EUROPEAN SPACE AGENCY
DIRECTORATE OF OPERATIONS AND INFRASTRUCTURE
OPS-GI

Directorate of Operations and Infrastructure

EXTERNAL USER INTERFACE CONTROL DOCUMENT (EUICD)

EGOS Data Dissemination System (EDDS)

Reference: EGOS-GEN-EDDS-ICD-1001
Version: 15.0
Date: 2018-02-16

Document Title:	External User Interface Control Document (EUICD)		
Document Reference:	EGOS-GEN-EDDS-ICD-1001		
Document Version:	15.0	**Date:**	2018-02-16
Abstract			

Approval Table:

Action	Name	Function	Signature	Date
Prepared by:	Rokibul Uddin	EDDS Team	Signature numérique de Rokibul Uddin DN : cn=Rokibul Uddin, o=CS GmbH, ou=Space, email=rokibul.uddin@c-ssystems.de, c=DE Date : 2018.04.10 11:41:50 +02'00'	2018-02-16
Verified by:	Delphine Thomas	Application Quality Assurance Engineer (EDDS)	Signature numérique de Delphine THOMAS DN : cn=Delphine THOMAS, o=CS SI, ou, email=delphine.thomas@c-s.fr, c=FR Date : 2018.04.19 15:45:02 +02'00'	2018-02-16
Approved by:	Rui Santos	Technical Officer	Digitally signed by Rui Santos Date: 2018.05.16 13:09:51 +02'00'	2018-02-16

Authors and Contributors:

Name	Contact	Description	Date
M. Hawkshaw	michael.hawkshaw@cgi.com	Author	2016-10-19
R. Ots	rauno.ots@cgi.com	Contributer	2016-10-19
K. Panitzek	kamill.panitzek@cgi.com	Contributer	2016-10-19
Rokibul Uddin	rokibul.uddin@c-ssystems.de	Contributor	2017-05-05 2018-02-16

Distribution List:

Document Change Log

Issue	Date	Description
1.0	2007-04-13	Update in line with PDR outcome.
2.0	2010-10-11	Issue of document for PA delivery
2.1	2011-03-29	Issue for Final Acceptance Delivery
3.0	2011-09-09	Issue for EDDS v1.1.0i1
3.1	2011-10-24	Issue for EDDS v1.1.0i2
3.2	2011-11-14	Issue for EDDS v1.1.0i3
3.3	2012-02-29	Issue for EDDS v1.1.1i1
3.4	2012-04-04	Issue for EDDS v1.1.1i2
3.5	2012-05-18	Issue for EDDS v1.1.2i1
3.6	2012-09-17	Issue for EDDS v1.2.0i1
3.7	2012-09-27	Issue for EDDS v1.2.0i2
3.8	2012-10-12	Issue for EDDS v1.2.0i3
3.9	2013-03-21	Issue for EDDS v1.2.1i1
3.10	2013-04-10	Issue for EDDS v1.2.1i2

Issue	Date	Description
4.0	2013-06-21	Issue for EDDS v1.2.2i1
5.0	2013-12-03	Issue for EDDS v1.3.0i1
6.0	2014-05-28	Issue for EDDS v1.4.0i1
7.0	2014-12-17	Issue for EDDS v1.5.0i1
8.0	2016-02-04	Issue for EDDS v1.6.0i1
9.0	2016-07-04	Issue for EDDS v.2.0.0i1
11.0	2016-10-19	Issue for EDDS v.2.1.0i1
13.0	2017-05-05	Issue for EDDS v.2.2.0i1
15.0	2018-01-15	Issue for EDDS v2.3.0i1

Document Change Record

DCR No:	01		
Date:	2018-02-16		
Document Title:	External User Interface Control Document (EUICD)		
Document Reference:	EGOS-GEN-EDDS-ICD-1001		
Page	**Paragraph**	**Reason for Change**	
189	F.6	Updated for edds#929	
54, 92, 89, 102, 119, 133, 164	C.1, C.28, C.83, 9.3.86	Alias functionality	
54, 64, 88, 89, 92, 97, 101, 103, 131, 161, 164, 165, 169, 174, 178, 186, 214, 220, 272	9.3.84, 9.3.85, 9.3.88, C.23, C.75, C.76, C.84, C.87, D.8, D.23, D.32, D.35, E.7, G.45, H.12,	DCR edds#951	
9, 10, 21	4.3, 4.3.4, Table 3	Updated for edds#934	
45, 20	6.8, 6	Updated for edds#958	

TABLE OF CONTENTS

TABLE OF TABLES

TABLE OF FIGURES

1. Introduction

1.1 Purpose

This document defines the service interfaces available to clients of the EGOS Data Dissemination System (EDDS).

Note: In most cases, an XML schema file is associated to the interface. In the case of updates, the XML files shall be the first source of update. Only after the update of the XML schema file, the change can be reported back to the EUD document.

1.2 Scope

This document describes the service interfaces between the EDDS and its clients. The following interfaces are described:

- EDDS Web Services Interface
- Stream Web Service Interface

This document is a description of the interfaces detailing the common elements that are to be found for all missions. It is expected that other documentation will be available to describe mission-specific elements of the interfaces described.

The readership is expected to be software developers, ESA technical authorities with interest in the EDDS component and users of the EDDS.

It is expected that mission specific Data Definition Interface Documents (DDID) will be created based on the information given in this document (EUICD).

1.3 Document Overview

Section 1 - Introduction (this section) provides the purpose, scope and this document's overview.

Section 2 - References provides the list of reference documents.

Section 3 - Glossary provides a list of acronyms and terms used throughout this document.

Section 4 - Software Overview provides a brief description of the software system and its context.

Section 5 – Describes the data types that are supported by the interface.

Section 6 – Describes the formatting of the data provided through the interface.

Section 7 – Describes the naming conventions used by EDDS for requests and responses

Section 8 – Describes the Delivery mechanism provided by the EDDS.

Section 9 - Describes the underlying web service operations that provide service functionality.

Section 10 – Describes the EDDS Email interface

Section 11 – Describes the client hosted File Server interface

Section 12 – Describes the generic client Java API provided by the EDDS

Appendix A – Provides a list of XML schemas, which define the acknowledgement data types, as described in the acknowledgement.xsd.

Appendix B – Provides a list of XML schemas, which define the authentication data types, as described in the authentication.xsd.

Appendix C – Provides a list of XML schemas, which define the batch request data types, as defined in the batchRequest.xsd.

Appendix D – Provides a list of XML schemas, which define the common data types, as defined in the common.xsd.

Appendix E – Provides a list of XML schemas, which define the delivery data types, as defined in the delivery.xsd.

Appendix F – Provides a list of XML schemas, which define the file system data types, as defined in the filesystem.xsd

Appendix G – Provides a list of XML schemas, which define the filter data types, as defined in the filted.xsd.

Appendix H – Provides a list of XML schemas, which define the format data types, as defined in the format.xsd.

Appendix I – Provides a list of XML schemas, which define the report data types, as defined in the report.xsd.

Appendix J – Provides a list of XML schemas, which define the schedule data types, as defined in the schedule.xsd.

Appendix K – Provides a list of XML schemas, which define the user management data types, as defined in the userrequest.xsd.

Appendix L – Provides a list of XML schemas, which define the stream data types, as defined in the streamrequest.xsd.

Appendix M – Provides info about the EDDS Parameter Spreadsheet.

Appendix N – Provides info about Google Protocol Buffers

2. References

2.1 Applicable documents

Ref.	Document Title	Issue and Revision, Date
[AD-1]	CCSDS, XML Formatted Data Unit (XFDU) Structure and Construction Rules.	BLUE BOOK 661.0-B.1, Set 2008
[AD-2]	Generic Data Disposition System (GDDS) Generic Data Delivery Interface Document DDID (EGOS-MCS-GDDS-1003) Appendix F	Issue 1.4, 2008-04-11

2.2 Reference documents

Ref.	Document Title	Issue and Revision, Date
[RD-1]	EDDS Client Software User Manual [EGOS-GEN-EDDS-SUM-1001]	Version 14.0 2018-02-16
[RD-2]	Multipurpose Internet Mail Extensions (MIME) Part One: Format of Internet Message Bodies [RFC 2045]	November 1996
[RD-3]	Web Services Definition Language (WSDL) 1.1 [http://www.w3.org/TR/wsdl]	2001-03-15
[RD-4]	MUST Architectural Design Document [no reference]	Version 1.1, 2006-01-13
[RD-5]	XSL Transforms (XSLT) Version 1.0 [http://www.w3.org/TR/xslt]	1999-11-16
[RD-6]	Extensible Markup Language (XML) 1.0 (Fourth Edition) [http://www.w3.org/TR/xml]	2006-09-29
[RD-7]	SOAP Version 1.2 Part 1: Messaging Framework [http://www.w3.org/TR/soap12-part1/]	2003-06-24
[RD-8]	Deleted	Deleted
[RD-9]	Deleted	Deleted
[RD-10]	SCOS-2000 Glossary, Definitions and Acronyms [EGOS-MCS-S2K-GLO-0001]	Version 1.7 2007-03-20
[RD-11]	Software Requirements Specification (SRS), EGOS Data Dissemination System (EDDS) [EGOS-GEN-EDDS-SRS-1001]	Version 18.0, 2018-02-16
[RD-12]	Software Design Document (SDD), EGOS Data Dissemination System (EDDS) [EGOS-GEN-EDDS-SDD-1001]	Version 13.0, 2017-05-05
[RD-13]	Generic Data Disposition System (GDDS) Generic Data Delivery Interface Document (DDID) [EGOS-MCS-GDDS-1003]	Version 1.4, 2008-04-11
[RD-14]	TDRS External Interfaces Control Document [EGOS-MCS-S2K-ICD-0017]	Version 5.1, 2008-08-22
[RD-15]	EDDS Configuration and Installation Guide [EGOS-GEN-EDDS-CIG-1001]	Version 14.0, 2018-02-16
[RD-16]	Google Protocol Buffers [https://developers.google.com/protocol-buffers/]	2012-09-17

3. Glossary

3.1 Acronyms

Acronyms	Description
AES	Advanced Encryption Standard
APID	Application Identifier
ASCII	American Standard Code for Information Interchange
CCSDS	Consultative Committee for Space Data Systems
CDR	Critical Design Review
CUC	CCSDS Unsegmented Code
DARC	Parameter Archive
DDID	Data Definition Interface Document
DVD	Digital Video Disc (Digital Versatile Disc)
EDDS	EGOS Data Dissemination System
ESA	European Space Agency
EGOS	ESA Ground Operations System
ERT	Earth Reception Time (Note: SCOS-2000 uses the SCOS-2000 reception time stamp and not the ground station reception time stamp.)
EV	Event (Mission control system event)
FARC	File Archive
GDDS	Generic Data Disposition System
HTTP	Hypertext Transfer Protocol
HTTPS	HTTP over Secure socket layer
JMS	Java Messaging Service
MAS	Mission Automation System
MCS	Mission Control System
MMI	Man Machine Interface
MUST	Mission Utility & Support Tool
OBEV	On-board Event
OBQ	On-board Queue
OBSM	On-board Software Maintenance
OBT	On-board Time
OOL	Out Of Limit
PARC	Packet Archive
PID	Process Identifier
PUS	Packet Utilisation Standard
SCOS	Spacecraft Control and Operations System
SMTP	Simple Mail Transfer Protocol
SMTPs	Simple Mail Transfer Protocol Secure
SOAP	*Originally* - Simple Object Access Protocol - *but since version 1.2 it has no official meaning*
SPID	SCOS-2000 Packet Identifier
SUM	Software Users Manual
TBD	To be decided
TBW	To be written
TC	Telecommand
TM	Telemetry
WSDL	Web Services Description Language
XFDU	XML Formatted Data Unit
XML	Extensible Mark-up Language

Acronyms	Description
XSL	Extensible Stylesheet Language
XSLT	XSL Transformations

3.2 Definition of Terms

Terms	Description
Batch Service	An EDDS service that allows a user or client application to make a request for mission data and receive a (single) response from the EDDS that contains the complete set of data requested.
Stream Service	An EDDS service that allows a client application to receive a continuous stream of mission data rather than a finite stored data set.
SCOS-2000	SCOS-2000 is a generic configurable spacecraft control and monitoring system with multi-domain and multi-mission capabilities, which operates in a scalable distributed environment. It is intended to be taken by client missions and customised to meet their specific requirements. At the time of writing this document SCOS-2000 release 5 is expected to be the baseline.
Web Services	Web services provide a standard means of interoperating between different software applications, running on a variety of platforms and/or frameworks. Web services are characterized by their great interoperability and extensibility, as well as their machine-processable descriptions thanks to the use of XML. They can be combined in a loosely coupled way in order to achieve complex operations. Programs providing simple services can interact with each other in order to deliver sophisticated added-value services. [quoted from W3C Web services Activity Statement - http://www.w3.org/2002/ws/Activity]
SCOS-2000 Domain	The controlled or controlling entity whose packet data is stored in a dedicated SCOS-2000 multi-domain archive. For example, a spacecraft or a ground station or a controlling domain such as System Control. Each individual spacecraft will be mapped to a specific domain. Each domain will have its own instance of SCOS-2000. Taken from [RD-10].
SCOS-2000 Packet Identifier (SPID)	The identifier of the history file containing all instances of a SCOS-2000 packet. This is unique for each packet type (TM, TC and EV). In the telemetry processing, the SPID is also used as the unique identifier of the packet content for fixed packets. Taken from [RD10].
SCOS-2000 DataStream	A logical grouping of SCOS-2000 history files used to partition the historical data and to enable the selective retrieval and processing of them. Also refers to a stream of telemetry transfer frame data received by the SCOS-2000 Packetiser. Taken from [RD10].
SCET	Used in this document to indicate the value of the counter from the spacecraft's onboard clock (i.e. an integer value).
Onboard Time (OBT)	Used in this document to mean the UTC correlated onboard time corresponding to SCET. It should be noted that duplicate packets with the same SCET can have a different OBT if they are time-stamped using a different time-correlation.

4. Software Overview

The primary purpose of the EDDS is to provide controlled access to mission data for users who do not have access to the mission control system (MCS) monitoring and control facilities. The data includes science and non-science telemetry data, telecommand data, and auxiliary data such as flight dynamics data.

The EDDS offers data services and management services. A user interface is provided to these services through the EDDS Client Application see [RD-1]. This document (EUICD) describes the interfaces available to users and client applications. These external interfaces provide access to the EDDS data services:

- Authentication Services
- Batch Service
- Stream Service
- User Management Service

4.1 EDDS Services Overview

4.1.1 Authentication Services

The EDDS User Management Service allows a client application to make request for the log in and log out of the users from the EDDS. It also allows for a sufficiently privileged user to update the LDAP details held on themselves or others.

4.1.2 Batch Service

The batch service is intended to allow clients to make requests for mission data and receive a data response that contains a complete set of data that matches the request. In general, a request lists the data types and allows the user to apply a set of filters for each data type. The response data is sent to the client via the delivery method indicated in the request. A batch service request can be viewed as transient in the sense that the EDDS processes the request, builds the data set (by retrieving the relevant data) and then delivers the data set to the user. The request is then considered complete.

A batch request is submitted as an XML document. The data response can take one of these forms:

- A binary or ASCII file (can also be compressed or encrypted);
- An XML file displayed on the MMI.

The format of response content depends on the type of requests as described in Section 6

Each request always has associated acknowledgement data. The acknowledgement data can be return to the user on explicit user request. The acknowledgement data indicates the status of the request and gives details for failures. Acknowledgement data is received as an e-mail or can be downloaded as an XML document.

The Batch request response files can be compressed. The compressed files can be one of a number of file archive formats that are supported by the EDDS. By default, compression is applied to the data. Currently the supported compressed file types are:

- zip – ZIP;
- tar - USTAR format (uncompressed);
- tar.gz - USTAR format compressed with gzip.

On top to the data compression, the EDDS allows to encrypt the data according to the AES algorithm, which is enabled or disabled at mission level.

4.1.3 Stream Service

An EDDS stream service allows a client application to receive a continuous stream of mission data rather than a finite stored data set. A client application makes a request for the stream service to the EDDS and then actively requests the streamed data from the EDDS.

A stream request is submitted as an XML document. The data response can be obtained either directly from the ActiveMQ topic or via the client API via CometD. The EDDS MMI uses the client API. Each message on the topic represents one packet or parameter, each associated with the original request by the request ID header field on the message. Currently only streaming of parameters from the DARC is supported.

The payload of each message contains a framed byte array, which represents an encoded parameter using Google Protocol Buffers. The payload can be decoded using the provided helper classes found in the esa.egos.edds.types package. For DARC Parameters, the DarcParamRec class can be used. The proto files describing the data structure are stored in edds-ws-common and are compiled as a part of the build process.

4.1.3.1 DARC Streaming Support

Receiving streamed data from the DARC only works with compatible versions of DARC for SLES 12 (version 3.0.0 or later). EDDS expects the DARC to put parameters samples onto an ActiveMQ topic owned by the DARC with the following header data:

Header Field	Type	Description
ValueType	String	The type of the value field – one of BOOLEAN, INTEGER, STRING, DOUBLE, FLOAT, LONG
Value	Variable	Value of the parameter sample. Type can be either Boolean, Short, Integer, String, Double, Float. For Short values, the ValueType should be set to INTEGER.
Validity	Integer	One of 0 (Valid), 1 (Invalid), 2 (Unknown), 3 (Expired)
SystemElement	String	Any String value
Source	String	Any String value
State	Short	Any Short value
GenericData	String	Any String value
GenerationTime	Double	Unix time stamp of the generation time
Name	String	Name of the parameter

When a user requests live parameter samples from the DARC, EDDS will add a listener to this topic with a selector on these header fields based on the filter specified in the request. This improves performance, as only matching messages are delivered by the broker to EDDS for processing. EDDS then creates a new message with two header fields, “RequestId” (with the user’s original request ID) and “MISSION_NAME” for the mission the stream relates to. It then creates a Google Protocol Buffers encoded record (See Appendix N) with the data from the header fields of the original message and adds it to the payload of a new message. These messages are received by the web server (or a 3rd party application – as they are placed on a topic, this doesn’t interfere with the web server) and passed on to any listening clients via CometD.

4.1.3.2 PARC Streaming Support

Streaming of live packet data is supported only via the Data Provision Services that can be found in SCOS 5.4.21 and 5.5.4 or later. Once a request for live streaming data is submitted (can be for TM, TC, EV (SCOS Event Log) or OOL) and the request becomes ACTIVE, then the data is put onto the ActiveMQ message bus onto a topic for any 3rd party application to listen to directly. Alternatively, if access to the message bus is not possible, then the method “startStreamData(RequestID)” on the WSDL can be executed to start streaming the live data to clients over HTTP(S) via CometD. The client must then listen to the relevant Bayeux channel as specified in the next subsections.

4.1.3.3 Packet TM

Messages are placed onto the topic “esa.egos.edds.packet.stream.tm” with the following header values:

Header Field	Type	Description
RequestID	String	The ID of the request from the client that initiated the live streaming
MISSION_NAME	String	The name of the mission that the request relates to

The body of the message is as defined for binary Packet TM Report response files and can be found in 6.7.3.3.7

Clients must call startStreamData(RequestID) on the EDDS Web Server and then listen on the Bayeux channel NotificationChannel.PKT_TM_STREAM. When finished, clients should call the stopStreamData(RequestId) method.

A code example for streaming live telemetry data can be found under edds-ws-client/src/test/java in packet esa.egos.edds.ws.client.examples.StreamLivePktTmData

4.1.3.4 Packet TC

Messages are placed onto the topic “esa.egos.edds.packet.stream.tc” with the following header values:

Header Field	Type	Description
RequestID	String	The ID of the request from the client that initiated the live streaming
MISSION_NAME	String	The name of the mission that the request relates to

The body of the message is as defined for binary Packet TC Report response files and can be found in 6.7.3.3.6

Clients must call startStreamData(RequestID) on the EDDS Web Server and then listen on the Bayeux channel NotificationChannel.PKT_TC_STREAM. When finished, clients should call the stopStreamData(RequestId) method.

4.1.3.5 Packet EV

Messages are placed onto the topic “esa.egos.edds.packet.stream.ev” with the following header values:

Header Field	Type	Description
RequestID	String	The ID of the request from the client that initiated the live streaming
MISSION_NAME	String	The name of the mission that the request relates to

The body of the message is as defined for binary Event Record Report response files and can be found in 6.7.3.3.4

Clients must call startStreamData(RequestID) on the EDDS Web Server and then listen on the Bayeux channel NotificationChannel.PKT_EV_STREAM. When finished, clients should call the stopStreamData(RequestId) method.

4.1.3.6 Out of Limits

Messages are placed onto the topic “esa.egos.edds.packet.stream.ool” with the following header values:

Header Field	Type	Description
RequestID	String	The ID of the request from the client that initiated the live streaming
MISSION_NAME	String	The name of the mission that the request relates to

The body of the message is as defined for binary Out of Limits Record Report response files and can be found in 6.7.3.3.9

Clients must call startStreamData(RequestID) on the EDDS Web Server and then listen on the Bayeux channel NotificationChannel.PKT_OOL_STREAM. When finished, clients should call the stopStreamData(RequestId) method.

4.1.4 User Management Service

An EDDS User Management Service allows a client application to make request for the management of the users, which have access to the EDDS services. Only authorised users can access the User Management services.

A user management request is submitted as an XML document.

4.2 EDDS Interface Overview

The following section describes the interfaces to the EDDS services.

4.2.1 EDDS Web Service Interface

Provides the set of operations that can be used by client applications to inject batch requests, user management requests and receive responses. The interface uses Web Services technology [RD-3][RD-7].

4.2.2 Stream Web Service Interface

Provides a set of operations that can be used by client applications to make stream requests and receive data in response. The interface uses Web Services technology [RD-3][RD-7].

4.2.3 Email Interface

Used by the EDDS components to deliver acknowledgement data to the user.

4.2.4 Delivery Manager Interface

The Delivery Manager is used by the EDDS to delivery batch data responses to a client hosted file server.

4.3 Client Applications Overview

A user can interact with the EDDS by use of several client application types:

- EDDS Client Application (via generic APIs)
- Web Services Application
- Email Server
- Delivery Manager
- Request Submitter
- EDDS stream client

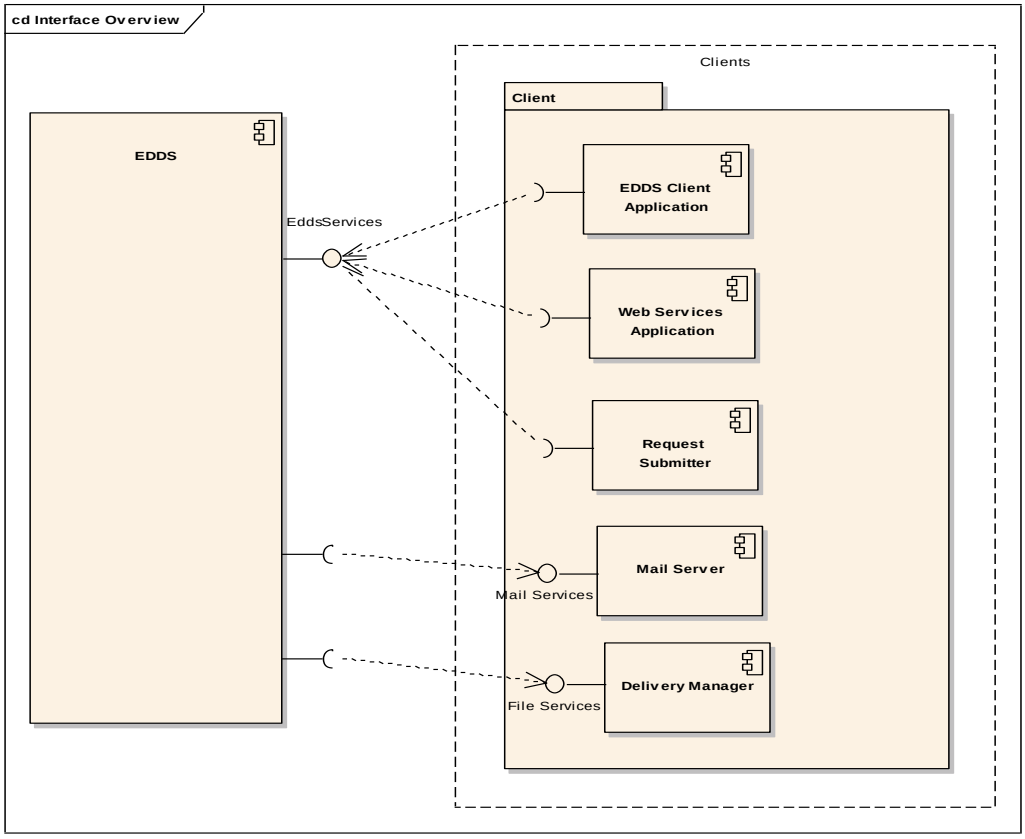


Figure 1 - Interface Overview

4.3.1 EDDS Client Application

The user interface to the EDDS is the EDDS Client Application. This application is downloaded from the EDDS web site and is run on the user's machine. See the EDDS Client SUM [RD-1] for more information on installation configuration and usage.

The EDDS Client provides GUI access to all available EDDS services (management and batch and streaming services). This document (EUICD) describes the format of the data that is sent to the Web Services Application.

The EDDS provides access to its management batch and stream services through Web Services technology. Users can write their own client applications that access EDDS batch and stream services through the use of Web Services technology.

4.3.2 Email Server

EDDS is capable of delivering batch service acknowledgement messages to a client via email. This document (EUICD) describes the format of the acknowledgement message.

4.3.3 Delivery Manager

EDDS is capable of delivering batch service responses to a file server running on the client's site. This document (EUICD) describes the format of batch service requests and the format of the resultant response. This document (EUICD) also describes the format of acknowledgement messages.

4.3.4 EDDS Stream client

The EDDS Stream client is a lightweight standalone application that is able to store stream data in files, in various format, from an existing edds stream request. This client is designed to be easily extensible, so that third-party users can extend the client to support new output file formats. Please refer to the CIG for further information.

4.3.5 Request Submitter

The EDDS Request Submitter is a lightweight standalone application that polls a configurable directory for request files. Any valid request files either in EDDS or GDDS format will be processed and submitted to the EDDS Server. Responses are returned as defined in the request.

All valid requests are submitted through the same user account specified in the RS configuration.

The EDDS request file must conform to the schema element RequestMessagePart as defined in [C.64]

4.3.5.1 GDDS Requests

EDDS Request Submitter also accepts requests in GDDS format [RD-13]. All GDDS request types are supported, but there are additional restrictions, because GDDS requests and EDDS requests are not fully compatible. The GDDS request is transformed into EDDS request via XSL Transformation (XSLT) and submitted for processing via EDDS client API. The mapping of GDDS schema elements is described below.

The response data will be in EDDS format and delivered by EDDS.

