengis.net/sps/2.0"
schemaLocation="Order_spsCommon.xsd"/>
	<xs:import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="../gml/3.2.1/base/gml.xsd"/>
	<xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsExceptionReport.xsd"/>
…
<xs:element name="DescribeResultAccess" type="DescribeResultAccessRequestType"/>
<xs:complexType name="DescribeResultAccessRequestType">
	<xs:complexContent>
	<xs:extension base="OrderRequestBaseType">
		<xs:sequence>
		<xs:element name="timeStamp" type="xs:dateTime" minOccurs="0"/>
		<xs:element ref="orderId"/>
		<xs:element name="subFunction">
			<xs:simpleType>
				<xs:restriction base="xs:string">
					<xs:enumeration value="allReady"/>
					<xs:enumeration value="nextReady"/>
				</xs:restriction>
			</xs:simpleType>
		</xs:element>
		</xs:sequence>
	</xs:extension>
</xs:complexContent>
</xs:complexType>
….
</schema>
```

15.3.3 DescribeResultAccess request example

The following is an example of DescribeResultAccess request.

Note that the example includes only the DescribeResultAcces element without the SOAP envelope.

```
<DescribeResultAccess xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd " service="OS" version="1.0.0">
	<orderId>urn:ESA:EECF:order_id_0001</orderId>
	<subFunction>allReady</subFunction>
</DescribeResultAccess>
```

15.4 DescribeResultAccess operation response

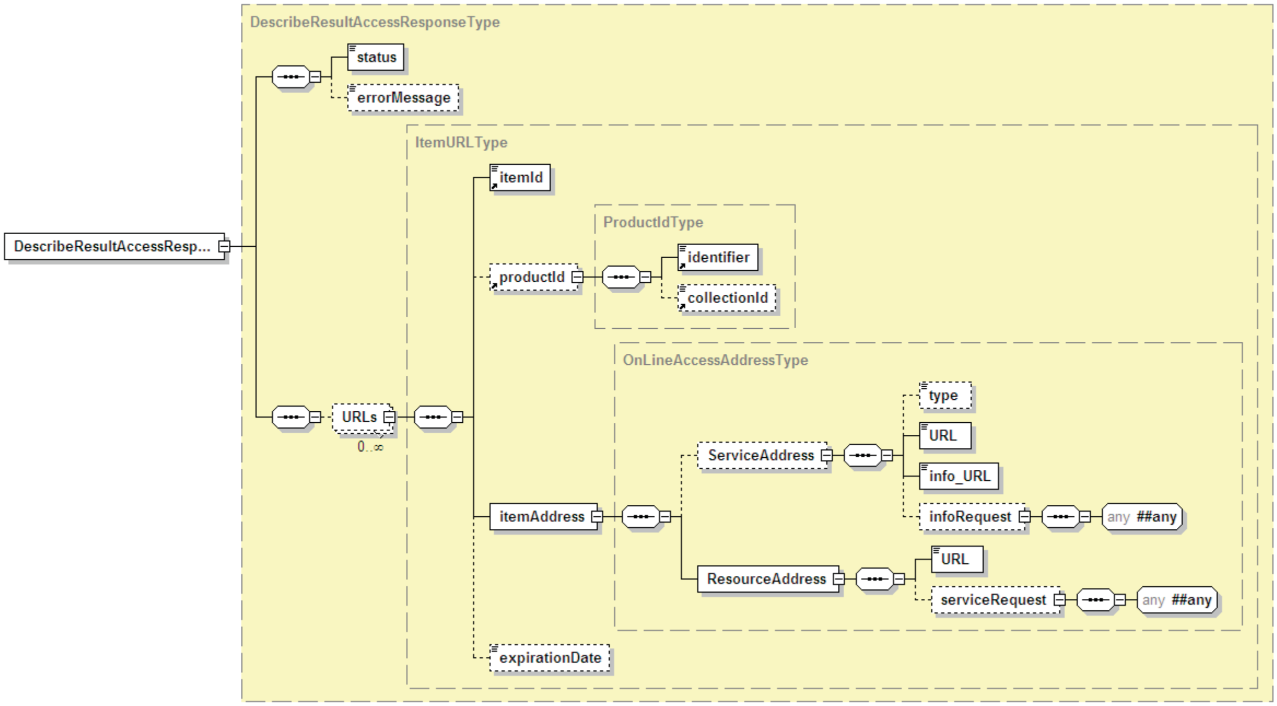


Figure 15-2 - DescribeResultAccessResponse diagram.

15.4.1 DescribeResultAccess response parameters

The following table presents all parameters within a DescribeResultAccess response.

Names	Definition	Data type and values	Multiplicity and use
status	Completion result of the operation: – success: operation successfully executed; – partial: some error occurred during the processing of the request which lead to an incomplete response. If a blocking error occurred the request will be aborted and it will be generated a SOAP-Fault.	Type: String Permitted Values: success, partial	One (mandatory)
errorMessage	Message to explain the reasons of the partial result.	Type: Not empty string (max 255 chars).	One (optional)
URLs		Type: ItemURLType (see Table 15-4)	Zero or more (optional)

Table 15-3 - DescribeResultAccessResponse description.

The following table presents the entity ItemURLType description used within DescribeResultAccess response.

Names	Definition	Data type and values	Multiplicity and use
itemId	Order item identifier specified in the Submit request.	Type: non-empty string (max 80 chars)	One (mandatory)
productId	Identifier of the product the URL is related to.	Type: ProductIdType (see Table 7-23)	One (optional)
itemAddress	This is the address of a single “item” to be retrieved; according to the settings of “packaging” in Submit an “item” might be the whole result of the Submit operation or a subset. If “allReady” flag is set to true, this list will contain the URL(s) of all the items currently available: if there are still items being processed these will NOT be available. According to the settings of packaging flag in Submit this might yield a single item or a bunch of items (packaging=None). If “nextReady” flag is set to true, this list will contain all the products available since last call (or from the beginning of the processing if called for the first time). According to the settings of packaging flag in Submit, this might yield a single item or a bunch of items (packaging=None).	Type: OnLineAccessAddressType (see Table 15-5)	One (mandatory)
expirationDate	Date and time at which the URL will expire.	Type: xs:dateTime	One (optional)

Table 15-4 - ItemURLType description.

The following table presents the entity OnLineAccessAddressType description used within ItemURLType entity. OnLineAccessAddressType defines the full information for accessing an on-line ordered item. In particular it provides information about the ordered resource and also about the server that host it.

Names	Definition	Data type and values	Multiplicity and use

Names	Definition	Data type and values	Multiplicity and use
ServiceAddress	This element provides full information about the server hosting the order item. This element is useful e.g. in case of items ordered through a WCS / WMS / etc.	Type: ServiceAddress	One (optional)
type	This field specifies the type of the service hosting the ordered item e.g.: WCS, WMS, etc.	Type String	One (optional)
URL	URL of the server hosting the ordered item.	Type: anyURI	One (mandatory)
info_URL	URL of a document providing metadata about the server. In case of OGC Web Services, it refers to the GetCapabilities operation (HTTP GET, KVP binding).	Type: anyURI	One (mandatory)
infoRequest	In case the server information are returned via HTTP POST request, this element specifies the message to send (e.g. in case the service supports GetCapabilities with SOAP binding, then the GetCapabilities XML message is specified in this tag)	Type: ##any	One (optional)
ResourceAddress	This element provides full information for accessing directly the ordered item.	Type:ResourceAddress	One (mandatory)
URL	URL of the ordered item.	Type: anyURI	One (mandatory)
serviceRequest	In case the ordered item is accessible via HTTP POST protocol, this element specifies the message to send for getting the ordered item.	Type: ##any	One (optional)

Table 15-5 - OnLineAccessAddressType description.

15.4.2 DescribeResultAccess response XML encoding

A XML schema fragment for this operation encoded in XML

```
<xs:schema xmlns="http://www.opengis.net/oseo/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:sps="http://www.opengis.net/sps/2.0"
targetNamespace="http://www.opengis.net/oseo/1.0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
      <xs:import namespace="http://www.opengis.net/swe/2.0"
schemaLocation="../swe/sweCommon/2.0.0/swe.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsGetCapabilities.xsd"/>
      <xs:import namespace="http://www.opengis.net/sps/2.0"
schemaLocation="Order_spsCommon.xsd"/>
      <xs:import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="../gml/3.2.1/base/gml.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsExceptionReport.xsd"/>
…
<xs:element name="DescribeResultAccessResponse"
type="DescribeResultAccessResponseType"/>
```

```
<xs:complexType name="DescribeResultAccessResponseType">
      <xs:complexContent>
      <xs:extension base="OrderResponseBaseType">
      <xs:sequence>
      <xs:element name="URLs" type="ItemURLType" minOccurs="0"
maxOccurs="unbounded"/>
      </xs:sequence>
      </xs:extension>
      </xs:complexContent>
</xs:complexType>
….
</schema>
```

15.4.3 DescribeResultAccess response example

The following is an example of DescribeResultAccess response.

Note that the example includes only the DescribeResultAccessResponse element without the SOAP envelope.

```
<DescribeResultAccessResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd ">
      <status>success</status>
      <URLs>
      <itemId>item_0001</itemId>
      <itemAddress>
      <ResourceAddress>
<URL>ftp://user:online@eoaserver:2121/0c88028bf3aa6a6a143ed846f2be1ea4</URL>
      </ResourceAddress>
      </itemAddress>
      </URLs>
      <URLs>
      <itemId>item_0002</itemId>
      <itemAddress>
      <ResourceAddress>
<URL>ftp://user:online@eoaserver:2121/0c88028bf3aa6a6a143ed846f2be1ea5</URL>
      </ResourceAddress>
      </itemAddress>
      </URLs>
      <URLs>
      <itemId>item_0003</itemId>
      <itemAddress>
      <ResourceAddress>
<URL>ftp://user:online@eoaserver:2121/0c88028bf3aa6a6a143ed846f2be1ea6</URL>
      </ResourceAddress>
      </itemAddress>
      </URLs>
</DescribeResultAccessResponse>
```

15.4.4 Exceptions

In the event that an Order Server encounters an error servicing a DescribeResultAccess, it shall return an exception within a SOAP Fault (in particular an ows:ExceptionReport according to the clause 8 of [NR9]).

The SOAP Fault Message (see §7.3.14) shall have one of the following exceptionCode value:

"exceptionCode" value	Meaning of code	"locator" value	"ExceptionText" value
MissingParameterValue	Operation request does not include a parameter value, and this server did not declare a default value for that parameter	Name of missing parameter	"Missing value for Parameter"
InvalidParameterValue	Operation request contains an invalid parameter value	Name of parameter with invalid value	"Invalid value for Parameter"
NoApplicableCode	No other exceptionCode specified by this service and server applies to this exception	None, omit "locator" parameter	"Code not applicable"
InvalidBehaviourValue	Operation request contains an invalid value for the behaviour. Allowed values are "allReady" or "nextReady".	"subFunction" parameter	"Invalid value for behaviour"
InvalidOrderIdentifier	Operation request contains an invalid order identifier.	"orderId" parameter	"Invalid value for order"
AuthenticationFailed	Invalid or missing identity information	"identity_token"	"Invalid or missing identity information"
AuthorizationFailed	The client is not authorized to call the operation or it is not authorized to provide the specified parameters.	Possible values: □ "DescribeResultAccess" □ "orderId"	Text describing the item not authorized. E.g.: □ "The client is not authorized to call the operation." □ Etc.

Table 15-6 – Exception codes for DescribeResultAcces operation.

16 Cancel Operation

16.1 Introduction

This operation allows cancelling a previously submitted order. In case of EO product orders this operation triggers the cancellation of the order items; in case of subscriptions it means to unsubscribe from them.

The cancellation of product order items is not always possible, and then the operation returns the following results:

– "success", if all items can be cancelled;

– "incomplete", in case of partial cancellation

The cancellation is not a real-time process, and then the operation is asynchronous.

16.2 Requirements

Req. id	**Requirement URI**	**Requirement Class**	**Requirement Text**
Req 113	http://www.opengis.net/spec/OSEO/1.0/req/Cancellation/Cancel	Cancellation	The Order Server shall implement the Cancel operation.
Req 114	http://www.opengis.net/spec/OSEO/1.0/req/Cancellation/Cancel/req/schema	Cancellation	The Cancel request shall consist of an XML instance document validated by the entity Cancel in the oseo.xsd XML Schema.
Req 115	http://www.opengis.net/spec/OSEO/1.0/req/Cancellation/Cancel/req/sync	Cancellation	The Order Server shall accept cancellation requests including: □ The identifier of the order to be cancelled: Cancel/orderId □ No notification: Cancel/statusNotification = None
Req 116	http://www.opengis.net/spec/OSEO/1.0/req/Cancellation/Cancel/ack_schema	Cancellation	The message output generated by the Cancel request shall consist of an XML instance document validated by the entity CancelAck in the oseo.xsd XML Schema.
Req 117	http://www.opengis.net/spec/OSEO/1.0/req/Cancellation/Cancel/ack	Cancellation	The response to a successful Cancel shall include: CancelAck/status = success
Req 118	http://www.opengis.net/spec/OSEO/1.0/req/Cancellation/Cancel/function	Cancellation	On successful acceptance of the Cancel request, the Order Server shall start the cancellation of the specified order. Comment: For verifying whether the Order Server is actually cancelling the order the GetStatus operation can be used.
Req 119	http://www.opengis.net/spec/OSEO/1.0/req/Cancellation/Cancel/exception	Cancellation	When a Order Server encounters an error while performing a Cancel operation, it shall return an ows: ExceptionReport according to the clause 8 of [NR9].

Req. id	Requirement URI	Requirement Class	Requirement Text
Req 120	http://www.opengis.net/spec/OSEO/1.0/req/AsyncCancellation/Cancel/Final	AsyncCancellation	The Order Server shall accept cancellation requests including: □ The identifier of the order to be cancelled: Cancel/orderId □ Notification at cancellation completion: Submit/statusNotification = Final soapenv:Envelope/soapenv:Header/wsa:ReplyTo
Req 121	http://www.opengis.net/spec/OSEO/1.0/req/AsyncCancellation/Cancel/All	AsyncCancellation	The Order Server shall accept cancellation requests including: □ The identifier of the order to be cancelled: Cancel/orderId □ All notifications: Submit/statusNotification = All soapenv:Envelope/soapenv:Header/wsa:ReplyTo
Req 122	http://www.opengis.net/spec/OSEO/1.0/req/AsyncCancellation/CancelResponse/schema	AsyncCancellation	The asynchronous notification (CancelResponse) to a successful Cancel request shall consist of an XML instance document as validated by the entity CancelResponse in the oseo.xsd XML Schema.
Req 123	http://www.opengis.net/spec/OSEO/1.0/req/AsyncCancellation/CancelResponse/final	AsyncCancellation	Ths asynchronous notification (CancelResponse) to a Cancel request with: Cancel/statusNotification = Final shall be sent by the Order Server only once at cancellation completion.
Req 124	http://www.opengis.net/spec/OSEO/1.0/req/AsyncCancellation/CancelResponse/all	AsyncCancellation	Ths asynchronous notification (CancelResponse) to a Cancel request having: Cancel/statusNotification = All shall be sent by the Order Server whenever the status of at least one item in the order is changed.
Req 125	http://www.opengis.net/spec/OSEO/1.0/req/AsyncCancellation/CancelResponse/resp	AsyncCancellation	The asynchronous notification (CancelResponse) to a successful Submit request shall contain: □ The order date and time: SubmitResponse/timeStamp □ Overal order status info □ One status info for each ordered item: SubmitResponse/orderMonitorSpecification/orderItem
Req 126	http://www.opengis.net/spec/OSEO/1.0/req/AsyncCancellation/CancelResponse/ack_schema	AsyncCancellation	The message output generated by the CancelResponse request shall consist of an XML instance document validated by the entity CancelResponseAck in the oseo.xsd XML Schema. The Order Server shall accept the CancelResponseAck.

Table 16-1: Cancel requirements.

16.3 Cancel operation request

The type of Cancel is CancelRequestType. The following figure provides a graphical representation of this type.

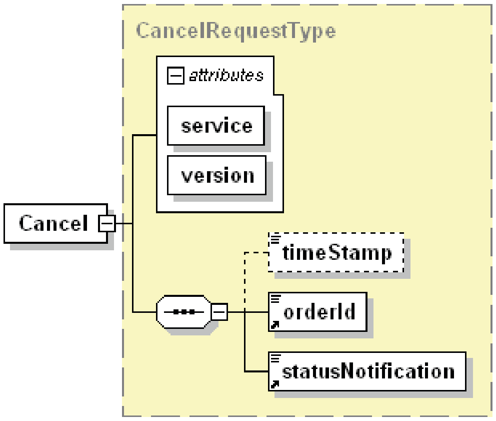


Figure 16-1 - Cancel diagram.

16.3.1 Cancel request parameters

The Cancel Operation uses the Cancel Request element.
The following table presents all parameters used within a Cancel request.

Names	Definition	Data type and values	Multiplicity and use
service	Service type identifier	Type: non-empty string Allowed values: OS	One (mandatory)
version	Specification version for operation	Type: non-empty string. Format: <x>.<y>.<z>	One (mandatory)
timeStamp	It is the time when the request has been issued.	Type: xs:DateTime	One (optional)
orderId	Order identification number unique for this Provider.	Type: xs: anyURI	One (mandatory)
statusNotification	This element specifies how many status notifications are sent back to the client (see §7.4.4)	Type: StatusNotificationType (see §7.4.4)	One (mandatory)

Table 16-2 - Cancel description.

16.3.2 Cancel request XML encoding

A XML schema fragment for this operation encoded in XML

```
<xs:schema xmlns="http://www.opengis.net/oseo/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:sps="http://www.opengis.net/sps/2.0"
targetNamespace="http://www.opengis.net/oseo/1.0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
```

```
      <xs:import namespace="http://www.opengis.net/swe/2.0"
schemaLocation="../swe/sweCommon/2.0.0/swe.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsGetCapabilities.xsd"/>
      <xs:import namespace="http://www.opengis.net/sps/2.0"
schemaLocation="Order_spsCommon.xsd"/>
      <xs:import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="../gml/3.2.1/base/gml.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsExceptionReport.xsd"/>
…
      <xs:element name="Cancel" type="CancelRequestType"/>
<xs:complexType name="CancelRequestType">
      <xs:complexContent>
      <xs:extension base="OrderRequestBaseType">
             <xs:sequence>
                    <xs:element name="timeStamp" type="xs:dateTime" minOccurs="0"/>
                    <xs:element ref="orderId"/>
                    <xs:element ref="statusNotification"/>
                    </xs:sequence>
             </xs:extension>
      </xs:complexContent>
      </xs:complexType>
….
</schema>
```

16.3.3 Cancel request example

The following is an example of Cancel request.

Note that the example includes only the Cancel element without the SOAP envelope.

```
<Cancel xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd " service="OS" version="1.0.0">
      <orderId>urn:ESA:EECF:order_id_0001</orderId>
      <statusNotification>Final</statusNotification>
</Cancel>
```

16.4 Cancel operation response

The synchronous output message is defined by the XML CancelAck entity. The type of CancelAck is CancelRequestAckType. The following figure provides a graphical representation of this type

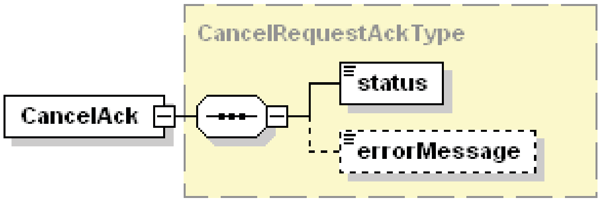


Figure 16-2 - CancelAck diagram.

16.4.1 Cancel response parameters

CancelAck defines the acknowledge for submitted order.

Names	Definition	Data type and values	Multiplicity and use
status	Completion result of the operation. If a blocking error occurred the request will abort and it will be generated a SOAP-Fault.	Type: String Permitted Values: success, partial	One (mandatory)
errorMessage	Message to explain a partial result.	Type: Not empty string (max 255 chars).	One (optional)

Table 16-3 - CancelAck description.

16.4.2 Cancel response XML encoding

A XML schema fragment for this operation encoded in XML

```
<xs:schema xmlns="http://www.opengis.net/oseo/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:sps="http://www.opengis.net/sps/2.0"
targetNamespace="http://www.opengis.net/oseo/1.0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
      <xs:import namespace="http://www.opengis.net/swe/2.0"
schemaLocation="../swe/sweCommon/2.0.0/swe.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsGetCapabilities.xsd"/>
      <xs:import namespace="http://www.opengis.net/sps/2.0"
schemaLocation="Order_spsCommon.xsd"/>
      <xs:import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="../gml/3.2.1/base/gml.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsExceptionReport.xsd"/>

…
<xs:element name="CancelAck" type="CancelRequestAckType"/>
<xs:complexType name="CancelRequestAckType">
             <xs:complexContent>
                   <xs:extension base="OrderResponseBaseType"/>
             </xs:complexContent>
      </xs:complexType>
      <xs:complexType name="StatusNotificationType">
             <xs:complexContent>
                   <xs:extension base="OrderRequestBaseType">
                          <xs:sequence>
                                <xs:element name="timeStamp" type="xs:dateTime"
minOccurs="0"/>
                                <xs:element ref="orderMonitorSpecification"/>
                          </xs:sequence>
                   </xs:extension>
             </xs:complexContent>
      </xs:complexType>
….
</schema>
```

16.4.3 Cancel response example

The following is an example of Cancel response (asynchronous).

Note that the example includes only the Cancel element without the SOAP envelope.

```
<CancelAck xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd">
      <status>success</status>
</CancelAck>
```

16.4.4 Exceptions

In the event that an Order Server encounters an error servicing a Cancel, it shall return an exception within a SOAP Fault (in particular an ows:ExceptionReport according to the clause 8 of [NR9]).

The SOAP Fault Message (see §7.3.14) shall have one of the following exceptionCode value:

"exceptionCode" value	Meaning of code	"locator" value	"ExceptionText" value
MissingParameterValue	Operation request does not include a parameter value, and this server did not declare a default value for that parameter	Name of missing parameter	"Missing value for Parameter"
InvalidParameterValue	Operation request contains an invalid parameter value	Name of parameter with invalid value	"Invalid value for Parameter"
NoApplicableCode	No other exceptionCode specified by this service and server applies to this exception	None, omit "locator" parameter	"Code not applicable"
AuthenticationFailed	Invalid or missing identity information	"identity_token"	"Invalid or missing identity information"
AuthorizationFailed	The client is not authorized to call the operation or it is not authorized to provide the specified parameters.	Possible values: □ "Cancel" □ "orderId"	Text describing the item not authorized. E.g.: □ "The client is not authorized to call the operation." □ "The Client is not authorized to cancel that order".
InvalidNotificationValue	In asynchronous mode, the operation request has the status notification active (with "All" or "Final" value) but an address information to which the server has to notify the ordering status that is empty or invalid.	"ws-address" parameter	"Invalid value for notification"
InvalidOrderIdentifier	Operation request contains an invalid order identifier.	"orderId" parameter	"Invalid value for order"

Table 16-4 – Exception codes for Cancel operation.

17 CancelResponse Operation

17.1 Introduction

Cancel is an asynchronous operation, then if the server supports asynchronous notification, two operations are called:

- Cancel, from client to server, for cancelling the order.
  - This operation is composed of two messages:
    - Cancel, it is the order cancel request
    - CancelAck, it is the acknowledge the server returns in real time to the request.
- CancelResponse, from server to client, for sending the notification to the client. Then this is a service to be implemented on the client.
  - This operation is composed of two messages:
    - CancelResponse, sending the order cancel status notification to the client;
    - CancelResponseAck, just returning the acknowledge on CancelResponse message receiption.

17.2 Requirements

See §16.2

17.3 CancelResponse operation request

This operation has to be implemented by a client of Ordering service supporting asynchronous operations.

This operation allows the Ordering Service to send to the client notifications about the progress of cancellation of submitted orders.

Because the CancelResponse operation carries the same information returned by the GetStatus operation, then the CancelResponse element uses the OrderMonitorSpecification, which is used into the GetStatusResponse element.

The following figure provides a graphical representation of this element.

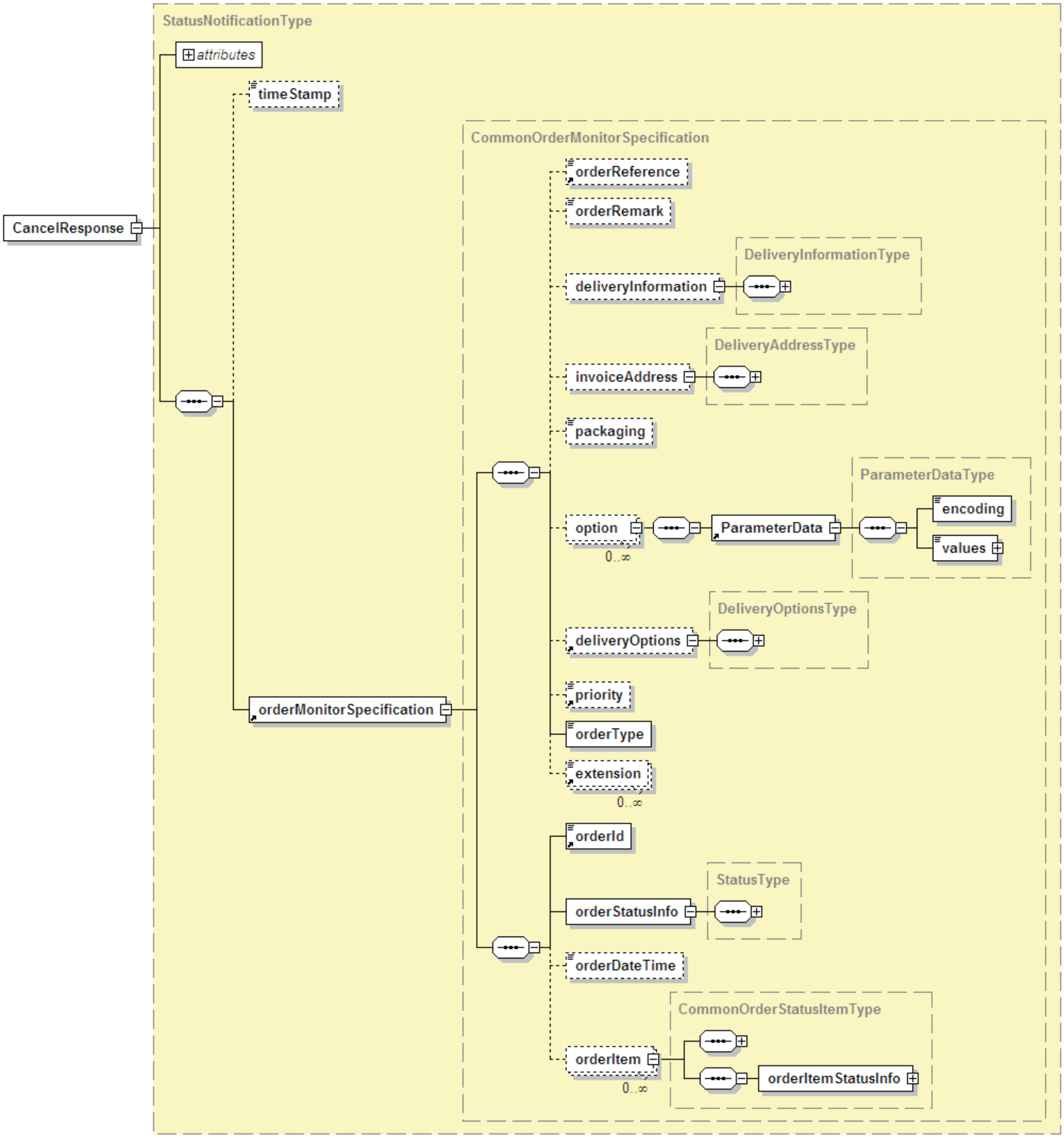


Figure 17-1 - CancelResponse diagram.

17.3.1 CancelResponse request parameters

The following table describes all information within a asynchronous Cancel response.

Names	Definition	Data type and values	Multiplicity and use
service	Service type identifier	Type: non-empty string Allowed values: OS	One (mandatory)
version	Specification version for operation	Type: non-empty string. Format: <x>.<y>.<z>	One (mandatory)
timeStamp	It is the time when the request has been issued.	Type: xs:DateTime	One (optional)
orderMonitorSpecification	Order Monitor Specification element. The content of this element correspond to that returned by GetStatus with **full** presentation.	Type: CommonOrderMonitorSpecification (see §7.3.12.1)	One (mandatory)

Table 17-1 - CancelResponse description.

17.3.2 CancelResponse request XML encoding

A XML schema fragment for this operation encoded in XML

```
<xs:schema xmlns="http://www.opengis.net/oseo/1.0"
xmlns:xs="http://www.w3.org/2001/XMLSchema"
xmlns:ows="http://www.opengis.net/ows/2.0" xmlns:gml="http://www.opengis.net/gml/3.2"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:sps="http://www.opengis.net/sps/2.0"
targetNamespace="http://www.opengis.net/oseo/1.0" elementFormDefault="qualified"
attributeFormDefault="unqualified">
      <xs:import namespace="http://www.opengis.net/swe/2.0"
schemaLocation="../swe/sweCommon/2.0.0/swe.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsGetCapabilities.xsd"/>
      <xs:import namespace="http://www.opengis.net/sps/2.0"
schemaLocation="Order_spsCommon.xsd"/>
      <xs:import namespace="http://www.opengis.net/gml/3.2"
schemaLocation="../gml/3.2.1/base/gml.xsd"/>
      <xs:import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="../ows/2.0/owsExceptionReport.xsd"/>
…
<xs:element name="CancelResponse" type="StatusNotificationType"/>
      <xs:complexType name="StatusNotificationType">
      <xs:complexContent>
      <xs:extension base="OrderRequestBaseType">
             <xs:sequence>
                    <xs:element name="timeStamp" type="xs:dateTime" minOccurs="0"/>
                    <xs:element ref="orderMonitorSpecification"/>
             </xs:sequence>
      </xs:extension>
      </xs:complexContent>
      </xs:complexType>
…
<xs:element name="CancelResponseAck" type="StatusNotificationAckType"/>
<xs:complexType name="StatusNotificationAckType">
             <xs:complexContent>
```

```
                  <xs:extension base="OrderResponseBaseType"/>
            </xs:complexContent>
      </xs:complexType>….
</schema>
```

17.3.3 CancelResponse request example

The following is an example of CancelResponse request.

Note that the example includes only the CancelResponse element without the SOAP envelope.

```
<CancelResponse xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0" xmlns:sps="http://www.opengis.net/sps/2.0"
xmlns:swe="http://www.opengis.net/swe/2.0" xmlns:gml="http://www.opengis.net/gml"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd" service="OS" version="1.0.0">
      <timeStamp>2006-11-18T21:08:00.000</timeStamp>
      <orderMonitorSpecification>
            <orderReference>example_0001</orderReference>
            <orderRemark>example</orderRemark>
            <deliveryInformation>
                  <mailAddress>
                        <recipient>Mr. John Smith</recipient>
                        <companyRef>DTMT</companyRef>
                        <postalAddress>
                              <streetAddress>Esrin Esa No. 1</streetAddress>
                              <city>Frascati</city>
                              <state>-</state>
                              <postalCode>00100</postalCode>
                              <country>IT</country>
                              <postBox>-</postBox>
                        </postalAddress>
                        <telephoneNumber>00390694180999</telephoneNumber>
                  </mailAddress>
            </deliveryInformation>
            <orderType>PRODUCT_ORDER</orderType>
            <orderId>123456-001</orderId>
            <orderStatusInfo>
                  <status>Cancelled</status>
                  <additionalStatusInfo>The order is
accomplished</additionalStatusInfo>
            </orderStatusInfo>
            <orderItem>
                  <itemId>item_0001</itemId>
                  <productOrderOptionsId>Level 1,Product SLC
(ASA_IMS)</productOrderOptionsId>
                  <orderItemRemark>First product</orderItemRemark>
                  <option>
                        <sps:ParameterData>
                              <sps:encoding>
                                    <swe:XMLEncoding
defaultNamespace="http://www.opengis.net/oseo/1.0/dataset"/>
                              </sps:encoding>
                              <sps:values
xmlns:ns="http://www.opengis.net/oseo/1.0/dataset">
                                    <ns:processingLevel>1B</ns:processingLevel>
                                    <ns:productType>ASA_IMS_1P</ns:productType>

      <ns:qualityOfService>STANDARD</ns:qualityOfService>
                              </sps:values>
                        </sps:ParameterData>
                  </option>
                  <sceneSelection>
```

```
                          <sps:ParameterData>
                                 <sps:encoding>
                                        <swe:XMLEncoding
defaultNamespace="http://www.opengis.net/oseo/1.0/dataset"/>
                                 </sps:encoding>
                                 <sps:values
xmlns:ns="http://www.opengis.net/oseo/1.0/dataset">
                                  <ns:sceneType>FloatingScene</ns:sceneType>
                                  <ns:sceneStartStopTime>
                                    <ns:min>2010-01-01T11:55:00.00Z</ns:min>
                                    <ns:max>2010-01-01T12:00:00.00Z</ns:max>
                                   </ns:sceneStartStopTime>
                                   <ns:frame>7191</ns:frame>
                                 </sps:values>
                          </sps:ParameterData>
                   </sceneSelection>
                   <deliveryOptions>
                   <mediaDelivery>
                          <packageMedium>CD-ROM</packageMedium>
                   </mediaDelivery>
                   </deliveryOptions>
                   <payment>
                          <paymentMethod>quota</paymentMethod>
                          <orderAccount>project_10000</orderAccount>
                   </payment>
                   <productId>
                          <identifier>EN1-05121921114570-3322.XI</identifier>

      <collectionId>http://www.opengis.net/def/EOP/ESA/0/ESA.EECF.ENVISAT_ASA_IMx_xS<
/collectionId>
                   </productId>
             </orderItem>
      </orderMonitorSpecification>
</CancelResponse>
```

17.4 *CancelResponse operation response*

The output message that client sends when it receives the outcome of cancel response is defined by the XML CancelResponseAck entity. The type of CancelResponseAck is StatusNotificationAckType. The following figure provides a graphical representation of this type:

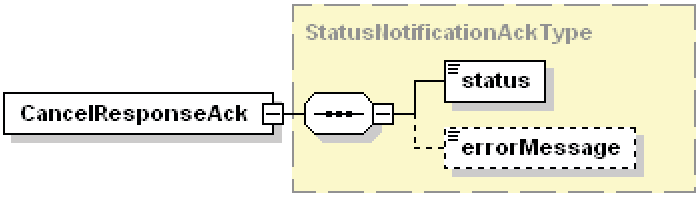


Figure 17-2 - CancelResponseAck diagram.

17.4.1 CancelResponse response parameters

The following table describes all information within a the output response to the Cancel response.

Names	Definition	Data type and values	Multiplicity and use
status	Status of reception of CancelResponse message. If a blocking error occurred the request will abort and it will be generated a SOAP-Fault.	Type: String Permitted Values: success, partial	One (mandatory)
errorMessage	Message to explain the partial result.	Type: Not empty string (max 255 chars).	One (optional)

Table 17-2 - CancelResponseAck description.

17.4.2 CancelResponse response XML encoding

See §17.3.2

17.4.3 CancelResponse response example

The following is an example of CancelResponse response.

Note that the example includes only the CancelResponseAck element without the SOAP envelope.