XSL transformation can be also performed independently of Request Submitter. On SLES 11 run from the terminal:

xsltproc path/to/transformation.xsl path/to/gddsRequest.xml

For example, to transform one of the test requests and output it to the console, you can run from edds source directory:

xsltproc edds-request-submitter/src/main/resources/gdds/GDDS2EDDS.xsl edds-request-submitter/src/test/resources/GDDSRequests/PktTcReportRequest.xml

4.3.5.1.1 Common request fields

Element in GDDS request	**Is mapped to**
userRequestId attribute in scheduleRequest and onlineRequest	Not used, but are added to the EDDS request comment
schedule/firstStartTime	If set, used as execution time of the first request
schedule/interval	Number of days between scheduled requests; GDDS request doesn't have end date in the schema, so 10 requests will always be created for scheduled requests
schedule/acknowledgement	Not used
general/comment	EDDS request comment field
general/userInfo/username and general/userInfo/FTPpassword, general/destInfo/FTP/target	Username and password are used as credentials for target FTP server if latter is provided; otherwise EDDS server delivery is used.
general/destInfo/RDM and subelements	Not used
general/destInfo/Online and subelements	Not used
general/destInfo/FTP/filename	Used as the target filename
general/destInfo/FTP/directory	Target directory, if file server delivery is used
general/formatInfo/compression	EDDS compression option
general/formatInfo/SFDUrequired	Not used
general/formatInfo/missionFormat	Not used
general/dataInfo/earliestStart	If set, the execution time of the (first) request; schedule/firstStartTime overrides this value

item	Exactly one item should be specified, contains the request type and filter elements. The filter tree must not contain any OR binary operation nodes inside AND nodes

4.3.5.1.1.1 Mission and Domain

By default, the request is submitted to the default mission and domain as configured in the file poller settings. Different mission and domain can be used for the request by having extra lines in the GDDS comment field. Keyword Mission or Domain followed by a colon character, and then the EDDS mission or domain name.

Example:

```
...
        <general>
                <comment>
                        Mission:TEST_MISSION
                        Domain:0
                          Example GDDS request comment
                </comment>
                ...
```

4.3.5.1.1.2 MultiTime elements

All the Multitime elements (i.e. SourcePktsGenTime, ExecutionTime, UplinkTime) should be in absolute time format, not in duration. Except when scheduleRequest is sent, then the required boundaries must be set as duration – start time (lower boundary) as relative time from the execution and end time as the duration after start time. Other MultiTime elements should still be in absolute time.

Example:

The following GDDS request will create scheduled request that will execute 10 requests in 5 days intervals. The packet data is retrieved starting from 3 months before the execution time and ending 21 days after start time. Additonal filter is applied with UplinkTime >= 2002-01-01T12:00:00Z.

The relevant fragments of the request:

```
<scheduleRequest xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation='GDDSScheduleRequest.xsd'
userRequestId="20020101GDDSScheduleRequestExample">
<schedule>
   <firstStartTime>2002-01-01T20:00:00Z</firstStartTime>
   <interval>5</interval>
   <acknowledgement>true</acknowledgement>
</schedule>
<onlineRequest  userRequestId="20020101GDDSRequestExample">
...
        <item>
                <dataType>CMH</dataType>
                <dataSource>CMDH</dataSource>
                <catalogueRequest>false</catalogueRequest>
                <filter>
                        <bin operation="OP_AND">
                                <lhs>
                                        <leaf operation="OP_GTE">
                                                <valuePair>
                                                        <ExecutionTime>
                                                                <a_duration>-P3M</a_duration>
                                                        </ExecutionTime>
                                                </valuePair>
```

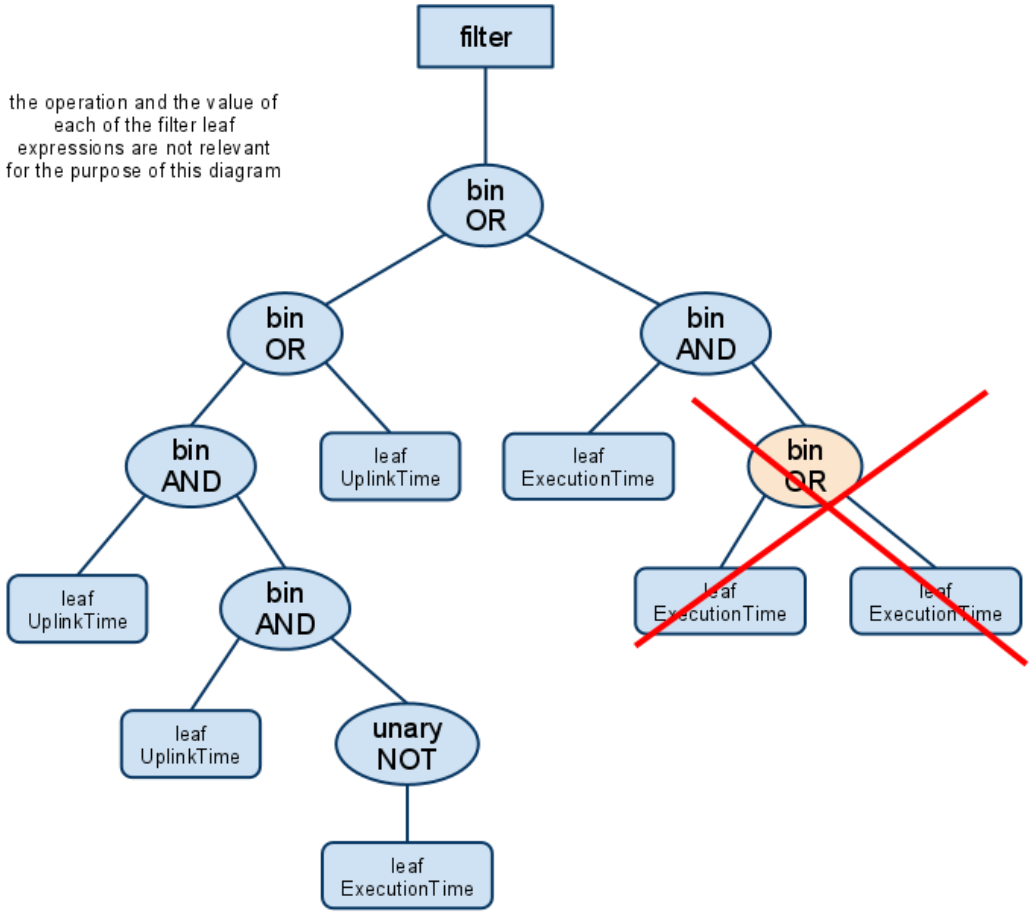


```
                                        </leaf>
                                </lhs>
                                <rhs>
                                        <bin operation="OP_AND">
                                                <lhs>
                                                        <leaf operation="OP_GTE">
                                                                <valuePair>
                                                                        <UplinkTime>
                                                <a_dateTime>2002-01-01T12:00:00Z</a_dateTime>
                                                                        </UplinkTime>
                                                                </valuePair>
                                                        </leaf>
                                                </lhs>
                                                <rhs>
                                                        <leaf operation="OP_LTE">
                                                                <valuePair>
                                                                        <ExecutionTime>
                                                                <a_duration>P21D</a_duration>
                                                                        </ExecutionTime>
                                                                </valuePair>
                                                        </leaf>
                                                </rhs>
                                        </bin>
                                </rhs>
                        </bin>
                </filter>
        </item>
</onlineRequest>
</scheduleRequest>
```

4.3.5.1.1.3 Filter expressions

Filter expressions should be simple: there should be no OR operations inside AND operations. So all the OR expressions should be towards the filter root. If there are any of those “illegal” expressions, the whole subtree will be ignored.

Example:



4.3.5.1.2 TM Packet request

This section describes the trypes of data request available in GDDS and how they map to the EDDS schema (where possible to map)

Item subelements	Description
dataType='TLM'	Determines the TM packet request type
dataSource	Filter packets by this Apid value
catalogueRequest	Not used
keyword	Not used
onEvent	Not used
filter	Required. Must contain a leaf element with operation OP_GTE or OP_GT and with keyword SourcePktsGenTime and another one with operation OP_LTE or OP_LT. These will be the lower and upper bound of the timerange for the query.

Supported keywords in filter are

Filter keyword	Mapped to	Allowed operations
SourcePktsGenTime and S2KpktsGenTime	GenerationTime	All operations
VolumeSize	MaxVolumeSize	All operations
Type, SubType, P1Val, P2Val		OP_EQ and negation in unary node with OP_NOT. One

		keyword can only appear once in a conjunctive clause (leafs separated with only AND expressions).

4.3.5.1.3 TC Packet History

Item subelements	Description
dataType='CMH'	Determines the TC packet request type
dataSource='CMDH'	Determines the raw TC packet request type
catalogueRequest	Not used
keyword	Not used
onEvent	Not used
filter	Required. Must contain a leaf element with operation OP_GTE or OP_GT and with keyword ExecutionTime and another one with operation OP_LTE or OP_LT. These will be the lower and upper bound of the timerange for the query.

Supported keywords in filter are

Filter keyword	Mapped to	Allowed operations
ExecutionTime	ExecutionTime	All operations
UplinkTime	UplinkTime	All operations
VolumeSize	MaxVolumeSize	All operations

4.3.5.1.4 TC Packet brief and full report

Item subelements	Description
dataType='CMH'	Determines the TC packet request type
dataSource='CMDPB' or dataSource='CMDPF'	Whether we want to get the brief report ('CMDPB') or full report with command parameters ('CMDPF').
catalogueRequest	Not used
keyword	Not used
onEvent	Not used
filter	Required. Must contain a leaf element with keyword ExecutionTime and with operation OP_GTE or OP_GT and another one with operation OP_LTE or OP_LT. These will be the lower and upper bound of the timerange for the query.

Supported keywords in filter are

Filter keyword	Mapped to	Allowed operations
ExecutionTime	ExecutionTime	All operations
UplinkTime	UplinkTime	All operations

VolumeSize	MaxVolumeSize	All operations

4.3.5.1.5 ArchiveFile request

Item subelements	**Description**
dataType='AUX'	Determines the request type
dataSource	Not used
catalogueRequest	Not used
keyword	Not used
onEvent	Not used
filter	Required.

Supported keywords in filter are

Filter keyword	**Mapped to**	**Allowed operations**
Name	Name	OP_EQ
Release, Issue	Version (Release and Issue are concatenated with a dot '.' character)	OP_EQ
CreationTime	CreationTime	Not supported
TimeSpan	ValidityTime	Not supported

5. Data Types

This section outlines a user's perspective of the data types that the EDDS supports. Table 1 (below) gives an overview of the data types and the services which provide the data type.

Data Type \ Delivery		File Server	EDDS Server	Stream		Email
				Online	Offline	
Packet	*TM (EDDS Raw)*	✓	✓	✓	✓	-
	TM (PARC Raw)	✓	✓	✓	✓	-
	TC (EDDS Raw)	✓	✓	✓	✓	-
	TC (PARC Raw)	✓	✓	✓	✓	-
	EV (EDDS Raw)	✓	✓	✓	✓	-
	EV (PARC Raw)	✓	✓	✓	✓	-
	Statistics	✓	✓	-	-	-
Param-eter	*TM*	✓	✓	✓	✓	-
	Statistics	✓	✓	-	-	-
	Preview	✓	✓	-	-	-
	Definition	✓	✓	-	-	-
Report	*MCS*	✓	✓	-	-	-
	EDDS	✓	✓	-	-	-
Archived Files	*File*	✓	✓	-	-	-
	Catalogue	✓	✓	-	-	-
File System	*File Catalogue*	✓	✓	-	-	-
	Folder Catalogue	✓	✓	-	-	-
	File	✓	✓	-	-	-
Acknowledgement		✓	✓	-	-	✓

Table 1 - Data Type & Service Overview

5.1 Packet

Packet data provided by the EDDS consists of telemetry (TM), telecommand (TC) and MCS event (EV) packets. The packets contain binary data as processed by the MCS in Network endian order. In addition packet statistics data is available that provides statistical information about packets stored in the MCS (e.g. the number of packets that match a given criterion for a given time span). Packets are extracted by using either the PARC Manager interface of a dedicated packet archive (PARC) or the Data Provision Services. The TM/TC/EV Raw format retrieves the packet data from the PARC exactly as it is stored as a hexadecimal string, and wraps each packet in an XML tag so that they can be identified. The data is not decoded, and so cannot be filtered beyond the basics of start time, end time and (for TM and EV) SPID list.

Packet statistics data are available that provides statistical information about stored packets. Packet Reports are available that decode the encoded packet data and store the data in an XML or ASCII formatted file.

5.2 Parameter

TM parameter data can be obtained through EDDS services. Parameter data is available from a dedicated parameter archive (DARC or SMON).

The parameter archive is populated as TM source packets are received by the MCS and is tailored to provide fast access to the parameter data stored. Due to the large number of TM parameters defined for a mission and their associated data, it may be impractical to store all parameters in a dedicated parameter archive. The mission will be able to configure which subset of parameters is stored in the dedicated parameter archive.

The ordering of parameter samples is not guaranteed (although usually the data is ordered by sample generation time). EDDS tries to use optimal retrieval mechanisms based on the API provided by backend and does not provide any ordering on its own.

Duplicate entries might exist in special cases where request for all DARC parameter samples is being resumed (after an explicit client request to suspend/resume or after EDDS server has been stopped/killed before processing of the request was finished). To avoid duplicates, split size of the result files can be set to 0, so the entire request will be restarted from the beginning.

Parameter statistics data is available that provides statistical information about stored parameters (e.g. the number of parameters that match a given criterion for a given time span).

Parameter Definition data type is used to provide parameter static definition data (e.g. description, unit, subsystem, etc) which is derived from the spacecraft database.

5.3 Report

This data type covers both reports generated by the MCS and those generated by the EDDS. A report is always in XML, ASCII, Binary or spreadsheet format.

MCS reports can be classified into TM, TC, EV and other reports. The availability of such reports is mission dependent and includes:

- TM Packet Report
- TM Packet Gap Report
- TC Packet Report
- OOL Data Report
- Event Record Report

EDDS reports are generated by the EDDS on user request. They are:

- Status Reports (not currently supported in this version)
- Request Summary Reports (not currently supported in this version)
- System Log Reports
- EDDS Usage Report

The content of EDDS reports is given in Section 6.2.3

Reports supported by the EDDS will be in XML format, with missions providing XSLT for transforming the reports into their final delivery format. This also allows missions to specialise reports with full EDDS support.

For Out of Limit Reports, EDDS supports the following packet versions from the PARC:

- 5: For SCOS versions 5.4.19 or 5.5.2 or earlier
- 6: For SCOS versions 5.4.20 or 5.5.3 or later.
- 7: For SCOS versions 5.4.20 or 5.5.3 or later with change request GAIA-SPR-694 applied.

5.4 Archived Files

The Archived Files data type contains files stored in the MCS archive including files received from Flight Dynamics, Mission Planning and TT&C stations. Additionally catalogue information is also available in XML format. The file data and catalogue are retrieved from the FARC. For FARC versions that support it, it is possible to register interest in items in the FARC so that when they change, a notification is received by EDDS. EDDS will then automatically retrieve the requested file from the FARC.

5.5 Acknowledgement

Acknowledgement data is data generated by EDDS and sent/taken to the user on request. Acknowledgement data contains status information. Acknowledgement data are produced by the EDDS at the different stages of the request processing and execution when the request for data is received by the EDDS or when the request has been completed successfully.

5.6 Request and Response Mappings

The table below maps the XML responses to each request so that the expected XML response types can be determined based on the request submitted. Note that not all requests will result in an XML file being returned.

Request type (BatchRequest/DataRequest)	Response type (ResponsePart/Response)
Param	ParamResponse C.44
SMONParam	SMONParamResponse C.72
ParamStatistics	ParamStatisticsResponse C.50
ParamDefinition	ParamDefinitionResponse C.35
ParamPreview	ParamPreviewResponse C.41
PktEv	N/A – binary response format
PktTm	N/A – binary response format
PktTc	N/A – binary response format
PktEvRaw	PktRawResponse C.54
PktTmRaw	PktRawResponse C.54
PktTcRaw	PktRawResponse C.54
PktTmStatistics	PacketStatisticsResponse C.27
PktTcStatistics	PacketStatisticsResponse C.27
PktEvStatistics	PacketStatisticsResponse C.27
PktTmReport	PktTmReportResponse I.24
PktTmGapReport	PktTmGapReportResponse I.29
PktTcReport	PktTcReportResponse I.20
EventRecordReport	EventRecordReportResponse I.9
ArchiveCatalogue	CatalogueResponse C.14
ArchiveFile	N/A – binary response format
ArchiveSubscription	N/A – binary response format
OolRecordReport	OolDataReportResponse I.11
EddsUsageReport	EddsUsageReportResponse I.5
FileSystemFileCatalogue	FileSystemFileCatalogueResponse F.7
FileSystemFolderCatalogue	FileSystemFolderCatalogueResponse F.10
FileSystemFile	N/A – binary response format
FileSystemFileSubscription	N/A – binary response format

Table 2 – Request and Response Mappings

6. Formats

The following section describes the formatting of data. The EDDS supports following formats:

- Binary
- EDDS Binary
- XML (and XML Transform)
- SFDU [AD-2]
- ASCII
- RawSourceBinary

Different data types support various formatting. Table 3 provides a summary of the possible formats that are supported by each request type.

Format / ***Request Type***		*EDDS Binary*	*Binary*	*SFDU*	*XML (XML Transform)*	*ASCII*	*Others*	*RawSourceBinary*
Packet (From PARC)	*TM*	✓*		✓			GDDS_BINARY*	
	TM (PARC Raw)		✓*		✓*			✓
	TM Report		✓*		✓*	✓*		
	TM Gap Report		✓		✓			
	TM Statistics				✓			
	TC	✓*		✓			GDDS_BINARY*	
	TC (PARC Raw)		✓*		✓*			
	TC Report		✓*		✓*	✓*		
	TC Statistics				✓			
	EV	✓*						
	EV (PARC Raw)		✓*		✓*			
	Event Record Report		✓*		✓*	✓*		
	EV Statistics				✓			
	OOL Record				✓*	✓*		
Parameter (from SMON)	*TM*		✓*		✓*		TDRS	
Parameter (From DARC)	*TM*		✓*		✓*		TDRS	
	Statistics				✓*	✓*		
	Preview				✓*	✓*		
	Definition		✓*		✓*			
Report (From EDDS)	*EddsUsage*				✓			
Archived Files (From FARC)	*File (and File Subscription)*		✓					
	Catalogue				✓	✓		
File System	*File Catalogue*				✓	✓		
	Folder Catalogue				✓	✓		
	File (and Subscription)		✓					
Acknowledgement					✓			

Table 3 - Data Type Formatting

The formats marked with * support splitting the response files into smaller chunks. The resulting files are still compliant with the format with appropriate header and tail in the files. To enable splitting, the configuration must be done by EDDS admin.

The following sections detail the formats and the applicable data types.

6.1 EDDS Binary

The format for EDDS raw data is both self-describing and extensible, and allows a mission to choose which of the standard EDDS header fields are included in that file. For a response consisting of *N* elements, the basic structure is as follows:

EDDS Raw Header
EDDS Header Fields (1)
Data Element (1)
EDDS Header Fields (2)
Data Element (2)
…
EDDS Header Fields (*N*)
Data Element (*N*)

Table 4 - Structure of EDDS Binary Data

The response data consists of a single EDDS Raw Header (based on what has been selected as the header fields), followed by the data element (this is the case both when the data is stored in a raw (ASCII hexadecimal) file, and when raw data is returned via a streaming service). Each data element is preceded by the EDDS header fields.

6.1.1 EDDS Raw Header

The EDDS Raw Header is a variable-size header, which serves the following purpose:

- Allows a user to uniquely identify binary data as being EDDS binary data
- Allows a user to find out which header fields are present in the binary data

The format of the EDDS binary header is as follows:

Field	Size (octets)	Type	Description
N	1	Integer	Total number of header fields defined for this data type. This is the total number of EDDS header fields (can be zero)
Size_1	1	Integer	Number of octets used for Header Field 1
Size_2	1	Integer	Number of octets used for Header Field 2
…	…	…	…
Size_*N*	1	Integer	Number of octets used for Header Field *N*

Table 5 - Format of the EDDS File Header

If should be underlined that in case one of the fields is a string the Size field of the header will not represent its length (since the size of a string is known only at run time) but will indicate if the field has to be included or not in the generated file.

The actual header fields, which can be present in EDDS binary data, depend on the data type; these will be defined in the following sections of this document.

Note: this section will contain the value “00” if the values of size1, size2, …, size *N* (defined in the EDDS File Header) are all zero.

6.1.2 Packet Data

The packet data depends on the data type. This will be defined in the following sections for each data type. The Appendix “Packet Data” in the Configuration and Installation Guide (CIG) [RD-15] contains detailed instruction on the implementation of the header structures for the information files.

6.1.2.1 TM

The EDDS defines the following fields, which can be included in the header for TM packet data (see section 6.1.1):

Number	Field	Type	Description
1	Source Sequence Counter	Unsigned Integer	The sequential binary count of each source packet generated by an application process identified by a unique application process identifier.
2	APID	Unsigned Integer	The PUS Service Application Process Identifier, uniquely identifies the on board source of the packet.
3	PID	Unsigned Integer	The Packet ID, calculated from the APID
4	Category	Unsigned Integer	The Packet Category, calculated from the APID
5	P1val	Unsigned Integer	SCOS-2000 P1 Value (the meaning is mission-specific)
6	P2val	Unsigned Integer	SCOS-2000 P2 Value (the meaning is mission-specific)
7	Data Stream	Unsigned Integer	SCOS-2000 Data Stream
8	Generation Time	CCSDS CUC	Time-correlated On-Board Generation Time
9	Reception Time	CCSDS CUC	Time at which the TM packet was extracted from its Transfer Frame on-ground
10	Packet Length	Unsigned Integer	Length of the packet. Added for backward compatibility with the GDDS
11	SPID	Unsigned Integer	SCOS-2000 Packet ID
12	Ground Station	Unsigned Integer	Id of the ground station which downlinked this telemetry packet
13	Virtual Channel	Unsigned Integer	Virtual Channel number of the on-board data source.
14	Data Unit type	Unsigned Integer	Identifies the data unit type
15	Type	Unsigned Integer	PUS Service Type
16	Subtype	Unsigned Integer	PUS Service Subtype
17	SleServiceID	Unsigned	An 8-bit integer value that defines the SLE service used. Added for backward compatibility with the

Number	Field	Type	Description
		Integer	GDDS
18	Time Quality	Unsigned Integer	The value of the time quality flag 0 = GOOD, 1 = INACCURATE, 2 = BAD.
19	Quality Flag / Time Stamp Type	Unsigned Integer	Indicates the data quality of the TM packet 0 = Packet Reception (PR), 1 = Frame Transmission (FT), 2 = Packet Generation (PG)
20	Database Version	Unsigned Integer	Version of the SCOS-2000 database used to extract this packet from its transfer frame
21	Domain	Unsigned Integer	The SCOS-2000 domain in which this TM packet was generated

Table 6 - TM Packet Raw Header Fields

Note: while the data in some of these header fields may be contained in the raw TM packet data itself, these header fields are still defined by EDDS for the convenience of missions which wish to include this information also in the EDDS header for each TM packet.

For a specific telemetry packet data file, the EDDS file header determines which fields are included in that file, and the corresponding field lengths.

Note any string field will be translated in the packet as follow:

<string length in hexadecimal >+<string represented in hexadecimal >

In the case of the TM packet data type, the data field itself consists of the raw packet as generated by the spacecraft but preceded by an EDDS sequence counter and the packet length:

Field	Content	Size	Description
Count	EDDS Sequence Count	1 byte	Data element sequence count applied by EDDS. Client applications can use this to detect missing data. Note that the counter is reset to 0 once 255 is reached. If suspending and resuming a request, the counter is reset on resume. The counter continues when the file is split.
Length	Data Length	4 bytes	Size of the Data field in octets
Data	TM packet	Raw	Raw packet in network endian order

Table 7 - TM Packet Raw Data Element

6.1.2.2 TC

The EDDS defines the following fields, which may be included in the header for TC packet data (see section 6.1.1):

Number	Field	Type	Description
1	Type	Unsigned Integer	PUS Service Type
2	Source Type	Byte	Type of originating command source
3	Subtype	Unsigned Integer	PUS Service Subtype

Number	Field	Type	Description
4	APID	Unsigned Integer	Application Process Identifier
5	PID	Unsigned Integer	Packet ID, calculated from the APID
6	Category	Unsigned Integer	Packet Category, calculated from the APID
7	Virtual Channel	Unsigned Integer	The Virtual Channel ID
8	Verification Status	Unsigned Integer	Verification Status. This corresponds to the Command Verification Stages defined in field 13. (The meaning of the verification status is mission-specific.)
9	Uplink Time	CCSDS CUC	Time of command uplink (if any)
10	Execution Time	CCSDS CUC	Predicted/actual time of command execution
11	Last Update Time	CCSDS CUC	Time the command was last updated
12	Release Time	CCSDS CUC	Time the command was released
13	Sequence Name	String	Name of the top-level parent sequence to which this instance of the command belongs (if any)
14	Subschedule	Unsigned Integer	On-board Sub Schedule to which this command belongs
15	Subsystem	String	Associated Sub-system
16	Command Source	String	The name of the workstation used to release the telecommand
17	Command Acknowledgement	Unsigned Integer	Four-bit representation of the Service 1 report packets to be generated by the spacecraft
18	Command Verification Stages	String	The verification stages of the telecommand as supported by infrastructure (R, GTO or AC).
19	Data Stream	Unsigned Integer	SCOS-2000 Data Stream
20	Domain	Unsigned Integer	SCOS-2000 Domain
21	Database Version	String	Version of the SCOS-2000 database used to encode this telecommand
22	PUS Source Sequence Count	Unsigned Integer	PUS Source Sequence Counter
23	CUV	Unsigned Integer	Command UV Onboard verification stage
24	CAV VC0	Unsigned Integer	Command EV Acceptance verification stage
25	CEV VC0	Unsigned Integer	Command EV Execution start verification stage
26	CAV VC1	Unsigned Integer	Command EV Acceptance verification stage
27	CEV VC1	Unsigned Integer	Command EV Execution start verification stage

Number	Field	Type	Description
28	Packet Length	Unsigned Integer	Size of the packet
29	Command Name	String	The name of the command

Table 8 - Possible TC Packet Raw Header Fields

Note: while the data in some of these header fields may be contained in the raw TC packet data itself, these header fields are still defined by EDDS for the convenience of missions which wish to include this information also in the EDDS header for each TC packet.

For a specific telecommand packet data file, the EDDS file header determines which fields are included in that file, and the corresponding field widths.

Note any string field will be translated in the packet as follow:

<string length in hexadecimal >+<string represented in hexadecimal >

In the case of the TC packet data type, the data field itself consists of the raw telecommand packet sent to the spacecraft:

Field	Content	Size	Description
Count	EDDS Sequence Count	Unsigned Integer	Data element sequence count applied by EDDS. Client applications can use this to detect missing data.
Length	Data Length	Unsigned Integer	Size of the Data field in octets
Data	TC packet	Raw	Raw packet in network endian order

Table 9 - TC Packet raw Data Element

6.1.2.3 EV

EDDS defines the following fields, which can be included in the header for SCOS-2000 EV packet data (see section 6.1.1):

Number	Field	Type	Description
1	Event Type	String	Type of the event
2	Event Id	String	Event Id
3	Source	String	The source of the event
4	Category	String	Category of the event
5	Generation Time	CCSDS CUC	Generation time of the event
6	Domain	Unsigned Integer	SCOS-2000 domain

Table 10 - Possible EV Packet Raw Header Fields

For a specific event packet data file, the EDDS file header determines which fields are included in that file, and the corresponding field widths.