```
<CancelResponseAck xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://www.opengis.net/oseo/1.0"
xsi:schemaLocation="http://www.opengis.net/oseo/1.0
http://schemas.opengis.net/oseo/1.0/oseo.xsd">
      <status>success</status>
</CancelResponseAck>
```

17.4.4 Exceptions

This operation has to be implemented by a client of Ordering service**.**

In the event that an Order Server encounters an error servicing a CancelResponse, it shall return an exception within a SOAP Fault (in particular an ows:ExceptionReport according to the clause 8 of [NR9]).

The SOAP Fault Message (see §7.3.14) shall have one of the following exceptionCode value:

"exceptionCode" value	Meaning of code	"locator" value	"ExceptionText" value
MissingParameterValue	Operation request does not include a parameter value, and this server did not declare a default value for that parameter	Name of missing parameter	"Missing value for Parameter"
InvalidParameterValue	Operation request contains an invalid parameter value	Name of parameter with invalid value	"Invalid value for Parameter"
NoApplicableCode	No other exceptionCode specified by this service and server applies to this exception	None, omit "locator" parameter	"Code not applicable"
AuthenticationFailed	Invalid or missing identity information	"identity_token"	"Invalid or missing identity information"

“exceptionCode” value	Meaning of code	“locator” value	“ExceptionText” value
AuthorizationFailed	The client is not authorized to call the operation or it is not authorized to provide the specified parameters.	None, omit “locator” parameter	“The client is not authorized to call the operation.”

Table 17-3 – Exception codes for CancelResponse operation.

APPENDIX A (normative) Abstract test suite

A.1 General

This section describes the Abstract Test Suite (ATS) of the OGC Ordering Services for Earth Observation Products. An ATS provides a basis for developing an executable test suite (ETS) to verify that an Implementation Under Test (IUT) conforms to all relevant functional specifications.

The assertions are gleaned from a set of specification documents; the dependencies among these specifications are shown in the figure below where each specification is represented as a UML package.

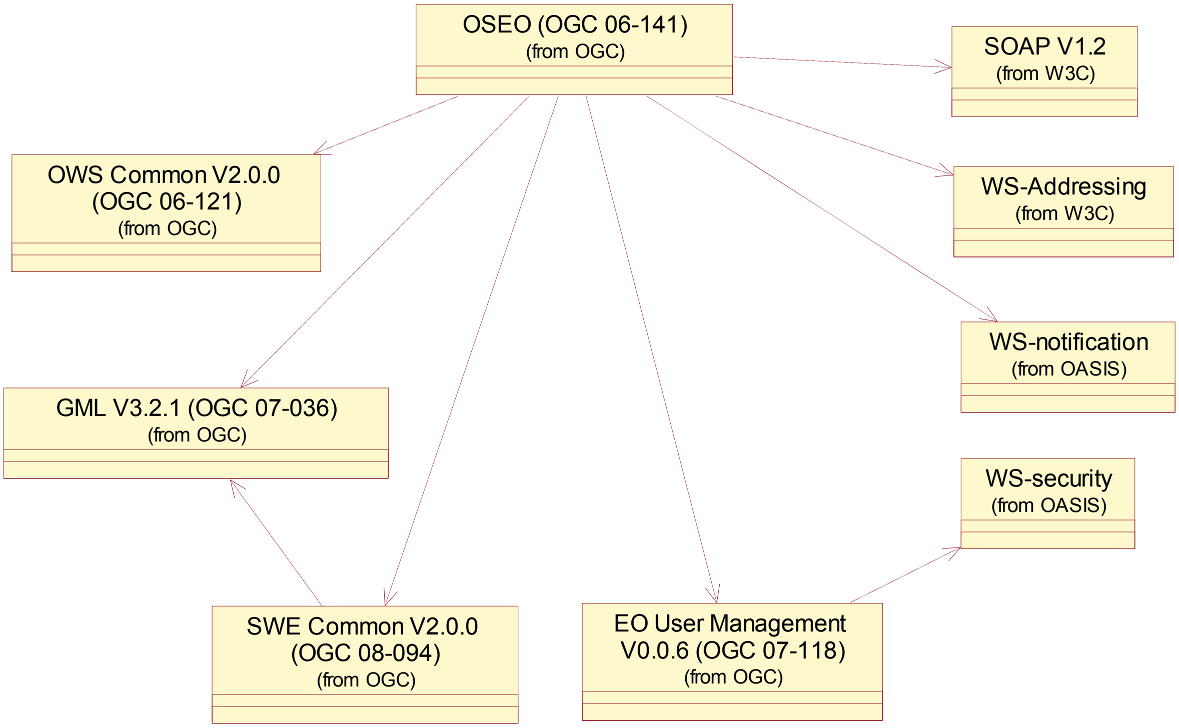


Figure 17-3 - Package Dependency of OSEO Specification.

A.2 Overview of the Conformance Classes

Due to the complexity of the specification a non negligible number of Requirements / Conformance Classes have been defined. In fact:

- The specification involves 7 operations, each having several possible different behaviours and only some of them can be implemented by an Order Server;
- Operations are Synchronous and Asynchronous;
- There are dependencies between the operations i.e. the execution of an operation is affected by other ones previously executed;
- Different items can be ordered through the same operations.

Because Order Servers can implemented only some of these functions, and servers implementing all functions are very rare, then a number of conformance classes have been defined regrouping the most correlated functions.

Hereafter the list of conformance classes is reported:

Requirement Class	Requirement Class URI	Dependency
AsyncCancellation	http://www.opengis.net/spec/OSEO/1.0/AsyncCancellation	Cancellation
AsyncSubmit	http://www.opengis.net/spec/OSEO/1.0/AsyncSubmit	Core
Cancellation	http://www.opengis.net/spec/OSEO/1.0/Cancellation	Core
Core	http://www.opengis.net/spec/OSEO/1.0/Core	
MediaDelivery	http://www.opengis.net/spec/OSEO/1.0/MediaDelivery	Core
Notification	http://www.opengis.net/spec/OSEO/1.0/Notification	Core
OnlineDataAccess	http://www.opengis.net/spec/OSEO/1.0/OnlineDataAccess	Core
OnlineDataDelivery	http://www.opengis.net/spec/OSEO/1.0/OnlineDataDelivery	Core
ProductOrder	http://www.opengis.net/spec/OSEO/1.0/ProductOrder	Core
Quotation	http://www.opengis.net/spec/OSEO/1.0/Quotation	Core
QuotationAsync	http://www.opengis.net/spec/OSEO/1.0/QuotationAsync	Quotation
QuotationMonitoring	http://www.opengis.net/spec/OSEO/1.0/QuotationMonitoring	Quotation
QuotationOffLine	http://www.opengis.net/spec/OSEO/1.0/QuotationOffLine	Quotation
QuotationSync	http://www.opengis.net/spec/OSEO/1.0/QuotationSync	Quotation
SceneSelection	http://www.opengis.net/spec/OSEO/1.0/SceneSelection	ProductOrder
SubscriptionOrder	http://www.opengis.net/spec/OSEO/1.0/SubscriptionOrder	Core
TaskingOrder	http://www.opengis.net/spec/OSEO/1.0/TaskingOrder	Core

Table 17-4: Requirements Classes.

Provided that the full compliance is very difficult to achieve, in the following possible examples of compliance are reported:

- An Order Server supporting product ordering for scientific users should comply with:

  - Core
  - MediaDelivery
  - ProductOrder
  - SceneSelection

  Optionally it should support also on-line delivery:

  - OnlineDataAccess
  - OnlineDataDelivery

- An Order Server supporting product ordering for commercial users should comply with:
  - Core
  - MediaDelivery
  - ProductOrder
  - SceneSelection
  - Quotation and at least one of:
    - QuotationAsync
    - QuotationMonitoring
    - QuotationOffLine
    - QuotationSync

  Optionally it should support also on-line delivery:

  - OnlineDataAccess
  - OnlineDataDelivery

Conformance classes have been defined with 1:1 mapping with respect to the Requirements Classes: each conformance class is in charge of validating the corresponding Requirement Class:

Requirement Class	Conformance Class	Conformance Class URI
AsyncCancellation	AsyncCancellation	http://www.opengis.net/spec/OSEO/1.0/conf/AsyncCancellation
AsyncSubmit	AsyncSubmit	http://www.opengis.net/spec/OSEO/1.0/conf/AsyncSubmit
Cancellation	Cancellation	http://www.opengis.net/spec/OSEO/1.0/conf/Cancellation
Core	Core	http://www.opengis.net/spec/OSEO/1.0/conf/Core
MediaDelivery	MediaDelivery	http://www.opengis.net/spec/OSEO/1.0/conf/MediaDelivery
Notification	Notification	http://www.opengis.net/spec/OSEO/1.0/conf/Notification
OnlineDataAccess	OnlineDataAccess	http://www.opengis.net/spec/OSEO/1.0/conf/OnlineDataAccess
OnlineDataDelivery	OnlineDataDelivery	http://www.opengis.net/spec/OSEO/1.0/conf/OnlineDataDelivery
ProductOrder	ProductOrder	http://www.opengis.net/spec/OSEO/1.0/conf/ProductOrder
Quotation	Quotation	http://www.opengis.net/spec/OSEO/1.0/conf/Quotation
QuotationAsync	QuotationAsync	http://www.opengis.net/spec/OSEO/1.0/conf/QuotationAsync
QuotationMonitoring	QuotationMonitoring	http://www.opengis.net/spec/OSEO/1.0/conf/QuotationMonitoring
QuotationOffLine	QuotationOffLine	http://www.opengis.net/spec/OSEO/1.0/conf/QuotationOffLine
QuotationSync	QuotationSync	http://www.opengis.net/spec/OSEO/1.0/conf/QuotationSync
SceneSelection	SceneSelection	http://www.opengis.net/spec/OSEO/1.0/conf/SceneSelection
SubscriptionOrder	SubscriptionOrder	http://www.opengis.net/spec/OSEO/1.0/conf/SubscriptionOrder
TaskingOrder	TaskingOrder	http://www.opengis.net/spec/OSEO/1.0/conf/TaskingOrder

Table 17-5: Requirements vs. Conformance Classes.

Each Conformance Class is composed of a set of tests, each verifying one or more requirements of the corresponding Requirements Class.

Each Conformance Class covers all requirements of the corresponding Requirements Class.

It has to be noted that the tests reported in the Conformance classes have “temporal dependencies”: in fact for running one test another specific tests might be needed (e.g. to test order status at least one order needs to be created in the Order Server by Submitting an order). Then:

- The tests specified in a Conformance Class must be executed in the order they are specified in the document;
- The tests of a Conformance Class can be started only if the tests of the parent class have been completed.

A.3 Conformance Test Classes

A.3.1 Conformance Class Core (http://www.opengis.net/spec/OSEO/1.0/conf/Core)

This Conformance Class is in charge of verifying the compliance of the Order Server under test with respect to the **Core** Requirement Class, which includes the basic functions that every Order Server shall implement.

A.3.1.1 GetCapabilities

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Core/GetCapabilities**

b) **Test purpose**: To verify that he Order Server under test correctly supports the OSEO GetCapabilities operation.

c) **Test method**:

Verify that the Order Server under test accepts a valid OSEO GetCapabilities request with servicename “OS”, complying with GetCapabilities element of oseo.xsd schema sent via HTTP/XML/POST/SOAP 1.2.

Verify that the SOAP Body of the response message:

□ complies with the Capabilities element of oseo.xsd schema

□ the ows:OperationsMetadata element is filled-in with the list of supported operations

□ all attributes of Capabilities/Contents/GetStatusCapabilities element are set to true

□ at least one collection identifier is returned

□ the supported SWE encoding (Capabilities/Contents/ContentsType supportedEncoding) is XMLEncoding

Pass if the assertion is satisfied; fail otherwise.

d) **References**:

□ Req 1

□ Req 2

□ Req 3

□ Req 4

□ Req 5

□ Req 6

□ Req 7

e) **Test type**: Capability

A.3.1.2 GetCapabilities non nominal conditions

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Core/GetCapabilities-non-nominal**

b) **Test purpose:**

Verification that the Order Server under test throws a correct error message when incorrect GetCapabilities request is received.

c) **Test method:**

Send an incorrect OSEO GetCapabilities request via HTTP/XML/POST/SOAP 1.2 without a servicename. Verify that the response to the request is a SOAP Fault including an ows: ExceptionReport according to the clause 8 of [NR9]

d) **References:**

□ Req 8

e) **Test type: Capability**

A.3.1.3 GetOptions - Core

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Core/GetOptions**

b) **Test purpose:**

To verify whether the Order Server under test supports the OSEO GetOptions operation.

Note that the request message shall be prepared in order to allow the Order Server under test to return a non empty response (i.e. it has to specifiy a supported product OR collection OR task id).

c) **Test method:**

To send a correct GetOptions request for an item supported by the Order Server (either a product or a collection or a tasking request) via HTTP/XML/POST/SOAP 1.2.

Verify that:

- the returned GetOptionsResponse is valid against oseo.xsd XML Schema.
- At least one <option> element is returned

d) **References:**

- Req 19
- Req 20
- Req 21

e) **Test type: Capability**

A.3.1.4 GetOptions non nominal conditions

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Core/GetOptions-non-nominal**

b) **Test purpose:**

To verify whether the Order Server under test is able to handle incorrect calls to OSEO GetOptions operation.

c) **Test method:**

To send different GetOptions requests via HTTP/XML/POST/SOAP 1.2 with different errors:

- Message with wrong element: GetOptions including the following wrong element GetOptions/WRONG_TAG.
- Message with unsupported item:
  - GetOptions/collectionId = WRONG_COLLECTION_ID
- Message with unsupported item:
  - GetOptions/identifier = WRONG_PRODUCT_ID
- Message with unsupported item:
  - GetOptions/taskingRequest/ID = WRONG_TASK_ID

Verify that a correct SOAP Fault is returned by Order Server under test.

d) **References:**

- Req 22

e) **Test type: Capability**

A.3.1.5 Submit - Core

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Core/Submit**

b) Test purpose:

To verify that the Order Server under test is able to manage calls to OSEO Submit operation (any supported order type).

Note that the request message shall be prepared in order to allow the Order Server under test to accept the order (i.e. it has to specifiy a supported product OR collection OR task id).

c) Test method:

To send 2 Submit requests via HTTP/XML/POST/SOAP 1.2:

- Including the order specification (any supported order type):

  Submit/orderSpecification
- with no order status notification:

  Submit/statusNotification = None
- valid against oseo.xsd schema;
- specifying 2 different orderReference values (to be re-used in later tests).

Verify that the returned SubmitAck:

- is valid against oseo.xsd XML Schema;
- SubmitAck/status = success**;**
- Has a unique order identifier:

  SubmitAck/orderId

d) References:

- Req 56
- Req 57
- Req 58
- Req 59
- Req 60
- Req 61

e) Test type: Capability

A.3.1.6 Submit - non nominal conditions

a) Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Core/Submit-non-nominal

b) Test purpose:

To verify that the Order Server under test is able to manage incorrect calls to OSEO Submit operation.

c) Test method:

To send different Submit requests via HTTP/XML/POST/SOAP 1.2 with different errors:

- Message with wrong element:
  - Submit including the following wrong element Submit/WRONG_TAG.

Verify that a correct SOAP Fault is returned by Order Server under test.

d) References:

- Req 62

e) **Test type: Capability**

A.3.1.7 GetStatus – order search by LastUpdate, BRIEF

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Core/GetStatus-search-brief**

b) **Test purpose:**

To verify whether the Order Server under test is able to manage calls to OSEO GetStatus operation, for any supported order type, order search mode, brief presentation.

c) **Test method:**

To send GetStatus request via HTTP/XML/POST/SOAP 1.2:

- Including lastUpdate as order search filter:

  GetStatus/filteringCriteria/lastUpdate

  Specifying a very old date in order to get a non empty answer.

  GetStatus/filteringCriteria/lastUpdateEnd

  Specifying today in order to get a non empty answer.

- with brief presentation:

  GetStatus/presentation = brief

- valid against oseo.xsd schema;

Verify that the returned GetStatusResponse:

- is valid against oseo.xsd XML Schema;
- GetStatusResponse/status = success**;**
- includes at least one order, i.e. at least one instance of:

  GetStatusResponse/orderMonitorSpecification

- the returned order(s) has modification date later than that specified in the request:

  GetStatusResponse/orderMonitorSpecification/orderDateTime

  Is later than:

  GetStatus/filteringCriteria/lastUpdate

- the returned order(s) monitoring info includes:
  - all information specified when the order has been submitted
  - GetStatusResponse/orderMonitorSpecification/orderId
  - GetStatusResponse/orderMonitorSpecification/orderStatusInfo
  - GetStatusResponse/orderMonitorSpecification/orderDateTime
- the returned order(s) monitoring info does NOT include:
  - GetStatusResponse/orderMonitorSpecification/orderItem

d) **References:**

- Req 79
- Req 80
- Req 82

- Req 83
- Req 86
- Req 88
- Req 91
- Req 93
- Req 95
- Req 98

e) **Test type: Capability**

A.3.1.8 GetStatus – order search by LastUpdate, FULL

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Core/GetStatus-search-full**

b) **Test purpose:**

To verify whether the Order Server under test is able to manage calls to OSEO GetStatus operation, for any supported order type, order search mode, FULL presentation.

c) **Test method:**

To send GetStatus request via HTTP/XML/POST/SOAP 1.2:

- Including lastUpdate as order search filter:

  GetStatus/filteringCriteria/lastUpdate

  Specifying a very old date in order to get a non empty answer.
- with FULL presentation:

  GetStatus/presentation = full
- valid against oseo.xsd schema;

Verify that the returned GetStatusResponse:

- is valid against oseo.xsd XML Schema;
- GetStatusResponse/status = success**;**
- includes at least one order, i.e. at least one instance of:

  GetStatusResponse/orderMonitorSpecification
- the returned order(s) has modification date later than that specified in the request:

  GetStatusResponse/orderMonitorSpecification/orderDateTime

  Is later than:

  GetStatus/filteringCriteria/lastUpdate
- the returned order(s) monitoring info includes:
  - all information specified when the order has been submitted
  - GetStatusResponse/orderMonitorSpecification/orderId
  - GetStatusResponse/orderMonitorSpecification/orderStatusInfo
  - GetStatusResponse/orderMonitorSpecification/orderDateTime
  - GetStatusResponse/orderMonitorSpecification/orderItem

- The returned orderItem status (GetStatusResponse/orderMonitorSpecification/orderItem/orderItemStatusInfo/status) is:
    - **Completed**, for ordered items that have been completed
    - **Submitted** or **Accepted** or **InProduction** for order items not yet ready

d) **References:**

- Req 79
- Req 80
- Req 82
- Req 83
- Req 87
- Req 88
- Req 91
- Req 94
- Req 95
- Req 96
- Req 97
- Req 98

e) **Test type: Capability**

A.3.1.9 GetStatus – order search by OrderStatus, FULL

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Core/GetStatus-search-status-full**

b) **Test purpose:**

To verify whether the Order Server under test is able to manage calls to OSEO GetStatus operation, for any supported order type, order search mode, FULL presentation.

c) **Test method:**

To send GetStatus request via HTTP/XML/POST/SOAP 1.2:

- Including OrderStatus as order search filter (2 instances):

  GetStatus/filteringCriteria/orderStatus

  Specifying order statuses returned by the previous test §A.3.1.8 in order to get a non empty answer.
- with FULL presentation:

  GetStatus/presentation = full
- valid against oseo.xsd schema;

Verify that the returned GetStatusResponse:

- is valid against oseo.xsd XML Schema;
- GetStatusResponse/status = success**;**
- includes at least one order, i.e. at least one instance of:

  GetStatusResponse/orderMonitorSpecification

- the returned order(s) has order status specified in the request:

  GetStatusResponse/orderMonitorSpecification/orderStatusInfo/status

  Is equal to one of:

  GetStatus/filteringCriteria/orderStatus
- the returned order(s) monitoring info includes:
  - all information specified when the order has been submitted
  - GetStatusResponse/orderMonitorSpecification/orderId
  - GetStatusResponse/orderMonitorSpecification/orderStatusInfo
  - GetStatusResponse/orderMonitorSpecification/orderDateTime
  - GetStatusResponse/orderMonitorSpecification/orderItem

d) References:

- Req 79
- Req 80
- Req 82
- Req 84
- Req 87
- Req 88
- Req 91
- Req 94
- Req 95
- Req 98

e) Test type: Capability

A.3.1.10 GetStatus – order search by OrderReference, FULL

a) Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Core/GetStatus-search-ref-full

b) Test purpose:

To verify whether the Order Server under test is able to manage calls to OSEO GetStatus operation, for any supported order type, order search mode, FULL presentation, orderReference search criterion.

c) Test method:

To send GetStatus request via HTTP/XML/POST/SOAP 1.2:

- Including OrderReference as order search filter:

  GetStatus/filteringCriteria/orderReference

  Specifying a value present in the Order Server (i.e. included in the response of test A.3.1.8) in order to get an answer.
- with FULL presentation:

  GetStatus/presentation = full
- valid against oseo.xsd schema;

Verify that the returned GetStatusResponse:

- is valid against oseo.xsd XML Schema;
- GetStatusResponse/status = success**;**
- includes orders:

  GetStatusResponse/orderMonitorSpecification
- the returned order(s) has order reference specified in the request:

  GetStatusResponse/orderMonitorSpecification/orderReference

  Is equal to one of:

  GetStatus/filteringCriteria/orderStatus

d) **References:**

- Req 79
- Req 80
- Req 82
- Req 85
- Req 87
- Req 88
- Req 92

e) **Test type: Capability**

A.3.1.11 GetStatus – by ORDER ID, FULL

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Core/GetStatus-retrieve-full**

b) **Test purpose:**

To verify whether the Order Server under test is able to manage calls to OSEO GetStatus operation, for any supported order type, order retrieve mode (i.e. by ORDER_ID), FULL presentation.

c) **Test method:**

To send GetStatus request via HTTP/XML/POST/SOAP 1.2:

- Including the order identifier:

  GetStatus/orderId

  Specifying a value present in the Order Server (i.e. included in the response of test A.3.1.8) in order to get a non empty answer.
- with FULL presentation:

  GetStatus/presentation = full
- valid against oseo.xsd schema;

Verify that the returned GetStatusResponse:

- is valid against oseo.xsd XML Schema;
- GetStatusResponse/status = success**;**

- includes ONE order, i.e. one instance of:

  GetStatusResponse/orderMonitorSpecification
- the returned order has the order identifier specified in the request:

  GetStatusResponse/orderMonitorSpecification/orderId

  Is equal to:

  GetStatus/orderId
- the returned order monitoring info shall include:
  - all information specified when the order was submitted
  - GetStatusResponse/orderMonitorSpecification/orderId
  - GetStatusResponse/orderMonitorSpecification/orderStatusInfo
  - GetStatusResponse/orderMonitorSpecification/orderDateTime
  - GetStatusResponse/orderMonitorSpecification/orderItem

d) **References:**

- Req 79
- Req 80
- Req 81
- Req 87
- Req 88
- Req 89
- Req 94
- Req 95

e) **Test type: Capability**

A.3.1.12 GetStatus – by Not Existing ORDER ID, BRIEF, SOAP Fault

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Core/GetStatus-retrieve-brief-empty**

b) **Test purpose:**

To verify whether the Order Server returns correctly a SOAP fault when retrieving a non existing order.

c) **Test method:**

To send GetStatus request via HTTP/XML/POST/SOAP 1.2:

- Including the identifier of a non existing order:

  GetStatus/orderId = NOT_EXISTING_ORDER
- with BRIEF presentation:

  GetStatus/presentation = brief
- valid against oseo.xsd schema;

Verify that the returned GetStatusResponse:

- SOAP Fault

d) References:

- Req 79
- Req 80
- Req 81
- Req 86
- Req 88
- Req 90
- Req 99

e) Test type: Capability

A.3.1.13 GetStatus - non nominal conditions

a) Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Core/GetStatus-non-nominal

b) Test purpose:

To verify whether the Order Server under test is able to manage incorrect calls to OSEO GetStatus operation.

c) Test method:

To send GetStatus request via HTTP/XML/POST/SOAP 1.2 with error:

- Message with wrong element:
  - GetStatus including the following wrong element GetStatus/WRONG_TAG.

Verify that a correct SOAP Fault is returned by Order Server under test.

d) References:

- Req 99

e) Test type: Capability

A.3.2 Conformance Class ProductOrder (http://www.opengis.net/spec/OSEO/1.0/conf/ProductOrder)

This Conformance Class is in charge of verifying the compliance of the Order Server under test with respect to the **ProductOrder** Requirement Class, which specifies all requirements an Order Server has to comply with for claiming the support of Ordering for Earth Observation Products.

A.3.2.1 GetCapabilities

a) Test id: http://www.opengis.net/spec/OSEO/1.0/conf/ProductOrder/GetCapabilities

b) **Test purpose**:

To verify that the Order Server under test correctly supports the OSEO GetCapabilities operation returning the additional information necessary for Product Orders.

c) **Test method**:

Verify that the Order Server under test accepts a valid OSEO GetCapabilities request and returns a valid Capabilities element including:

- Either Capabilities/Contents/ProductOrders/@supported = true
- Or at least one element Capabilities/Contents/SupportedCollection/ProductOrders/@supported = true

Pass if the assertion is satisfied; fail otherwise.

d) **References**:

- Req 9

e) **Test type**: Capability

A.3.2.2 GetOptions – By Product

a) Test id: http://www.opengis.net/spec/OSEO/1.0/conf/ProductOrder/GetOptions-product

b) Test purpose:

To verify whether the Order Server under test supports the OSEO GetOptions operation returning order options for Product Ordering.

c) Test method:

To send a GetOptions request via HTTP/XML/POST/SOAP 1.2 asking for the Order Options of 2 EO Products of different collections i.e. send a message including 2 instances of:

GetOptions/identifier

Verify that at least one order option for each identified product is returned.

Verify that the order options are flagged for product orders i.e. the response messge shall include:

GetOptionsResponse/orderOptions/orderType = PRODUCT_ORDER

d) References:

- Req 23
- Req 24
- Req 27

e) Test type: Capability

A.3.2.3 GetOptions – By Collection

a) Test id: http://www.opengis.net/spec/OSEO/1.0/conf/ProductOrder/GetOptions-collection

b) Test purpose:

To verify whether the Order Server under test supports the OSEO GetOptions operation for Product Ordering by collection id.

c) Test method:

To send a GetOptions request via HTTP/XML/POST/SOAP 1.2 asking for the Order Options of 1 collection i.e. send a message including 1 instance of:

GetOptions/collection

Verify that at least one order option is returned.

Verify that the order options are flagged for product orders i.e. the response messge shall include:

GetOptionsResponse/orderOptions/orderType = PRODUCT_ORDER

d) References:

- Req 25
- Req 26
- Req 27

e) **Test type: Capability**

A.3.2.4 Submit – Product Order

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/ProductOrder/Submit-product-order**

b) **Test purpose:**

To verify that the Order Server under test is able to manage calls to OSEO Submit operation for Earth Observation Products orders.

c) **Test method:**

To send Submit requests via HTTP/XML/POST/SOAP 1.2:

- Including the order specification:

  Submit/orderSpecification

- Specifying product order:

  Submit/orderType = “PRODUCT_ORDER”

- Specifying at least one Earth Observation Product:

  Submit/orderSpecification/orderItem/productId

- with no order status notification:

  Submit/statusNotification = None

- valid against oseo.xsd schema;

Verify that the returned SubmitAck:

- is valid against oseo.xsd XML Schema;
- SubmitAck/status = success**;**
- has a unique order identifier:

  SubmitAck/orderId

d) **References:**

- Req 70

e) **Test type: Capability**

A.3.2.5 Submit - non nominal conditions

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/ProductOrder/Submit-non-nominal**

b) **Test purpose:**

To verify whether the Order Server under test is able to manage incorrect calls to OSEO Submit operation.

c) **Test method:**

To send Submit different requests via HTTP/XML/POST/SOAP 1.2 with different errors:

- Message with unsupported item:

    - Submit/orderSpecification/orderItem/productId/identifier = WRONG_PRODUCT_ID
  - Message with unsupported item:
    - Submit/orderSpecification/orderItem/productId/collectionId = WRONG_COLLECTION_ID
  - Order Options different from those returned by a previous call to GetOptions.

  Verify that a correct SOAP Fault is returned by Order Server under test.

d) **References:**

  - Req 62

e) **Test type: Capability**

A.3.2.6 GetStatus – by ORDER ID, FULL

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/ProductOrder/GetStatus-retrieve-full**

b) **Test purpose:**

To verify whether the Order Server under test is able to retrieve Product Orders by ID, FULL presentation.

c) **Test method:**

To send GetStatus request via HTTP/XML/POST/SOAP 1.2:

- Including the order identifier:

  GetStatus/orderId

  Specifying a value present in the Order Server in order to get a non empty answer.
- with FULL presentation:

  GetStatus/presentation = full
- valid against oseo.xsd schema;

Verify that the returned GetStatusResponse:

- is valid against oseo.xsd XML Schema;
- GetStatusResponse/status = success**;**
- includes ONE order, i.e. one instance of:

  GetStatusResponse/orderMonitorSpecification
- the returned order has the order identifier specified in the request:

  GetStatusResponse/orderMonitorSpecification/orderId

  Is equal to:

  GetStatus/orderId
- the returned order monitoring info shall include:
  - all information specified when the order was submitted
  - GetStatusResponse/orderMonitorSpecification/orderId
  - GetStatusResponse/orderMonitorSpecification/orderStatusInfo
  - GetStatusResponse/orderMonitorSpecification/orderDateTime

  - GetStatusResponse/orderMonitorSpecification/orderItem
  - GetStatusResponse/orderMonitorSpecification/orderItem/productId

d) **References:**

- Req 100

e) **Test type: Capability**

A.3.3 Conformance Class SceneSelection (http://www.opengis.net/spec/OSEO/1.0/conf/SceneSelection)

This Conformance Class is in charge of verifying the compliance of the Order Server under test with respect to the **SceneSelection** Requirement Class, which specifies all requirements an Order Server has to comply with for claiming the support of Ordering for Earth Observation Products with scene selection options.

A.3.3.1 GetOptions – Scene Selection

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/SceneSelection/GetOptions**

b) **Test purpose:**

To verify whether the Order Server under test supports scene selection options in OSEO GetOptions operation.

c) **Test method:**

To send a GetOptions request via HTTP/XML/POST/SOAP 1.2 asking for the Order Options of 1 collection or 1 product for which scene selection options are supported (it is not required that ALL products or collections can be ordered specifying the scene selection, but it shall be possible for at least one).

Verify that at least one order option including scene selection options is returned i.e.

GetOptionsResponse/orderOptions/sceneSelectionOptions is non empty

d) **References:**

- Req 28

e) **Test type: Capability**

A.3.3.2 Submit – Product with Scenes

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/SceneSelection/Submit**

b) **Test purpose:**

To verify that the Order Server under test is able to manage calls to OSEO Submit operation for Earth Observation Products orders with Scene Selection.

c) **Test method:**

To send Submit requests via HTTP/XML/POST/SOAP 1.2:

- Including the order specification:

Submit/orderSpecification

- Specifying product order:

  Submit/orderType = “PRODUCT_ORDER”
- Specifying at least one Earth Observation Product:

  Submit/orderSpecification/orderItem/productId
- Specifying one scene selection for the specified order item:

  Submit/orderSpecification/orderItem/sceneSelection
- with no order status notification:

  Submit/statusNotification = None
- valid against oseo.xsd schema;

Verify that the returned SubmitAck:

- is valid against oseo.xsd XML Schema;
- SubmitAck/status = success**;**
- Has a unique order identifier:

  SubmitAck/orderId

d) **References:**

- Req 71

e) **Test type: Capability**

A.3.3.3 Submit - non nominal conditions

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/SceneSelection/Submit-non-nominal**

b) **Test purpose:**

To verify whether the Order Server under test is able to manage calls to OSEO Submit operation including incorrect scene selection options.

c) **Test method:**

To send Submit request via HTTP/XML/POST/SOAP 1.2 with error:

- Scene Selection Options different from those returned by a previous call to GetOptions.

Verify that a correct SOAP Fault is returned by Order Server under test.

d) **References:**

- Req 72

e) **Test type: Capability**

A.3.3.4 GetStatus – by ORDER ID, FULL

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/SceneSelection/GetStatus-retrieve-full**

b) **Test purpose:**

To verify whether the Order Server under test is able to retrieve Product Orders with scene selection by ID, FULL presentation.

c) **Test method:**

To send GetStatus request via HTTP/XML/POST/SOAP 1.2:

- Including the order identifier:

  GetStatus/orderId

  Specifying a value present in the Order Server in order to get a non empty answer.
- with FULL presentation:

  GetStatus/presentation = full
- valid against oseo.xsd schema;

Verify that the returned GetStatusResponse:

- is valid against oseo.xsd XML Schema;
- GetStatusResponse/status = success**;**
- includes ONE order, i.e. one instance of:

  GetStatusResponse/orderMonitorSpecification
- the returned order has the order identifier specified in the request:

  GetStatusResponse/orderMonitorSpecification/orderId

  Is equal to:

  GetStatus/orderId
- the returned order monitoring info shall include:
  - all information specified when the order was submitted
  - GetStatusResponse/orderMonitorSpecification/orderId
  - GetStatusResponse/orderMonitorSpecification/orderStatusInfo
  - GetStatusResponse/orderMonitorSpecification/orderDateTime
  - GetStatusResponse/orderMonitorSpecification/orderItem
  - GetStatusResponse/orderMonitorSpecification/orderItem/productId
- the returned order monitoring info shall include:
  - GetStatusResponse/orderMonitorSpecification/orderItem/sceneSelection

d) References:

- Req 101

e) Test type: Capability

A.3.4 Conformance Class SubscriptionOrder (http://www.opengis.net/spec/OSEO/1.0/conf/SubscriptionOrder)

This Conformance Class is in charge of verifying the compliance of the Order Server under test with respect to the **SubscriptionOrder** Requirement Class, which specifies all requirements an Order Server has to comply with for claiming the support of Subscriptions for Earth Observation Products.

A.3.4.1 GetCapabilities

a) Test id: http://www.opengis.net/spec/OSEO/1.0/conf/SubscriptionOrder/GetCapabilities

b) Test purpose:

To verify that the Order Server under test correctly supports the OSEO GetCapabilities operation returning the additional information necessary for Subscription Orders.

c) **Test method**:

Verify that the Order Server under test accepts a valid OSEO GetCapabilities request and returns a valid Capabilities element including:

- Either Capabilities/Contents/SubscriptionOrders/@supported = true
- Or at least one element Capabilities/Contents/SupportedCollection/SubscriptionOrders/@supported = true

Pass if the assertion is satisfied; fail otherwise.

d) **References**:

- Req 10

e) **Test type**: Capability

A.3.4.2 GetOptions – Subscription Order

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/SubscriptionOrder/GetOptions**

b) **Test purpose:**

To verify whether the Order Server under test supports the OSEO GetOptions operation for Subscription Orders.

c) **Test method:**

To send a correct GetOptions request via HTTP/XML/POST/SOAP 1.2 asking for the Order Options of one subscription i.e. send a message including 1 instance of:

GetOptions/collectionId

Verify that at least one order option is returned.

Verify that the order options are flagged for subscription orders i.e. the response messge shall include:

GetOptionsResponse/orderOptions/orderType = SUBSCRIPTION_ORDER

d) **References:**

- Req 29
- Req 30
- Req 31

e) **Test type: Capability**

A.3.4.3 Submit – Subscription Order

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/SubscriptionOrder/Submit**

b) **Test purpose:**

To verify that the Order Server under test is able to manage calls to OSEO Submit operation for Subscribing to Earth Observation Products.

c) **Test method:**

To send Submit requests via HTTP/XML/POST/SOAP 1.2:

- Including the order specification:

  Submit/orderSpecification

- Specifying subscription order:

  Submit/orderType = “SUBSCRIPTION_ORDER”

- Specifying one Subscription to Earth Observation Products:

  Submit/orderSpecification/orderItem/subscriptionId

- with no order status notification:

  Submit/statusNotification = None

- valid against oseo.xsd schema;

Verify that the returned SubmitAck:

- is valid against oseo.xsd XML Schema;
- SubmitAck/status = success**;**
- Has a unique order identifier:

  SubmitAck/orderId

d) **References:**

- Req 73

e) **Test type: Capability**

A.3.4.4 Submit - non nominal conditions

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/SubscriptionOrder/Submit-non-nominal**

b) **Test purpose:**

To verify that the Order Server under test is able to manage incorrect Subscription requests.

c) **Test method:**

To send Submit different requests via HTTP/XML/POST/SOAP 1.2 with different errors:

- Message with unsupported item:
  - Submit/orderSpecification/orderItem/subscriptionId/collectionId = WRONG_COLLECTION_ID
- Order Options different from those returned by a previous call to GetOptions.

Verify that a correct SOAP Fault is returned by Order Server under test.

d) **References:**

- Req 62

e) **Test type: Capability**

A.3.4.5 GetStatus – by ORDER ID, FULL

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/SubscriptionOrder/GetStatus-retrieve-full**

b) Test purpose:

To verify whether the Order Server under test is able to retrieve Subscription Orders by ID, FULL presentation.

c) Test method:

To send GetStatus request via HTTP/XML/POST/SOAP 1.2:

- Including the order identifier:

  GetStatus/orderId

  Specifying a value present in the Order Server in order to get a non empty answer.
- with FULL presentation:

  GetStatus/presentation = full
- valid against oseo.xsd schema;

Verify that the returned GetStatusResponse:

- is valid against oseo.xsd XML Schema;
- GetStatusResponse/status = success**;**
- includes ONE order, i.e. one instance of:

  GetStatusResponse/orderMonitorSpecification
- the returned order has the order identifier specified in the request:

  GetStatusResponse/orderMonitorSpecification/orderId

  Is equal to:

  GetStatus/orderId
- the returned order monitoring info shall include:
  - all information specified when the order was submitted
  - GetStatusResponse/orderMonitorSpecification/orderId
  - GetStatusResponse/orderMonitorSpecification/orderStatusInfo
  - GetStatusResponse/orderMonitorSpecification/orderDateTime
  - GetStatusResponse/orderMonitorSpecification/orderItem
  - GetStatusResponse/orderMonitorSpecification/orderItem/subscriptionId

d) References:

- Req 102

e) Test type: Capability

A.3.5 Conformance Class TaskingOrder (http://www.opengis.net/spec/OSEO/1.0/conf/TaskingOrder)

This Conformance Class is in charge of verifying the compliance of the Order Server under test with respect to the **TaskingOrder** Requirement Class, which specifies all requirements an Order Server has to comply with for claiming the support of Tasking Orders for Earth Observation Products.

A.3.5.1 GetCapabilities

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/TaskingOrder/GetCapabilities**

b) **Test purpose**:

To verify that the Order Server under test correctly supports the OSEO GetCapabilities operation returning the additional information necessary for Tasking Orders.

c) **Test method**:

Verify that the Order Server under test accepts a valid OSEO GetCapabilities request and returns a valid Capabilities element including:

□ Capabilities/Contents/ProgrammingOrders/@supported = true and Capabilities/Contents/ProgrammingOrders/@SPS_URL set.

Pass if the assertion is satisfied; fail otherwise.

d) **References**:

□ Req 11

e) **Test type**: Capability

A.3.5.2 GetOptions – By TASK ID

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/TaskingOrder/GetOptions**

b) **Test purpose:**

To verify whether the Order Server under test supports the OSEO GetOptions operation for tasking orders.

c) **Test method:**

To send a GetOptions request via HTTP/XML/POST/SOAP 1.2 asking for the Order Options of 1 tasking request i.e. send a message including 1 instance of:

GetOptions/taskingRequest/ID

Verify that at least one order option is returned.

Verify that the order options are flagged for product orders i.e. the response messge shall include:

GetOptionsResponse/orderOptions/orderType = PRODUCT_ORDER

d) **References:**

□ Req 32

□ Req 33

□ Req 34

e) **Test type: Capability**

A.3.5.3 Submit – By TASK ID

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/TaskingOrder/Submit**

b) **Test purpose:**

To verify that the Order Server under test is able to manage calls to OSEO Submit operation for ordering future products via a tasking request.

c) **Test method:**

To send Submit requests via HTTP/XML/POST/SOAP 1.2:

- Including the order specification:

  Submit/orderSpecification

- Specifying tasking order:

  Submit/orderType = "TASKING_ORDER"

- Specifying one tasking request:

  Submit/orderSpecification/orderItem/taskingRequestId

- with no order status notification:

  Submit/statusNotification = None

- valid against oseo.xsd schema;

Verify that the returned SubmitAck:

- is valid against oseo.xsd XML Schema;
- SubmitAck/status = success**;**
- Has a unique order identifier:

  SubmitAck/orderId

d) **References:**

- Req 74

e) **Test type: Capability**

A.3.5.4 Submit - non nominal conditions

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/TaskingOrder/Submit-non-nominal**

b) **Test purpose:**

To verify whether the Order Server under test is able to manage incorrect order for future products.

c) **Test method:**

To send Submit request via HTTP/XML/POST/SOAP 1.2 with error:

- Message with unsupported item:
  - Submit/orderSpecification/orderItem/taskingRequestId/ID = WRONG_TASK_ID
- Order Options different from those returned by a previous call to GetOptions.

Verify that a correct SOAP Fault is returned by Order Server under test.

d) **References:**

- Req 62

e) **Test type: Capability**

A.3.5.5 GetStatus – by ORDER ID, FULL

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/TaskingOrder/GetStatus-retrieve-full**

b) Test purpose:

To verify whether the Order Server under test is able to retrieve Tasking Orders by ID, FULL presentation.

c) Test method:

To send GetStatus request via HTTP/XML/POST/SOAP 1.2:

- Including the order identifier:

  GetStatus/orderId

  Specifying a value present in the Order Server in order to get a non empty answer.
- with FULL presentation:

  GetStatus/presentation = full
- valid against oseo.xsd schema;

Verify that the returned GetStatusResponse:

- is valid against oseo.xsd XML Schema;
- GetStatusResponse/status = success**;**
- includes ONE order, i.e. one instance of:

  GetStatusResponse/orderMonitorSpecification
- the returned order has the order identifier specified in the request:

  GetStatusResponse/orderMonitorSpecification/orderId

  Is equal to:

  GetStatus/orderId
- the returned order monitoring info shall include:
  - all information specified when the order was submitted
  - GetStatusResponse/orderMonitorSpecification/orderId
  - GetStatusResponse/orderMonitorSpecification/orderStatusInfo
  - GetStatusResponse/orderMonitorSpecification/orderDateTime
  - GetStatusResponse/orderMonitorSpecification/orderItem
  - GetStatusResponse/orderMonitorSpecification/orderItem/taskingRequestId

d) References:

- Req 103

e) Test type: Capability

A.3.6 Conformance Class AsyncSubmit (http://www.opengis.net/spec/OSEO/1.0/conf/AsyncSubmit)

This Conformance Class is in charge of verifying the compliance of the Order Server under test with respect to the **AsyncSubmit** Requirement Class.

A server conforming with this class is able to accept orders and reply with asynchronous notifications informing the client about the progress of the submitted order.

A.3.6.1 GetCapabilities

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/AsyncSubmit/GetCapabilities**

b) **Test purpose**: To verify that the Order Server under test correctly supports the OSEO GetCapabilities operation returning the additional information necessary for asynchronous order submission.

c) **Test method**:

Verify that the Order Server under test accepts a valid OSEO GetCapabilities request and returns a valid Capabilities element including:

- Capabilities/Contents/SubmitCapabilities/@asynchronous = true

Pass if the assertion is satisfied; fail otherwise.

d) **References**:

- Req 12

e) **Test type**: Capability

A.3.6.2 Submit – Single Notification

a) Test id: http://www.opengis.net/spec/OSEO/1.0/conf/AsyncSubmit/Submit-single

b) Test purpose:

To verify that the Order Server under test is able to manage calls to OSEO Submit operation (any supported order type) asking for a single asynchronous notification at order completion.

c) Test method:

To send Submit requests via HTTP/XML/POST/SOAP 1.2:

- Including an order (any supported order type)
- with order status notification:

  Submit/statusNotification = Final
- with reply address pointing to a valid service:

  soapenv:Envelope/soapenv:Header/wsa:ReplyTo
- valid against oseo.xsd schema;

Verify that the returned SubmitAck:

- is valid against oseo.xsd XML Schema;
- SubmitAck/status = success;
- has a unique order identifier:

  SubmitAck/orderId

d) References:

- Req 63

e) Test type: Capability

A.3.6.3 SubmitResponse – Single Notification

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/AsyncSubmit/SubmitResponse-single**

b) **Test purpose:**

To verify that the Order Server under test is able to send a Single Notification – at order completion – to the specified reply address.

c) **Test method:**

To verify whether the Order Server has called the SubmitResponse operation on the reply address specified at §A.3.6.2 and to verify that:

- The SubmitResponse message is valid against oseo.xsd schema;
- the SubmitResponse/orderMonitorSpecification/orderId is the same as returned in SubmitAck §A.3.6.2;
- SubmitResponse/timestamp is set with the date & time of the order status change;
- only ONE call has been done to SubmitResponse for that order identifier;
- the returned order items are the same as specified in Submit of §A.3.6.2.

Verify that the received SubmitResponseAck:

- SubmitResponseAck/status = success;

d) **References:**

- Req 65
- Req 66
- Req 68
- Req 69

e) **Test type: Capability**

A.3.6.4 Submit –Multiple Notifications

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/AsyncSubmit/Submit-multiple**

b) **Test purpose:**

To verify that the Order Server under test is able to manage calls to OSEO Submit operation (any supported order type) asking for multiple asynchronous notification until order completion.

c) **Test method:**

To send Submit requests via HTTP/XML/POST/SOAP 1.2:

- Including an order (any supported order type)
- with order status notification:

  Submit/statusNotification = All
- with reply address pointing to a valid service:

  soapenv:Envelope/soapenv:Header/wsa:ReplyTo
- valid against oseo.xsd schema;

Verify that the returned SubmitAck:

- is valid against oseo.xsd XML Schema;

- SubmitAck/status = success;
- has a unique order identifier:

SubmitAck/orderId

d) **References:**

- Req 64

e) **Test type: Capability**

A.3.6.5 SubmitResponse – Multiple Notifications

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/AsyncSubmit/SubmitResponse-multiple**

b) **Test purpose:**

To verify that the Order Server under test is able to send Multiple Notifications about the order status change to the specified reply address until order completion.

c) **Test method:**

To verify whether the Order Server has called the SubmitResponse operation on the reply address specified at §A.3.6.2 and to verify that:

- The SubmitResponse message is valid against oseo.xsd schema;
- the SubmitResponse/orderMonitorSpecification/orderId is the same as returned in SubmitAck §A.3.6.2;
- SubmitResponse/timestamp is set with the date & time of the order status change;
- AT LEAST one call has been done to SubmitResponse for that order identifier;
- the returned order items are the same as specified in Submit of §A.3.6.2.

Verify that the received SubmitResponseAck:

- SubmitResponseAck/status = success;

d) **References:**

- Req 67
- Req 68

e) **Test type: Capability**

A.3.7 Conformance Class Quotation (http://www.opengis.net/spec/OSEO/1.0/conf/Quotation)

This Conformance Class is in charge of verifying the compliance of the Order Server under test with respect to the **Quotation** Requirement Class.

A server complying with this class is able to accept payment options, is able to provide the quotation of orders going to be submitted and it is able to accept submission of already quoted orders.

A.3.7.1 GetCapabilities

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Quotation/GetCapabilities**

b) **Test purpose**:

To verify that the Order Server under test correctly supports the OSEO GetCapabilities operation returning the additional information notifying the support of order quotation.

c) **Test method**:

Verify that the Order Server under test accepts a valid OSEO GetCapabilities request and returns a valid Capabilities element including:

□ Capabilities/Contents/GetQuotationCapabilities/@supported = true

and at least one of:

□ Capabilities/Contents/GetQuotationCapabilities/@synchronous

□ Capabilities/Contents/GetQuotationCapabilities/@asynchronous

□ Capabilities/Contents/GetQuotationCapabilities/@monitoring

□ Capabilities/Contents/GetQuotationCapabilities/@off-line

set to true.

Pass if the assertion is satisfied; fail otherwise.

d) **References**:

□ Req 13

e) **Test type**: Capability

A.3.7.2 GetOptions – Payment Options

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Quotation/GetOptions**

b) **Test purpose:**

To verify whether the Order Server under test supports payment options in OSEO GetOptions operation.

c) **Test method:**

To send a GetOptions request via HTTP/XML/POST/SOAP 1.2 asking for the Order Options of any supported item (either an EO product or a subscription or a tasking).

□ Verify that at least one order option is returned.

□ Verify that at least one instance of the following element is returned:

GetOptionsResponse/orderOptions/paymentOptions

d) **References:**

□ Req 35

e) **Test type: Capability**

A.3.7.3 GetQuotation - Basic

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Quotation/GetQuotation**

b) **Test purpose:**

To verify whether the Order Server under test supports the OSEO GetQuotation operation.

c) **Test method:**

To send a correct GetQuotation request (i.e. valid against GetQuotation element of oseo.xsd) for an item supported by the Order Server (either a product or a collection or a tasking request) via HTTP/XML/POST/SOAP 1.2.

Verify that the returned GetQuotationAck is valid against oseo.xsd XML Schema.

d) **References:**

- Req 39
- Req 40
- Req 41
- Req 42

e) **Test type: Capability**

A.3.7.4 GetQuotation - non nominal conditions

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Quotation/GetQuotation-non-nominal**

b) **Test purpose:**

To verify whether the Order Server under test is able to handle incorrect calls to OSEO GetQuotation operation.

c) **Test method:**

To send GetQuotation requests via HTTP/XML/POST/SOAP 1.2 with errors:

- Message with wrong element: GetQuotation including the following wrong element GetQuotation/WRONG_TAG.
- Message with unsupported item:
    - GetQuotation/quotationId = WRONG_QUOTATION_ID

Verify that a correct SOAP Fault is returned by Order Server under test.

d) **References:**

- Req 43

e) **Test type: Capability**

A.3.7.5 Submit – By quotation ID

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Quotation/Submit-quotation-id**

b) **Test purpose:**

To verify whether the Order Server under test is able to manage calls to OSEO Submit operation for already quoted orders.

c) **Test method:**

To send Submit requests via HTTP/XML/POST/SOAP 1.2:

- Including a valid quotation identifier returned by a previous call to GetQuotation operation: Submit/quotationId

- with no order status notification:

  Submit/statusNotification = None
- valid against oseo.xsd schema;

Verify that the returned SubmitAck:

- is valid against oseo.xsd XML Schema;
- SubmitAck/status = success**;**
- Has a unique order identifier:

  SubmitAck/orderId

d) References:

- Req 75

e) Test type: Capability

A.3.8 Conformance Class QuotationSync (http://www.opengis.net/spec/OSEO/1.0/conf/QuotationSync)

This Conformance Class is in charge of verifying the compliance of the Order Server under test with respect to the **QuotationSync** Requirement Class.

A server complying with this conformance class is able to quote orders in real time, without the need to wait for a delayed answer.

A.3.8.1 GetCapabilities

a) Test id: http://www.opengis.net/spec/OSEO/1.0/conf/QuotationSync/GetCapabilities

b) Test purpose:

To verify that the Order Server under test correctly supports the OSEO GetCapabilities operation returning the additional information notifying the support for synchronous order quotation.

c) Test method:

Verify that the Order Server under test accepts a valid OSEO GetCapabilities request and returns a valid Capabilities element including:

- Capabilities/Contents/GetQuotationCapabilities/@supported = true
- Capabilities/Contents/GetQuotationCapabilities/@synchronous = true

Pass if the assertion is satisfied; fail otherwise.

d) References:

- Req 14

e) Test type: Capability

A.3.8.2 GetQuotation - synchronous

a) Test id: http://www.opengis.net/spec/OSEO/1.0/conf/QuotationSync/GetQuotation-sync

b) Test purpose:

To verify whether the Order Server under test is able to handle synchronous calls to OSEO GetQuotation operation (any type of order).

c) **Test method:**

To send GetQuotation requests via HTTP/XML/POST/SOAP 1.2 including the order specification (any supported order type):

- GetQuotation/orderSpecification

Verify that the returned GetQuotationAck:

- is valid against oseo.xsd XML Schema;
- GetQuotationAck/status = **success;**
- GetQuotationAck/quotation is non empty and reports the order quotation;

d) **References:**

- Req 44

e) **Test type: Capability**

A.3.9 Conformance Class QuotationMonitoring (http://www.opengis.net/spec/OSEO/1.0/conf/QuotationMonitoring)

This Conformance Class is in charge of verifying the compliance of the Order Server under test with respect to the **QuotationMonitoring** Requirement Class.

A server complying with this conformance class allows a client to ask and then retrieve the quotation of orders going to be submitted.

A.3.9.1 GetCapabilities

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/QuotationMonitoring/GetCapabilities**

b) **Test purpose**:

To verify that the Order Server under test correctly supports the OSEO GetCapabilities operation returning the additional information notifying the support for monitoring order quotation.

c) **Test method**:

Verify that the Order Server under test accepts a valid OSEO GetCapabilities request and returns a valid Capabilities element including:

- Capabilities/Contents/GetQuotationCapabilities/@supported = true
- Capabilities/Contents/GetQuotationCapabilities/@monitoring = true

Pass if the assertion is satisfied; fail otherwise.

d) **References**:

- Req 15

e) **Test type**: Capability

A.3.9.2 GetQuotation - asynchronous with monitoring – initial call

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/QuotationMonitoring/GetQuotation-mon**

b) **Test purpose:**

To verify whether the Order Server under test is able to handle asynchronous with monitoring calls to OSEO GetQuotation operation (any type of order).

c) **Test method:**

To send GetQuotation requests via HTTP/XML/POST/SOAP 1.2 including the order specification (any supported order type):

- GetQuotation/orderSpecification
- Empty reply address:

  soapenv:Envelope/soapenv:Header/wsa:ReplyTo =
  http://www.w3.org/2005/08/addressing/anonymous

Verify that the returned GetQuotationAck:

- is valid against oseo.xsd XML Schema;
- GetQuotationAck/status = **success;**
- GetQuotationAck/quotationId is non empty and reports the quotation identifier;

d) **References:**

- Req 45

e) **Test type: Capability**

A.3.9.3 GetQuotation - asynchronous with monitoring – next call

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/QuotationMonitoring/GetQuotation-mon-next-call**

b) **Test purpose:**

To verify whether the Order Server under test is able to handle asynchronous with monitoring calls to OSEO GetQuotation operation (any type of order).

c) **Test method:**

To send GetQuotation requests via HTTP/XML/POST/SOAP 1.2 including:

- a valid quotation identifier got from a previous GetQuotation call:

  GetQuotation/quotationId
- Empty reply address:

  soapenv:Envelope/soapenv:Header/wsa:ReplyTo =
  http://www.w3.org/2005/08/addressing/anonymous

Verify that the returned GetQuotationAck:

- is valid against oseo.xsd XML Schema;
- GetQuotationAck/status = **success;**
- Either GetQuotationAck/quotationId is non empty and reports the quotation identifier. If it is the case repeat the test.
- Or GetQuotationAck/quotation is filled with the order quotation.

The test is successful when a GetQuotationAck/quotation properly filled is received.

d) **References:**

- Req 46
- Req 47

e) **Test type: Capability**

A.3.10 Conformance Class QuotationAsync (http://www.opengis.net/spec/OSEO/1.0/conf/QuotationAsync)

This Conformance Class is in charge of verifying the compliance of the Order Server under test with respect to the **QuotationAsync** Requirement Class.

A server complying with this conformance class provides the order quotation to the users with a later notification.

A.3.10.1 GetQuotation - asynchronous with notification – initial call

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/QuotationAsync/GetQuotation-init**

b) **Test purpose:**

To verify whether the Order Server under test is able to handle asynchronous calls with notification to OSEO GetQuotation operation (any type of order).

c) **Test method:**

To send GetQuotation requests via HTTP/XML/POST/SOAP 1.2 including the order specification (any supported order type):

- GetQuotation/orderSpecification
- reply address, which is the address the server will send the quotation, set in the following tag:

soapenv:Envelope/soapenv:Header/wsa:ReplyTo

Verify that the returned GetQuotationAck:

- is valid against oseo.xsd XML Schema;
- GetQuotationAck/status = **success;**
- GetQuotationAck/quotationId is non empty and reports the quotation identifier;

d) **References:**

- Req 48
- Req 49

e) **Test type: Capability**

A.3.10.2 GetQuotation - asynchronous with notification – notification

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/QuotationAsync/GetQuotation-notification**

b) Test purpose:

To verify whether the Order Server under test is able to send asynchronous notifications reporting the order quotation.

c) Test method:

To verify that a GetQuotationResponse message has been sent to the address specified in the test at §A.3.10.1 complying with the following conditions:

- It is valid against GetQuotationResponse element of oseo.xsd XML Schema;
- GetQuotationResponse/status = **success;**
- GetQuotationResponse/quotation reports the order quotation.

Verify that the returned GetQuotationResponseAck complies with the following conditions:

- It is valid against GetQuotationResponseAck element of oseo.xsd XML Schema;
- GetQuotationResponseAck/status = **success;**

d) References:

- Req 50
- Req 51
- Req 52

e) Test type: Capability

A.3.11 Conformance Class QuotationOffLine (http://www.opengis.net/spec/OSEO/1.0/conf/QuotationOffLine)

This Conformance Class is in charge of verifying the compliance of the Order Server under test with respect to the **QuotationOffLine** Requirement Class.

A server complying with this conformance class provides the order quotation to the users off-line via mail or other non-on-line mechanisms.

A.3.11.1 GetQuotation – off-line

a) Test id: http://www.opengis.net/spec/OSEO/1.0/conf/QuotationOffLine/GetQuotation

b) Test purpose:

To verify whether the Order Server under test is able to handle calls to OSEO GetQuotation operation (any type of order) asking for off-line quotation.

c) Test method:

To send GetQuotation requests via HTTP/XML/POST/SOAP 1.2 including the order specification (any supported order type):

- GetQuotation/orderSpecification
- Empty reply address:

  soapenv:Envelope/soapenv:Header/wsa:ReplyTo =
  http://www.w3.org/2005/08/addressing/anonymous

Verify that the returned GetQuotationAck:

- is valid against oseo.xsd XML Schema;
- GetQuotationAck/status = **success;**
- GetQuotationAck/quotationId is non empty and reports the quotation identifier;

Verify that a document including the order quotation is sent via mail to the following address:

- GetQuotation/orderSpecification/invoiceAddress

d) References:

- Req 53
- Req 54
- Req 55

e) Test type: Capability

A.3.12 Conformance Class OnlineDataAccess (http://www.opengis.net/spec/OSEO/1.0/conf/OnlineDataAccess)

This Conformance Class is in charge of verifying the compliance of the Order Server under test with respect to the **OnlineDataAccess** Requirement Class.

A server complying with this conformance class allows the users to submit orders having on-line delivery. Then the retrieval of ordered products has to be done by calling a dedicated operation returning the corresponding URLs.

A.3.12.1 GetCapabilities

a) Test id: http://www.opengis.net/spec/OSEO/1.0/conf/OnlineDataAccess/GetCapabilities

b) Test purpose:

To verify that the Order Server under test correctly supports the OSEO GetCapabilities operation returning the additional information notifying the support of online access.

c) Test method:

Verify that the Order Server under test accepts a valid OSEO GetCapabilities request and returns a valid Capabilities element including:

- Either Capabilities/Contents/ DescribeResultAccessCapabilities/@supported = true
- Or at least one element:

  Capabilities/Contents/SupportedCollection/DescribeResultAccessCapabilities/@supported = true

Pass if the assertion is satisfied; fail otherwise.

d) References:

- Req 16

e) Test type: Capability

A.3.12.2 GetOptions – online data access options

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/OnlineDataAccess/GetOptions**

b) **Test purpose:**

To verify whether the Order Server under test supports the OSEO GetOptions operation for orders with online access.

c) **Test method:**

To send a GetOptions request via HTTP/XML/POST/SOAP 1.2 asking for the Order Options of a supported item (either a product, or a collection, or a subscription, or a tasking request).

Verify that at least one order option is returned.

Verify that the order options are flagged for on line data access i.e. the response message shall include at least one instance of:

GetOptionsResponse/orderOptions/productDeliveryOptions/onlineDataAccess/protocol

d) **References:**

- Req 36

e) **Test type: Capability**

A.3.12.3 Submit – on line data access

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/OnlineDataAccess/Submit**

b) **Test purpose:**

To verify whether the Order Server under test is able to manage calls to OSEO Submit operation for products with online access.

c) **Test method:**

To send Submit requests via HTTP/XML/POST/SOAP 1.2:

- Including at least 2 order items, in order to allow the retrieval of them with 2 separated calls to DescribeResultAccess (see A.3.12.4, A.3.12.5).
- Including options for online access:

  Either:

  Submit/orderSpecification/deliveryOptions/onlineDataAccess/protocol

  Or:

  Submit/orderSpecification/orderItem/deliveryOptions/onlineDataAccess/protocol

  For both items.
- with no order status notification:

  Submit/statusNotification = None
- valid against oseo.xsd schema;

Verify that the returned SubmitAck:

- is valid against oseo.xsd XML Schema;
- SubmitAck/status = success**;**
- Has a unique order identifier:

SubmitAck/orderId

d) **References:**

- Req 76

e) **Test type: Capability**

A.3.12.4 DescribeResultAccess – retrieve all available items

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/OnlineDataAccess/DRA-all**

b) **Test purpose:**

To verify whether the Order Server under test is able to manage calls to OSEO DescribeResultAccess operation with subFunction = allReady.

c) **Test method:**

To send DescribeResultAccess requests via HTTP/XML/POST/SOAP 1.2:

- Including the identifier of an order with on line access (i.e. the order submitted at §A.3.12.3):

  DescribeResultAccess/orderId

- Specifying sub-function all ready:

  DescribeResultAccess/subFunction = allReady

- valid against oseo.xsd schema;

Verify that the returned DescribeResultAccessResponse:

- is valid against oseo.xsd XML Schema;
- DescribeResultAccessResponse/status = success**;**
- For each ordered item ready for download one instance of the following element shall be returned:

  DescribeResultAccessResponse/URLs

- If there are items with limited time availability, then the following element has to be provided:

  DescribeResultAccessResponse/URLs/expirationDate

d) **References:**

- Req 104
- Req 105
- Req 106
- Req 108
- Req 109
- Req 112

e) **Test type: Capability**

A.3.12.5 DescribeResultAccess – retrieve new available items

a) Test id: http://www.opengis.net/spec/OSEO/1.0/conf/OnlineDataAccess/DRA-new

b) Test purpose:

To verify whether the Order Server under test is able to manage calls to OSEO DescribeResultAccess operation with subFunction = nextReady.

c) Test method:

To send DescribeResultAccess requests via HTTP/XML/POST/SOAP 1.2:

- Including the identifier of the order with on line access used in the previous test §A.3.12.3:

  DescribeResultAccess/orderId

- Specifying sub-function next ready:

  DescribeResultAccess/subFunction = nextReady

- valid against oseo.xsd schema;

Verify that the returned DescribeResultAccessResponse:

- is valid against oseo.xsd XML Schema;
- DescribeResultAccessResponse/status = success**;**
- There are new items, different with respect to the previous call (§A.3.12.4), reported in the following element:

  DescribeResultAccessResponse/URLs

Note: the call with subFunction = nextReady has to be performed with care otherwise no items are returned. In fact: if the previous test (A.3.12.4) is executed when the Order Server has processed all items, then no items are returned by this call.

d) References:

- Req 104
- Req 105
- Req 107
- Req 108
- Req 110

e) Test type: Capability

A.3.12.6 DescribeResultAccess – non nominal conditions

a) Test id: http://www.opengis.net/spec/OSEO/1.0/conf/OnlineDataAccess/DRA-non-nominal

b) Test purpose:

To verify whether the Order Server under test is able to manage incorrect calls to OSEO DescribeResultAccess operation.

c) Test method:

To send several DescribeResultAccess requests via HTTP/XML/POST/SOAP 1.2 including the following errors:

- Including the identifier of a non existing order:

DescribeResultAccess/orderId = NOT_EXISTING_ORDER

- Including a wrong tag:

DescribeResultAccess/WRONG_TAG

Verify that the returned DescribeResultAccessResponse:

- Reports a SOAP Fault

d) **References:**

- Req 111

e) **Test type: Capability**

A.3.13 Conformance Class OnlineDataDelivery (http://www.opengis.net/spec/OSEO/1.0/conf/OnlineDataDelivery)

This Conformance Class is in charge of verifying the compliance of the Order Server under test with respect to the **OnlineDataDelivery** Requirement Class.

A server complying with this conformance class allows the users to submit orders having on-line delivery, but the address where the ordered products are made available is specified by the client itself and then it has not to be retrieved from the Order Server.

A.3.13.1 GetOptions – online delivery options

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/OnlineDataDelivery/GetOptions**

b) **Test purpose:**

To verify whether the Order Server under test supports the OSEO GetOptions operation for orders with online delivery.

c) **Test method:**

To send a GetOptions request via HTTP/XML/POST/SOAP 1.2 asking for the Order Options of a supported item (either a product, or a collection, or a subscription, or a tasking request).

Verify that at least one order option is returned.

Verify that the order options are flagged for on line data delivery i.e. the response message shall include at least one instance of:

GetOptionsResponse/orderOptions/productDeliveryOptions/onlineDataDelivery/protocol

d) **References:**

- Req 37

e) **Test type: Capability**

A.3.13.2 Submit – on line delivery

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/OnlineDataDelivery/Submit**

b) **Test purpose:**

To verify whether the Order Server under test is able to manage calls to OSEO Submit operation for products with online delivery.

c) **Test method:**

To send Submit requests via HTTP/XML/POST/SOAP 1.2:

- Including options for online access:

  Either:

  Submit/orderSpecification/deliveryOptions/onlineDataDelivery/protocol

  Or:

  Submit/orderSpecification/orderItem/deliveryOptions/onlineDataDelivery/protocol
- Additionally the delivery address must be specified:

  Submit/orderSpecification/deliveryInformation/onlineAddress
- with no order status notification:

  Submit/statusNotification = None
- valid against oseo.xsd schema;

Verify that the returned SubmitAck:

- is valid against oseo.xsd XML Schema;
- SubmitAck/status = success**;**
- Has a unique order identifier:

  SubmitAck/orderId

d) **References:**

- Req 77

e) **Test type: Capability**

A.3.14 Conformance Class MediaDelivery (http://www.opengis.net/spec/OSEO/1.0/conf/MediaDelivery)

This Conformance Class is in charge of verifying the compliance of the Order Server under test with respect to the **MediaDelivery** Requirement Class.

A server complying with this conformance class allows the users to submit orders having delivery on media.

A.3.14.1 GetOptions – media delivery options

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/MediaDelivery/GetOptions**

b) **Test purpose:**

To verify whether the Order Server under test supports the OSEO GetOptions operation for orders with delivery on media.

c) **Test method:**

To send a GetOptions request via HTTP/XML/POST/SOAP 1.2 asking for the Order Options of a supported item (either a product, or a collection, or a subscription, or a tasking request).

Verify that at least one order option is returned.

Verify that the order options are flagged for delivery on media i.e. the response message shall include at least one instance of:

GetOptionsResponse/orderOptions/productDeliveryOptions/mediaDelivery/packageMedium

d) References:

□ Req 38

e) Test type: Capability

A.3.14.2 Submit – delivery on media

a) Test id: http://www.opengis.net/spec/OSEO/1.0/conf/MediaDelivery/Submit

b) Test purpose:

To verify whether the Order Server under test is able to manage calls to OSEO Submit operation for products with delivery on media.

c) Test method:

To send Submit requests via HTTP/XML/POST/SOAP 1.2:

□ Including options for online access:

Either:

Submit/orderSpecification/deliveryOptions/mediaDelivery/packageMedium

Or:

Submit/orderSpecification/orderItem/deliveryOptions/mediaDelivery/packageMedium

□ with no order status notification:

Submit/statusNotification = None

□ valid against oseo.xsd schema;

Verify that the returned SubmitAck:

□ is valid against oseo.xsd XML Schema;

□ SubmitAck/status = success**;**

□ Has a unique order identifier:

SubmitAck/orderId

d) References:

□ Req 78

e) Test type: Capability

A.3.15 Conformance Class Cancellation (http://www.opengis.net/spec/OSEO/1.0/conf/Cancellation)

This Conformance Class is in charge of verifying the compliance of the Order Server under test with respect to the **Cancellation** Requirement Class.

A server compliant with this class allows the on-line cancellation of submitted orders.

A.3.15.1 GetCapabilities

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Cancellation/GetCapabilities**

b) **Test purpose**:

To verify that the Order Server under test correctly supports the OSEO GetCapabilities operation returning the additional information notifying the support of order cancellation.

c) **Test method**:

Verify that the Order Server under test accepts a valid OSEO GetCapabilities request and returns a valid Capabilities element including:

- Either Capabilities/Contents/CancelCapabilities/@supported = true
- Or at least one element Capabilities/Contents/SupportedCollection/CancelCapabilities/@supported = true

Pass if the assertion is satisfied; fail otherwise.

d) **References**:

- Req 17

e) **Test type**: Capability

A.3.15.2 Cancel – no notification

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Cancellation/Cancel**

b) **Test purpose:**

To verify whether the Order Server under test is able to manage synchronous calls to OSEO Cancel operation (any supported order type).

c) **Test method:**

To send Cancel request via HTTP/XML/POST/SOAP 1.2:

- Including the identifier of an existing order (e.g. returned by test A.3.1.5):

  Cancel/orderId

- Without notification:

  Cancel/statusNotification = None

  soapenv:Envelope/soapenv:Header/wsa:ReplyTo = http://www.w3.org/2005/08/addressing/anonymous

- valid against oseo.xsd schema;

Verify that the returned CancelAck:

- is valid against oseo.xsd XML Schema;
- CancelAck/status = success**;**

d) **References:**

- Req 113
- Req 114
- Req 115

- Req 116
- Req 117

e) **Test type: Capability**

A.3.15.3 Order Cancellation Monitoring

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Cancellation/Cancel-monitoring**

b) **Test purpose:**

To verify whether the Order Server under test is actually cancelling the order specified at test A.3.15.2. The verification is performed checking the order status.

c) **Test method:**

To send GetStatus request via HTTP/XML/POST/SOAP 1.2:

- Including the identifier of the order cancelled at previous test A.3.15.2.

  GetStatus/orderId

- Presentation full:

  GetStatus/presentation = full

Verify that the returned GetStatusResponse reports:

- GetStatusResponse/status = success**;**
- GetStatusResponse/orderMonitorSpecification/orderId = GetStatus/orderId
- GetStatusResponse/orderMonitorSpecification/orderStatusInfo/status = Cancelled

  If the order status is not "Cancelled", it means that the server has not yet cancelled the order then repeat the test.
- GetStatusResponse/orderMonitorSpecification/orderItem/orderItemStatusInfo/status = Cancelled

  For at least one item.

The test is successful if:

GetStatusResponse/orderMonitorSpecification/orderStatusInfo/status = Cancelled

d) **References:**

- Req 118

e) **Test type: Capability**

A.3.15.4 Cancel - non nominal conditions

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Cancellation/Cancel-non-nominal**

b) **Test purpose:**

To verify whether the Order Server under test is able to handle incorrect calls to OSEO Cancel operation.

c) **Test method:**

To send Cancel requests via HTTP/XML/POST/SOAP 1.2 with errors:

- Message with wrong element:
  - Cancel including the following wrong element

    Cancel/WRONG_TAG.
- Message with non existing order:
  - Cancel/orderId = WRONG_ORDER_ID
- To send the cancellation for an order that cannot be cancelled (it depends on the current status of the order: the Order Server can refuse cancellation if the resources for processing the order have been already allocated).

Verify that a correct SOAP Fault is returned by the Order Server under test.

d) **References:**

- Req 119

e) **Test type: Capability**

A.3.16 Conformance Class AsyncCancellation (http://www.opengis.net/spec/OSEO/1.0/conf/AsyncCancellation)

This Conformance Class is in charge of verifying the compliance of the Order Server under test with respect to the **AsyncCancellation** Requirement Class.

A.3.16.1 Cancel – Single Notification

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/AsyncCancellation/Cancel-single**

b) **Test purpose:**

To verify whether the Order Server under test is able to manage calls to OSEO Cancel operation (any supported order type) asking for a single asynchronous notification at cancellation completion.

c) **Test method:**

To send Cancel requests via HTTP/XML/POST/SOAP 1.2:

- Including the identifier of an existing order (any supported order type)

  Cancel/orderId
- with order status notification:

  Cancel/statusNotification = Final
- with reply address pointing to a valid service:

  soapenv:Envelope/soapenv:Header/wsa:ReplyTo
- valid against oseo.xsd schema;

Verify that the returned CancelAck:

- is valid against oseo.xsd XML Schema;
- CancelAck/status = success;

d) **References:**

- Req 120

e) **Test type: Capability**

A.3.16.2 CancelResponse – Single Notification

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/AsyncCancellation/CancelResponse-single**

b) **Test purpose:**

To verify whether the Order Server under test is able to send a Single Notification – at cancellation completion – to the specified reply address.

c) **Test method:**

To verify whether the Order Server has called the CancelResponse operation on the reply address specified at §A.3.16.1 and verify that:

- □ CancelResponse message is valid against oseo.xsd schema;
- □ CancelResponse/orderMonitorSpecification/orderId is the same as that specified in:

  Cancel/orderId §A.3.16.1
- □ CancelResponse/orderMonitorSpecification/orderStatusInfo/status = Cancelled
- □ CancelResponse/timestamp is set with the date & time of the order status change;
- □ only ONE call has been done to CancelResponse for that order identifier;

Verify that the Order Server accepts the acknowledge on the asynchronous notification (CancelResponseAck).

If no notification has been received, it may happen depending on the Cancellation performance of the Order Server, then repeat this test again.

d) **References:**

- □ Req 120
- □ Req 122
- □ Req 123
- □ Req 125
- □ Req 126

e) **Test type: Capability**

A.3.16.3 Cancel –Multiple Notifications

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/AsyncCancellation/Cancel-multiple**

b) **Test purpose:**

To verify whether the Order Server under test is able to manage calls to OSEO Cancel operation (any supported order type) asking for multiple asynchronous notification until cancellation completion.

c) **Test method:**

To send Cancel requests via HTTP/XML/POST/SOAP 1.2:

- Including the identifier of an existing order (any supported order type)

  Cancel/orderId

- with order status notification:

  Cancel/statusNotification = All

- with reply address pointing to a valid service:

  soapenv:Envelope/soapenv:Header/wsa:ReplyTo

- valid against oseo.xsd schema;

Verify that the returned CancelAck:

- is valid against oseo.xsd XML Schema;
- CancelAck/status = success;

d) **References:**

- Req 121

e) **Test type: Capability**

A.3.16.4 CancelResponse – Multiple Notifications

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/AsyncCancellation/CancelResponse-multiple**

b) **Test purpose:**

To verify whether the Order Server under test is able to send Multiple Notifications about the order status change to the specified reply address until cancellation completion.

c) **Test method:**

To verify whether the Order Server has called the CancelResponse operation on the reply address specified at §A.3.16.3 and to verify that:

- The CancelResponse message is valid against oseo.xsd schema;
- CancelResponse/orderMonitorSpecification/orderId is the same as that specified in:

  Cancel/orderId §A.3.16.3

- CancelResponse/timestamp is set with the date & time of the order status change;

If CancelResponse/orderMonitorSpecification/orderStatusInfo/status = Cancelled

Then the test is successful,

Else this test has to be executed again waiting for that the Order Server under test has actually cancelled the order.

Verify that the Order Server accept the acknowledge on the asynchronous notification (CancelResponseAck).

d) **References:**

- Req 121
- Req 122
- Req 124
- Req 125

□ Req 126

e) **Test type: Capability**

A.3.17 Conformance Class Notification (http://www.opengis.net/spec/OSEO/1.0/conf/Notification)

This Conformance Class is in charge of verifying the compliance of the Order Server under test with respect to the **Notification** Requirement Class.

A.3.17.1 GetCapabilities

a) **Test id: http://www.opengis.net/spec/OSEO/1.0/conf/Notification/GetCapabilities**

b) **Test purpose**:

To verify that the Order Server under test correctly supports the OSEO GetCapabilities operation returning the additional information notifying the support of WS-notification.

c) **Test method**:

Verify that the Order Server under test accepts a valid OSEO GetCapabilities request and returns a valid Capabilities element including a non empty:

□ Capabilities/Notifications

Pass if the assertion is satisfied; fail otherwise.

d) **References**:

□ Req 18

e) **Test type**: Capability

A.4 Traceability Matrix

A.4.1 Requirements vs. Conformance Tests Traceability Matrix

Requirement	Conformance Test
Req 1	A.3.1.1GetCapabilities
Req 2	A.3.1.1GetCapabilities
Req 3	A.3.1.1GetCapabilities
Req 4	A.3.1.1GetCapabilities
Req 5	A.3.1.1GetCapabilities
Req 6	A.3.1.1GetCapabilities
Req 7	A.3.1.1GetCapabilities
Req 8	A.3.1.2GetCapabilities non nominal conditions
Req 9	A.3.2.1GetCapabilities
Req 10	A.3.4.1GetCapabilities
Req 11	A.3.5.1GetCapabilities

Requirement	Conformance Test
Req 12	A.3.6.1GetCapabilities
Req 13	A.3.7.1GetCapabilities
Req 14	A.3.8.1GetCapabilities
Req 15	A.3.9.1GetCapabilities
Req 16	A.3.12.1GetCapabilities
Req 17	A.3.15.1GetCapabilities
Req 18	A.3.17.1GetCapabilities
Req 19	A.3.1.3GetOptions - Core
Req 20	A.3.1.3GetOptions - Core
Req 21	A.3.1.3GetOptions - Core
Req 22	A.3.1.4GetOptions non nominal conditions
Req 23	A.3.2.2GetOptions – By Product
Req 24	A.3.2.2GetOptions – By Product
Req 25	A.3.2.3GetOptions – By Collection
Req 26	A.3.2.3GetOptions – By Collection
Req 27	A.3.2.2GetOptions – By Product
Req 27	A.3.2.3GetOptions – By Collection
Req 28	A.3.3.1GetOptions – Scene Selection
Req 29	A.3.4.2GetOptions – Subscription Order
Req 30	A.3.4.2GetOptions – Subscription Order
Req 31	A.3.4.2GetOptions – Subscription Order
Req 32	A.3.5.2GetOptions – By TASK ID
Req 33	A.3.5.2GetOptions – By TASK ID
Req 34	A.3.5.2GetOptions – By TASK ID
Req 35	A.3.7.2GetOptions – Payment Options
Req 36	A.3.12.2GetOptions – online data access options
Req 37	A.3.13.1GetOptions – online delivery options
Req 38	A.3.14.1GetOptions – media delivery options
Req 39	A.3.7.3GetQuotation - Basic
Req 40	A.3.7.3GetQuotation - Basic
Req 41	A.3.7.3GetQuotation - Basic
Req 42	A.3.7.3GetQuotation - Basic
Req 43	A.3.7.4GetQuotation - non nominal conditions
Req 44	A.3.8.2GetQuotation - synchronous
Req 45	A.3.9.2GetQuotation - asynchronous with monitoring – initial call

Requirement	Conformance Test
Req 46	A.3.9.3GetQuotation - asynchronous with monitoring – next call
Req 47	A.3.9.3GetQuotation - asynchronous with monitoring – next call
Req 48	A.3.10.1GetQuotation - asynchronous with notification – initial call
Req 49	A.3.10.1GetQuotation - asynchronous with notification – initial call
Req 50	A.3.10.2GetQuotation - asynchronous with notification – notification
Req 51	A.3.10.2GetQuotation - asynchronous with notification – notification
Req 52	A.3.10.2GetQuotation - asynchronous with notification – notification
Req 53	A.3.11.1GetQuotation – off-line
Req 54	A.3.11.1GetQuotation – off-line
Req 55	A.3.11.1GetQuotation – off-line
Req 56	A.3.1.5Submit - Core
Req 57	A.3.1.5Submit - Core
Req 58	A.3.1.5Submit - Core
Req 59	A.3.1.5Submit - Core
Req 60	A.3.1.5Submit - Core
Req 61	A.3.1.5Submit - Core
Req 62	A.3.1.6Submit - non nominal conditions
Req 62	A.3.2.5Submit - non nominal conditions
Req 62	A.3.4.4Submit - non nominal conditions
Req 62	A.3.5.4Submit - non nominal conditions
Req 63	A.3.6.2Submit – Single Notification
Req 64	A.3.6.4Submit –Multiple Notifications
Req 65	A.3.6.3SubmitResponse – Single Notification
Req 66	A.3.6.3SubmitResponse – Single Notification
Req 67	A.3.6.5SubmitResponse – Multiple Notifications
Req 68	A.3.6.3SubmitResponse – Single Notification
Req 68	A.3.6.5SubmitResponse – Multiple Notifications
Req 69	A.3.6.3SubmitResponse – Single Notification
Req 70	A.3.2.4Submit – Product Order
Req 71	A.3.3.2Submit – Product with Scenes
Req 72	A.3.3.3Submit - non nominal conditions
Req 73	A.3.4.3Submit – Subscription Order
Req 74	A.3.5.3Submit – By TASK ID
Req 75	A.3.7.5Submit – By quotation ID
Req 76	A.3.12.3Submit – on line data access

Requirement	Conformance Test
Req 77	A.3.13.2Submit – on line delivery
Req 78	A.3.14.2Submit – delivery on media
Req 79	A.3.1.10GetStatus – order search by OrderReference, FULL
Req 79	A.3.1.11GetStatus – by ORDER ID, FULL
Req 79	A.3.1.12GetStatus – by Not Existing ORDER ID, BRIEF, SOAP Fault
Req 79	A.3.1.7GetStatus – order search by LastUpdate, BRIEF
Req 79	A.3.1.8GetStatus – order search by LastUpdate, FULL
Req 79	A.3.1.9GetStatus – order search by OrderStatus, FULL
Req 80	A.3.1.10GetStatus – order search by OrderReference, FULL
Req 80	A.3.1.11GetStatus – by ORDER ID, FULL
Req 80	A.3.1.12GetStatus – by Not Existing ORDER ID, BRIEF, SOAP Fault
Req 80	A.3.1.7GetStatus – order search by LastUpdate, BRIEF
Req 80	A.3.1.8GetStatus – order search by LastUpdate, FULL
Req 80	A.3.1.9GetStatus – order search by OrderStatus, FULL
Req 81	A.3.1.11GetStatus – by ORDER ID, FULL
Req 81	A.3.1.12GetStatus – by Not Existing ORDER ID, BRIEF, SOAP Fault
Req 82	A.3.1.10GetStatus – order search by OrderReference, FULL
Req 82	A.3.1.7GetStatus – order search by LastUpdate, BRIEF
Req 82	A.3.1.8GetStatus – order search by LastUpdate, FULL
Req 82	A.3.1.9GetStatus – order search by OrderStatus, FULL
Req 83	A.3.1.7GetStatus – order search by LastUpdate, BRIEF
Req 83	A.3.1.8GetStatus – order search by LastUpdate, FULL
Req 84	A.3.1.9GetStatus – order search by OrderStatus, FULL
Req 85	A.3.1.10GetStatus – order search by OrderReference, FULL
Req 86	A.3.1.12GetStatus – by Not Existing ORDER ID, BRIEF, SOAP Fault
Req 86	A.3.1.7GetStatus – order search by LastUpdate, BRIEF
Req 87	A.3.1.10GetStatus – order search by OrderReference, FULL
Req 87	A.3.1.11GetStatus – by ORDER ID, FULL
Req 87	A.3.1.8GetStatus – order search by LastUpdate, FULL
Req 87	A.3.1.9GetStatus – order search by OrderStatus, FULL
Req 88	A.3.1.10GetStatus – order search by OrderReference, FULL
Req 88	A.3.1.11GetStatus – by ORDER ID, FULL
Req 88	A.3.1.12GetStatus – by Not Existing ORDER ID, BRIEF, SOAP Fault
Req 88	A.3.1.7GetStatus – order search by LastUpdate, BRIEF
Req 88	A.3.1.8GetStatus – order search by LastUpdate, FULL

Requirement	Conformance Test
Req 88	A.3.1.9GetStatus – order search by OrderStatus, FULL
Req 89	A.3.1.11GetStatus – by ORDER ID, FULL
Req 90	A.3.1.12GetStatus – by Not Existing ORDER ID, BRIEF, SOAP Fault
Req 91	A.3.1.7GetStatus – order search by LastUpdate, BRIEF
Req 91	A.3.1.8GetStatus – order search by LastUpdate, FULL
Req 91	A.3.1.9GetStatus – order search by OrderStatus, FULL
Req 92	A.3.1.10GetStatus – order search by OrderReference, FULL
Req 93	A.3.1.7GetStatus – order search by LastUpdate, BRIEF
Req 94	A.3.1.11GetStatus – by ORDER ID, FULL
Req 94	A.3.1.8GetStatus – order search by LastUpdate, FULL
Req 94	A.3.1.9GetStatus – order search by OrderStatus, FULL
Req 95	A.3.1.11GetStatus – by ORDER ID, FULL
Req 95	A.3.1.7GetStatus – order search by LastUpdate, BRIEF
Req 95	A.3.1.8GetStatus – order search by LastUpdate, FULL
Req 95	A.3.1.9GetStatus – order search by OrderStatus, FULL
Req 96	A.3.1.8GetStatus – order search by LastUpdate, FULL
Req 97	A.3.1.8GetStatus – order search by LastUpdate, FULL
Req 98	A.3.1.7GetStatus – order search by LastUpdate, BRIEF
Req 98	A.3.1.8GetStatus – order search by LastUpdate, FULL
Req 98	A.3.1.9GetStatus – order search by OrderStatus, FULL
Req 99	A.3.1.12GetStatus – by Not Existing ORDER ID, BRIEF, SOAP Fault
Req 99	A.3.1.13GetStatus - non nominal conditions
Req 100	A.3.2.6GetStatus – by ORDER ID, FULL
Req 101	A.3.3.4GetStatus – by ORDER ID, FULL
Req 102	A.3.4.5GetStatus – by ORDER ID, FULL
Req 103	A.3.5.5GetStatus – by ORDER ID, FULL
Req 104	A.3.12.4DescribeResultAccess – retrieve all available items
Req 104	A.3.12.5DescribeResultAccess – retrieve new available items
Req 105	A.3.12.4DescribeResultAccess – retrieve all available items
Req 105	A.3.12.5DescribeResultAccess – retrieve new available items
Req 106	A.3.12.4DescribeResultAccess – retrieve all available items
Req 107	A.3.12.5DescribeResultAccess – retrieve new available items
Req 108	A.3.12.4DescribeResultAccess – retrieve all available items
Req 108	A.3.12.5DescribeResultAccess – retrieve new available items
Req 109	A.3.12.4DescribeResultAccess – retrieve all available items

Requirement	Conformance Test
Req 110	A.3.12.5DescribeResultAccess – retrieve new available items
Req 111	A.3.12.6DescribeResultAccess – non nominal conditions
Req 112	A.3.12.4DescribeResultAccess – retrieve all available items
Req 113	A.3.15.2Cancel – no notification
Req 114	A.3.15.2Cancel – no notification
Req 115	A.3.15.2Cancel – no notification
Req 116	A.3.15.2Cancel – no notification
Req 117	A.3.15.2Cancel – no notification
Req 118	A.3.15.3Order Cancellation Monitoring
Req 119	A.3.15.4Cancel - non nominal conditions
Req 120	A.3.16.1Cancel – Single Notification
Req 120	A.3.16.2CancelResponse – Single Notification
Req 121	A.3.16.3Cancel –Multiple Notifications
Req 121	A.3.16.4CancelResponse – Multiple Notifications
Req 122	A.3.16.2CancelResponse – Single Notification
Req 122	A.3.16.4CancelResponse – Multiple Notifications
Req 123	A.3.16.2CancelResponse – Single Notification
Req 124	A.3.16.4CancelResponse – Multiple Notifications
Req 125	A.3.16.2CancelResponse – Single Notification
Req 125	A.3.16.4CancelResponse – Multiple Notifications
Req 126	A.3.16.2CancelResponse – Single Notification
Req 126	A.3.16.4CancelResponse – Multiple Notifications

Table 17-6: Requirements vs. Conformance Tests Traceability Matrix.

A.4.2 Conformance Tests vs. Requirements Traceability Matrix

Conformance Test	Requirement
A.3.1.1GetCapabilities	Req 1
A.3.1.1GetCapabilities	Req 2
A.3.1.1GetCapabilities	Req 3
A.3.1.1GetCapabilities	Req 4
A.3.1.1GetCapabilities	Req 5
A.3.1.1GetCapabilities	Req 6
A.3.1.1GetCapabilities	Req 7
A.3.1.2GetCapabilities non nominal conditions	Req 8

Conformance Test	Requirement
A.3.1.3GetOptions - Core	Req 19
A.3.1.3GetOptions - Core	Req 20
A.3.1.3GetOptions - Core	Req 21
A.3.1.4GetOptions non nominal conditions	Req 22
A.3.1.5Submit - Core	Req 56
A.3.1.5Submit - Core	Req 57
A.3.1.5Submit - Core	Req 58
A.3.1.5Submit - Core	Req 59
A.3.1.5Submit - Core	Req 60
A.3.1.5Submit - Core	Req 61
A.3.1.6Submit - non nominal conditions	Req 62
A.3.1.7GetStatus – order search by LastUpdate, BRIEF	Req 79
A.3.1.7GetStatus – order search by LastUpdate, BRIEF	Req 80
A.3.1.7GetStatus – order search by LastUpdate, BRIEF	Req 82
A.3.1.7GetStatus – order search by LastUpdate, BRIEF	Req 83
A.3.1.7GetStatus – order search by LastUpdate, BRIEF	Req 86
A.3.1.7GetStatus – order search by LastUpdate, BRIEF	Req 88
A.3.1.7GetStatus – order search by LastUpdate, BRIEF	Req 91
A.3.1.7GetStatus – order search by LastUpdate, BRIEF	Req 93
A.3.1.7GetStatus – order search by LastUpdate, BRIEF	Req 95
A.3.1.7GetStatus – order search by LastUpdate, BRIEF	Req 98
A.3.1.8GetStatus – order search by LastUpdate, FULL	Req 79
A.3.1.8GetStatus – order search by LastUpdate, FULL	Req 80
A.3.1.8GetStatus – order search by LastUpdate, FULL	Req 82
A.3.1.8GetStatus – order search by LastUpdate, FULL	Req 83
A.3.1.8GetStatus – order search by LastUpdate, FULL	Req 87
A.3.1.8GetStatus – order search by LastUpdate, FULL	Req 88
A.3.1.8GetStatus – order search by LastUpdate, FULL	Req 91
A.3.1.8GetStatus – order search by LastUpdate, FULL	Req 94
A.3.1.8GetStatus – order search by LastUpdate, FULL	Req 95
A.3.1.8GetStatus – order search by LastUpdate, FULL	Req 96
A.3.1.8GetStatus – order search by LastUpdate, FULL	Req 97
A.3.1.8GetStatus – order search by LastUpdate, FULL	Req 98
A.3.1.9GetStatus – order search by OrderStatus, FULL	Req 79
A.3.1.9GetStatus – order search by OrderStatus, FULL	Req 80

Conformance Test	Requirement
A.3.1.9GetStatus – order search by OrderStatus, FULL	Req 82
A.3.1.9GetStatus – order search by OrderStatus, FULL	Req 84
A.3.1.9GetStatus – order search by OrderStatus, FULL	Req 87
A.3.1.9GetStatus – order search by OrderStatus, FULL	Req 88
A.3.1.9GetStatus – order search by OrderStatus, FULL	Req 91
A.3.1.9GetStatus – order search by OrderStatus, FULL	Req 94
A.3.1.9GetStatus – order search by OrderStatus, FULL	Req 95
A.3.1.9GetStatus – order search by OrderStatus, FULL	Req 98
A.3.1.10GetStatus – order search by OrderReference, FULL	Req 79
A.3.1.10GetStatus – order search by OrderReference, FULL	Req 80
A.3.1.10GetStatus – order search by OrderReference, FULL	Req 82
A.3.1.10GetStatus – order search by OrderReference, FULL	Req 85
A.3.1.10GetStatus – order search by OrderReference, FULL	Req 87
A.3.1.10GetStatus – order search by OrderReference, FULL	Req 88
A.3.1.10GetStatus – order search by OrderReference, FULL	Req 92
A.3.1.11GetStatus – by ORDER ID, FULL	Req 79
A.3.1.11GetStatus – by ORDER ID, FULL	Req 80
A.3.1.11GetStatus – by ORDER ID, FULL	Req 81
A.3.1.11GetStatus – by ORDER ID, FULL	Req 87
A.3.1.11GetStatus – by ORDER ID, FULL	Req 88
A.3.1.11GetStatus – by ORDER ID, FULL	Req 89
A.3.1.11GetStatus – by ORDER ID, FULL	Req 94
A.3.1.11GetStatus – by ORDER ID, FULL	Req 95
A.3.1.12GetStatus – by Not Existing ORDER ID, BRIEF, SOAP Fault	Req 79
A.3.1.12GetStatus – by Not Existing ORDER ID, BRIEF, SOAP Fault	Req 80
A.3.1.12GetStatus – by Not Existing ORDER ID, BRIEF, SOAP Fault	Req 81
A.3.1.12GetStatus – by Not Existing ORDER ID, BRIEF, SOAP Fault	Req 86
A.3.1.12GetStatus – by Not Existing ORDER ID, BRIEF, SOAP Fault	Req 88
A.3.1.12GetStatus – by Not Existing ORDER ID, BRIEF, SOAP Fault	Req 90
A.3.1.12GetStatus – by Not Existing ORDER ID, BRIEF, SOAP Fault	Req 99
A.3.1.13GetStatus - non nominal conditions	Req 99
A.3.2.1GetCapabilities	Req 9
A.3.2.2GetOptions – By Product	Req 23
A.3.2.2GetOptions – By Product	Req 24
A.3.2.2GetOptions – By Product	Req 27

Conformance Test	Requirement
A.3.2.3GetOptions – By Collection	Req 25
A.3.2.3GetOptions – By Collection	Req 26
A.3.2.3GetOptions – By Collection	Req 27
A.3.2.4Submit – Product Order	Req 70
A.3.2.5Submit - non nominal conditions	Req 62
A.3.2.6GetStatus – by ORDER ID, FULL	Req 100
A.3.3.1GetOptions – Scene Selection	Req 28
A.3.3.2Submit – Product with Scenes	Req 71
A.3.3.3Submit - non nominal conditions	Req 72
A.3.3.4GetStatus – by ORDER ID, FULL	Req 101
A.3.4.1GetCapabilities	Req 10
A.3.4.2GetOptions – Subscription Order	Req 29
A.3.4.2GetOptions – Subscription Order	Req 30
A.3.4.2GetOptions – Subscription Order	Req 31
A.3.4.3Submit – Subscription Order	Req 73
A.3.4.4Submit - non nominal conditions	Req 62
A.3.4.5GetStatus – by ORDER ID, FULL	Req 102
A.3.5.1GetCapabilities	Req 11
A.3.5.2GetOptions – By TASK ID	Req 32
A.3.5.2GetOptions – By TASK ID	Req 33
A.3.5.2GetOptions – By TASK ID	Req 34
A.3.5.3Submit – By TASK ID	Req 74
A.3.5.4Submit - non nominal conditions	Req 62
A.3.5.5GetStatus – by ORDER ID, FULL	Req 103
A.3.6.1GetCapabilities	Req 12
A.3.6.2Submit – Single Notification	Req 63
A.3.6.3SubmitResponse – Single Notification	Req 65
A.3.6.3SubmitResponse – Single Notification	Req 66
A.3.6.3SubmitResponse – Single Notification	Req 68
A.3.6.3SubmitResponse – Single Notification	Req 69
A.3.6.4Submit –Multiple Notifications	Req 64
A.3.6.5SubmitResponse – Multiple Notifications	Req 67
A.3.6.5SubmitResponse – Multiple Notifications	Req 68
A.3.7.1GetCapabilities	Req 13
A.3.7.2GetOptions – Payment Options	Req 35

Conformance Test	Requirement
A.3.7.3GetQuotation - Basic	Req 39
A.3.7.3GetQuotation - Basic	Req 40
A.3.7.3GetQuotation - Basic	Req 41
A.3.7.3GetQuotation - Basic	Req 42
A.3.7.4GetQuotation - non nominal conditions	Req 43
A.3.7.5Submit – By quotation ID	Req 75
A.3.8.1GetCapabilities	Req 14
A.3.8.2GetQuotation - synchronous	Req 44
A.3.9.1GetCapabilities	Req 15
A.3.9.2GetQuotation - asynchronous with monitoring – initial call	Req 45
A.3.9.3GetQuotation - asynchronous with monitoring – next call	Req 46
A.3.9.3GetQuotation - asynchronous with monitoring – next call	Req 47
A.3.10.1GetQuotation - asynchronous with notification – initial call	Req 48
A.3.10.1GetQuotation - asynchronous with notification – initial call	Req 49
A.3.10.2GetQuotation - asynchronous with notification – notification	Req 50
A.3.10.2GetQuotation - asynchronous with notification – notification	Req 51
A.3.10.2GetQuotation - asynchronous with notification – notification	Req 52
A.3.11.1GetQuotation – off-line	Req 53
A.3.11.1GetQuotation – off-line	Req 54
A.3.11.1GetQuotation – off-line	Req 55
A.3.12.1GetCapabilities	Req 16
A.3.12.2GetOptions – online data access options	Req 36
A.3.12.3Submit – on line data access	Req 76
A.3.12.4DescribeResultAccess – retrieve all available items	Req 104
A.3.12.4DescribeResultAccess – retrieve all available items	Req 105
A.3.12.4DescribeResultAccess – retrieve all available items	Req 106
A.3.12.4DescribeResultAccess – retrieve all available items	Req 108
A.3.12.4DescribeResultAccess – retrieve all available items	Req 109
A.3.12.4DescribeResultAccess – retrieve all available items	Req 112
A.3.12.5DescribeResultAccess – retrieve new available items	Req 104
A.3.12.5DescribeResultAccess – retrieve new available items	Req 105
A.3.12.5DescribeResultAccess – retrieve new available items	Req 107
A.3.12.5DescribeResultAccess – retrieve new available items	Req 108
A.3.12.5DescribeResultAccess – retrieve new available items	Req 110
A.3.12.6DescribeResultAccess – non nominal conditions	Req 111

Conformance Test	Requirement
A.3.13.1GetOptions – online delivery options	Req 37
A.3.13.2Submit – on line delivery	Req 77
A.3.14.1GetOptions – media delivery options	Req 38
A.3.14.2Submit – delivery on media	Req 78
A.3.15.1GetCapabilities	Req 17
A.3.15.2Cancel – no notification	Req 113
A.3.15.2Cancel – no notification	Req 114
A.3.15.2Cancel – no notification	Req 115
A.3.15.2Cancel – no notification	Req 116
A.3.15.2Cancel – no notification	Req 117
A.3.15.3Order Cancellation Monitoring	Req 118
A.3.15.4Cancel - non nominal conditions	Req 119
A.3.16.1Cancel – Single Notification	Req 120
A.3.16.2CancelResponse – Single Notification	Req 120
A.3.16.2CancelResponse – Single Notification	Req 122
A.3.16.2CancelResponse – Single Notification	Req 123
A.3.16.2CancelResponse – Single Notification	Req 125
A.3.16.2CancelResponse – Single Notification	Req 126
A.3.16.3Cancel –Multiple Notifications	Req 121
A.3.16.4CancelResponse – Multiple Notifications	Req 121
A.3.16.4CancelResponse – Multiple Notifications	Req 122
A.3.16.4CancelResponse – Multiple Notifications	Req 124
A.3.16.4CancelResponse – Multiple Notifications	Req 125
A.3.16.4CancelResponse – Multiple Notifications	Req 126
A.3.17.1GetCapabilities	Req 18

Table 17-7: Conformance Tests vs. Requirements Traceability Matrix.

APPENDIX B (normative) XML Schema definitions

In addition to this document, this standard includes several normative XML Schema Documents. These XML Schema Documents are bundled in a zip file with the present document. After OGC acceptance of a Version 1.0.0 of this standard, these XML Schema Documents will also be posted online at the URL http://schemas.opengeospatial.net/oseo/1.0. In the event of a discrepancy between the bundled and online versions of the XML Schema Documents, the online files shall be considered authoritative.

The Order Service for Earth Observation Products now specified in this document uses the XML Schema Document included in the zip file with this document. This XML Schema Document combines the XML schema fragments listed in various subclauses of this document, eliminating duplications. This XML Schema Document is named:

- oseo.xsd

These XML Schema Documents use and build on the following XML Schema Documents:

- sweCommon/2.0/swe.xsd
- swes/2.0/swes.xsd
- ows/2.0/owsGetCapabilities.xsd

APPENDIX C (informative) Example XML documents

This annex provides information to access to the example XML documents than given in the body of this document.

The Order Service for Earth Observation Products examples specified in this document use the specified files that may be included in the zip file with this document. These XML examples are within the directory:

schema\oseo\SampleMessages

and can be found online at http://schemas.opengis.net/oseo/1.0/SampleMessages/

APPENDIX D (informative) WSDL Specification