Note any string field will be translated in the packet as follow:

<string length in hexadecimal >+<string represented in hexadecimal >

In the case of the EV packet data type, the data field itself consists of the raw SCOS-2000 event packet:

Field	Content	Size	Description
Count	EDDS Sequence Count	Unsigned Integer	Data element sequence count applied by EDDS. Client applications can use this to detect missing data.
Length	Data Length	Unsigned Integer	Size of the Data field in octets
Data	EV packet	Raw	Raw SCOS-2000 event packet data

Table 11 - EV Packet Raw Data Element

6.1.3 GDDS Binary File Example

6.1.3.1 TM Example

This section contains an example which shows how a GDDS TM packet binary data file would be represented using the EDDS binary file structure. The structure is based on the information contained in the GDDS ICD [RD-13]. The example provided assumes two TM packets (data elements) are present in the EDDS TM packet binary file.

GDDS telemetry packet header records contain the following fields:

- SCET (8 octets)
- Packet length (4 octets)
- Ground station ID (2 octets)
- Virtual Channel ID (2 octets)
- SLE Service (1 octet)
- Time Quality (1 octet)

These correspond to the following header EDDS header fields:

- Field 8 (Generation Time)
- Field 10 (Packet Length)
- Field 12 (Ground Station ID.)
- Field 13 (Virtual Channel)
- Field 17 (SLE Service ID)
- Field 18 (Time Quality Flag)

The following table illustrates the sequence and contents of each of the sections of the EDDS TM Packet binary file (the values are given in Hexadecimal. Note that two hecadecimal characters represent one octet):

Packet Header Fields (packet 1)	Field_1 (Generation Time)	4E94 46BE A7B8 D000
	Field_2 (Packet Length)	0000 0050
	Field_3 (Ground Station ID.)	0000
	Field_4 (Virtual Channel)	0000
	Field_5 (SLE Service ID)	00

	Field_6 (Time Quality Flag)	00
Packet Data (packet 1)	TM packet	0000 0000 FFFF FFFF 0000 C800 0100 C8C8 0000 0012 5800 0000 2100 0000 1BB5 A0A6 42AF 0400 0000 0000 0000 0000 0000 0000 5000 0000 0000 0000 5900 0000 B0E9 9B08 E0B4 9208 FFFF FFFF 5550 4441 5445 2025
Packet Header Fields (packet 2)	Field_1 (Generation Time)	4E94 46BE 6334 3000
	Field_2 (Packet Length)	0000 0050
	Field_3 (Ground Station ID.)	0000
	Field_4 (Virtual Channel)	0000
	Field_5 (SLE Service ID)	00
	Field_6 (Time Quality Flag)	00
Packet Data (packet 2)	TM packet	0000 0000 FFFF FFFF 0000 C800 0200 C8C8 0000 0012 5800 0000 2100 0000 1B51 D154 B6AF 0400 0000 0000 0000 0000 0000 0000 5000 0000 0000 0000 5900 0000 0000 0000 4E94 5872 0006 BF8B 4E94 5872 0006 BF8B

Table 12 - Example TM Packet Binary Format

Note that the generation time is in two parts. The first part (octets 0-3) is the number of seconds since epoch (1st Jan 1970) and the second part (octets 4-7) is the number of microseconds.

6.1.3.2 TC Example

This section contains an example which shows how a GDDS TC packet binary data file would be represented using the EDDS binary file structure. The structure is based on the information contained in the GDDS ICD [RD-13]. The example provided assumes two TC packets (data elements) are present in the EDDS TC packet binary file.

GDDS telecommand packet header records contain the following fields:

- Uplink Time (8 octets)
- Execution Time (8 octets)
- Command Uplink Stage CUV (2 octets)
- Command Acceptance Stage Real-Time CAV (VC0) (2 octets)
- Command Execution Stage Real-Time CEV (VC0) (2 octets)
- Command Acceptance Stage Playback CAV (VC1) (2 octets)
- Command Execution Stage Playback CEV (VC1) (2 octets)
- Packet Length of the DDS telecommand packet body (4 octets)
- Command ID (8 octets)

- Parent Sequence ID (8 octets)

These correspond to the following header EDDS header fields:

- Field 9 (Uplink Time)
- Field 10 (Execution Time)
- Field 23 (CUV)
- Field 24 (CAV VC0)
- Field 25 (CEV VC0)
- Field 26 (CAV VC1)
- Field 27 (CEV VC1)
- Field 28 (Packet Length)
- Field 29 (Command Name)
- Field 13 (Sequence Name)

The following table illustrates the sequence and contents of each of the sections of the EDDS TC Packet binary file (the values are given in Hexadecimal. Note that two hecadecimal characters represent one octet):

Packet Header Fields (packet 1)	Field 6 (Uplink Time)	4FB6 5AD2 000E 9A94
	Field 7 (Execution Time)	4FB6 5BE4 0006 9398
	Field 22 (CUV)	0004
	Field 23 (CAV VC0)	1000
	Field 24 (CEV VC0)	0000
	Field 25 (CAV VC1)	1000
	Field 26 (CEV VC1)	0000
	Field 27 (Packet Length)	0000 000D
	Field 28 (Command Name)	5A41 4330 3330 3036
	Field 10 (Sequence Name)	0000 0000 0000 0000
Packet Data (packet 1)	TC packet	18BC C1F1 0006 1903 0600 0084 C0
Packet Header Fields (packet 2)	Field 6 (Uplink Time)	4FB6 5AD2 000E 9A95
	Field 7 (Execution Time)	4FB6 5AD2 000E 9A95
	Field 22 (CUV)	0004
	Field 23 (CAV VC0)	0020
	Field 24 (CEV VC0)	0000
	Field 25 (CAV VC1)	0020
	Field 26 (CEV VC1)	0000
	Field 27 (Packet Length)	0000 0021
	Field 28 (Command Name)	5A4D 5431 3130 3034
	Field 10 (Sequence Name)	0000 0000 0000 0000

Packet Data (packet 2)	TC packet	18FC C012 001A 190B 0400 0007 4FB6 5BE4 6E56 18BC C1F1 0006 1903 0600 0084 C0B5 39

Table 13 - Example TC Packet Binary Format

Note that the generation time is in two parts. The first part (octets 0-3) is the number of seconds since epoch (1st Jan 1970) and the second part (octets 4-7) is the number of microseconds.

6.2 XML

This section identifies the EDDS data types that support XML formatting and provides references to associated schemas (found inAppendix C)

6.2.1 Packet

- Statistics - The packet statistics data type supports XML formatting, see the PacketStatistics schema Appendix C
- TC / TM / EV PARC Raw – This data type supports XML formatting only. See the PktRawResponse schema Appendix C
- TC / TM / EV / OOL Reports – These data types supports XML formatting only. See the PktTcReportResponse, PktTmReportResponse, EventRecordReportResponse and OolDataReportResponse schemas in Appendix C respectively.

6.2.2 Parameter

- Param – The parameter data type supports XML formatting, see the Param schema Appendix C.
- Statistics - The parameter statistics data type supports XML formatting, see the ParameterStatistics schema Appendix C.
- Definition - The parameter definition data type supports XML formatting, see the ParameterDefinitionType schema Appendix C.
- Preview - The parameter preview data type supports XML formatting, see the ParameterPreview schema Appendix C
- SMON Parameter - The parameter data type supports XML formatting, see the SMONParamResponse schema Appendix C

6.2.2.1 A special note on times shown in the response files

There are a number of times shown in the output of the DARC Param response file:

- Generation time: for DARC sources, this corresponds to the parameter generation time. For SMON sources, this corresponds to the sample time (i.e. the packet generation time (filing time) plus an offset for supercommutated packets
- Storage time: for DARC sources, this corresponds to the DARC storage time. For SMON sources, this corresponds to the create time (i.e. the PARC storage time)
- Parent generation time: for DARC sources, this corresponds to the packet generation time. For SMON sources, this corresponds to the packet generation time (filing time)

6.2.3 Report

The EDDS provides access to EDDS generated reports.

6.2.3.1.1 EDDS Report

EDDS reports are generated by the EDDS on user request. All EDDS reports are in XML format and are packaged within an EDDS XML EddsReport type. The XML schema of the report type can be found in Appendix I. The reports that may be packaged within the EddsReport are as follows:

6.2.3.1.1.1 Request Summary Report

The XML schema AcknowledgementPart defines the structure and content of an EDDS XML request summary report type.

EDDS provides two views of Request Summary Reports: a brief and a full summary report. The full summary report displays the complete acknowledgement message, while the brief is a subset of info from the acknowledgement message.

The detailed structure of the AcknowledgementPart is defined in the Section A.1

6.2.3.1.1.2 EDDS Usage Report

The XML schema EddsUsageReport defines the structure and content of an EDDS XML usage report type. See I.2.

A usage report contains information about the usage of the EDDS services over a given period:

Field	***Type***	***Description***
Total Deleted Requests	Integer	Total number of deleted requests over the period
Total Processed Requests	Integer	Total number of processed requests over the period.
Total Failed Requests	Integer	Total number of failed requests over the period
The following is within the EddsUsageReportList element, within zero or more EddsUsageReportListElement elements.		
RequestId	String	The ID of the request
User	String	The username of the user who submitted the request
Role	String	The name of the role used for the request
Status	String	The final status of the request
SizeOfResponseFile	BigInteger	The size of the response file in bytes
SubmissionDate	Date	The date the request was submitted to the server
ExecutionDate	Date	The date the request was executed
CompletionData	Date	The date the request was completed

Table 14 - EDDS Usage Report

6.2.4 Archived Files

The EDDS provides access to catalogues and files of MCS file archives. The catalogue data type supports XML formatting. The corresponding XML schema, Catalogue, defines the structure and content of an EDDS XML catalogue data type. See C.1, C.5 and C.6.

Note: Files from an MCS archive are passed to the end user as retrieved from the MCS file archive. This implies that a file could of course be an XML document. The definition of the content of MCS files is mission-specific and is outside the scope of this ICD.

6.2.5 File System

The EDDS provides access to a designated directory on the file system of EDDS Server. The File and Folder Catalogue requests support XML formatting option.

6.2.6 Acknowledgement

An EDDS acknowledgement is an XML document. The structure and content is described in the XML schema acknowledgement.xsd. See A.1 for detailed information.

6.3 XML Transform

All the requests that support XML format also support XML transformations. Custom stylesheets can be added to the EDDS Server for transforming the XML responses. Available list of transformations can be retrieved through the EddsService.getTransformations() method in the Client API.

6.4 XFDU

The XFDU format is a schema that allows existing data to be packaged up with further information such as the original system used, the filename and a checksum. The XFDU schema is an external schema and can be found in the document “CCSDS, XML Formatted Data Unit (XFDU) Structure and Construction Rules” [AD-1]. EDDS supports a post processing option that, when selected, will create an XML file compliant with this schema.

6.5 TDRS Spreadsheet

The EDDS supports the formatting of data types (Parameters) into the TDRS spreadsheet format. The format is a text file containing tab delimited fields (columns) with new lines separating rows. See Appendix M for more information.

6.6 ASCII

The EDDS supports ASCII representation of data types. The contents of the ASCII files is similar to the equivalent XML files, but without the XML markup.

For the TC Report, the ASCII format is based on the Command History application found in SCOS 5.3/5.4. Full information on this format can be found in the SCOS 2000 Command History Operator User Manual, with the most useful information copied here for convenience.

The following snippet shows an example of this format:

```
Name     Description              S D C G B IL ST Source         FC TC Q R GTO A SS 1122 CC


S2KTC007 TC(3,3)                  E E E     SR MS de-vm140       00 01   S SSS S         S
  S2KCP013 Housekeeping SID         Eng      Dec                   HK_SID_1
```

To save space, the columns “Sequence, Domain, Release Time and Execution Time” have been removed from the output. The first line provides the column headings, and the following line is the values for a TC packet. The final indented line shows the parameters for the TC packet. These can be removed from the output by selecting “Get brief summary only” in the TC Report request.

The following table describes the column headings after “Description”:

Column	***Description***
S	Static PTV Check State (E => Enabled, D => Disabled or O => Overridden)
D	Dynamic PTV Check State (E => Enabled, D => Disabled or O => Overridden)
C	CEV Check State (E => Enabled or D => Disabled)
G	Group flag value (G => in a group, E => last in a group)
B	Block flag value (B => in a block, E => last in a block)

<table>
<tr><td>IL</td><td colspan="2">Interlock Status, two characters, the first reports the interlock type and scope, the second, the interlock stage type.</td></tr>
<tr><td></td><td colspan="2">Interlock Type and Scope</td></tr>
<tr><td></td><td>L</td><td>Wait-Fail Local</td></tr>
<tr><td></td><td>G</td><td>Wait-Fail Global</td></tr>
<tr><td></td><td>S</td><td>Wait-Fail Subsystem Local</td></tr>
<tr><td></td><td>F</td><td>Fail-Only Local</td></tr>
<tr><td></td><td>T</td><td>Fail-Only Subsystem Local</td></tr>
<tr><td></td><td colspan="2">Interlock Stage Type</td></tr>
<tr><td></td><td>R</td><td>1st UV stage: Ground Stn. Reception</td></tr>
<tr><td></td><td>U</td><td>2nd UV stage: Uplink</td></tr>
<tr><td></td><td>O</td><td>3rd UV stage: On Board Reception</td></tr>
<tr><td></td><td>A</td><td>Execution Ver.: Acceptance</td></tr>
<tr><td></td><td>C</td><td>Execution Ver.: Completion</td></tr>
<tr><td>ST</td><td colspan="2">Source Type (MS => Man.Stack, AS => Auto Stack)</td></tr>
<tr><td>Source</td><td colspan="2">Source workstation ID</td></tr>
<tr><td>FC</td><td colspan="2">Frame Counter: indicating current frame count. Only applicable for CLTU AD mode</td></tr>
<tr><td>TC</td><td colspan="2">Transmission Counter: indicating the current number of (re)transmission. Only applicable for CLTU AD mode.</td></tr>
<tr><td>Q</td><td colspan="2">This column will show an asterisk (*) should any of the verification stages be anything other than “Success (S)” or not applicable (blank).</td></tr>
<tr><td colspan="3">Verification Status Headings</td></tr>
<tr><td></td><td colspan="2">The status of each individual verification stage may have one of the following values:
F Fail
U Unverified
X Unknown
P Pending
T Time-out
S Success
I Idle
A Assumed Passed
C Affected
V Uncertain Successful
N Uncertain Failed
E Superseded
Blank N/A
The following behaviour of a time-tagged commands verification status are extended:
• If a time-tagged command is deleted prior to execution, a string “DELETED” is showed in verification stage field
• If a time-tagged command is disabled at execution, EV stages are marked as “D”</td></tr>
</table>

R	Release from SCOS-2000 to the SLE User (NCTRS/NIS). Note that a failure for this stage may be caused by one of the following reasons: • PTV NOT OK at release time (the PTV state is visible in the CQD • PTV/CEV pane, see Section 3.1.3.3 below) • PTV OK (e.g. dynamic PTV was overridden) but no TCP/IP connection between the releaser and the NCTRS was available at release time • The command was successfully released but eventually rejected by the NCTRS e.g. because the required connection to the station equipment was not available at release time. In this case an administrative message is also received from the NCTRS and logged in the HFA
G	Ground station reception (UV 1st stage)
T	Uplink a.k.a. radiation (UV 2nd stage) and in case of Throw Event Command it corresponds to the Execution stage.
O	On-board reception a.k.a. transfer (3rd UV stage, only applicable in AD mode)
A	On-board application acceptance
SS	Execution start – two instances of the stage are shown, one corresponding to updates from Real Time and the other corresponding to updates from Playback TM
1122	Execution step „n" – two instances of each stage are shown, one corresponding to updates from Real Time and the other corresponding to updates from Playback TM
CC	Execution completion – two instances of the stage are shown, one corresponding to updates from Real Time and the other corresponding to updates from Playback TM

Table 15 – TC Report ASCII column output

6.7 Binary

Binary format is specific to each of the request types.

6.7.1 Archive File

The file is stored as received from the mission File Archive, packaged in a TAR file.

6.7.2 File System

The files are stored as received from the file system, packaged in a TAR file.

6.7.3 Google Proto Buffers

Google Protocol Buffers binary format is used to deliver data in an efficient format. Compared to XML format it is more compact and faster to process in case of large data volumes. The proto files describing the data structure are stored in edds-ws-common and are compiled as a part of the build process. Data is encoded as a framed byte array. Helper classes are provided in project edds-ws-client in package esa.egos.edds.ws.client.binary to simplify working with binary format.

6.7.3.1 Header line

Each EDDS binary file has ASCII encoded header line included to provide simple interaction by external applications. Format of header line is simple key value pairs separated by semicolons:

```
# Key1: Value1; ... ; KeyN: ValueN \n
```

- Header line starts with # (byte value 23 in HEX) and space (byte value 20 in HEX)
- Each key value pair ends with semicolon (byte value 3B in HEX) and optional space (byte value 20 in HEX)

- Key is separated from value with colon (byte value 3A in HEX) and optional space (byte value 20 in HEX)
- Header line ends with single line feed character (byte value A0 in HEX)
- Currently supported keys are:
    - Type – data type in binary format (current supported values are Parameter and SMON Parameter)
    - Version – version of binary format

6.7.3.2 Data

Data format is based on Google Protocol Buffers (GPB, see [RD-16]). The file contains the list of serialized GPB DarcParameterBinary objects, each of them prefixed with its size in bytes, the value is in GPB varint format.

Note that Google Protocol Buffer encoded files are not indended to be decoded manually. Instead, the original Proto file describing the format in plain text should be compiled and the Google Protocol Buffers library should be used instead. The Proto file describing this format is provided in the EDDS source code within the edds-ws-common package. For more information on Google Protocol Buffers, see [RD-16].

6.7.3.3 Specific formats

6.7.3.3.1 Parameter TM

Protobuf format is used to deliver Parameter TM data efficiently. Binary is one of the available formats Parameter TM request submitter can choose from.

6.7.3.3.1.1 Data

Following is the GPB DarcParameterBinary message definition:

```
import "ParamValidity.proto";


message DarcParameterBinary {


      enum Type {
            // Special type indicating that the proto type definition
should be updated to include a new type
            UNKNOWN = 0;
            BIT = 1;
            UTINYINT = 2;
            STINYINT = 3;
            USMALLINT = 4;
            SSMALLINT = 5;
            UMEDIUMINT = 6;
            SMEDIUMINT = 7;
            SINT = 8;
            UINT = 9;
            FLOAT = 10;
            DOUBLE = 11;
```



```
            STRING = 12;
            DATETIME = 13;
            JOB = 14;
            LOG = 15;
      }

      required string param_name = 1;
      required int64 gen_time = 2;
      required int64 storage_time = 3;
      required string value = 4;
      required ParamValidity validity = 5;
      required Type type = 6;

      // Following are not present in DARC v2.2.2 and earlier
      optional int64    parentGenTime = 7; // SCOS TM packet generation time
      optional int64    parentId = 8; // SPID in PARC
      optional Type     rawValType = 9;
      optional bool     boolRawVal = 10;
      optional string   strRawVal = 11;
      optional double   dblRawVal = 12;
      optional float    fltRawVal = 13;
      optional sint64   longRawVal = 14;
}
```

The above message imports the ParamValidity definition which is specified as follows:

```
package esa.egos.edds.model;

option java_outer_classname = "ParamValidityProto";
enum ParamValidity {
      VALID = 0;
      INVALID = 1;
      UNKNOWN = 2;
      EXPIRED = 3;
}
```

6.7.3.3.1.2 Decoding Files

Note that it is not necessary to manually decode these files! EDDS provides an API in the edds-ws-client to decode them. It is also possible to decode the files using Google Protocol Buffers libraries for many other languages including Python, Perl and C++.

First line	**# Type: Parameter; Version: 1;**	**Header line for the binary file format**
Rest of file	Binary data	Framed Google Protocol Buffers encoded data**.** The encoded data is prefixed with the length of

		the message so that multiple messages can be included in one file. This functionality is provided by both the Apache ActiveMQ and the Google Protobuf implementations. Example code on how to read the data with either implementation are provided in the esa.egos.edds.ws.client.examples test package of edds-ws-client.

6.7.3.3.2 Parameter TM Definitions

Protobuf format is used to deliver Parameter TM definitions data efficiently. Binary is one of the available formats Parameter TM Definitions request submitter can choose from.

6.7.3.3.2.1 Data

Following is the GPB `ParamDefinitions` message definition:

```
message ParamDefinitionRecord
{
    optional string name = 1;
    optional string description = 2;
    optional string unit = 3;
    optional string type = 4;
    optional bool active = 5;
}

message ParamDefinitions
{
    repeated ParamDefinitionRecord entry = 1;
}
```

6.7.3.3.3 SMON Parameter

The Protobuf format is used to deliver Parameter TM data from SMON efficiently. Binary is one of the available formats that the SMON Parameter request can choose from. It is also possible to choose the DARC Binary format, so that the output from the SMON Parameter request is compatible with programs that can read DARC Binary data files. See Section 6.7.3.3.1 for details of that format.

6.7.3.3.3.1 Data

Following is the GPB SMONParameter message definition:

```
message SMONParameterBinary
{
      required string name = 1;
      required string calib_id = 2;
      repeated PacketInstanceBinary packet_instances = 3;
      repeated RepresentationSampleBinary representation_samples = 4;

    enum ValidityValueBinary {
```



```
      UNKNOWN = 0;
      VALID = 1;
      INVALID = 3;
      EXPIRED = 4;
    }

    message PacketInstanceBinary {
      required string reception_time = 1;
      required string generation_time = 2;
      required int32 sample_offset = 3;
      required int32 sequence_counter = 4;
      required int32 spacecraft_id = 5;
      required SMONParameterBinary.ValidityValueBinary validity = 6;
      required int32 data_stream = 7;
      required int32 domain = 8;
      required int32 packet_size = 9;
      required string ground_station = 10;
      required string packet_mnemonic = 11;
    }

    message RepresentationSampleBinary {
      required string representation_name = 1;
      required string sample_time = 2;
      required string value = 3;

        message Validity {

            required SMONParameterBinary.ValidityValueBinary validity = 1;
            required string validity_details = 2;

        }
      required Validity validity = 4;

        message BehaviourState {
            required string status_type = 1;

            enum GeneralBehaviourStateBinary {
                INCONSISTENT_STATUS = 0;
                NOMINAL = 1;
                OUT_OF_LIMIT = 2;
```



```
            UNCHECKED = 3;
            UNDEFINED = 4;
            VIOLATED = 5;
            WARNING = 6;
        }
        required GeneralBehaviourStateBinary ool_state = 2;

        required int32 nr_of_violations = 3;
        required string value = 4;

        message DetailedBehaviourStateBinary {
            optional LimitBehaviourStateBinary limitState = 1;
            optional StatusBehaviourStateBinary statusState = 2;
            optional SCCBehaviourStateBinary sccState = 3;
            optional DeltaBehaviourStateBinary deltaState = 4;

            enum LimitBehaviourStateBinary {
                WITHIN_RANGE = 0;
                LOW = 1;
                HIGH = 2;
                LOW_LOW = 3;
                HIGH_HIGH = 4;
            }

            enum StatusBehaviourStateBinary {
                ALLOWED = 0;
                UNEXPECTED = 1;
            }

            enum SCCBehaviourStateBinary {
                SCC_ENABLE = 0;
                SCC_UNINIT = 1;
                SCC_DISABLE = 2;
                SCC_OFF = 3;
                SCC_FAIL = 4;
            }

            enum DeltaBehaviourStateBinary {
                DELTA_ALLOWED = 0;
                DELTA_EQUAL = 1;
```



```
                    MAX_DELTA_LOW = 2;
                    MAX_DELTA_HIGH = 3;
                    MIN_DELTA_LOW = 4;
                    MIN_DELTA_HIGH = 5;
                }
            }
            required DetailedBehaviourStateBinary detailed_state = 5;
        }
      repeated BehaviourState behaviour_states = 5;
    }

}
```

6.7.3.3.4 Event Record Report

The Protobuf format is used to deliver packet TM data from PARC efficiently. Binary is one of the available formats that the Event Record Report request can choose from.

6.7.3.3.4.1 Data

Following is the GPB `EventRecordReport` message definition:

```
message EventRecordReportBinary
{
    optional string id = 1;
    optional string domain = 2;
    optional int64 generation_time = 3;
    optional string application = 4;
    optional string severity = 5;
    optional string type = 6;
    optional string workstation = 7;
    optional string message = 8;
}
```

6.7.3.3.5 Packet Raw

The Protobuf format is used to deliver raw packet data from PARC efficiently. Binary is one of the available formats that the Packet TC Raw, Packet TM Raw and Packet EV Raw requests can choose from.

6.7.3.3.5.1 Data

Following is the GPB `PacketRaw` message definition:

```
message PacketRawBinary
{
    optional string packet = 1;
    optional int64 packet_id = 2;
}
```

6.7.3.3.6 Packet TC Report

The Protobuf format is used to deliver packet TC report data from PARC efficiently. Binary is one of the available formats that the Packet TC Report request can choose from.

6.7.3.3.6.1 Data

Following is the GPB `PacketTCReport` message definition:

```
message PacketTCBinary
{
    optional string command_name = 1;
    optional string domain = 2;
    optional string source = 3;
    optional string description = 4;
    optional string sequence_name = 5;
    optional string release_time = 6;
    optional string uplink_time = 7;
    optional string execution_time = 8;
    optional string sub_system_id = 9;
    optional int64 sub_schedule_id = 10;
    optional int32 p_uSService_type = 11;
    optional int32 p_uSSub_service_type = 12;
    optional int32 apid = 13;
    optional int32 pid = 14;
    optional int32 vcid = 15;
    optional int32 category = 16;
    optional int32 ground_station_id = 17;
    optional int32 ssc = 18;
    optional string db_version = 19;
    optional string uplink_mode = 20;
    optional string static_ptv_check_state = 21;
    optional string dynamic_ptv_check_state = 22;
    optional string cev_check_state = 23;
    optional bool is_in_group = 24;
    optional bool is_group_end = 25;
    optional bool is_in_block = 26;
    optional bool is_block_end = 27;
    optional string interlock_type = 28;
    optional string interlock_stage_type = 29;
    optional string src_type = 30;
    optional string frame_count = 31;
    optional string release_state = 32;
    optional string ground_state = 33;
    optional string uplink_state = 34;
    optional string on_board_state = 35;
    optional string on_board_acc_state = 36;
    optional string exec_start_rt_state = 37;
    optional string exec_start_pb_state = 38;
    optional string exec_step1_rt_state = 39;
    optional string exec_step1_pb_state = 40;
    optional string exec_step2_rt_state = 41;
    optional string exec_step2_pb_state = 42;
    optional string exec_comp_state = 43;
    optional string exec_comp_pb_state = 44;
    optional string tt_obq_status = 45;
    optional string tt_del_status = 49;
    optional string tt_load_status = 50;
    repeated PktTcReportParameterBinary pkt_tc_report_parameters = 46;
    optional int32 trans_count = 47;
    repeated MissionSpecificField mission_specific_fields = 48;
    optional bytes body = 51;

    message PktTcReportParameterBinary {
        optional string parameter_name = 1;
        optional string parameter_description = 2;
        optional string parameter_unit = 3;
```



```
        optional string param_type = 4;
        optional string param_rep = 5;
        optional bool fix_edit = 6;
        optional bool has_changed = 7;
        optional string radix = 8;
        optional bool is_editable = 9;
        optional bool manually_edited = 10;
        optional string param_value = 11;
    }

    message MissionSpecificField {
        optional string name = 1;
        optional string type = 2;
        optional string value = 3;
    }
}
```

6.7.3.3.7 Packet TM Report

The Protobuf format is used to deliver packet TM report data from PARC efficiently. Binary is one of the available formats that the Packet TM Report request can choose from.