```
<?xml version="1.0" encoding="UTF-8"?>
<!--
File:           oseo.wsdl
File Type:      XML instance of W3C WSDL 1.1 Schema
Abstract:       WSDL file for OSEO
Uses:           oseo.xsd
Author:         Marchionni Daniele
-->
<definitions xmlns="http://schemas.xmlsoap.org/wsdl/"
xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
xmlns:tns="http://www.opengis.net/oseo/1.0"
xmlns:ns1="http://www.w3.org/2001/XMLSchema"
xmlns:wsa="http://www.w3.org/2005/08/addressing"
xmlns:ns="http://www.isotc211.org/2005/gco"
xmlns:ns2="http://www.isotc211.org/2005/gmd"
xmlns:ns3="http://www.isotc211.org/2005/gsr"
xmlns:ns4="http://www.isotc211.org/2005/gss"
xmlns:ns5="http://www.isotc211.org/2005/gts"
xmlns:ns6="http://www.opengis.net/gml/3.2" xmlns:ns7="http://www.opengis.net/ows/2.0"
xmlns:ns8="http://www.opengis.net/sps/2.0" xmlns:ns9="http://www.opengis.net/swe/2.0"
xmlns:ns10="http://www.opengis.net/swe/st/0"
xmlns:ns11="http://www.w3.org/2001/SMIL20/"
xmlns:ns12="http://www.w3.org/2001/SMIL20/Language"
xmlns:ns13="http://schemas.xmlsoap.org/soap/encoding/"
xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata" name="OSEO_V100"
targetNamespace="http://www.opengis.net/oseo/1.0">
      <types>
             <schema attributeFormDefault="qualified" elementFormDefault="qualified"
targetNamespace="http://schemas.xmlsoap.org/wsdl/"
xmlns="http://www.w3.org/2001/XMLSchema">
                   <import namespace="http://www.opengis.net/oseo/1.0"
schemaLocation="http://schemas.opengis.net/oseo/1.0/oseo.xsd"/>
                   <import namespace="http://www.opengis.net/ows/2.0"
schemaLocation="http://schemas.opengis.net/ows/2.0/owsAll.xsd"/>
             </schema>
      </types>
      <message name="GetCapabilitiesRequest">
             <part name="parameters" element="tns:GetCapabilities"/>
      </message>
      <message name="GetCapabilitiesResponse">
             <part name="parameters" element="tns:Capabilities"/>
      </message>
      <message name="GetOptionsRequestParameter">
             <part name="parameters" element="tns:GetOptions"/>
      </message>
      <message name="GetOptionsResponseParameter">
             <part name="parameters" element="tns:GetOptionsResponse"/>
      </message>
      <message name="GetQuotationRequestParameter">
             <part name="parameters" element="tns:GetQuotation"/>
      </message>
      <message name="GetQuotationAckParameter">
             <part name="parameters" element="tns:GetQuotationAck"/>
      </message>
      <message name="GetQuotationResponseParameter">
             <part name="parameters" element="tns:GetQuotationResponse"/>
      </message>
      <message name="GetQuotationResponseAckParameter">
```