6.7.3.3.7.1 Data

Following is the GPB `PacketTMReport` message definition:

```
message PacketTMBinary
{
    optional int32 apid = 1;
    optional int32 pid = 2;
    optional int32 category = 3;
    optional string database_version = 4;
    optional string generation_time = 5;
    optional string data_stream = 6;
    optional string partition_name = 25;
    optional string domain = 7;
    optional string ground_station = 8;
    optional string quality_flag = 9;
    optional string reception_time = 10;
    optional string sle_service_id = 11;
    optional int32 source_sequence_counter = 12;
    optional int64 spid = 13;
    optional int32 type = 14;
    optional int32 sub_type = 15;
    optional string time_quality = 16;
    optional string virtual_channel = 17;
    optional string data_unit_type = 18;
    optional int32 pi1 = 19;
    optional int32 pi2 = 20;
    optional int32 packet_length = 21;
    optional string mnemonic = 22;
    optional string description = 23;
    optional sint64 id = 24;
    optional int32 spacecraftId = 26;
    optional bytes body = 28;

    repeated PktTmReportParameterBinary pkt_tm_report_parameters = 27;

    message PktTmReportParameterBinary {
        optional string parameter_name = 1;
        optional string parameter_description = 2;
        optional string parameter_units = 3;
```



```
        optional string parameter_radix = 4;
        optional int64 filing_time = 5;
        optional int32 bit_offset = 6;
        optional int32 byte_offset = 7;
        optional string parameter_value = 8;
        optional int64 length = 9;
    }

    enum TimeQuality {
      GOOD = 0;
      INACCURATE = 1;
      BAD = 2;
      NA = 255;
    }
}
```

6.7.3.3.8 Packet TM Gap Report

The Protobuf format is used to deliver packet TM gap report data from PARC efficiently. Binary is one of the available formats that the Packet TM Gap Report request can choose from.

6.7.3.3.8.1 Data

Following is the GPB `PacketTMGapBinary` message definition:

```
message PacketTMGapBinary
{
    optional int64 start_gap_time = 1;
    optional int64 end_gap_time = 2;
    optional int32 data_partition = 3;
    optional int32 apid = 4;
    optional int32 start_ssc_pkt = 5;
    optional int32 end_ssc_pkt = 6;
    optional int32 num_pkts_missing = 7;
}
```

6.7.3.3.9 Out of Limit Record Report

The Protobuf format is used to deliver packet OOL report data from PARC efficiently. Binary is one of the available formats that the OOL Report request can choose from.

6.7.3.3.9.1 Data

Following is the GPB `OOLReport` message definition:

```
package esa.egos.edds.model;

enum OOLRecordType {
    NO_VALUE = 0;
    LIMIT = 1;
    STATE = 2;
    STATUS_CONSISTENCY = 3;
    SOFT_HARD_LIMIT = 4;
    LIMIT_MO = 5;
    DELTA = 6;
}

message OOLReportBinary {
    message OOLReportRecord {
        enum OOLParameterState {
            NOMINAL = 0;
            WARNING = 1;
            OUT_OF_LIMITS = 2;
```



```
            SCC = 3;
            SCC_DISABLE = 4;
            SCC_OFF = 5;
            SCC_UNINIT = 6;
            VIOLATION = 7;
            NO_VALUE = 8;
        }

        enum OOLBehaviourLimitState {
            LIMIT_HIGH = 0;
                       LIMIT_HIGH_HIGH = 1;
                       LIMIT_LOW = 2;
                       LIMIT_LOW_LOW = 3;
                       LIMIT_WITHIN_RANGE = 4;
        }

        enum OOLBehaviourSccState {
            B_SCC_ENABLE = 0;
            B_SCC_DISABLE = 1;
            B_SCC_FAIL = 2;
            B_SCC_OFF = 3;
            B_SCC_UNINIT = 4;
        }

        enum OOLBehaviourDeltaState {
            DELTA_ALLOWED = 0;
            DELTA_EQUAL = 1;
            MAX_DELTA_HIGH = 2;
            MAX_DELTA_LOW = 3;
            MIN_DELTA_HIGH = 4;
            MIN_DELTA_LOW = 5;
        }

        enum OOLBehaviourStatusState {
            STATUS_ALLOWED = 0;
            STATUS_UNEXPECTED = 1;
        }

        optional OOLRecordType record_type = 1;
        optional int64 generation_time = 2;
        optional int64 parameter_changed_time = 3;
        optional string parameter_name = 4;
        optional OOLParameterState parameter_state = 5;
        optional string ool_state = 6;
        optional string value = 7;
        optional string lower_limit = 8;
        optional string upper_limit = 9;
        optional string allowed_values = 10;
        optional string scc_limit = 11;
        optional string reference_value = 12;
        optional string delta_value = 13;
        optional string min_value = 14;
        optional string last_value = 15;
        optional int32 violation_number = 16;
        optional int32 violation_number_hard = 17;
        optional int32 violation_number_soft = 18;
        optional string description = 19;
        optional string unit = 20;
        optional OOLBehaviourLimitState behaviour_limit_state = 21;
        optional OOLBehaviourSccState behaviour_scc_state = 22;
        optional OOLBehaviourDeltaState behaviour_delta_state = 23;
```



```
        optional OOLBehaviourStatusState behaviour_status_state = 24;
        optional string violation_description = 25;
    }

    repeated OOLReportRecord records = 1;
}
```

6.8 RawSourceBinary

This format is the simplest format we currently have in EDDS and can only be requested for TM Report request type, both parc and dataprovision service. In practical terms, for TM packets, this is nothing more than a continuous data message of CCSDS TM source packets (without any additional header).

7. The output is a binary file of the continuous raw data as provided by RawBodyData attribute.

7. Naming Conventions

7.1 Request ID Convention

The Request ID is formed using the information in the following table:

Field	*Type*	*Possible Values*
RequestType	String	BatchRequest, StreamRequest, Cancel, DeleteData, DeleteRequest, AccountRequest
Subtype	String	Taken from DataAccessDataElement (See Section D.23)
Mission	String	Any alphanumeric value
Domain	String	Typically numbers from 0 – 51
Username	String	The user ID of the user performing the request
Time	String	Time of the creation of the request in DOY form in GMT

Table 16 – Request ID Convention

The Request ID can either be on of the following two formats:

```
[RequestType].[Username].[yyyy.DDD.hh.mm.ss.SSS]
```

Or

```
[RequestType].[Subtype].[Mission].[Domain].[yyyy.DDD.hh.mm.ss.SSS]
```

The first form is used for changes to user and mission LDAP account information (AccountRequest types) and the other form is used for all other request types. The Request ID is created by the EDDS Archiver (by the ScheduleUtility class) when the request is added to the database.

7.2 Response ID Convention

The Response ID is formed using the information in the following table:

Field	*Type*	*Possible Values*
RequestId	RequestId	Same as Section 7
Time	String	Time the response file was created in DOY form in GMT
Counter	Int	The response file number, starting at 1.
Extension	String	The file type

Table 17 – Response ID Convention

The Response ID is in the following format:

```
[RequestType].[Subtype].[Mission].[Domain].[yyyy.DDD.hh.mm.ss.SSS]@[yyyy.DDD
.hh.mm.ss.SSS].[Counter].[Extension]
```


8. Delivery

8.1 Delivery Mechanism

The EDDS supports the following delivery mechanisms for the delivery of data to the end user.

The batch request response data can be delivered to the user via File Server or EDDS Server each file separately as soon as EDDS Server is finished processing a file and it is available for Delivery Manager.

8.1.1 File Server

Data types that support the 'File Server' delivery mechanism are delivered over SFTP or FTP using the EDDS web server. Direct delivery requires the end user to be running a server that supports SFTP/FTP.

In order to user SFTP the user has to provide username and password of the remote file server. In case these data is not provided, the web server will automatically try to use FTP.

8.1.2 EDDS Server

Data types that support the EDDS Server delivery mechanism are stored on an EDDS server. Also when the File Server delivery fails, the data is still stored in EDDS file server.

8.1.3 Stream

Data types that support Stream delivery require the end user to run a stream client application that can be downloaded from the EDDS web site and installed on the user's machine. Stream delivery supports online and offline data where, in the context of the EDDS, offline is taken to refer to the retrieval of previously archived data and online is taken to refer to the routing of data as it is received on the relevant control system. In addition, online data delivery supports two modes of operation: timely and complete. Timely mode ensures data is delivered within a specified time and will drop data if this criterion cannot be met. Complete mode ensures all data is delivered, although there may be considerable delay in the arrival of the complete data set.

It is possible to use EDDS stream client to save streaming data in files, read chapter 4.3.4 for more information.

9. Web Services Description

The EDDS uses web services as the fundamental mechanism to provide the functionality behind EDDS batch services and EDDS stream services. This section of the EUICD details the structure of the EDDS Web Services.

9.1 Web Service Introduction

The EDDS provides access to data via Web Services that are defined using Web Services Description Language (WSDL) see [RD-3]. The Web Services described in this section of the document allows client applications access to batch and user management services.

The services provide operations that allow a client to be authenticated, make requests for batch data, make requests for stream data, perform user management and receive data in response to those requests.

A WSDL service description is an XML document that contains a set of definitions for:

- Port Types
- Messages
- Bindings
- Services

A port type is a named set of abstract operations and the abstract messages involved. Message definitions are always considered to be an abstract definition of the message content. A message binding describes how the abstract content is mapped into a concrete format. However, in the case of the EDDS the content is intended to map directly onto the XML schemas. The bindings provide the protocol details for operations and messages defined by a particular port type. In the case of the EDDS, the bindings are to a SOAP protocol using XML messages. Finally, a service definition groups a set of related ports together. The definition of the EDDS Web Services is defined in the XML documents,

Port operations defined by EDDS provide a means to indicate if an error has occurred during the request of an operation.) An exception is raised when a service request cannot be executed with success.

The WSDL file can be viewed in a web browser by simply entering the end point URL normally passed to the client application into the web browser's address bar.

9.2 PortTypes

A port type is a named set of abstract operations and the abstract messages involved. The port type name attribute provides a unique name among all port types defined within in the enclosing service definition document. WSDL has four transmission primitives that an endpoint can support:

- One-way - The endpoint receives a message.
- Request-response - The endpoint receives a message, and sends a correlated message.
- Solicit-response - The endpoint sends a message, and receives a correlated message.
- Notification - The endpoint sends a message.

WSDL refers to these primitives as operations.

The operation instances defined in the EDDS port types are based on 'one-way' and 'request-response' transmission primitives. The following sections describe the port types that have been defined for the EDDS.

9.2.1 EDDS Port Type

The EDDS port type provides the operations that allow the authentication of a user and subsequent allocation of a session identifier. All users of the EDDS are assigned a session and it is under this

session that a series of operations are made. A session will be timed out by the EDDS server if inactive, however it is strongly recommended that applications should actively close the session when appropriate. This ensures that server resources are freed as soon as possible to service new session requests. Not following this guideline will result in the potential delaying (or rejection) of session requests. The timeout period of sessions can be changed by the EDDS Administrator in the EDDS Web Server configuration in the file "web.xml". The authentication port type also provides operations for the client to query the roles and missions available within the session.

The EDDS port type is a component that is used by both batch and stream services.

It provides the set of operations available for the request and receipt of data (and acknowledgements) within the scope of an EDDS batch service.

It provides the set of operations available for the execution of user management requests.

9.2.1.1 EddsPortType

The EDDS port type describes the named set of messages that a service sends and/or receives

operations	**batchRequest** *The batch request operation* input **ws:batchRequest** output **ws:batchRequestResponse** fault **ws:requestFault** *An error occurs during the elaboration of the request* fault **ws:authorizationFault** *If the user has no priviledges to perform the request* fault **ws:sessionFault** *If the user is not logged in* **streamRequest** *The stream request operation* input **ws:streamRequest** output **ws:streamRequestResponse** fault **ws:requestFault** *An error occurs during the elaboration of the request* fault **ws:authorizationFault** *If the user has no priviledges to perform the request* fault **ws:sessionFault** *If the user is not logged in*

getStatus

The get status operation

input **ws:getStatus**

output **ws:getStatusResponse**

fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

reSubmit

The re-submit operation

input **ws:reSubmit**

output **ws:reSubmitResponse**

fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

getStatuses

Gets the status of a list of jobs

input **ws:getStatuses**

output **ws:getStatusesResponse**

fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

getResponse

The get response operation

input **ws:getResponse**
output **ws:getResponseResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

getRequest

The get request operation

input **ws:getRequest**
output **ws:getRequestResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

getStreamRequest

The get stream request operation

input **ws:getStreamRequest**
output **ws:getStreamRequestResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

startStreamData

Starts the incoming stream of the specified active stream request

input **ws:startStreamData**
output **ws:startStreamDataResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no privileges to perform the request

fault **ws:sessionFault**

If the user is not logged in

stopStreamData

Stops the incoming stream of the specified active stream request

input **ws:stopStreamData**
output **ws:stopStreamDataResponse**
fault **ws:sessionFault**

If the user is not logged in

getParamDefinitions

Gets the parameter definitions for the specified mission

input **ws:getParamDefinitions**
output **ws:getParamDefinitionsResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no privileges to perform the request

fault **ws:sessionFault**

If the user is not logged in

getCatalogue

Gets the catalogue for the specified data source

input **ws:getCatalogue**
output **ws:getCatalogueResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no privileges to perform the request

fault **ws:sessionFault**

If the user is not logged in

getFarcCatalogue

Gets the FARC catalogue for the specified mission and domain
@deprecated use getCatalogue instead

input **ws:getFarcCatalogue**
output **ws:getFarcCatalogueResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no privileges to perform the request

fault **ws:sessionFault**

If the user is not logged in

getDataspaces

Gets the dataspaces for the specified mission and request type

input **ws:getDataspaces**
output **ws:getDataspacesResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no privileges to perform the request

fault **ws:sessionFault**

If the user is not logged in

getAlias

Gets aliases for the specified mission

input **ws:getAlias**
output **ws:getAliasResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

getLastConsolidation

Gets the last consolidation time for the specified mission, request type and dataspace name if given.

input **ws:getLastConsolidation**
output **ws:getLastConsolidationResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

getAllowedRequestTypes

Gets the request types that the currently logged-in user has permission to issue.

input **ws:getAllowedRequestTypes**
output **ws:getAllowedRequestTypesResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

cancel

The cancel operation

input **ws:cancel**
output **ws:cancelResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

suspend

The suspend operation

input **ws:suspend**
output **ws:suspendResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

resume

The resume operation

input **ws:resume**
output **ws:resumeResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

deleteData

The operation to delete response data
input **ws:deleteData**
output **ws:deleteDataResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

deleteRequest

The operation to delete a request
input **ws:deleteRequest**
output **ws:deleteRequestResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

getJobs

The get jobs operation

input **ws:getJobs**

output **ws:getJobsResponse**

fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

getJobsWithStatusAndLimit

input **ws:getJobsWithStatusAndLimit**

output **ws:getJobsWithStatusAndLimitResponse**

fault **ws:authorizationFault**

If the user has no privileges to perform the request

fault **ws:requestFault**

An error occurs in the elaboration of the request

fault **ws:sessionFault**

If the user is not logged in

getJobsWithStatus

Gets the jobs for the specified start and end date. The message contains a java.util.ArrayList of type RequestInfoRec encoded in Base64

input **ws:getJobs**

output **ws:getJobsWithStatusResponse**

fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

getHistoricalLog

Gets the historical system logs

input **ws:getHistoricalLog**
output **ws:getHistoricalLogResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

stopHistoricalLogRetrieval

Stops the retrieval of historical log messages

input **ws:stopHistoricalLogRetrieval**
output **ws:stopHistoricalLogRetrieval Response**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

getTransformations

Gets the available XSL Transformations for the given mission and request type

input **ws:getTransformations**
output **ws:getTransformationsResponse**
fault **ws:requestFault**

An error occurs during the elaboration of the request

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:sessionFault**

If the user is not logged in

logIn

Authenticate the user and start the session

input **ws:logIn**

The credentials of the user: username and password

output **ws:logInResponse**

No response given

fault **ws:authenticationFault**

If the username or password are incorrect or the LDAP server could not be reached

fault **ws:passwordExpiredFault**

If the user's password has expired and needs to be changed

hasSessionExpired

Checks if the user's session has expired, and return true if it has. This operation has a one-way transport.

input **ws:checkSession**

Input is not used. JAX-WS doesn't support notification style messages.

output **ws:checkSessionResponse**

Returns true if the session has expired or has not been started

logOut

Close the session for the user. This operation has a one-way transport

input **ws:logOut**

checkUserPassword

Check if the password supplied matches the user's current password. Used to double check the user's current password before allowing the user to change their password.

input **ws:checkUserPassword**

The user's current password in plain text

output **ws:checkUserPasswordResponse**

No output given

fault **ws:authenticationFault**

If the password doesn't match or the LDAP server could not be reached

fault **ws:sessionFault**

If the user is not logged in

accountRequest

The account request operation

input **ws:accountRequest**

The actual request

output **ws:accountRequestResponse**

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:requestFault**

An error occurs in the elaboration of the request

fault **ws:sessionFault**

If the user is not logged in

getUsers

Retrives the list of users

input **ws:getUsers**

Flag to indicate whether to fetch all the users

output **ws:getUsersResponse**

The list of users

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:requestFault**

An error occurs in the elaboration of the request

fault **ws:sessionFault**

If the user is not logged in

getMissions

the list of missions

input **ws:getMissions**

Empty message

output **ws:getMissionsResponse**

The list of missions

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:requestFault**

An error occurs in the elaboration of the request

fault **ws:sessionFault**

If the user is not logged in

getMissionsAndDomains

the list of missions with their domains

input **ws:getMissionsAndDomains**

Empty message

output **ws:getMissionsAndDomainsResponse**

The list of missions

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:requestFault**

An error occurs in the elaboration of the request

fault **ws:sessionFault**

If the user is not logged in

getRoles

Returns the roles the user has permission to use for the specified mission

input **ws:getRoles**

The name of the mission

output **ws:getRolesResponse**

The role details

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:requestFault**

An error occurs in the elaboration of the request

fault **ws:sessionFault**

If the user is not logged in

getUserAccountDetails

Returns the details for the specified user

input **ws:getUserAccountDetails**

The name of the user

output **ws:getUserAccountDetailsResponse**

The user's details

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:requestFault**

An error occurs in the elaboration of the request

fault **ws:sessionFault**

If the user is not logged in

getMissionDetails

Returns the details for the specified mission

input **ws:getMissionDetails**

The name of the mission

output **ws:getMissionDetailsResponse**

The mission details

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:requestFault**

An error occurs in the elaboration of the request

fault **ws:sessionFault**

If the user is not logged in

getUserQuota

Returns the quota usage for the specified mission and logged on user

input **ws:getUserQuota**

The name of the mission

output **ws:getUserQuotaResponse**

The quota details for the mission and logged on user

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:requestFault**

An error occurs in the elaboration of the request

fault **ws:sessionFault**

If the user is not logged in

getQuotaDetails

Returns the quota usage for the specified mission and logged on user.

input **ws:getQuotaDetails**

The name of the mission

output **ws:getQuotaDetailsResponse**

The quota details for the mission and user

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:requestFault**

An error occurs in the elaboration of the request

fault **ws:sessionFault**

If the user is not logged in

hasPermissionToSeeAllUsers

Checks if user has privileges to see all users data.

input **ws:hasPermissionToSeeAllUsers**

Username

output **ws:permissionToSeeAllUsersResponse**

Returns true if the user has permission to see other users data

fault **ws:authorizationFault**

If the user has no priviledges to perform the request

fault **ws:requestFault**

An error occurs in the elaboration of the request

fault **ws:sessionFault**

	If the user is not logged in
used by	binding **EddsBinding**

9.3 Messages

This section describes the XML messages that EDDS sends and receive in order to provide services.

9.3.1 message accountRequest

The message contains the details of a account management request

parts	**accountRequest** element **model:AccountRequestMessagePart**
used by	Operation **accountRequest** in PortType **EddsPortType**
source	<wsdl:message name="accountRequest"> <wsdl:documentation>The message contains the details of a account management request</wsdl:documentation> <wsdl:part name="accountRequest" element="model:AccountRequestMessagePart"/> </wsdl:message>

9.3.2 message accountRequestResponse

Empty message

parts	
used by	Operation **accountRequest** in PortType **EddsPortType**
source	<wsdl:message name="accountRequestResponse"> <wsdl:documentation>Empty message</wsdl:documentation> </wsdl:message>

9.3.3 message authenticationFault

The message contains details related to a authentication fault

parts	**authenticationFault** element **model:AuthenticationFault**
used by	Operation **logIn** in PortType **EddsPortType** Operation **checkUserPassword** in PortType **EddsPortType**
source	<wsdl:message name="authenticationFault"> <wsdl:documentation>The message contains details related to a authentication fault</wsdl:documentation> <wsdl:part name="authenticationFault" element="model:AuthenticationFault"/> </wsdl:message>

9.3.4 message authorizationFault

The message contains details related to a authorization fault

parts	**authorizationFault** element **model:AuthorizationFault**
used by	Operation **batchRequest** in PortType **EddsPortType** Operation **streamRequest** in PortType **EddsPortType** Operation **getStatus** in PortType **EddsPortType** Operation **getStatuses** in PortType **EddsPortType** Operation **getResponse** in PortType **EddsPortType** Operation **getRequest** in PortType **EddsPortType** Operation **getStreamRequest** in PortType **EddsPortType** Operation **startStreamData** in PortType **EddsPortType** Operation **getParamDefinitions** in PortType **EddsPortType** Operation **getFarcCatalogue** in PortType **EddsPortType** Operation **cancel** in PortType **EddsPortType** Operation **suspend** in PortType **EddsPortType** Operation **resume** in PortType **EddsPortType** Operation **deleteData** in PortType **EddsPortType** Operation **deleteRequest** in PortType **EddsPortType** Operation **getJobs** in PortType **EddsPortType** Operation **getJobsWithStatus** in PortType **EddsPortType** Operation **getJobsWithStatusAndLimit** in PortType **EddsPortType** Operation **getHistoricalLog** in PortType **EddsPortType** Operation **stopHistoricalLogRetrieval** in PortType **EddsPortType** Operation **getTransformations** in PortType **EddsPortType** Operation **accountRequest** in PortType **EddsPortType** Operation **getUsers** in PortType **EddsPortType** Operation **getMissions** in PortType **EddsPortType** Operation **getMissionsAndDomains** in PortType **EddsPortType** Operation **getRoles** in PortType **EddsPortType** Operation **getUserAccountDetails** in PortType **EddsPortType** Operation **getMissionDetails** in PortType **EddsPortType** Operation **getUserQuota** in PortType **EddsPortType** Operation **hasPermissionToSeeAllUsers** in PortType **EddsPortType**
source	<wsdl:message name="authorizationFault"> <wsdl:documentation>The message contains details related to a authorization fault</wsdl:documentation> <wsdl:part name="authorizationFault" element="model:AuthorizationFault"/> </wsdl:message>

9.3.5 message batchRequest

The message contains the details of a batch request

parts	**batchRequest** element **model:RequestMessagePart**
used by	Operation **batchRequest** in PortType **EddsPortType**
source	<wsdl:message name="batchRequest"> <wsdl:documentation>The message contains the details of a batch request</wsdl:documentation> <wsdl:part name="batchRequest" element="model:RequestMessagePart"/> </wsdl:message>

9.3.6 message batchRequestResponse

The message contains job ID associated to a batch request

parts	**batchRequestResponse** element **model:JobListIdPart**
used by	Operation **batchRequest** in PortType **EddsPortType**
source	<wsdl:message name="batchRequestResponse"> <wsdl:documentation>The message contains job ID associated to a batch request</wsdl:documentation> <wsdl:part name="batchRequestResponse" element="model:JobListIdPart"/> </wsdl:message>

9.3.7 message cancel

The message contains the details of "cancel" requests (the job IDs)

parts	**cancel** element **model:CancelPartList**
used by	Operation **cancel** in PortType **EddsPortType**
source	<wsdl:message name="cancel"> <wsdl:documentation>The message contains the details of "cancel" requests (the job IDs)</wsdl:documentation> <wsdl:part name="cancel" element="model:CancelPartList"/> </wsdl:message>

9.3.8 message suspend

The message contains the details of "suspend" request (the job IDs)

parts	**suspend** element **model:SuspendPartList**
used by	Operation **suspend** in PortType **EddsPortType**
source	<wsdl:message name="suspend"> <wsdl:documentation> The message contains the details of "suspend" requests (the job IDs)</wsdl:documentation> <wsdl:part name="suspend" element="model:SuspendPartList"/> </wsdl:message>

9.3.9 message resume

The message contains the details of "resume" requests (the job IDs)

parts	**resume** element **model:ResumePartList**
used by	Operation **resume** in PortType **EddsPortType**

source	<wsdl:message name="resume"> <wsdl:documentation>The message contains the details of "resume" requests (the job IDs)</wsdl:documentation> <wsdl:part name="resume" element="model:ResumePartList"/> </wsdl:message>

9.3.10 message cancelResponse

The message contains the job IDs associated with the cancel requests

parts	**cancelResponse** element **model:AcknowledgementPartList**
used by	Operation **cancel** in PortType **EddsPortType**
source	<wsdl:message name="cancelResponse"> <wsdl:documentation>The message contains the job IDs associated with the cancel requests</wsdl:documentation> <wsdl:part name="cancelResponse" element="model:AcknowledgementPartList"/> </wsdl:message>

9.3.11 message suspendResponse

The message contains the job IDs associated with the suspend requests

parts	**suspendResponse** element **model:AcknowledgementPartList**
used by	Operation **suspend** in PortType **EddsPortType**
source	<wsdl:message name="suspendResponse"> <wsdl:documentation>The message contains the job IDs associated with the suspend requests</wsdl:documentation> <wsdl:part name="suspendResponse" element="model:AcknowledgementPartList"/> </wsdl:message>

9.3.12 message resumeResponse

The message contains the job IDs associated with the resume requests

parts	**resumeResponse** element **model:AcknowledgementPartList**
used by	Operation **resume** in PortType **EddsPortType**
source	<wsdl:message name="resumeResponse"> <wsdl:documentation>The message contains the job IDs associated with the resume requests</wsdl:documentation> <wsdl:part name="resumeResponse" element="model:AcknowledgementPartList"/> </wsdl:message>

9.3.13 message checkSession

Empty message

parts	
used by	Operation **hasSessionExpired** in PortType **EddsPortType**
source	<wsdl:message name="checkSession"> <wsdl:documentation>Empty message</wsdl:documentation> </wsdl:message>