```
             <part name="parameters" element="tns:GetQuotationResponseAck"/>
      </message>
      <message name="SubmitRequestParameter">
             <part name="parameters" element="tns:Submit"/>
      </message>
      <message name="SubmitAckParameter">
             <part name="parameters" element="tns:SubmitAck"/>
      </message>
      <message name="SubmitResponseParameter">
             <part name="parameters" element="tns:SubmitResponse"/>
      </message>
      <message name="SubmitResponseAckParameter">
             <part name="parameters" element="tns:SubmitResponseAck"/>
      </message>
      <message name="GetStatusRequestParameter">
             <part name="parameters" element="tns:GetStatus"/>
      </message>
      <message name="GetStatusResponseParameter">
             <part name="parameters" element="tns:GetStatusResponse"/>
      </message>
      <message name="CancelRequestParameter">
             <part name="parameters" element="tns:Cancel"/>
      </message>
      <message name="CancelAckParameter">
             <part name="parameters" element="tns:CancelAck"/>
      </message>
      <message name="CancelResponseParameter">
             <part name="parameters" element="tns:CancelResponse"/>
      </message>
      <message name="CancelResponseAckParameter">
             <part name="parameters" element="tns:CancelResponseAck"/>
      </message>
      <message name="DescribeResultAccessParameter">
             <part name="parameters" element="tns:DescribeResultAccess"/>
      </message>
      <message name="DescribeResultAccessResponseParameter">
             <part name="parameters" element="tns:DescribeResultAccessResponse"/>
      </message>
      <message name="StartHeader">
             <part name="MessageID" element="wsa:MessageID"/>
             <part name="ReplyTo" element="wsa:ReplyTo"/>
      </message>
      <message name="ContinueHeader">
             <part name="RelatesTo" element="wsa:RelatesTo"/>
      </message>
      <message name="ServiceExceptionReport">
             <part name="fault" element="ns7:ExceptionReport"/>
      </message>
      <portType name="OSEO_V100_Service">
             <operation name="GetCapabilities">
                    <input message="tns:GetCapabilitiesRequest"/>
                    <output message="tns:GetCapabilitiesResponse"/>
                    <fault name="ServiceExceptionReport"
message="tns:ServiceExceptionReport"/>
             </operation>
             <operation name="GetOptions">
                    <input message="tns:GetOptionsRequestParameter"/>
                    <output message="tns:GetOptionsResponseParameter"/>
                    <fault name="ServiceExceptionReport"
message="tns:ServiceExceptionReport"/>
             </operation>
             <operation name="GetQuotation">
```