9.3.14 message checkSessionResponse

True if session has expired (or not started)

parts	**sessionResponse** element **model:SessionExpired**
used by	Operation **hasSessionExpired** in PortType **EddsPortType**
source	<wsdl:message name="checkSessionResponse"> <wsdl:documentation>True if session has expired (or not started)</wsdl:documentation> <wsdl:part name="sessionResponse" element="model:SessionExpired"/> </wsdl:message>

9.3.15 message checkUserPassword

The message contains the password to be verified

parts	**checkUserPassword** element **model:Password**
used by	Operation **checkUserPassword** in PortType **EddsPortType**
source	<wsdl:message name="checkUserPassword"> <wsdl:documentation>The message contains the password to be verified</wsdl:documentation> <wsdl:part name="checkUserPassword" element="model:Password"/> </wsdl:message>

9.3.16 message checkUserPasswordResponse

Empty message

parts	
used by	Operation **checkUserPassword** in PortType **EddsPortType**
source	<wsdl:message name="checkUserPasswordResponse"> <wsdl:documentation>Empty message</wsdl:documentation> </wsdl:message>

9.3.17 message deleteData

The message contains the details of a request to delete response data (a job ID)

parts	**deleteData** element **model:DeleteDataPart**
used by	Operation **deleteData** in PortType **EddsPortType**
source	<wsdl:message name="deleteData"> <wsdl:documentation>The message contains the details of a request to delete response data (a job ID)</wsdl:documentation> <wsdl:part name="deleteData" element="model:DeleteDataPart"/> </wsdl:message>

9.3.18 message deleteDataResponse

The message contains the job IDs associated to a request to delete response data

parts	**deleteDataResponse** element **model:AcknowledgementPartList**
used by	Operation **deleteData** in PortType **EddsPortType**
source	<wsdl:message name="deleteDataResponse"> <wsdl:documentation>The message contains the job IDs associated to a request to delete response data</wsdl:documentation> <wsdl:part name="deleteDataResponse" element="model:AcknowledgementPartList"/> </wsdl:message>

9.3.19 message deleteRequest

The message contains the details of a request to delete requests (the job IDs)

parts	**deleteRequest** element **model:DeleteRequestPart**
used by	Operation **deleteRequest** in PortType **EddsPortType**
source	<wsdl:message name="deleteRequest"> <wsdl:documentation>The message contains the details of a request to delete requests (the job IDs)</wsdl:documentation> <wsdl:part name="deleteRequest" element="model:DeleteRequestPart"/> </wsdl:message>

9.3.20 message deleteRequestResponse

The message contains the job IDs associated to the deletion requests

parts	**deleteRequestResponse** element **model:AcknowledgementPartList**
used by	Operation **deleteRequest** in PortType **EddsPortType**
source	<wsdl:message name="deleteRequestResponse"> <wsdl:documentation>The message contains the job IDs associated to the deletion requests</wsdl:documentation>

	<wsdl:part name="deleteRequestResponse" element="model:AcknowledgementPartList"/> </wsdl:message>

9.3.21 message getAllowedRequestTypes

Gets the request types the logged-in user has permission to issue

parts	
used by	Operation **getAllowedRequestTypes** in PortType **EddsPortType**
source	<wsdl:message name="getAllowedRequestTypes"> <wsdl:documentation> Gets the request types the logged-in user has permission to issue </wsdl:documentation> </wsdl:message>

9.3.22 message getAllowedRequestTypesResponse

Returns the list of request types the logged-in user has permission to issue

parts	**allowedRequestTypesResponse** element **model:AllowedRequestTypes**
used by	Operation **getAllowedRequestTypes** in PortType **EddsPortType**
source	<wsdl:message name="getAllowedRequestTypesResponse"> <wsdl:documentation>Returns the list of request types the logged-in user has permission to issue</wsdl:documentation> <wsdl:part name="allowedRequestTypesResponse" element="model:AllowedRequestTypes"/> </wsdl:message>

9.3.23 message getCatalogue

Gets the catalogue for the specified data source.

parts	**getCatalogue** element **model:CatalogueRequest**
used by	Operation **getCatalogue** in PortType **EddsPortType**
source	<wsdl:message name="getCatalogue"> <wsdl:documentation>Gets the catalogue for the specified data source.</wsdl:documentation> <wsdl:part name="getCatalogue" element="model:CatalogueRequest"/> </wsdl:message>

9.3.24 message getCatalogueResponse

Returns the allocated request ID for the request. This is used to take the asynchronous messages off the topic.

parts	**catalogueResponse** element **model:JobIdPart**
used by	Operation **getCatalogue** in PortType **EddsPortType**

source	<wsdl:message name="getCatalogueResponse"> <wsdl:documentation>Returns the allocated request ID for the request. This is used to take the asynchronous messages off the topic.</wsdl:documentation> <wsdl:part name="catalogueResponse" element="model:JobIdPart"/> </wsdl:message>

9.3.25 message getDataspaces

Gets the list of available dataspaces. For DARC, the first entry in the list is the currently active one.

parts	**getDataspaces** element **model:DataspacesRequest**
used by	Operation **getDataspaces** in PortType **EddsPortType**
source	<wsdl:message name="getDataspaces"> <wsdl:documentation>Gets the list of available dataspaces. For DARC, the first entry in the list is the currently active one.</wsdl:documentation> <wsdl:part name="getDataspaces" element="model:DataspacesRequest"/> </wsdl:message>

9.3.26 message getDataspacesResponse

Returns the list of available dataspaces. For DARC, the first entry in the list is the currently active one

parts	**dataspacesResponse** element **model:ListString**
used by	Operation **getDataspaces** in PortType **EddsPortType**
source	<wsdl:message name="getDataspacesResponse"> <wsdl:documentation>Returns the list of available dataspaces. For DARC, the first entry in the list is the currently active one</wsdl:documentation> <wsdl:part name="dataspacesResponse" element="model:ListString"/> </wsdl:message>

9.3.27 message getFarcCatalogue

Gets the FARC catalogue for the specified mission and domain (-1 represents the "No domain").
@deprecated use getCatalogue instead

parts	**getFarcCatalogue** element **model:FarcCatalogueRequest**
used by	Operation **getFarcCatalogue** in PortType **EddsPortType**
source	<wsdl:message name="getFarcCatalogue"> <wsdl:documentation>Gets the FARC catalogue for the specified mission and domain (-1 represents the "No domain"). @deprecated use getCatalogue instead</wsdl:documentation> <wsdl:part name="getFarcCatalogue" element="model:FarcCatalogueRequest"/> </wsdl:message>

9.3.28 message getFarcCatalogueResponse

Returns the allocated request ID for the request. This is used to take the messages off the topic.
@deprecated

parts	**farcCatalogueResponse** element **model:JobIdPart**
used by	Operation **getFarcCatalogue** in PortType **EddsPortType**
source	<wsdl:message name="getFarcCatalogueResponse"> <wsdl:documentation>Returns the allocated request ID for the request. This is used to take the messages off the topic. @deprecated</wsdl:documentation> <wsdl:part name="farcCatalogueResponse" element="model:JobIdPart"/> </wsdl:message>

9.3.29 message getHistoricalLog

The message contains the start and end date to retrieve the historical log from EDDS

parts	**getHistoricalLog** element **model:TimeWindow**
used by	Operation **getHistoricalLog** in PortType **EddsPortType**
source	<wsdl:message name="getHistoricalLog"> <wsdl:documentation>The message contains the start and end date to retrieve the historical log from EDDS</wsdl:documentation> <wsdl:part name="getHistoricalLog" element="model:TimeWindow"/> </wsdl:message>

9.3.30 message getHistoricalLogResponse

Returns the allocated request ID for the request. This is used to take the messages off the topic.

parts	**getHistoricalLogResponse** element **model:JobIdPart**
used by	Operation **getHistoricalLog** in PortType **EddsPortType**
source	<wsdl:message name="getHistoricalLogResponse"> <wsdl:documentation>Returns the allocated request ID for the request. This is used to take the messages off the topic.</wsdl:documentation> <wsdl:part name="getHistoricalLogResponse" element="model:JobIdPart"/> </wsdl:message>

9.3.31 message getJobs

The message contains the start and end date to retrieve the jobs from EDDS

parts	**getJobs** element **model:TimeWindow**
used by	Operation **getJobs** in PortType **EddsPortType** Operation **getJobsWithStatus** in PortType **EddsPortType**

source	<wsdl:message name="getJobs"> <wsdl:documentation>The message contains the start and end date to retrieve the jobs from EDDS</wsdl:documentation> <wsdl:part name="getJobs" element="model:TimeWindow"/> </wsdl:message>

9.3.32 message getJobsResponse

The list of jobs

parts	**getJobsResponse** element **model:JobListIdPart**
used by	Operation **getJobs** in PortType **EddsPortType**
source	<wsdl:message name="getJobsResponse"> <wsdl:documentation>The list of jobs</wsdl:documentation> <wsdl:part name="getJobsResponse" element="model:JobListIdPart"/> </wsdl:message>

9.3.33 message getJobsWithStatus

The message contains the start and end date to retrieve the jobs from EDDS

parts	**getJobsWithStatus** element **model:TimeWindow**
source	<wsdl:message name="getJobsWithStatus"> <wsdl:documentation>The message contains the start and end date to retrieve the jobs from EDDS</wsdl:documentation> <wsdl:part name="getJobsWithStatus" element="model:TimeWindow"/> </wsdl:message>

9.3.34 message getJobsWithStatusAndLimit

parts	**getJobsWithStatusAndLimit** element **model:TimeWindowWithLimit**
used by	Operation **getJobsWithStatusAndLimit** in PortType **EddsPortType**
source	<wsdl:message name="getJobsWithStatusAndLimit"> <wsdl:part name="getJobsWithStatusAndLimit" element="model:TimeWindowWithLimit"/> </wsdl:message>

9.3.35 message getJobsWithStatusAndLimitResponse

parts	**jobIdPart** element **model:JobIdPart**
used by	Operation **getJobsWithStatusAndLimit** in PortType **EddsPortType**
source	<wsdl:message name="getJobsWithStatusAndLimitResponse"> <wsdl:part name="jobIdPart" element="model:JobIdPart"/>

	</wsdl:message>

9.3.36 message getJobsWithStatusResponse

Returns the allocated unique ID for the request. This is used to take the messages off the topic. The ID is a random string, not a Request ID.

parts	**getJobsWithStatusResponse** element **model:JobIdPart**
used by	Operation **getJobsWithStatus** in PortType **EddsPortType**
source	<wsdl:message name="getJobsWithStatusResponse"> <wsdl:documentation>Returns the allocated unique ID for the request. This is used to take the messages off the topic. The ID is a random string, not a Request ID.</wsdl:documentation> <wsdl:part name="getJobsWithStatusResponse" element="model:JobIdPart"/> </wsdl:message>

9.3.37 message getLastConsolidation

Gets the last consolidation time for the specified mission, request type and dataspace name if given.

parts	**getLastConsolidation** element **model:LastConsolidationRequest**
used by	Operation **getLastConsolidation** in PortType **EddsPortType**
source	<wsdl:message name="getLastConsolidation"> <wsdl:documentation>Gets the list of available dataspaces. For DARC, the first entry in the list is the currently active one.</wsdl:documentation> <wsdl:part name="getLastConsolidation" element="model:LastConsolidationRequest"/> </wsdl:message>

9.3.38 message getLastConsolidationResponse

Gets the last consolidation time for the specified mission, request type and dataspace name if given.

parts	**lastConsolidationResponse** element **model:Long**
used by	Operation **getLastConsolidation** in PortType **EddsPortType**
source	<wsdl:message name="getLastConsolidationResponse"> <wsdl:documentation>Returns the list of available dataspaces. For DARC, the first entry in the list is the currently active one</wsdl:documentation> <wsdl:part name="lastConsolidationResponse" element="model:Long"/> </wsdl:message>

9.3.39 message getMissionDetails

The message contains a mission's name

parts	**getMissionDetails** element **model:MissionName**
used by	Operation **getMissionDetails** in PortType **EddsPortType**
source	<wsdl:message name="getMissionDetails">

	<wsdl:documentation>The message contains a mission's name</wsdl:documentation> <wsdl:part name="getMissionDetails" element="model:MissionName"/> </wsdl:message>

9.3.40 message getMissionDetailsResponse

The message contains the missions details associated to the mission's name as defined in the LDAP

parts	**getMissionDetailsResponse** element **model:MissionDetails**
used by	Operation **getMissionDetails** in PortType **EddsPortType**
source	<wsdl:message name="getMissionDetailsResponse"> <wsdl:documentation>The message contains the missions details associated to the mission's name as defined in the LDAP</wsdl:documentation> <wsdl:part name="getMissionDetailsResponse" element="model:MissionDetails"/> </wsdl:message>

9.3.41 message getMissions

Empty message

parts	
used by	Operation **getMissions** in PortType **EddsPortType**
source	<wsdl:message name="getMissions"> <wsdl:documentation>Empty message</wsdl:documentation> </wsdl:message>

9.3.42 message getMissionsResponse

The list of missions

parts	**getMissionsResponse** element **model:MissionsList**
used by	Operation **getMissions** in PortType **EddsPortType**
source	<wsdl:message name="getMissionsResponse"> <wsdl:documentation>The list of missions</wsdl:documentation> <wsdl:part name="getMissionsResponse" element="model:MissionsList"/> </wsdl:message>

9.3.43 message getMissionsAndDomains

Empty message

parts	

used by	Operation **getMissionsAndDomains** in PortType **EddsPortType**
source	<wsdl:message name="getMissionsAndDomains"> <wsdl:documentation>Empty message</wsdl:documentation> </wsdl:message>

9.3.44 message getMissionsAndDomainsResponse

The list of missions with their domains

parts	**getMissionsAndDomainsResponse** element **model:MissionsDomainList**
used by	Operation **getMissionsAndDomains** in PortType **EddsPortType**
source	<wsdl:message name="getMissionsAndDomainsResponse"> <wsdl:documentation>The list of missions with their domains</wsdl:documentation> <wsdl:part name="getMissionsAndDomainsResponse" element="model:MissionsDomainList"/> </wsdl:message>

9.3.45 message getParamDefinitions

Gets the parameter definitions for the specified mission.

parts	**getParamDefinitions** element **model:MissionName**
used by	Operation **getParamDefinitions** in PortType **EddsPortType**
source	<wsdl:message name="getParamDefinitions"> <wsdl:documentation>Gets the parameter definitions for the specified mission.</wsdl:documentation> <wsdl:part name="getParamDefinitions" element="model:MissionName"/> </wsdl:message>

9.3.46 message getParamDefinitionsResponse

Returns the allocated request ID for the request. This is used to take the messages off the topic.

parts	**paramDefinitionsResponse** element **model:JobIdPart**
used by	Operation **getParamDefinitions** in PortType **EddsPortType**
source	<wsdl:message name="getParamDefinitionsResponse"> <wsdl:documentation>Returns the allocated request ID for the request. This is used to take the messages off the topic.</wsdl:documentation> <wsdl:part name="paramDefinitionsResponse" element="model:JobIdPart"/> </wsdl:message>

9.3.47 message getQuotaDetails

Get the quota details for the specified user and mission

parts	**getQuotaDetailsMission** element **model:UserMission**
used by	Operation **getQuotaDetails** in PortType **EddsPortType**
source	<wsdl:message name="getQuotaDetails"> <wsdl:documentation>Get the quota details for the specified user and mission</wsdl:documentation> <wsdl:part name="getQuotaDetailsMission" element="model:UserMission"/> </wsdl:message>

9.3.48 message getQuotaDetailsResponse

The message contains the quota usage details

parts	**getQuotaDetailsResponse** element **model:UserQuotaDetails**
used by	Operation **getQuotaDetails** in PortType **EddsPortType**
source	<wsdl:message name="getQuotaDetailsResponse"> <wsdl:documentation>The message contains the quota usage details</wsdl:documentation> <wsdl:part name="getQuotaDetailsResponse" element="model:UserQuotaDetails"/> </wsdl:message>

9.3.49 message getRequest

The message contains the details of a "request" request (a job ID)

parts	**getRequest** element **model:JobIdPart**
used by	Operation **getRequest** in PortType **EddsPortType**
source	<wsdl:message name="getRequest"> <wsdl:documentation>The message contains the details of a "request" request (a job ID)</wsdl:documentation> <wsdl:part name="getRequest" element="model:JobIdPart"/> </wsdl:message>

9.3.50 message getRequestResponse

The message contains the original request associated to the given job ID

parts	**getRequestResponse** element **model:RequestMessagePart**
used by	Operation **getRequest** in PortType **EddsPortType**
source	<wsdl:message name="getRequestResponse"> <wsdl:documentation>The message contains the original request associated to the given job ID</wsdl:documentation> <wsdl:part name="getRequestResponse" element="model:RequestMessagePart"/> </wsdl:message>

9.3.51 message getResponse

The message contains the details of a "response" request (a job ID)

parts	**getResponse** element **model:DownloadList**
used by	Operation **getResponse** in PortType **EddsPortType**
source	<wsdl:message name="getResponse"> <wsdl:documentation> Get the response file(s) for a request. If the file list is empty, all response files are returned.</wsdl:documentation> <wsdl:part name="getResponse" element="model:DownloadList"/> </wsdl:message>

9.3.52 message getResponseResponse

The message contains the response file associated to a specific request

parts	**getResponseResponse** element **model:ResponseData**
used by	Operation **getResponse** in PortType **EddsPortType**
source	<wsdl:message name="getResponseResponse"> <wsdl:documentation>The message contains the response file associated to a specific request</wsdl:documentation> <wsdl:part name="getResponseResponse" element="model:ResponseData"/> </wsdl:message>

9.3.53 message getRoles

The message contains the name of the mission to get the roles for

parts	**getRoles** element **model:MissionName**
used by	Operation **getRoles** in PortType **EddsPortType**
source	<wsdl:message name="getRoles"> <wsdl:documentation>The message contains the name of the mission to get the roles for</wsdl:documentation> <wsdl:part name="getRoles" element="model:MissionName"/> </wsdl:message>

9.3.54 message getRolesResponse

The list of roles for the specified mission

parts	**getRolesResponse** element **model:RolesList**
used by	Operation **getRoles** in PortType **EddsPortType**
source	<wsdl:message name="getRolesResponse"> <wsdl:documentation>The list of roles for the specified mission</wsdl:documentation> <wsdl:part name="getRolesResponse" element="model:RolesList"/>

	</wsdl:message>

9.3.55 message getStatus

The message contains the details of a "status" request (a job ID)

parts	**getStatus** element **model:JobIdPart**
used by	Operation **getStatus** in PortType **EddsPortType**
source	<wsdl:message name="getStatus"> <wsdl:documentation>The message contains the details of a "status" request (a job ID)</wsdl:documentation> <wsdl:part name="getStatus" element="model:JobIdPart"/> </wsdl:message>

9.3.56 message getStatuses

Gets the status of a list of jobs

parts	**getStatus** element **model:JobListIdPart**
used by	Operation **getStatuses** in PortType **EddsPortType**
source	<wsdl:message name="getStatuses"> <wsdl:documentation>Gets the status of a list of jobs</wsdl:documentation> <wsdl:part name="getStatus" element="model:JobListIdPart"/> </wsdl:message>

9.3.57 message getStatusesResponse

Returns the allocated unique ID for the request. This is used to take the messages off the topic. The ID is a random string, not a Request ID.

parts	**getStatusesResponse** element **model:JobIdPart**
used by	Operation **getStatuses** in PortType **EddsPortType**
source	<wsdl:message name="getStatusesResponse"> <wsdl:documentation>Returns the allocated unique ID for the request. This is used to take the messages off the topic. The ID is a random string, not a Request ID.</wsdl:documentation> <wsdl:part name="getStatusesResponse" element="model:JobIdPart"/> </wsdl:message>

9.3.58 message getStatusResponse

The message contains the acknowledgement of a specific request

parts	**getStatusResponse** element **model:AcknowledgementPart**

used by	Operation **getStatus** in PortType **EddsPortType**
source	<wsdl:message name="getStatusResponse"> <wsdl:documentation>The message contains the acknowledgement of a specific request</wsdl:documentation> <wsdl:part name="getStatusResponse" element="model:AcknowledgementPart"/> </wsdl:message>

9.3.59 message getTransformations

The message contains the mission name and request type

parts	**getTransformations** element **model:TransformationsRequest**
used by	Operation **getTransformations** in PortType **EddsPortType**
source	<wsdl:message name="getTransformations"> <wsdl:documentation>The message contains the mission name and request type</wsdl:documentation> <wsdl:part name="getTransformations" element="model:TransformationsRequest"/> </wsdl:message>

9.3.60 message getTransformationsResponse

The message contains a list of available transformations

parts	**getTransformationsResponse** element **model:TransformationsList**
used by	Operation **getTransformations** in PortType **EddsPortType**
source	<wsdl:message name="getTransformationsResponse"> <wsdl:documentation>The message contains a list of available transformations</wsdl:documentation> <wsdl:part name="getTransformationsResponse" element="model:TransformationsList"/> </wsdl:message>

9.3.61 message getUserAccountDetails

The message contains a username

parts	**getUserAccountDetails** element **model:UserName**
used by	Operation **getUserAccountDetails** in PortType **EddsPortType**
source	<wsdl:message name="getUserAccountDetails"> <wsdl:documentation>The message contains a username</wsdl:documentation> <wsdl:part name="getUserAccountDetails" element="model:UserName"/> </wsdl:message>

9.3.62 message getUserAccountDetailsResponse

The message contains the user details associated to the username as defined in the LDAP

parts	**getUserAccountDetailsResponse** element **model:UserAccountDetails**
used by	Operation **getUserAccountDetails** in PortType **EddsPortType**
source	<wsdl:message name="getUserAccountDetailsResponse"> <wsdl:documentation>The message contains the user details associated to the username as defined in the LDAP</wsdl:documentation> <wsdl:part name="getUserAccountDetailsResponse" element="model:UserAccountDetails"/> </wsdl:message>

9.3.63 message getUserQuota

The message contains a mission's name

parts	**getUserQuota** element **model:MissionName**
used by	Operation **getUserQuota** in PortType **EddsPortType**
source	<wsdl:message name="getUserQuota"> <wsdl:documentation>The message contains a mission's name</wsdl:documentation> <wsdl:part name="getUserQuota" element="model:MissionName"/> </wsdl:message>

9.3.64 message getUserQuotaResponse

The message contains the user quota usage details

parts	**getUserQuotaResponse** element **model:UserQuotaDetails**
used by	Operation **getUserQuota** in PortType **EddsPortType**
source	<wsdl:message name="getUserQuotaResponse"> <wsdl:documentation>The message contains the user quota usage details</wsdl:documentation> <wsdl:part name="getUserQuotaResponse" element="model:UserQuotaDetails"/> </wsdl:message>

9.3.65 message getUsers

Flag to indicate whether to fetch all users or not

parts	**getUsers** element **model:IncludeAll**
used by	Operation **getUsers** in PortType **EddsPortType**
source	<wsdl:message name="getUsers"> <wsdl:documentation>Flag to indicate whether to fetch all users or not</wsdl:documentation>

	<wsdl:part name="getUsers" element="model:IncludeAll"/> </wsdl:message>

9.3.66 message getUsersResponse

The list of users

parts	**getUsersResponse** element **model:UsersList**
used by	Operation **getUsers** in PortType **EddsPortType**
source	<wsdl:message name="getUsersResponse"> <wsdl:documentation>The list of users</wsdl:documentation> <wsdl:part name="getUsersResponse" element="model:UsersList"/> </wsdl:message>

9.3.67 message hasPermissionToSeeAllUsers

The message contains a username

parts	**hasPermissionToSeeAllUsers** element **model:UserName**
used by	Operation **hasPermissionToSeeAllUsers** in PortType **EddsPortType**
source	<wsdl:message name="hasPermissionToSeeAllUsers"> <wsdl:documentation>The message contains a username</wsdl:documentation> <wsdl:part name="hasPermissionToSeeAllUsers" element="model:UserName"/> </wsdl:message>

9.3.68 message logIn

The message contains the user credentials needed for the authentication

parts	**credentials** element **model:UserCredentials**
used by	Operation **logIn** in PortType **EddsPortType**
source	<wsdl:message name="logIn"> <wsdl:documentation>The message contains the user credentials needed for the authentication</wsdl:documentation> <wsdl:part name="credentials" element="model:UserCredentials"/> </wsdl:message>

9.3.69 message logInResponse

The message returns the user credentials enriched with the session ID

parts	**credentials** element **model:UserCredentials**
used by	Operation **logIn** in PortType **EddsPortType**

source	<wsdl:message name="logInResponse"> <wsdl:documentation>The message returns the user credentials enriched with the session ID</wsdl:documentation> <wsdl:part name="credentials" element="model:UserCredentials"/> </wsdl:message>

9.3.70 message logOut

Empty message

parts	
used by	Operation **logOut** in PortType **EddsPortType**
source	<wsdl:message name="logOut"> <wsdl:documentation>Empty message</wsdl:documentation> </wsdl:message>

9.3.71 message passwordExpiredFault

The message contains details related to a authentication fault

parts	**passwordExpiredFault** element **model:PasswordExpiredFault**
used by	Operation **logIn** in PortType **EddsPortType**
source	<wsdl:message name="passwordExpiredFault"> <wsdl:documentation>The message contains details related to a authentication fault</wsdl:documentation> <wsdl:part name="passwordExpiredFault" element="model:PasswordExpiredFault"/> </wsdl:message>

9.3.72 message permissionToSeeAllUsersResponse

True if session has expired (or not started)

parts	**permissionToSeeAllUsersResponse** element **model:PermissionToSeeAllUsers**
used by	Operation **hasPermissionToSeeAllUsers** in PortType **EddsPortType**
source	<wsdl:message name="permissionToSeeAllUsersResponse"> <wsdl:documentation>True if session has expired (or not started)</wsdl:documentation> <wsdl:part name="permissionToSeeAllUsersResponse" element="model:PermissionToSeeAllUsers"/> </wsdl:message>

9.3.73 message requestFault

The message contains details related to a request fault

parts	**requestFault** element **model:RequestFault**
used by	Operation **batchRequest** in PortType **EddsPortType** Operation **streamRequest** in PortType **EddsPortType** Operation **getStatus** in PortType **EddsPortType** Operation **getStatuses** in PortType **EddsPortType** Operation **getResponse** in PortType **EddsPortType** Operation **getRequest** in PortType **EddsPortType** Operation **getStreamRequest** in PortType **EddsPortType** Operation **startStreamData** in PortType **EddsPortType** Operation **getParamDefinitions** in PortType **EddsPortType** Operation **getFarcCatalogue** in PortType **EddsPortType** Operation **cancel** in PortType **EddsPortType** Operation **suspend** in PortType **EddsPortType** Operation **resume** in PortType **EddsPortType** Operation **deleteData** in PortType **EddsPortType** Operation **deleteRequest** in PortType **EddsPortType** Operation **getJobs** in PortType **EddsPortType** Operation **getJobsWithStatus** in PortType **EddsPortType** Operation **getJobsWithStatusAndLimit** in PortType **EddsPortType** Operation **getHistoricalLog** in PortType **EddsPortType** Operation **stopHistoricalLogRetrieval** in PortType **EddsPortType** Operation **getTransformations** in PortType **EddsPortType** Operation **accountRequest** in PortType **EddsPortType** Operation **getUsers** in PortType **EddsPortType** Operation **getMissions** in PortType **EddsPortType** Operation **getMissionsAndDomains** in PortType **EddsPortType** Operation **getRoles** in PortType **EddsPortType** Operation **getUserAccountDetails** in PortType **EddsPortType** Operation **getMissionDetails** in PortType **EddsPortType** Operation **getUserQuota** in PortType **EddsPortType** Operation **hasPermissionToSeeAllUsers** in PortType **EddsPortType**
source	<wsdl:message name="requestFault"> <wsdl:documentation>The message contains details related to a request fault</wsdl:documentation> <wsdl:part name="requestFault" element="model:RequestFault"/> </wsdl:message>

9.3.74 message reSubmit

The message contains the details of a request to re-submit(a job ID)

parts	**reSubmit** element **model:JobIdPart**
source	<wsdl:message name="reSubmit"> <wsdl:documentation>The message contains the details of a request to re-submit(a job ID)</wsdl:documentation> <wsdl:part name="reSubmit" element="model:JobIdPart"/> </wsdl:message>

9.3.75 message reSubmitResponse

The message contains job IDs associated to a batch request

parts	**reSubmitResponse** element **model:JobListIdPart**
source	<wsdl:message name="reSubmitResponse"> <wsdl:documentation>The message contains job IDs associated to a batch request</wsdl:documentation> <wsdl:part name="reSubmitResponse" element="model:JobListIdPart"/> </wsdl:message>

9.3.76 message sessionFault

The message contains details related to a session fault

parts	**sessionFault** element **model:SessionFault**
used by	Operation **batchRequest** in PortType **EddsPortType** Operation **streamRequest** in PortType **EddsPortType** Operation **getStatus** in PortType **EddsPortType** Operation **getStatuses** in PortType **EddsPortType** Operation **getResponse** in PortType **EddsPortType** Operation **getRequest** in PortType **EddsPortType** Operation **getStreamRequest** in PortType **EddsPortType** Operation **startStreamData** in PortType **EddsPortType** Operation **stopStreamData** in PortType **EddsPortType** Operation **getParamDefinitions** in PortType **EddsPortType** Operation **getFarcCatalogue** in PortType **EddsPortType** Operation **cancel** in PortType **EddsPortType** Operation **suspend** in PortType **EddsPortType** Operation **resume** in PortType **EddsPortType** Operation **deleteData** in PortType **EddsPortType** Operation **deleteRequest** in PortType **EddsPortType** Operation **getJobs** in PortType **EddsPortType** Operation **getJobsWithStatus** in PortType **EddsPortType** Operation **getJobsWithStatusAndLimit** in PortType **EddsPortType** Operation **getHistoricalLog** in PortType **EddsPortType** Operation **stopHistoricalLogRetrieval** in PortType **EddsPortType** Operation **getTransformations** in PortType **EddsPortType** Operation **checkUserPassword** in PortType **EddsPortType** Operation **accountRequest** in PortType **EddsPortType** Operation **getUsers** in PortType **EddsPortType** Operation **getMissions** in PortType **EddsPortType** Operation **getMissionsAndDomains** in PortType **EddsPortType** Operation **getRoles** in PortType **EddsPortType** Operation **getUserAccountDetails** in PortType **EddsPortType** Operation **getMissionDetails** in PortType **EddsPortType** Operation **getUserQuota** in PortType **EddsPortType** Operation **hasPermissionToSeeAllUsers** in PortType **EddsPortType**
source	<wsdl:message name="sessionFault"> <wsdl:documentation>The message contains details related to a session fault</wsdl:documentation> <wsdl:part name="sessionFault" element="model:SessionFault"/> </wsdl:message>

9.3.77 message streamRequest

The message contains the details of a stream request

parts	**streamRequest** element **model:StreamRequestMessagePart**
used by	Operation **streamRequest** in PortType **EddsPortType**
source	<wsdl:message name="streamRequest"> <wsdl:documentation>The message contains the details of a stream request</wsdl:documentation> <wsdl:part name="streamRequest" element="model:StreamRequestMessagePart"/> </wsdl:message>

9.3.78 message streamRequestResponse

The message contains job ID associated to a stream request

parts	**streamRequestResponse** element **model:JobIdPart**
used by	Operation **streamRequest** in PortType **EddsPortType**
source	<wsdl:message name="streamRequestResponse"> <wsdl:documentation>The message contains job ID associated to a stream request</wsdl:documentation> <wsdl:part name="streamRequestResponse" element="model:JobIdPart"/> </wsdl:message>

9.3.79 message startStreamData

Starts the incoming stream of the specified active stream request to the client.

parts	**startStreamData** element **model:JobIdPart**
used by	Operation **startStreamData** in PortType **EddsPortType**
source	<wsdl:message name="startStreamData"> <wsdl:documentation>Starts the incoming stream of the specified active stream request to the client.</wsdl:documentation> <wsdl:part name="startStreamData" element="model:JobIdPart"/> </wsdl:message>

9.3.80 message startStreamDataResponse

Empty message

parts	
used by	Operation **startStreamData** in PortType **EddsPortType**
source	<wsdl:message name="startStreamDataResponse"> <wsdl:documentation>Empty message</wsdl:documentation> </wsdl:message>

9.3.81 message stopHistoricalLogRetrieval

Stops the retrieval of historical log messages. Requires the request ID provided by getHistoricalLog.

parts	**stopHistoricalLogRetrieval** element **model:JobIdPart**
used by	Operation **stopHistoricalLogRetrieval** in PortType **EddsPortType**
source	<wsdl:message name="stopHistoricalLogRetrieval"> <wsdl:documentation>Stops the retrieval of historical log messages. Requires the request ID provided by getHistoricalLog.</wsdl:documentation> <wsdl:part name="stopHistoricalLogRetrieval" element="model:JobIdPart"/> </wsdl:message>

9.3.82 message stopHistoricalLogRetrievalResponse

Empty message

parts	
used by	Operation **stopHistoricalLogRetrievalResponse** in PortType **EddsPortType**
source	<wsdl:message name="stopHistoricalLogRetrievalResponse"> <wsdl:documentation>Empty message</wsdl:documentation> </wsdl:message>

9.3.83 message stopStreamData

Stops the incoming stream of the specified active stream request to the client.

parts	**stopStreamData** element **model:JobIdPart**
used by	Operation **stopStreamData** in PortType **EddsPortType**
source	<wsdl:message name="stopStreamData"> <wsdl:documentation>Stops the incoming stream of the specified active stream request to the client.</wsdl:documentation> <wsdl:part name="stopStreamData" element="model:JobIdPart"/> </wsdl:message>

9.3.84 message getStreamRequest

The message contains the details of a "stream request" request (a job ID)

parts	**getStreamRequest** element **model:JobIdPart**
used by	Operation **getStreamRequest** in PortType **EddsPortType**
source	<wsdl:message name="getStreamRequest"> <wsdl:documentation>The message contains the details of a "stream request" request (a job ID)</wsdl:documentation> <wsdl:part name="getStreamRequest" element="model:JobIdPart"/> </wsdl:message>

9.3.85 message getStreamRequestResponse

The message contains the original stream request associated to the given job ID

parts	**getStreamRequestResponse** element **model:StreamRequestMessagePart**

used by	Operation **getStreamRequest** in PortType **EddsPortType**
source	<wsdl:message name=" getStreamRequestResponse "> <wsdl:documentation> The message contains the original stream request associated to the given job ID </wsdl:documentation> <wsdl:part name=" getStreamRequestResponse " element="model: StreamRequestMessagePart"/> </wsdl:message>

9.3.86 message getAlias

Get aliases for the mission

parts	**getAlias** element **model:AliasRequest**
used by	Operation **getAlias** in PortType **EddsPortType**
source	<wsdl:message name=" getAlias"> <wsdl:documentation>Get aliases for the mission </wsdl:documentation> <wsdl:part name=" getAlias" element="model:AliasRequest "/> </wsdl:message>

9.3.87 message getAliasResponse

Returns the list of aliases for the mission

parts	**getAliasResponse** element **model:ListAliasInfo**
used by	Operation **getAlias** in PortType **EddsPortType**
source	<wsdl:message name=" getAliasResponse"> <wsdl:documentation>Returns the list of aliases for the mission.</wsdl:documentation> <wsdl:part name=" aliasResponse" element="model:ListAliasInfo"/> </wsdl:message>

9.3.88 message checkSession

Empty message

parts	
used by	Operation **hasSessionExpired** in PortType **EddsPortType**
source	<wsdl:message name=" checkSession"> <wsdl:documentation>Empty message</wsdl:documentation> </wsdl:message>

9.3.89 message stopStreamDataResponse

Empty message

parts	
used by	Operation **stopStreamData** in PortType **EddsPortType**
source	<wsdl:message name="stopStreamDataResponse"> <wsdl:documentation>Empty message</wsdl:documentation> </wsdl:message>

9.4 Bindings

The EDDS binding describes how the service is bound to the messaging protocol. Note that this EUICD assumes the use of SOAP as the underlying protocol for all services.

type	**ws:EddsPortType**
extensibility	<soap:binding style="document" transport="http://schemas.xmlsoap.org/soap/http"/>
operations	**batchRequest** *This operation perform a batch request on the EDDS erver. The request is asynchonous and its output is the job ID or an error in case of failure* extensibility: <soap:operation soapAction="http://edds.egos.esa/batchrequest/batchRequest" style="document"/> input: name="batchRequest"<soap:body use="literal"/> output: name="batchRequestResponse"<soap:body use="literal"/> **streamRequest** *This operation perform a stream request on the EDDS erver. The request is asynchonous and its output is the job ID or an error in case of failure* extensibility: <soap:operation soapAction="http://edds.egos.esa/streamrequest/streamRequest" style="document"/> input: name="streamRequest"<soap:body use="literal"/> output: name="streamRequestResponse"<soap:body use="literal"/> **getStatus** *This operation provides the status of a specific request. The operation takes as input a job ID and return an acknowledgement* extensibility: <soap:operation soapAction="http://edds.egos.esa/batchrequest/getStatus" style="document"/> input: name="getStatus"<soap:body use="literal"/> output: name="getStatusResponse"<soap:body use="literal"/> **reSubmit** *This operation re-submits a request. The operation takes as input a job ID and its output is the list of job IDs* extensibility: <soap:operation soapAction="http://edds.egos.esa/batchrequest/reSubmit" style="document"/> input: name="reSubmit"<soap:body use="literal"/> output: name="reSubmitResponse"<soap:body use="literal"/> **getStatuses** *Gets the status of a list of jobs. The operation takes as input a list of job IDs and returns a java.util.ArrayList of type RequestInfoRec encoded in Base64* extensibility: <soap:operation soapAction="http://edds.egos.esa/batchrequest/getStatuses" style="document"/>

input name="getStatuses"<soap:body use="literal"/>

output name="getStatusesResponse"<soap:body use="literal"/>

getResponse

This operation returns the result file of abatch request. The response is available if the delivery method of the associated batch request was "server delivery". The method is synchronous.

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/getResponse" style="document"/>

input name="getResponse"<soap:body use="literal"/>

output name="getResponseResponse"<soap:body use="literal"/>

getRequest

This operation provides the template of an existing batch request. The operation takes in input the batch request job ID and returns as output the corresponding XML. The method is synchronous.

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/getRequest" style="document"/>

input name="getRequest"<soap:body use="literal"/>

output name="getRequestResponse"<soap:body use="literal"/>

getStreamRequest

This operation provides the template of an existing batch request. The operation takes in input the batch request job ID and returns as output the corresponding XML. The method is synchronous.

extensibility <soap:operation soapAction="http://edds.egos.esa/streamrequest/getStreamRequest" style="document"/>

input name="getStreamRequest"<soap:body use="literal"/>

output name="getStreamRequestResponse"<soap:body use="literal"/>

startStreamData

Starts the CometD forwarder of the specified stream request

extensibility <soap:operation soapAction="http://edds.egos.esa/streamrequest/startStreamData" style="document"/>

input name="startStreamData"<soap:body use="literal"/>

output name="startStreamDataResponse"<soap:body use="literal"/>

stopStreamData

Stops the CometD forwarder of the specified stream request

extensibility <soap:operation soapAction="http://edds.egos.esa/streamrequest/stopStreamData" style="document"/>

input name="stopStreamData"<soap:body use="literal"/>

output name="stopStreamDataResponse"<soap:body use="literal"/>

getParamDefinitions

Gets the parameter definitions for the specified mission. The method is synchronous.

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/getParamDefinitions" style="document"/>

input name="getParamDefinitions"<soap:body use="literal"/>

output name="getParamDefinitionsResponse"<soap:body use="literal"/>

getCatalogue

Gets the catalogue for the specified data source. The method is asynchronous.

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/getCatalogue" style="document"/>

input name="getCatalogue"<soap:body use="literal"/>

output name="getCatalogueResponse"<soap:body use="literal"/>

getFarcCatalogue

Gets the FARC catalogue for the specified mission and domain. The method is asynchronous.
@deprecated

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/getFarcCatalogue" style="document"/>

input name="getFarcCatalogue"<soap:body use="literal"/>

output name="getFarcCatalogueResponse"<soap:body use="literal"/>

getDataspaces

Gets the dataspaces for the specified mission and request type. The method is synchronous.

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/getDataspaces" style="document"/>

input name="getDataspaces"<soap:body use="literal"/>

output name="getDataspacesResponse"<soap:body use="literal"/>

getAlias

Gets alias for the specified mission. The method is synchronous.

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/getAlias" style="document"/>

input name="getAlias"<soap:body use="literal"/>

output name="getAliasResponse"<soap:body use="literal"/>

getLastConsolidation

Gets the last consolidation time for the specified mission, request type and dataspace name if given. The method is synchronous.

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/getLastConsolidation" style="document"/>

input name="getLastConsolidation"<soap:body use="literal"/>

output name="getLastConsolidationResponse"<soap:body use="literal"/>

getAllowedRequestTypes

Gets the request types that the currently logged-in user has permission to issue. The method is synchronous..

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/getAllowedRequestTypes" style="document"/>

input name="getAllowedRequestTypes"<soap:body use="literal"/>

output name="getAllowedRequestTypesResponse"<soap:body use="literal"/>

cancel

This operation cancel a scheduled request or try to interrupt an already executing request. The operation takes in input the job ID of the operation to cancel and return the job ID of the cancel request. The operation is asyncronous.

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/cancel" style="document"/>

input name="cancel"<soap:body use="literal"/>

output name="cancelResponse"<soap:body use="literal"/>

suspend

This operation suspends an active request. The operation takes in input the job ID of the operation to suspend and return the job ID of the suspend request. The operation is asyncronous.

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/suspend" style="document"/>

input name="suspend"<soap:body use="literal"/>

output name="suspendResponse"<soap:body use="literal"/>

resume

This operation resumes a suspended or failed request. The operation takes in input the job ID of the operation to resume and return the job ID of the resumed request. The operation is asyncronous.

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/resume" style="document"/>

input name="resume"<soap:body use="literal"/>

output name="resumeResponse"<soap:body use="literal"/>

deleteData

This operation deletes the data related to a request. The operation is asynchronous.

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/deleteData" style="document"/>

input name="deleteData"<soap:body use="literal"/>

output name="deleteDataResponse"<soap:body use="literal"/>

deleteRequest

This operation deletes a request. The operation is asyncronous.

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/deleteRequest" style="document"/>

input name="deleteRequest"<soap:body use="literal"/>

output name="deleteRequestResponse"<soap:body use="literal"/>

getJobs

This operation returns the list of jobs over a defined time span. The operation takes in input the start and end date (where null is considered as ALL) and returns the list of operation that a user has right to read. The operation is syncronous.

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/getJobs" style="document"/>

input name="getJobs"<soap:body use="literal"/>

output name="getJobsResponse"<soap:body use="literal"/>

getJobsWithStatus

This operation returns the list of jobs with their status over a defined time span. The operation takes in input the start and end date (where null is considered as ALL) and returns the list of operation that a user has right to read. The operation is syncronous. The message contains a java.util.ArrayList of type RequestInfoRec encoded in Base64

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/getJobsWithStatus" style="document"/>

input name="getJobsWithStatus"<soap:body use="literal"/>

output name="getJobsWithStatusResponse"<soap:body use="literal"/>

getJobsWithStatusAndLimit

This operation returns the list of jobs with their status over a defined time span. The operation takes in input the start and end date (where null is considered as ALL) and returns the list of operation that a user has right to read. The number of results returned can be limited by specifying an upper limit. Ommitting the limit returns all matching rows. The operation is syncronous. The message contains a java.util.ArrayList of type RequestInfoRec encoded in Base64

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/getJobsWithStatusAndLimit" style="document"/>

input name="getJobsWithStatusAndLimit"<soap:body use="literal"/>

output name="getJobsWithStatusAndLimitResponse"<soap:body use="literal"/>

getHistoricalLog

The operation return the historical log messages over a defined time span. The operation takes as input the start and end date (where null is considered as ALL) and returns a request ID used to listen to the resulting stream. The operation is asyncronous.

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/getHistoricalLog" style="document"/>

input name="getHistoricalLog"<soap:body use="literal"/>

output name="getHistoricalLogResponse"<soap:body use="literal"/>

stopHistoricalLogRetrieval

The operation stops the historical log message retrieval. The operation takes as input the request ID of the stream.

extensibility <soap:operation soapAction="http://edds.egos.esa/batchrequest/stopHistoricalLogRetrieval" style="document"/>

input name="stopHistoricalLogRetrieval"<soap:body use="literal"/>

output name="stopHistoricalLogRetrievalResponse"<soap:body use="literal"/>

getTransformations

The operation returns the available transformations for the requet type.

extensibility `<soap:operation soapAction="http://edds.egos.esa/batchrequest/getTransformations" style="document"/>`

input `name="getTransformations"<soap:body use="literal"/>`

output `name="getTransformationsResponse"<soap:body use="literal"/>`

logIn

Login the user and start the session

extensibility `<soap:operation soapAction="http://edds.egos.esa/authentication/logIn" style="document"/>`

input `name="logIn"<soap:body use="literal"/>`

output `name="logInResponse"<soap:body use="literal"/>`

hasSessionExpired

Checks if the user's session has expired, and return true if it has

extensibility `<soap:operation soapAction="http://edds.egos.esa/authentication/hasSessionExpired" style="document"/>`

input `name="checkSession"<soap:body use="literal"/>`

output `name="checkSessionResponse"<soap:body use="literal"/>`

logOut

Logout the user and close the session

extensibility `<soap:operation soapAction="http://edds.egos.esa/authentication/logOut" style="document"/>`

input `name="logOut"<soap:body use="literal"/>`

output

checkUserPassword

Check if the password supplied matches the user's current password. Used to double check the user's current password before allowing the user to change their password.

extensibility `<soap:operation soapAction="http://edds.egos.esa/authentication/checkUserPassword" style="document"/>`

input `name="checkUserPassword"<soap:body use="literal"/>`

output `name="checkUserPasswordResponse"<soap:body use="literal"/>`

accountRequest

This operation performs an accaunt management request. The operation is asynchronous: the detail of the request are contained in the request itself and the result of the operation is a job id. In case of error the operation returns an error.

extensibility `<soap:operation soapAction="http://edds.egos.esa/userrequest/accountRequest" style="document"/>`

input `<soap:body use="literal"/>`

output `<soap:body use="literal"/>`

getUsers

The operation return the list of users. The operation is syncronous. An error on the request will throw a fault

extensibility: <soap:operation soapAction="http://edds.egos.esa/userrequest/getUsers" style="document"/>

input: name="getUsers"<soap:body use="literal"/>

output: name="getUsersResponse"<soap:body use="literal"/>

getRoles

The operation return the list of roles the user has permission to use for the specified mission. The operation is syncronous. An error on the request will throw a fault

extensibility: <soap:operation soapAction="http://edds.egos.esa/userrequest/getRoles" style="document"/>

input: name="getRoles"<soap:body use="literal"/>

output: name="getRolesResponse"<soap:body use="literal"/>

getMissions

The operation return the list of missions. The operation is syncronous. An error on the request will throw a fault

extensibility: <soap:operation soapAction="http://edds.egos.esa/userrequest/getMissions" style="document"/>

input: name="getMissions"<soap:body use="literal"/>

output: name="getMissionsResponse"<soap:body use="literal"/>

getMissionsAndDomains

The operation return the list of missions with their domains. The operation is syncronous. An error on the request will throw a fault

extensibility: <soap:operation soapAction="http://edds.egos.esa/userrequest/getMissionsAndDomains" style="document"/>

input: name="getMissionsAndDomains"<soap:body use="literal"/>

output: name="getMissionsAndDomainsResponse"<soap:body use="literal"/>

getUserAccountDetails

The operation return the details associated to a specific user as defined in the LDAP. The operation is syncronous. An error on the request will throw a fault

extensibility: <soap:operation soapAction="http://edds.egos.esa/userrequest/getUserAccountDetails" style="document"/>

input: name="getUserAccountDetails"<soap:body use="literal"/>

output: name="getUserAccountDetailsResponse"<soap:body use="literal"/>

getMissionDetails

The operation return the details associated to a specific mission as defined in the LDAP. The operation is syncronous. An error on the request will throw a fault

extensibility: <soap:operation soapAction="http://edds.egos.esa/userrequest/getMissionDetails" style="document"/>

	input name="getMissionDetails"<soap:body use="literal"/> output name="getMissionDetailsResponse"<soap:body use="literal"/> **getUserQuota** *Returns the details for the specified mission* extensibility <soap:operation soapAction="http://edds.egos.esa/userrequest/getUserQuota" style="document"/> input name="getUserQuota"<wsdl:documentation>The name of the mission</wsdl:documentation><soap:body use="literal"/> output name="getUserQuotaResponse"<wsdl:documentation>The quota details for the mission</wsdl:documentation><soap:body use="literal"/> **getQuotaDetails** *Returns the quota for the specified user and mission* extensibility <soap:operation soapAction="http://edds.egos.esa/userrequest/getQuotaDetails" style="document"/> input name="getQuotaDetails"<wsdl:documentation> The name of the user and mission</wsdl:documentation><soap:body use="literal"/> output name="getQuotaDetailsResponse"<wsdl:documentation> The quota details for the user and mission</wsdl:documentation><soap:body use="literal"/> **hasPermissionToSeeAllUsers** *Checks if user has permission to see other users data* extensibility <soap:operation soapAction="http://edds.egos.esa/userrequest/hasPermissionToSeeAllUsers" style="document"/> input name="hasPermissionToSeeAllUsers"<wsdl:documentation>Username</wsdl:documentation><soap:body use="literal"/> output name="permissionToSeeAllUsersResponse"<wsdl:documentation>The result whether user has permission to see other users data or not</wsdl:documentation><soap:body use="literal"/>
used by	Port **EddsPort** in Service **EddsService**

10. E-mail Interface

10.1 Data Elements

The EDDS allows users to request the delivery of acknowledgement messages by email. This implies that the receiving client system has access to a mail server.

10.2 Message Description

The email contains a configurable message that can be tailored by the mission. The acknowledgement message is in attachment to the email in XML format.

10.3 Communication Protocol

SMTP or SMTPs are used by the EDDS.

10.4 Errors

A SMTP/SMTPs client cannot verify the correct reception of an email. It is assumed that the configuration for the email has been properly set by the mission and that the email address given by the user is valid.

11. File Server Interface

11.1 Data Elements

The EDDS allows users to request the delivery of batch responses to one or more client hosted file server(s). On completion of the batch request, the EDDS will try to deliver the response to the addresses given in the corresponding request.

11.2 Message Description

A description of a File server delivery request can be found in section Appendix C

A description of a batch response can be found in section Appendix C

11.3 Communication Protocol

The EDDS uses the FTP protocol (or SFTP depending on the mission configuration) to deliver the batch response using the PUT method. The client's file server must support the PUT action.

11.4 Errors

If a batch response cannot be delivered to one or more of the remote FTP server(s) (after a number of retries) it may be deleted or retained (for a limited period) on the EDDS server as described in the original batch request.

12. Client API

The section describes the Java APIs provided by the EDDS application. The purpose of the APIs are to allow a client application access to the EDDS services. After the “package” Maven target has been run within the EDDS source directories, the JAR file can be found within the “edds-ws-client/target” folder. After the “install” target has been run, the library is copied to the local Maven repository. See the Configuration and Installation Guide (CIG) [RD-15] for information on how to build and deploy EDDS.

The EDDS APIs are derived from the definition of the EDDS Web Services. The API’s implement a façade pattern on top of the classes generated from the WSDL files, which describe the EDDS services. They hide the implementation details of the services, like the connection to the web server.

EDDS Client API offers the following types of methods:

- EDDS Web Services client
- EDDS Push notifications client
- Utility methods

12.1 Edds Web Services Client API

The tables below describe the Client API. The class EddsService is a façade to the application. The methods available in the API match exactly those listed in Section 9.4.

12.1.1 Instantiating

Operation	Returns	Description
EddsService(String serverEndPoint)	EddsService	Creates an instance
setEndPointAddress(String address)	void	Sets the end-point to the address specified
connectToServer()	void	Initialises the service or reconnects to the server on a previously set address
setNotificationManager(NotificationManager notificationManager)	void	Sets an instance of the notification manager to be used with this EddsService

12.1.2 Session Management

Operation	**Returns**	**Description**
logIn(String userName, String password)	String	Creates session with server, also reconnects NotificationManager. Returns session ID.
hasSessionExpired()	boolean	Checks if the user's session has expired, or has not been created
checkUserPassword(String password)	void	Check if the password supplied matches the user's current password.
logOut()	void	Log out the user and close the session.

12.1.3 Issuing Requests

Operation	**Returns**	**Description**
batchRequest(RequestMessagePart parameters)	JobListIdPart	Submit a batch request.
streamRequest(StreamRequestMessagePart parameters)	JobIdPart	Perform a stream request.

12.1.4 Managing Requests

Operation	**Returns**	**Description**
cancel(CancelPartList cancelJobIds)	AcknowledgementPartList	Sends a Cancel request
reSubmit(JobIdPart job)	JobListIdPart	Resubmits the given request as a new request.
suspend(SuspendPartList suspendJobIds)	AcknowledgementPartList	Suspend request(s)
resume(ResumePartList resumeJobIds)	AcknowledgementPartList	Resume request(s)
deleteData(JobIdPartList requestIds)	AcknowledgementPartList	Delete response data
deleteRequest(JobIdPartList requestIds)	AcknowledgementPartList	Delete request(s) and their response data
startStreamData(JobIdPart jobIdPart)	void	Start the stream from the web server to the client using CometD for the specified active stream request
stopStreamData(JobIdPart jobIdPart)	void	Stop the stream from the web server to the client using CometD for the specified active stream request

12.1.5 Special Requests

Operation	Returns	Description
*getParamDefinitions(String mission)	JobIdPart	Initiates the retrieval of parameter definitions for the specified mission.
getDataspaces(String mission, String domain, DataSource dataSource)	List<String>	Gets the available dataspaces from the EDDS server for the specified mission, domain (optional) and data source.
*getFarcCatalogue(String mission, String domain, String folder)	JobIdPart	Initiates the retrieval of a FARC Catalogue for the specified mission, domain and folder.
*getFsCatalogue(String mission, String folder)	String - the request ID	Initiates the retrieval of a File System Catalogue for the specified mission and folder
*getHistoricalLog(Date from, Date to)	JobIdPart	Return a list of historical log messages between the from and to date.
stopHistoricalLogRetrieval(JobIdPart requestId)	void	Stops the retrieval of historical log messages identified by the requestId.
getLastConsolidation(String mission, DataSource dataSource, String dataspace)	long	Gets the last consolidation time for the specified mission, data source and dataspace.
getAlias(String mission)	List<AliasType>	Get aliases definition for the specified mission.