```
                  <input message="tns:GetQuotationRequestParameter"/>
                  <output message="tns:GetQuotationAckParameter"/>
                  <fault name="ServiceExceptionReport"
message="tns:ServiceExceptionReport"/>
            </operation>
            <operation name="Submit">
                  <input message="tns:SubmitRequestParameter"/>
                  <output message="tns:SubmitAckParameter"/>
                  <fault name="ServiceExceptionReport"
message="tns:ServiceExceptionReport"/>
            </operation>
            <operation name="GetStatus">
                  <input message="tns:GetStatusRequestParameter"/>
                  <output message="tns:GetStatusResponseParameter"/>
                  <fault name="ServiceExceptionReport"
message="tns:ServiceExceptionReport"/>
            </operation>
            <operation name="Cancel">
                  <input message="tns:CancelRequestParameter"/>
                  <output message="tns:CancelAckParameter"/>
                  <fault name="ServiceExceptionReport"
message="tns:ServiceExceptionReport"/>
            </operation>
            <operation name="DescribeResultAccess">
                  <input message="tns:DescribeResultAccessParameter"/>
                  <output message="tns:DescribeResultAccessResponseParameter"/>
                  <fault name="ServiceExceptionReport"
message="tns:ServiceExceptionReport"/>
            </operation>
      </portType>
      <portType name="OSEO_V100_AsyncClient">
            <operation name="GetQuotationResponse">
                  <input message="tns:GetQuotationResponseParameter"/>
                  <output message="tns:GetQuotationResponseAckParameter"/>
                  <fault name="ServiceExceptionReport"
message="tns:ServiceExceptionReport"/>
            </operation>
            <operation name="SubmitResponse">
                  <input message="tns:SubmitResponseParameter"/>
                  <output message="tns:SubmitResponseAckParameter"/>
                  <fault name="ServiceExceptionReport"
message="tns:ServiceExceptionReport"/>
            </operation>
            <operation name="CancelResponse">
                  <input message="tns:CancelResponseParameter"/>
                  <output message="tns:CancelResponseAckParameter"/>
                  <fault name="ServiceExceptionReport"
message="tns:ServiceExceptionReport"/>
            </operation>
      </portType>
      <binding name="OSEO_V100_ServicePortBinding" type="tns:OSEO_V100_Service">
            <soap:binding style="document"
transport="http://schemas.xmlsoap.org/soap/http"/>
            <operation name="GetCapabilities">
                  <soap:operation soapAction="GetCapabilities"/>
                  <input>
                        <soap:body use="literal"/>
                  </input>
                  <output>
                        <soap:body use="literal"/>
                  </output>
                  <fault name="ServiceExceptionReport">
```