* The response to this request is delivered through asynchronous notification to the push notifications client. See section 12.2.

12.1.6 Utility Methods for Requests

Operation	Returns	Description
getResponse(DownloadList downloadList)	ResponseData	Gets the response file(s) from EDDS File server.
getJobs(XMLGregorianCalendar start, XMLGregorianCalendar end)	JobListIdPart	Gets the jobs in the given time windows.
getJobsWithStatus(XMLGregorianCalendar start, XMLGregorianCalendar end)	JobIdPart	Gets the status of jobs in the given time windows.
getJobsWithStatusAndLimit(JobsFilter filter)	JobIdPart	Gets the status of jobs in the given time windows, limiting the number of rows returned by the limit specified.
getStatus(JobIdPart job)	AcknowledgementPart	Gets the acknowledgement.
getStatuses(JobListIdPart jobs)	JobIdPart	Gets the statuses of multiple jobs.
getRequest(JobIdPart job)	RequestMessagePart	Return the specified RequestMessagePart
getRequest(String job)	RequestMessagePart	Return the specified RequestMessagePart
getStreamRequest(JobIdPart job)	StreamRequestMessagePart	Return the specified StreamRequestMessagePart

12.1.7 Utility Methods for User Management

Operation	Returns	Description
getAllowedRequestTypes()	List <DataSourceAccess>	Gets the list of allowed request types with data sources that the user can submit
getMissions()	MissionsList	Gets all the mission names.
getMissionsAndDomains()	MissionsDomainList	Gets all the mission names and domains.

Operation	Returns	Description
getMissionDetails(String mission)	MissionDetailsList	Gets all the details for the specified mission.
getRoles(String missionName)	RolesList	Gets all the roles for the specified mission
getUsers(boolean includeAll)	UsersList	Returns a the list of EDDS users
getUserAccountDetails(String username)	UserAccountDetails	Returns the full account details of the specified user
getUserQuota(String mission)	UserQuotaDetailsList	Gets the current user quota usage for the specified mission.
hasPermissionToSeeAllUsers(String username)	boolean	Checks if user has permission to see other users data.
getQuotaDetails(String user, String mission)	UserQuotaDetailsList	Gets the quota usage for the specified user and mission.
getTransformations(String mission, DataAccessDataElement type)	List<String>	Retrieves the list of available transformations for given mission and request type
accountRequest(AccountRequestMessagePart parameters)	void	Performs the specified changes in LDAP according to the contents of the AccountRequestMessagePart

12.1.8 Edds Client API – code examples

This section provides some snippets of code in order to facilitate the developer in the usage of the library. The code is quite minimalistic since all the complexity of the SOAP protocol is hidden by the library. More examples can be found in the source code under edds-ws-client/src/test/java under esa.egos.edds.ws.client.examples.

12.1.8.1 LogIn

```
    /**
     * Perform the login of a user and open a session on the web server.
     *
     * @param username The username
     * @param password The password
     * @throws WebServiceException
     * @throws MalformedURLException
     * @throws PasswordExpiredFault
     */
    public void logIn(String username, String password) throws
MalformedURLException, WebServiceException, PasswordExpiredFault
    {
        String serverAddress =
"http://10.48.29.159:8080/edds/EddsService?wsdl";
        eddsService = new EddsService(serverAddress);
        // These two lines are necessary only if notifications are used
        notificationManager = new NotificationManager(eddsService,
serverAddress);
        eddsService.setNotificationManager(notificationManager);
        // Initialize the connection to the web service
        eddsService.connectToServer();

        try
        {
            eddsService.logIn(username, password);
        }
        catch (AuthenticationFault ex)
        {
            // exception management
```



```
        }
    }
```

12.1.8.2 LogOut

```
    /**
     * Close the session of the user
     *
     * @param username The username
     * @param password The password
     * @throws WebServiceException
     * @throws MalformedURLException
     */
    public void logOut(String username, String password) throws
MalformedURLException, WebServiceException
    {
        // we assume the login has been already performed
        eddsService.logOut();
    }
```

12.1.8.3 GetStatus

```
    /**
     * Return the status of a request
     *
     * @param jobIdPart The ID of the reqeust
     * @throws WebServiceException
     * @throws MalformedURLException
     */
    public void getStatus(JobIdPart jobIdPart) throws
MalformedURLException, WebServiceException
    {
        try
        {
            // we assume the login has been already performed

            // we have as return the complete acknowledgement of the
request
            AcknowledgementPart acknowledgment =
eddsService.getStatus(jobIdPart);

            // ...
        }
        catch (RequestFault ex)
        {
            // exception management
        }
        catch (SessionFault ex)
        {
            // exception management
        }
        catch (AuthorizationFault ex)
        {
            // exception management
        }
    }
```

12.1.8.4 Get request statuses

```
   /**
```



```
     * Return the request information of multiple requests
     *
     * @param jobListIdPart The IDs of the requests
     * @throws WebServiceException
     * @throws DatatypeConfigurationException
     * @throws ClassNotFoundException
     * @throws IOException
     * @throws TimeFormatException
     */
    public void getJobsWithStatuses(JobListIdPart jobListIdPart) throws
WebServiceException, DatatypeConfigurationException, IOException,
            ClassNotFoundException, TimeFormatException
    {
        try
        {
            // we assume the login has been already performed

            // Create a start time
            Date startTime = new Date(System.currentTimeMillis());
            GregorianCalendar startTimeGregorian = new
GregorianCalendar();
            startTimeGregorian.setTime(startTime);
            XMLGregorianCalendar startTimeXml =
DatatypeFactory.newInstance().newXMLGregorianCalendar(startTimeGregorian)
;

            // Create an end time
            Date endTime = new Date(System.currentTimeMillis());
            GregorianCalendar endTimeGregorian = new GregorianCalendar();
            endTimeGregorian.setTime(endTime);
            XMLGregorianCalendar endTimeXml =
DatatypeFactory.newInstance().newXMLGregorianCalendar(endTimeGregorian);

            // we have as return a job ID which can be used to identify
the response from CometD. The JobID is a random string, not a RequestId
            JobIdPart jobId = eddsService.getJobsWithStatus(startTimeXml,
endTimeXml);
        }
        catch (RequestFault ex)
        {
            // exception management
        }
        catch (SessionFault ex)
        {
            // exception management
        }
        catch (AuthorizationFault ex)
        {
            // exception management
        }
    }
```

12.1.8.5 Perform BatchRequest

```
    /**
     * Perform a batch request
     *
     * @param request The request message
     * @throws WebServiceException
     * @throws MalformedURLException
     */
```



```
    public void batchReqeust(RequestMessagePart request) throws
MalformedURLException, WebServiceException
    {
        try
        {
            // we assume the login has been already performed

            // the request is submitted and we receive the related ID
            JobListIdPart jobId = eddsService.batchRequest(request);

            // ...
        }
        catch (RequestFault ex)
        {
            // exception management
        }
        catch (SessionFault ex)
        {
            // exception management
        }
        catch (AuthorizationFault ex)
        {
            // exception management
        }
    }
```

12.1.8.6 Perform StreamRequest

```
    /**
     * Perform a stream request
     *
     * @param request The request message
     * @throws WebServiceException
     * @throws MalformedURLException
     */
    public void streamRequest(StreamRequestMessagePart request) throws
MalformedURLException, WebServiceException
    {
        try
        {
            // we assume the login has been already performed

            // the request is submitted and we receive the related ID
            JobListIdPart jobId = eddsService.streamRequest(request);

            // ...
        }
        catch (RequestFault ex)
        {
            // exception management
        }
        catch (SessionFault ex)
        {
            // exception management
        }
        catch (AuthorizationFault ex)
        {
            // exception management
        }
    }
```


To receive the stream data, follow the example in Section 12.2.2, using `NotificationChannel.PARAM_STREAM` as the channel to listen to, and implement `NotificationListener <DarcParamRec>`. Next call the method `startStreamData(requestId)` in the Client API to request that the Web Server starts to send the stream data to your application. When finished, ensure you call `stopStreamData(requestId)`. The RequestId to pass is the one returned to you from the above method - `streamRequest(request)`.

12.1.8.7 Cancel

```
    /**
     * Cancels request(s)
     *
     * @param jobIds The request IDs
     * @throws WebServiceException
     * @throws MalformedURLException
     */
    public void cancel(CancelPartList jobIds) throws
MalformedURLException, WebServiceException
    {
        try
        {
            // we assume the login has been already performed


            // we have as return the complete acknowledgement of the
request(s)
            AcknowledgementPartList acknowledgements =
eddsService.cancel(jobIds);

            // ...
        }
        catch (RequestFault ex)
        {
            // exception management
        }
        catch (SessionFault ex)
        {
            // exception management
        }
        catch (AuthorizationFault ex)
        {
            // exception management
        }
    }
```

12.1.8.8 Suspend

```
   /**
     * Suspends a request(s)
     *
     * @param jobIds The request IDs
     * @throws WebServiceException
     * @throws MalformedURLException
     */
    public void suspend(SuspendPartList jobIds) throws
MalformedURLException, WebServiceException
    {
        try
        {
            // we assume the login has been already performed
```



```
            // we have as return the complete acknowledgement of the
request(s)
            AcknowledgementPartList acknowledgements =
eddsService.suspend(jobIds);

            // ...
        }
        catch (RequestFault ex)
        {
            // exception management
        }
        catch (SessionFault ex)
        {
            // exception management
        }
        catch (AuthorizationFault ex)
        {
            // exception management
        }
    }
```

12.1.8.9 Resume

```
/**
     * Resumes a request(s)
     *
     * @param jobIds The request IDs
     * @throws WebServiceException
     * @throws MalformedURLException
     */
    public void resume(ResumePartList jobIds) throws
MalformedURLException, WebServiceException
    {
        try
        {
            // we assume the login has been already performed

            // we have as return the complete acknowledgement of the
request(s)
            AcknowledgementPartList acknowledgements =
eddsService.resume(jobIds);

            // ...
        }
        catch (RequestFault ex)
        {
            // exception management
        }
        catch (SessionFault ex)
        {
            // exception management
        }
        catch (AuthorizationFault ex)
        {
            // exception management
        }
    }
```


12.1.8.10 Get parameter definitions

```
    public void getParamDefinitions(String mission)
    {
        try
        {
            // we assume the login has been already performed

            JobIdPart jobId = eddsService.getParamDefinitions(mission);

            // ...
        }
        catch (RequestFault e)
        {
            // exception management
        }
        catch (SessionFault e)
        {
            // exception management
        }
        catch (AuthorizationFault e)
        {
            // exception management
        }
    }
```

12.1.8.11 Download response file

```
    /**
     * Save the response files for the given jobId locally
     *
     * @param jobId the request id
     */
    public void saveResponseFiles(String jobId) throws
WebServiceException, RequestFault, SessionFault, AuthorizationFault,
            FileNotFoundException, IOException
    {
        DownloadList downloadList = new DownloadList();
        downloadList.setJob(jobId);
        List <ResponseDataElement> dataHandlers = eddsService.
getResponse(downloadList).getResponseDataElement();

        // Save the response files
        for (ResponseDataElement responseElem : dataHandlers)
        {
            BufferedOutputStream bos = new BufferedOutputStream(new
FileOutputStream(new File(responseElem.getFileName())));
            responseElem.getResponse().writeTo(bos);
            bos.close();
        }
   }
```


12.2 EDDS Push notifications client

EDDS clients can register their listeners to receive push notification on messages from EDDS system. The client listeners are called asynchronously when any messages are received. Supported notification types are:

Channel (from enum NotificationChannel)	Return Type	Description
USER_MANAGEMENT	AccountRequestMessagePart	User management updates
STATUS_UPDATES	RequestInfoRec	Job updates
STATUS_RETRIEVAL	RequestInfoRecs	Used to get a list of jobs and their status
STATUS_RETRIEVAL_ERRORS	String	Error message if unable to retrieve status information
EDDS_LOGS	LoggingEvent	EDDS live system log messages
PARAM_DEFINITIONS	ParamDefRecs	Parameter definitions in response to the requests through the web service.
PARAM_DEFINITIONS_ERRORS	String	Parameter definition errors
ARCHIVE_CATALOGUE		The FARC catalogue entries
ARCHIVE_CATALOGUE_ERRORS	String	Any error information when retrieving the catalogue
ARCHIVE_UPDATES	FarcCatRec	Information about a new commit in the FARC
HISTORICAL_LOGS	SystemLogRec	Historical logs in response to the requests through the web service.
USER_QUOTA_USAGE	UserQuotaDetailsList	Quota updates
PARAM_STREAM	DarcParamRec	DARC parameter stream
PKT_TC_STREAM	PacketTCBinary	Telecommand stream
PKT_TM_STREAM	PacketTMBinary	Telemetry stream
PKT_EV_STREAM	EventRecordReportBinary	SCOS Event Log stream
PKT_OOL_STREAM	OOLReportBinary	Out of limits stream
FILE_SYSTEM_CATALOGUE	Catalogue	The File System Catalogue entries in response to the Special request for File System
FILE_SYSTEM_UPDATES	FsCatalogueEntry	Notifications about changes in designated File System directory.

12.2.1 Security

The same session key that is obtained during login of EDDS Web services is used to authenticate the client. If client fails to provide the server with an existing session key, the service is denied.

12.2.2 Example: Register notifications listener

```
/**
```



```
     * Register your listener to receive asynchronous notifications for
status updates
     *
     * @param listener the listener the will receive updates when they
are received from the EDDS Web Server
     * @see NotificationManager#registerListener(NotificationChannel,
NotificationListener) for the list supported types
     */
    public void registerStatusUpdatesListener(NotificationListener
<RequestInfoRec> listener)
    {
      notificationManager.
registerListener(NotificationChannel.STATUS_UPDATES, listener);
    }
```

12.3 EDDS Utility methods

12.3.1 Decoding EDDS Param Binary format

EDDS Client API provides utility functions to read and write the TM Parameter binary format. The implementation is in package esa.egos.edds.ws.client.binary. For more detailed documentation read the Javadoc of the provided classes.

12.3.1.1 Example: Reading Parameter TM data from EDDS Binary file

```
    /**
     * Read the Parameter BINARY response file. It contains a list of {@link
DarcParameterBinary} objects
     * encoded using Google Protocol Buffers. Each record is prepended with
it's length.
     * @param file the file object of the file being read.
     * @throws InvalidProtocolBufferException when something unexpected occurs
while reading the data
     * @throws IOException problem reading the file
     */
    public void readParamBinaryFormat(File file) throws
InvalidProtocolBufferException, IOException{
        DarcParameterBinaryInputSupport bin = null;
        try
        {
            bin = new DarcParameterBinaryInputSupport(file);
            DarcParameterBinary rec;
            while ( (rec = bin.next()) != null)
            {
                System.out.println(rec.getParamName() + " " + rec.getGenTime()
+ ": " + rec.getValue());
            }
        }
        finally
        {
            if (bin != null)
            {
                bin.close();
            }
        }
```


12.4 Dependencies

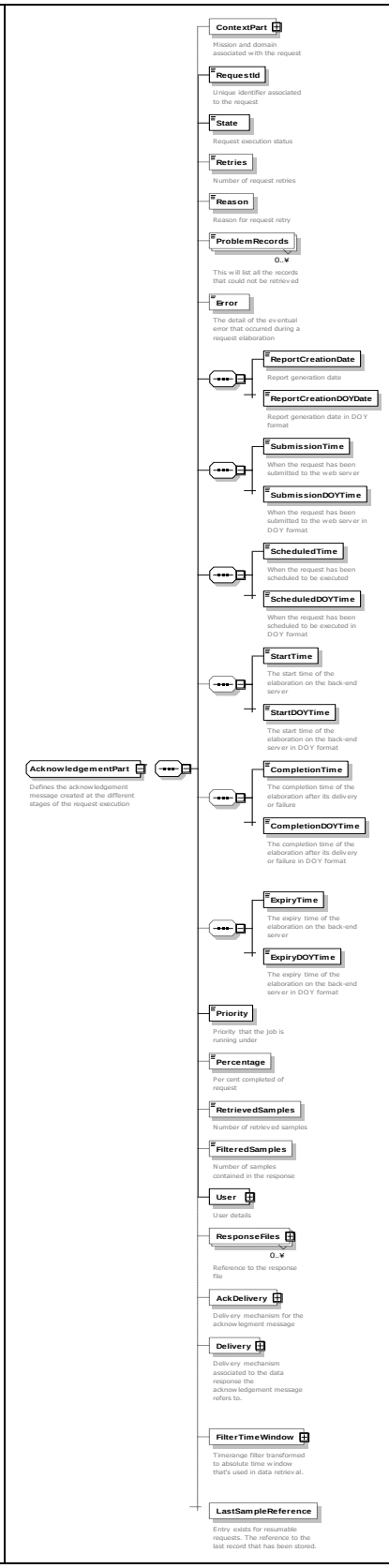
The EDDS Client API can be used by including the respective Maven artifact to the project. Maven then will resolve the dependencies and add all required libraries to the classpath. To use the EDDS Client API add the following Maven artifact to the project:

```
<dependency>
	<groupId>esa.egos.edds</groupId>
	<artifactId>edds-ws-client</artifactId>
	<version>2.3.0</version>
</dependency>
```


Appendix A Acknowledgement Data Type

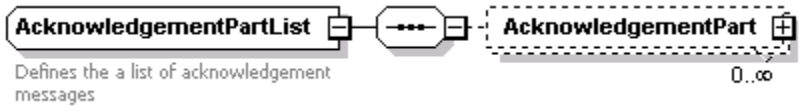
A.1 complexType AcknowledgementPart

diagram

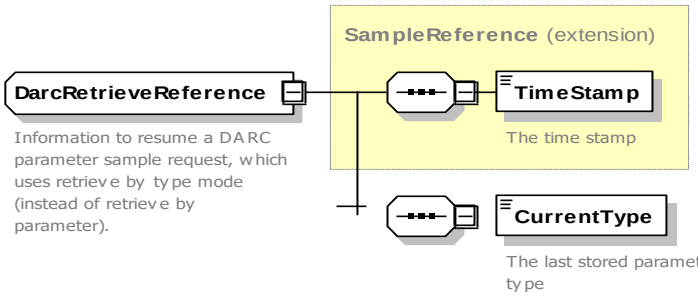


namespace	http://edds.egos.esa/model
children	**ContextPart RequestId State Retries Reason ProblemRecords Error ReportCreationDate ReportCreationDOYDate SubmissionTime SubmissionDOYTime ScheduledTime ScheduledDOYTime StartTime StartDOYTime CompletionTime CompletionDOYTime ExpiryTime ExpiryDOYTime Priority Percentage RetrievedSamples FilteredSamples User ResponseFiles AckDelivery Delivery FilterTimeWindow LastSampleReference**
used by	element **AcknowledgementPartList/AcknowledgementPart**
annotation	documentation Defines the acknowledgement message created at the different stages of the request execution

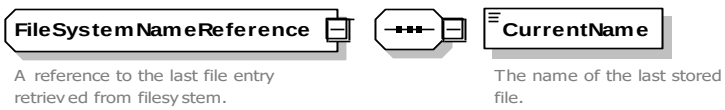
A.2 complexType AcknowledgementPartList

diagram	
namespace	http://edds.egos.esa/model
children	**AcknowledgementPart**
annotation	documentation Defines the a list of acknowledgement messages

A.3 complexType DarcRetrieveReference

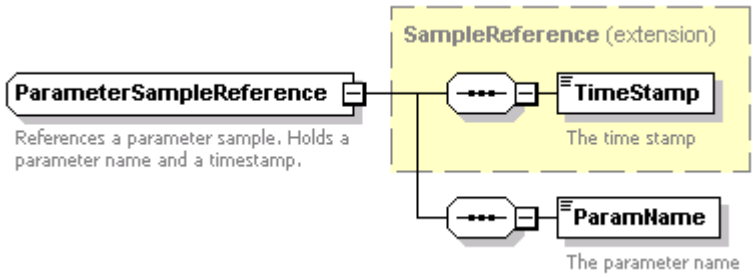
diagram	
namespace	http://edds.egos.esa/model
type	extension of **SampleReference**
properties	base SampleReference
children	**TimeStamp CurrentType**
annotation	documentation Information to resume a DARC parameter sample request, which uses retrieve by type mode (instead of retrieve by parameter).

A.4 complexType FileSystemNameReference

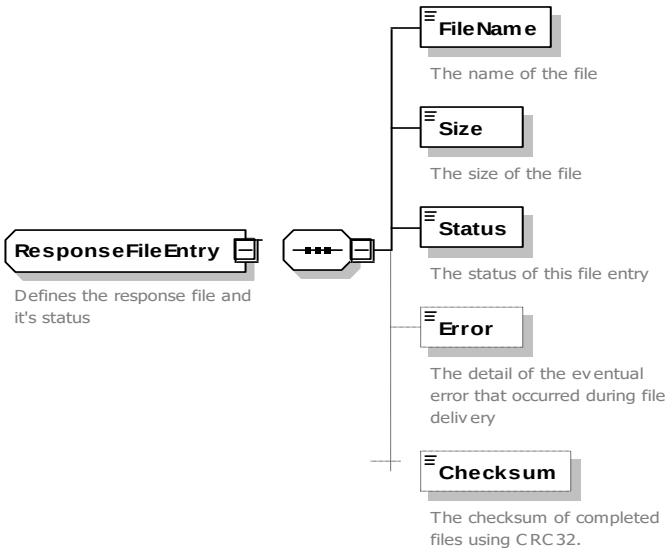
diagram	
namespace	http://edds.egos.esa/model
type	extension of **RetrievalReference**
properties	base RetrievalReference
children	**CurrentName**

annotation	documentation A reference to the last file entry retrieved from filesystem.

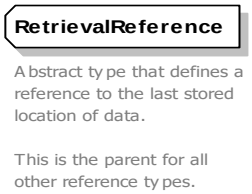
A.5 complexType ParameterSampleReference

diagram	
namespace	http://edds.egos.esa/model
type	extension of **SampleReference**
properties	base SampleReference
children	**TimeStamp ParamName**
annotation	documentation References a parameter sample. Holds a parameter name and a timestamp.

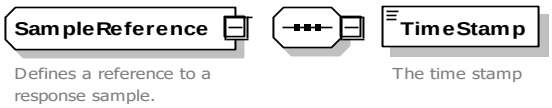
A.6 complexType ResponseFileEntry

diagram	
namespace	http://edds.egos.esa/model
children	**FileName Size Status Error Checksum**
used by	element **AcknowledgementPart/ResponseFiles**
annotation	documentation Defines the response file and it's status

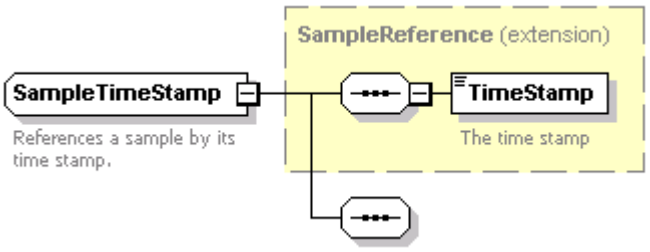
A.7 complexType RetrievalReference

diagram	
namespace	http://edds.egos.esa/model
properties	abstract true
used by	element **AcknowledgementPart/LastSampleReference** complexTypes FileSystemNameReference SampleReference
annotation	documentation Abstract type that defines a reference to the last stored location of data. This is the parent for all other reference types.

A.8 complexType SampleReference

diagram	
namespace	http://edds.egos.esa/model
type	extension of RetrievalReference
properties	base RetrievalReference
children	**TimeStamp**
used by	complexTypes DarcRetrieveReference ParameterSampleReference SampleTimeStamp
annotation	documentation Defines a reference to a response sample.

A.9 complexType SampleTimeStamp

diagram	
namespace	http://edds.egos.esa/model
type	extension of **SampleReference**
properties	base SampleReference
children	**TimeStamp**
annotation	documentation References a sample by its time stamp.

A.10 simpleType ResponseFileStatus

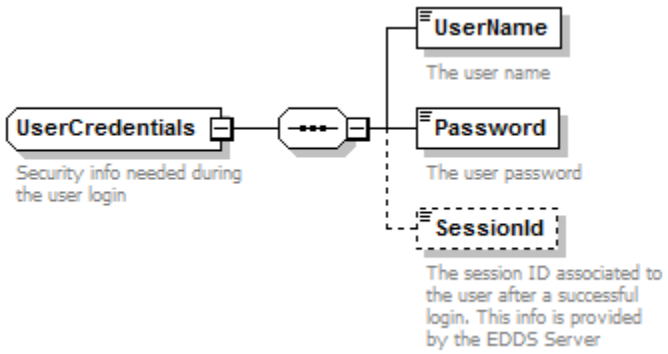
namespace	http://edds.egos.esa/model

type	restriction of **xs:string**
properties	base xs:string
used by	element **ResponseFileEntry/Status**
facets	Kind / Value / annotation enumeration ACTIVE — documentation: The file is being written by the EDDS Server enumeration SERVER_COMPLETED — documentation: EDDS Server completed writing the file, delivery-manager needs to deliver the file enumeration DELIVERED_LOCAL — documentation: The file is available in the EDDS Server FTP directory enumeration DELIVERED_LOCAL_REMOTE — documentation: The file is delivered to the remote server and is also available in the EDDS Server FTP directory enumeration DELIVERED_REMOTE — documentation: The delivery of the file is completed enumeration DELETED — documentation: The data on EDDS file server is deleted. Can be reached after DELIVERED_LOCALLY or DELIVERED_LOCAL_REMOTE enumeration ERROR — documentation: The file cannot be saved on EDDS file server
annotation	documentation Response file statuses

Appendix B Authentication Data Type

This section of the document describes the complex types used for authentication as defined in the authentication.xsd file.

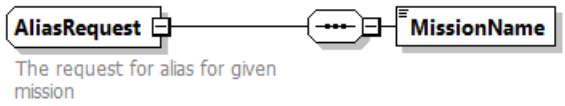
B.1 complexType UserCredentials

diagram	
namespace	http://edds.egos.esa/model
children	**UserName Password SessionId**
used by	element **UserCredentials**
annotation	documentation Security info needed during the user login

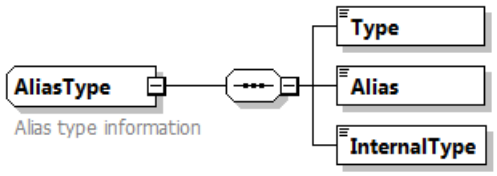
Appendix C Batch Request Data Type

This section of the document describes the complex types used for batch requests as defined in the batchrequest.xsd file.