```
                          <soap:fault name="ServiceExceptionReport" use="literal"/>
                   </fault>
             </operation>
             <operation name="GetOptions">
                   <soap:operation soapAction="GetOptions"/>
                   <input>
                          <soap:body use="literal"/>
                   </input>
                   <output>
                          <soap:body use="literal"/>
                   </output>
                   <fault name="ServiceExceptionReport">
                          <soap:fault name="ServiceExceptionReport" use="literal"/>
                   </fault>
             </operation>
             <operation name="GetQuotation">
                   <soap:operation soapAction="GetQuotation"/>
                   <input>
                          <soap:body use="literal"/>
                          <soap:header message="tns:StartHeader" part="ReplyTo"
use="literal"/>
                          <soap:header message="tns:StartHeader" part="MessageID"
use="literal"/>
                   </input>
                   <output>
                          <soap:body use="literal"/>
                   </output>
                   <fault name="ServiceExceptionReport">
                          <soap:fault name="ServiceExceptionReport" use="literal"/>
                   </fault>
             </operation>
             <operation name="Submit">
                   <soap:operation soapAction="Submit"/>
                   <input>
                          <soap:body use="literal"/>
                          <soap:header message="tns:StartHeader" part="ReplyTo"
use="literal"/>
                          <soap:header message="tns:StartHeader" part="MessageID"
use="literal"/>
                   </input>
                   <output>
                          <soap:body use="literal"/>
                   </output>
                   <fault name="ServiceExceptionReport">
                          <soap:fault name="ServiceExceptionReport" use="literal"/>
                   </fault>
             </operation>
             <operation name="GetStatus">
                   <soap:operation soapAction="GetStatus"/>
                   <input>
                          <soap:body use="literal"/>
                   </input>
                   <output>
                          <soap:body use="literal"/>
                   </output>
                   <fault name="ServiceExceptionReport">
                          <soap:fault name="ServiceExceptionReport" use="literal"/>
                   </fault>
             </operation>
             <operation name="Cancel">
                   <soap:operation soapAction="Cancel"/>
                   <input>
```

```
                        <soap:body use="literal"/>
                        <soap:header message="tns:StartHeader" part="ReplyTo" 
use="literal"/>
                        <soap:header message="tns:StartHeader" part="MessageID" 
use="literal"/>
                  </input>
                  <output>
                        <soap:body use="literal"/>
                  </output>
                  <fault name="ServiceExceptionReport">
                        <soap:fault name="ServiceExceptionReport" use="literal"/>
                  </fault>
            </operation>
            <operation name="DescribeResultAccess">
                  <soap:operation soapAction="DescribeResultAccess"/>
                  <input>
                        <soap:body use="literal"/>
                  </input>
                  <output>
                        <soap:body use="literal"/>
                  </output>
                  <fault name="ServiceExceptionReport">
                        <soap:fault name="ServiceExceptionReport" use="literal"/>
                  </fault>
            </operation>
      </binding>
      <binding name="OSEO_V100_AsyncClientPortBinding" 
type="tns:OSEO_V100_AsyncClient">
            <soap:binding style="document" 
transport="http://schemas.xmlsoap.org/soap/http"/>
            <operation name="GetQuotationResponse">
                  <soap:operation soapAction="GetQuotationResponse"/>
                  <input>
                        <soap:body use="literal"/>
                        <soap:header message="tns:ContinueHeader" part="RelatesTo" 
use="literal"/>
                  </input>
                  <output>
                        <soap:body use="literal"/>
                  </output>
                  <fault name="ServiceExceptionReport">
                        <soap:fault name="ServiceExceptionReport" use="literal"/>
                  </fault>
            </operation>
            <operation name="SubmitResponse">
                  <soap:operation soapAction="SubmitResponse"/>
                  <input>
                        <soap:body use="literal"/>
                        <soap:header message="tns:ContinueHeader" part="RelatesTo" 
use="literal"/>
                  </input>
                  <output>
                        <soap:body use="literal"/>
                  </output>
                  <fault name="ServiceExceptionReport">
                        <soap:fault name="ServiceExceptionReport" use="literal"/>
                  </fault>
            </operation>
            <operation name="CancelResponse">
                  <soap:operation soapAction="CancelResponse"/>
                  <input>
                        <soap:body use="literal"/>
```

```
                           <soap:header message="tns:ContinueHeader" part="RelatesTo"
use="literal"/>
                    </input>
                    <output>
                           <soap:body use="literal"/>
                    </output>
                    <fault name="ServiceExceptionReport">
                           <soap:fault name="ServiceExceptionReport" use="literal"/>
                    </fault>
             </operation>
      </binding>
      <service name="OSEO_V100">
             <port name="OSEO_V1.0.0_Port"
binding="tns:OSEO_V100_ServicePortBinding">
                    <soap:address location="No Target Address"/>
             </port>
      </service>
      <service name="OSEO_V100_AsyncClient">
             <port name="OSEO_V1.0.0_AsyncClientPort"
binding="tns:OSEO_V100_AsyncClientPortBinding">
                    <soap:address location="No Target Address"/>
             </port>
      </service>
</definitions>
```

APPENDIX E (informative) Implementation guidance

E.1 General

The following section gives developers help when setting up an order service instance that complies with this specification. Any information provided here is non-normative or is a detailing of former descriptions.

E.2 Semantic issues

None.

E.3 Technical issues

E.3.1 SOAP

This protocol specifies that only SOAP messaging (via HTTP/POST) with document/literal style has to be used. Messages must conform to SOAP 1.2 (http://www.w3.org/TR/SOAP/). The message payload will be in the body of the SOAP envelope.

However there are several implementations of Order Servers that cannot complies with the SOAP V1.2 requirement due to COTS / legacy software.It is assumed that also SOAP V1.1 is acceptable for a compliant Order server.

E.3.2 Service Implementation

The Ordering Service described in this document has been successfully implemented in Java using the following development environment:

□ Java 1.6

□ Netbeans 6.9.1 (IDE)

□ Axis2 (Web Service Framework)

□ XMLBEANS V2.4 (for data binding)

Additionally OSEO schema has been successfully converted in C# using the following tool:

□ XSD2CODE V3.4

The usage of other tools was not successful due to the import of GML schemas in SWE Common, which in turn is included in OSEO.

The following configurations were **not** successful:

□ JAXB, as data binding, it failed in conveting GML schemas;

□ Metro, as frame work for Web Service implementation, was not successful still due to GML;

□ XMLBEANS V2.5 was not able to convert GML (probably a regression with respect to V2.4).

E.4 Realization of WS-Notification interface

WS-Notifications is a group of specifications, that enables to set up a system in which a client application can request to be notified when a particular event occurs. The standardization of this process enables to create a publish-subscribe system, easily adding or removing "topics" for which a component wishes to receive notifications.

One common use for WS-Notifications is for one resource to subscribe to receive notifications when changes occur. For example, the subsciber might have a service that monitors the status of the order. That service will need to be notified in the event that the order changes its status.

In its simplest form, the system involves a NotificationProducer, which sends notifications to a NotificationConsumer. These notifications get sent in response to specific "situations", but only when the consumer has been subscribe to a particular "topic".

For example, the Subscriber could subscribe the topic "OrderFailure". When the topic happens a NotificationConsumer is just a Web service designed to receive specific messages. Often Subscriber and NotificationConsumer are the same entity. The following types of information can be found within a subscribe request:

- Where is the location of the notification consumer that will receive the notification messages?
- What is the situation of interest?
- Should all the related notification messages be sent, or just a subset of them?
- Do any other constraints or policies govern the circumstances under which notification messages should be sent to the notification consumer?

E.4.1 Example

To subscribe to a topic, the NotificationConsumer (Subscriber) sends a message to the NotificationProducer. The following example present the complete SOAP structure.

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
       xmlns:tns="http://www.opengis.net/swes/2.0"
       xmlns:wsn-b="http://docs.oasis-open.org/wsn/b-2"
       xmlns:os="http://www.opengis.net/oseo/1.0"
       xmlns:wsa="http://www.w3.org/2005/08/addressing"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation=http://docs.oasis-open.org/wsn/b-2 ../../wsn/b-2.xsd>
    <SOAP-ENV:Header>
       <wsa:Action>
           http://docs.oasis-open.org/wsn/2004/06/WS-
BaseNotification/Subscribe
       </wsa:Action>
       <wsa:To>http://example.com/OrderingService</wsa:To>
       <os:orderId>urn:ESA:EECF:order_id_0001</os:orderId>
    </SOAP-ENV:Header>
    <SOAP-ENV:Body>
    <wsn-b:Subscribe>
<wsn-b:ConsumerReference>
      <wsa:Address>
       http://www.opengis.net/oseo/1.0/OrderingConsumer
      </wsa:Address>
</wsn-b:ConsumerReference>
<wsn-b:Filter>
```

```
      <wsn-b:TopicExpression
Dialect="http://docs.oasis-open.org/wsn/2004/06/TopicExpression/Simple">
            os:OrderFailure
      </wsn-b:TopicExpression>
</wsn-b:Filter>
<wsn-b:SubscriptionPolicy>
      <wsn-b:FixedTopicSet>true</wsn-b:FixedTopicSet>
    </wsn-b:SubscriptionPolicy>
  </wsn-b:Subscribe>
</SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The Header includes the fact that it is trying to create a subscription, as well as the resource to which we are attempting to subscribe, the order identified as “urn:ESA:EECF:order_id_0001”.

The example presents a subscription requests that includes:

- **ConsumerReference**—Identifies the NotificationConsumer: the Web service that will receive the notification message associated with the situations of interest described by the subscription.
- **TopicExpression**—Identifies which topic or topics are of interest to the subscriber. The topic expression is associated with a dialect that describes the contents of the topic expression. The purpose of a TopicExpression is to describe a collection of zero or more topics. Each notification message associated with topic(s) identified by this expression will potentially be sent to the notification consumer. (e.g. the **OrderFailure** topic).

The NotificationProducer sends back a confirmation message to NotificationConsumer (Subscriber):

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
       xmlns:tns="http://www.opengis.net/swes/2.0"
       xmlns:wsn-b="http://docs.oasis-open.org/wsn/b-2"
       xmlns:os="http://www.opengis.net/oseo/1.0"
       xmlns:wsa="http://www.w3.org/2005/08/addressing"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation=http://docs.oasis-open.org/wsn/b-2 ../../wsn/b-2.xsd>
    <SOAP-ENV:Header>
       <wsa:Action>
           http://docs.oasis-open.org/wsn/2004/06/WS-
BaseNotification/SubscribeResponse
       </wsa:Action>
       <wsa:To>http://example.com/OrderingConsumer</wsa:To>
    </SOAP-ENV:Header>
    <SOAP-ENV:Body>
       <wsn-b:SubscribeResponse>
          <wsn-b:SubscriptionReference>
      <wsa:Address>
        http://www.opengis.net/oseo/1.0/OrderingSubscription
      </wsa:Address>
</wsn-b:SubscriptionReference>
       </wsn-b:SubscribeResponse>
    </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

The notification comes in the form of a SOAP message, the body of which might look something like this:

```
<wsn-b:Notify xmlns:swes="http://www.opengis.net/swes/2.0" xmlns:wsn-
b="http://docs.oasis-open.org/wsn/b-2"
xmlns:wsa="http://www.w3.org/2005/08/addressing"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:os="http://www.opengis.net/oseo/1.0"
xsi:schemaLocation="http://docs.oasis-open.org/wsn/b-2 ../../wsn/b-2.xsd">
<wsn-b:NotificationMessage>
      <wsn-b:Topic Dialect="http://docs.oasis-
open.org/wsn/2004/06/TopicExpression/Simple">
            os:OrderFailure
      </wsn-b:Topic>
      <wsn-b:ProducerReference>
            <wsa:Address>http://www.opengis.net/oseo/1.0</wsa:Address>
      </wsn-b:ProducerReference>
      <wsn-b:Message>
            <os:OrderFailure>
<os:code
codeSpace="http://www.opengis.net/oseo/1.0#EventCode">ORDERING_FAILED</os:code>
                  <os:orderId>urn:ESA:EECF:order_id_0001</os:orderId>
            </os:OrderFailure>
      </wsn-b:Message>
</wsn-b:NotificationMessage>
</wsn-b:Notify>
```

The message starts by specifying the topic for which this message contains notifications (`os:OrderFailure`).

Next, the message specifies the NotificationProducer from which the notification is being sent (wsn-b:ProducerReference). Finally, there is the message itself, which includes the message title and content.

Annex F – Revision History

viii. Revision history

Date	Release	Editor	Primary clauses modified	Description
01/04/2011	0.9.9	D.Marchionni	§7.3.2.1, §7.3.3, §9.4	Updated Order Options Definition (Public Comment#1.2.4)
01/04/2011	0.9.9	D.Marchionni	§7.4.3, §7.4.4, §10.1, §10.2, §A.3.9.1, A.3.9.2, A.3.11.1	Updated WS-addressing namespace (Public Comment#1.2.8)
01/04/2011	0.9.9	D.Marchionni	§16.2	Req.111 removed (Public Comment#1.2.10)
01/04/2011	0.9.9	D.Marchionni	§7.3.7, §7.3.8.1, §7.3.14, §14.3.1	Added extension element (anyType) at order level, order item level, order search (Public Comment#1.3.2)
01/04/2011	0.9.9	D.Marchionni	§7.3.9	Updated ParameterData element
01/04/2011	0.9.9	D.Marchionni	all	The size of the following elements has been increased to 4000 chars: orderRemark orderItemRemark productAnnotation specialInstructions sceneSelectionOption/description additionalStatusInfo (Public Comment#1.3.4)
01/04/2011	0.9.9	D.Marchionni	§7.3.2, §7.3.7, §7.3.8, all	“options” renamed to “option” (Public Comment#1.3.6)
01/04/2011	0.9.9	D.Marchionni	§7.3.8.1	productOrderOptionsId made optional in OrderSpecification. (Public Comment#1.3.8)
01/04/2011	0.9.9	D.Marchionni	§15.4.1	Added productId to ItemURLType (Public Comment#1.3.10)
01/04/2011	0.9.9	D.Marchionni	§14.2	Added requirements to GetStatus operation.
22/04/2011	1.0.0	D. Marchionni	All	Text review

Date	Release	Editor	Primary clauses modified	Description
22/04/2011	1.0.0	D. Marchionni	§9.4.4, §10.4.4, §11.4.4, §12.4.4, §13.4.4, §14.4.4, §15.4.4, §16.4.4, §17.4.4	Improved exceptions definition.
22/04/2011	1.0.0	D. Marchionni	§8.4.3	Removed ProgrammingOrders from: Capabilities/Contents/SupportedCollection
22/04/2011	1.0.0	D. Marchionni	§8.2	Added Requirements 14, 15
22/04/2011	1.0.0	D. Marchionni	§A.3.3.3, §A.3.8.1, §A.3.9.1	Added tests A.3.3.3, A.3.8.1, A.3.9.1
22/04/2011	1.0.0	D. Marchionni	§A.4	Updated Traceability Matrix
22/04/2011	1.0.0	D. Marchionni	§D	Updated WSDL
22/04/2011	1.0.0	D. Marchionni	§14.x, §A.3.1.7	Added lastUpdateEnd filtering parameter. Added TooManyHits exception code.