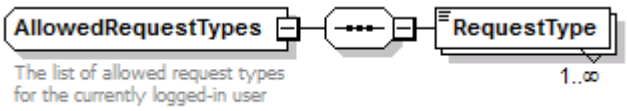
C.1 complexType AliasRequest

diagram	
namespace	http://edds.egos.esa/model
children	**string**
used by	
annotation	documentation The request for alias for given mission

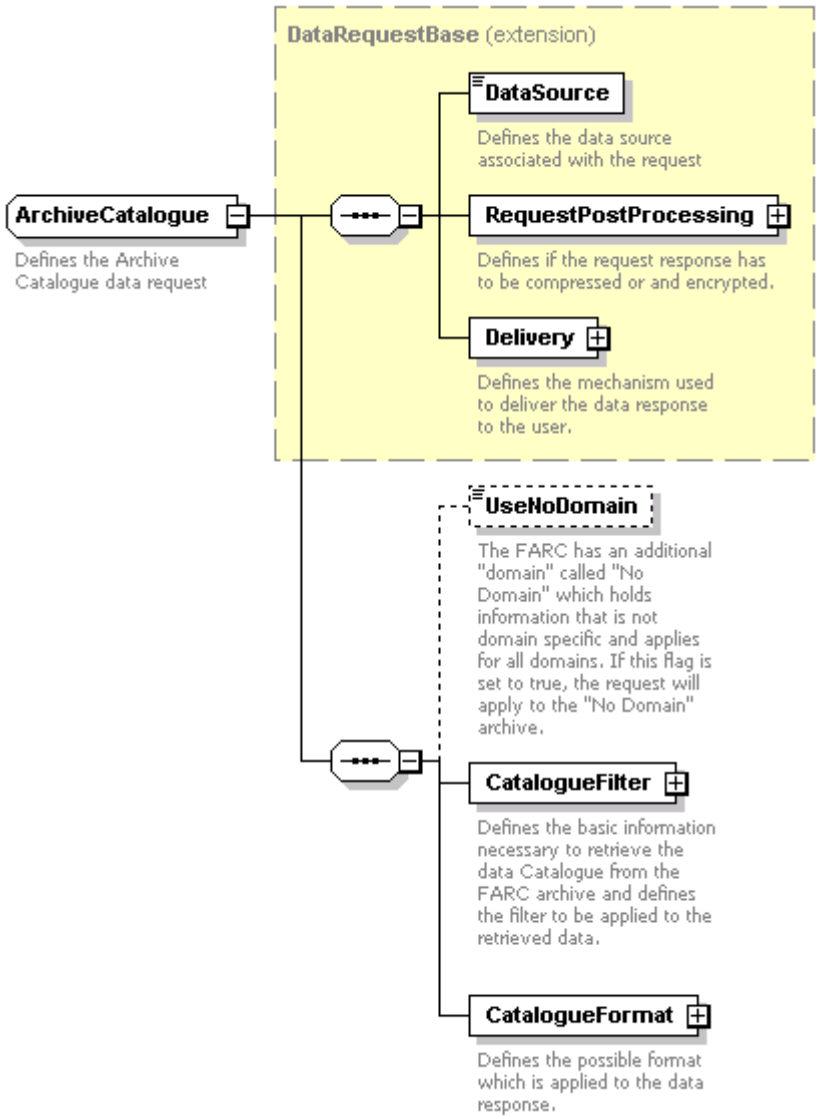
C.2 complexType AliasType

diagram	
namespace	http://edds.egos.esa/model
children	**string string AliasInternalType**
used by	**ListAliasInfo**
annotation	documentation Alias type information

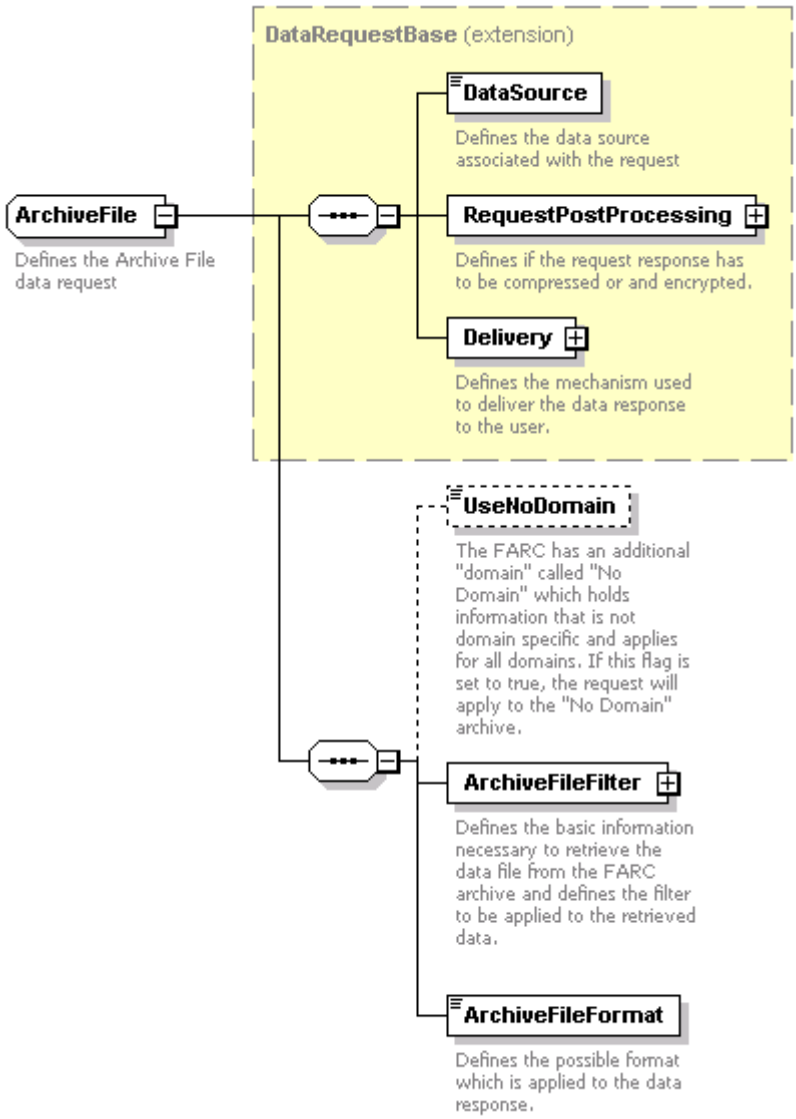
C.3 complexType AllowedRequestTypes

diagram	
namespace	http://edds.egos.esa/model
children	RequestType
used by	element AllowedRequestTypes
annotation	documentation The list of allowed request types for the currently logged-in user

C.4 complexType ArchiveCatalogue

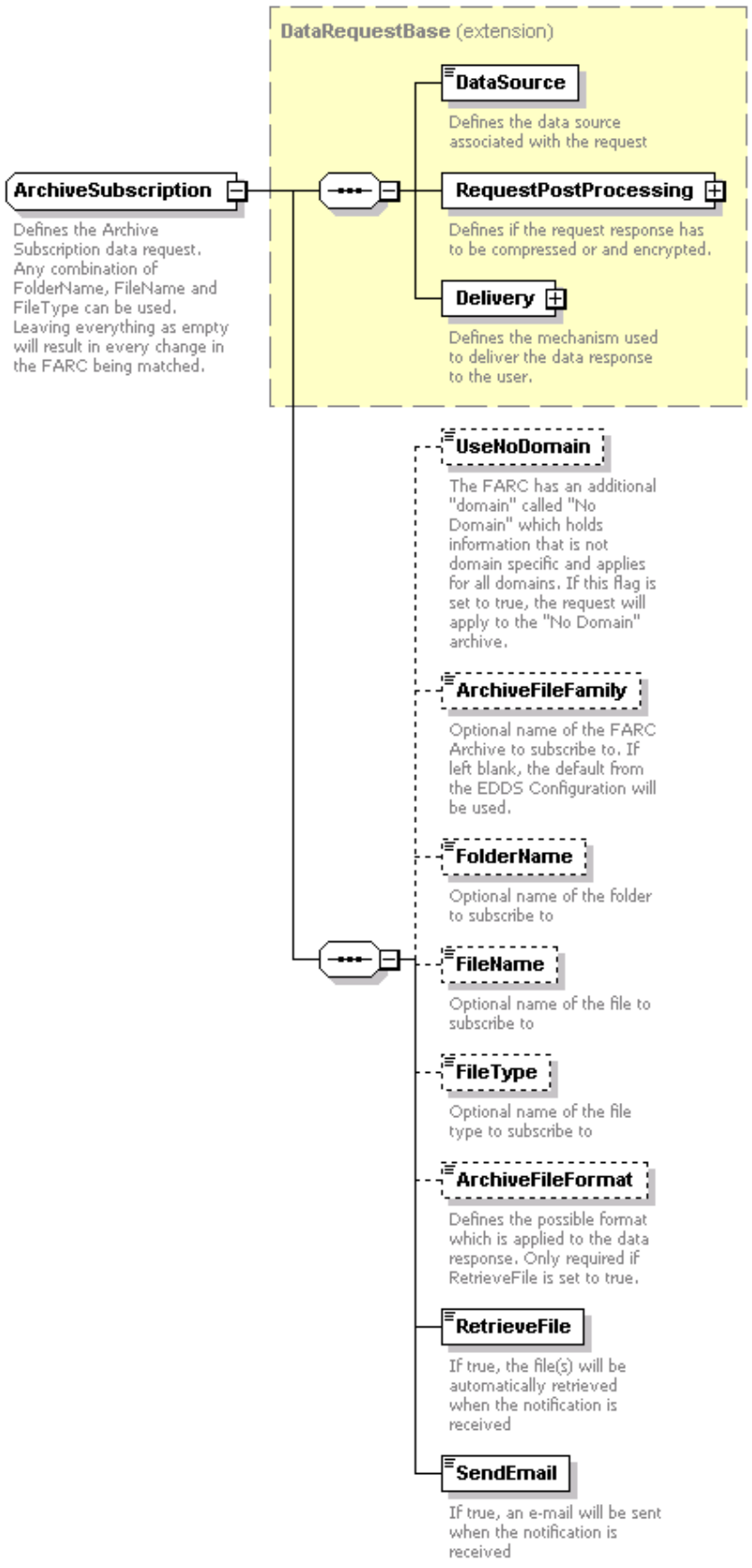
<table>
<tr><td>diagram</td><td> 
</td></tr>
<tr><td>namespace</td><td>http://edds.egos.esa/model</td></tr>
<tr><td>type</td><td>extension of DataRequestBase</td></tr>
<tr><td>properties</td><td>base DataRequestBase</td></tr>
<tr><td>children</td><td>RequestPostProcessing Delivery UseNoDomain CatalogueFilter CatalogueFormat</td></tr>
<tr><td>used by</td><td>elements ArchiveCatalogue DataRequest/ArchiveCatalogue</td></tr>
<tr><td>annotation</td><td>documentation
Defines the Archive Catalogue data request</td></tr>
</table>

C.5 complexType ArchiveFile

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**RequestPostProcessing Delivery UseNoDomain ArchiveFileFilter ArchiveFileFormat**
used by	elements **ArchiveFile DataRequest/ArchiveFile**
annotation	documentation Defines the Archive File data request

C.6 complexType ArchiveSubscription

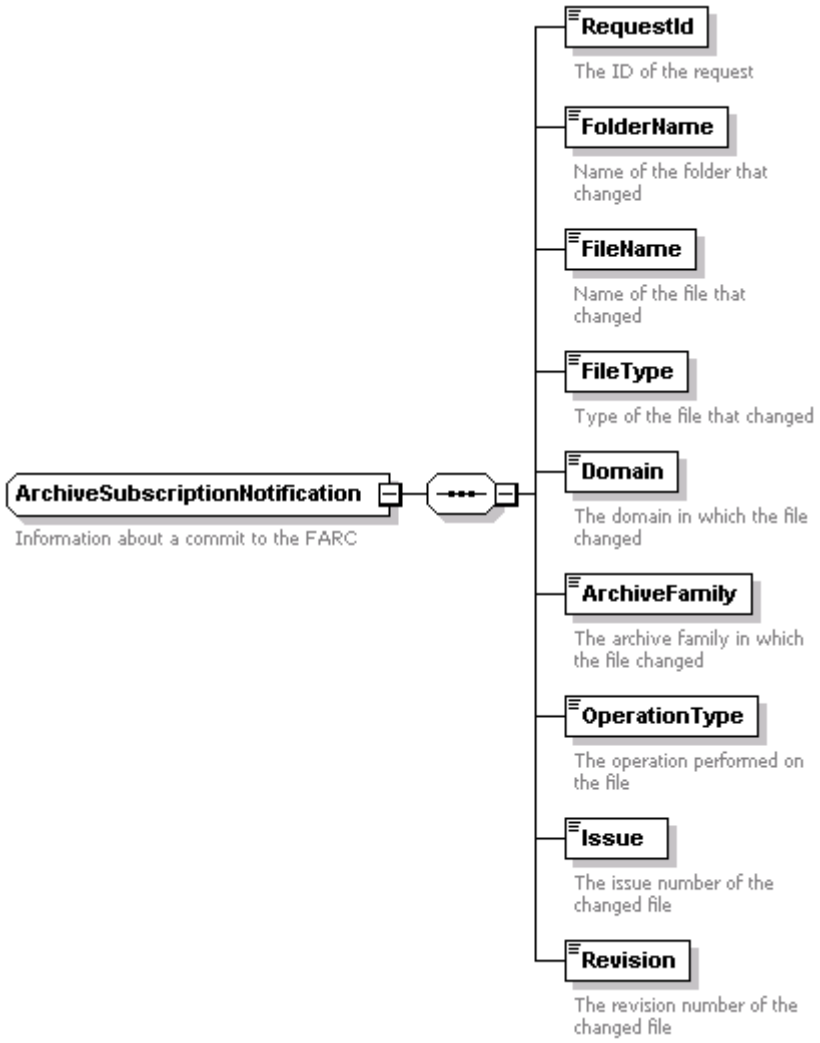
diagram



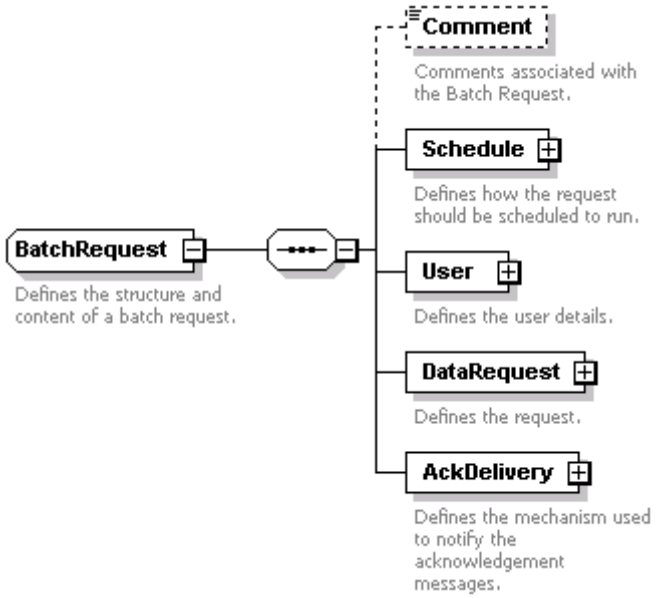
namespace	http://edds.egos.esa/model

type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**RequestPostProcessing Delivery UseNoDomain ArchiveFileFamily FolderName FileName FileType ArchiveFileFormat RetrieveFile SendEmail**
used by	elements **ArchiveSubscription DataRequest/ArchiveSubscription**
annotation	documentation Defines the Archive Subscription data request. Any combination of FolderName, FileName and FileType can be used. Leaving everything as empty will result in every change in the FARC being matched.

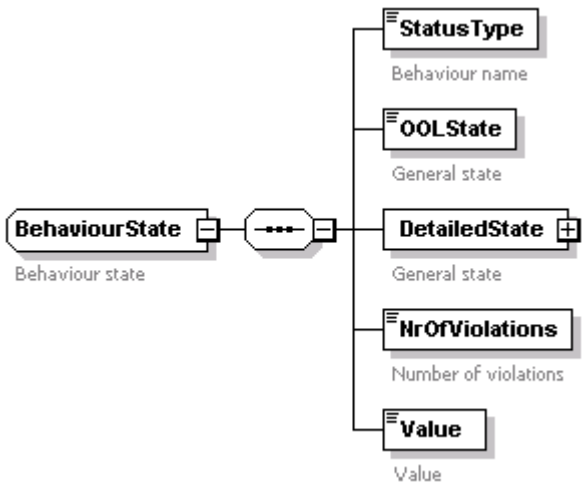
C.7 complexType ArchiveSubscriptionNotification

diagram	
namespace	http://edds.egos.esa/model
children	**RequestId FolderName FileName FileType Domain ArchiveFamily OperationType Issue Revision**
used by	element **ArchiveSubscriptionNotification**
annotation	documentation Information about a commit to the FARC

C.8 complexType BatchRequest

diagram	
namespace	http://edds.egos.esa/model
children	**Comment Schedule User DataRequest AckDelivery**
used by	elements **BatchRequest RequestMessagePart/BatchRequest**
annotation	documentation Defines the structure and content of a batch request.

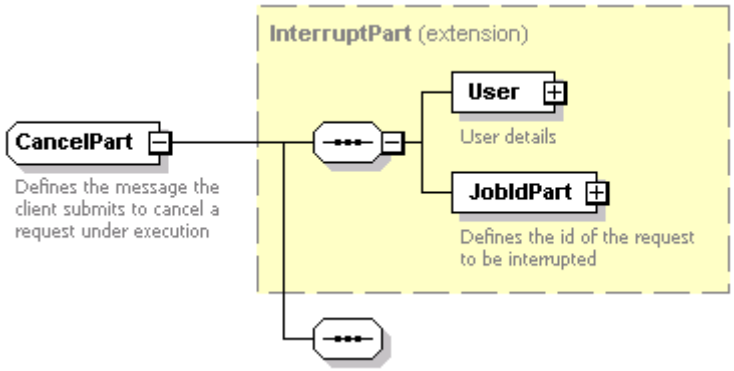
C.9 complexType BehaviourState

diagram	
namespace	http://edds.egos.esa/model
children	**StatusType OOLState DetailedState NrOfViolations Value**
used by	element **BehaviourStates/BehaviourState**
annotation	documentation Behaviour state

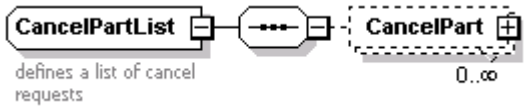
C.10 complexType BehaviourStates

diagram	
namespace	http://edds.egos.esa/model
children	**BehaviourState**
used by	element **ParamRepresentationSample/BehaviourStates**
annotation	documentation Behaviour states

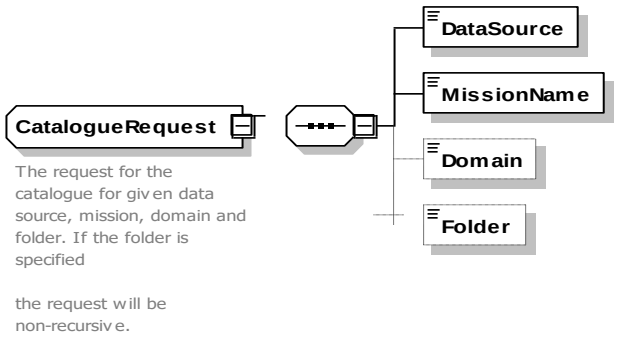
C.11 complexType CancelPart

diagram	
namespace	http://edds.egos.esa/model
type	extension of **InterruptPart**
properties	base InterruptPart
children	**User JobIdPart**
used by	element **CancelPart**
annotation	documentation Defines the message the client submits to cancel a request under execution

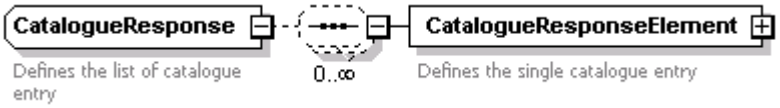
C.12 complexType CancelPartList

diagram	
namespace	http://edds.egos.esa/model
children	**CancelPart**
used by	element **CancelPartList**
annotation	documentation defines a list of cancel requests

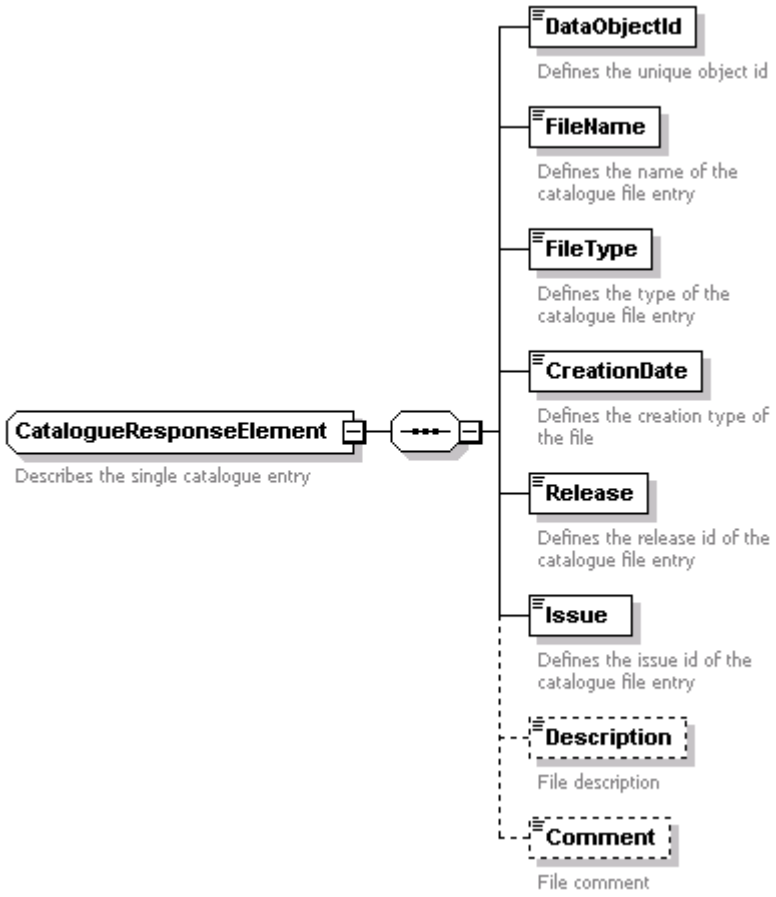
C.13 complexType CatalogueRequest

diagram	
namespace	http://edds.egos.esa/model
children	**DataSource MissionName Domain Folder**
used by	element **CatalogueRequest**
annotation	documentation The request for the catalogue for given data source, mission, domain and folder. If the folder is specified the request will be non-recursive.

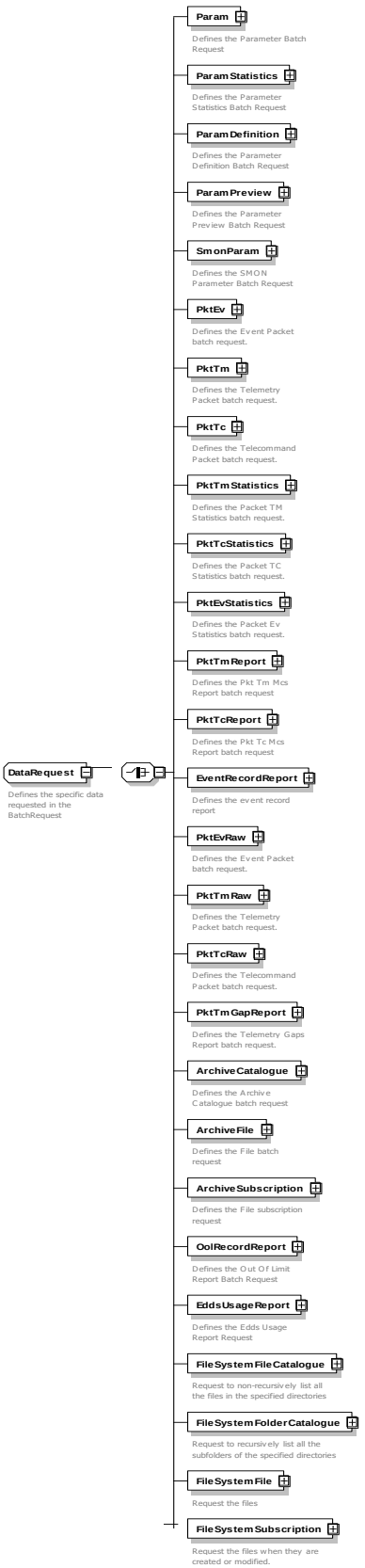
C.14 complexType CatalogueResponse

diagram	
namespace	http://edds.egos.esa/model
children	**CatalogueResponseElement**
used by	element **Response/CatalogueResponse**
annotation	documentation Defines the list of catalogue entry

C.15 complexType CatalogueResponseElement

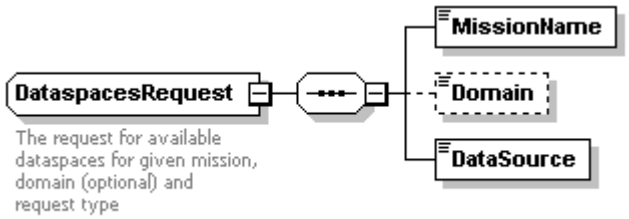
diagram	
namespace	http://edds.egos.esa/model
children	**DataObjectId FileName FileType CreationDate Release Issue Description Comment**
used by	element **CatalogueResponse/CatalogueResponseElement**
annotation	documentation Describes the single catalogue entry

C.16 complexType DataRequest

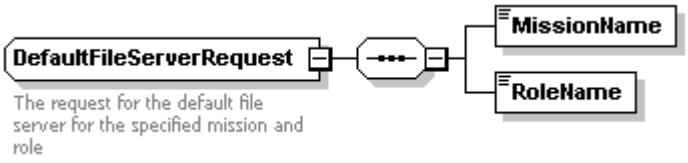
diagram	
namespace	http://edds.egos.esa/model
children	**Param ParamStatistics ParamDefinition ParamPreview SmonParam PktEv PktTm PktTc PktTmStatistics PktTcStatistics PktEvStatistics PktTmReport PktTcReport EventRecordReport PktEvRaw PktTmRaw PktTcRaw PktTmGapReport ArchiveCatalogue ArchiveFile ArchiveSubscription OolRecordReport EddsUsageReport FileSystemFileCatalogue FileSystemFolderCatalogue**

	FileSystemFile FileSystemSubscription
used by	elements **DataRequest BatchRequest/DataRequest**
annotation	documentation Defines the specific data requested in the BatchRequest

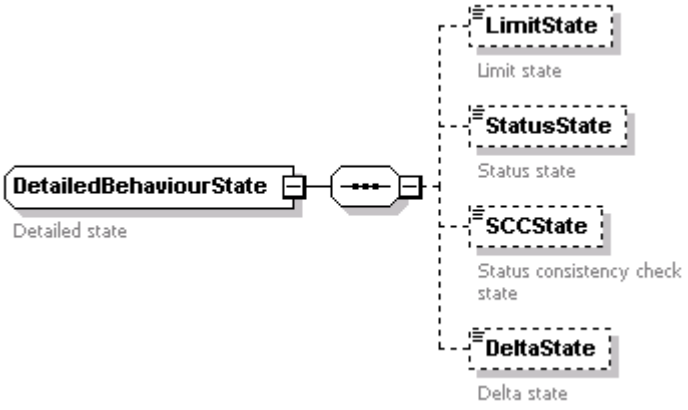
C.17 complexType DataspacesRequest

diagram	
namespace	http://edds.egos.esa/model
children	MissionName Domain DataSource
used by	element **DataspacesRequest**
annotation	documentation The request for available dataspaces for given mission, domain (optional) and request type

C.18 complexType DefaultFileServerRequest

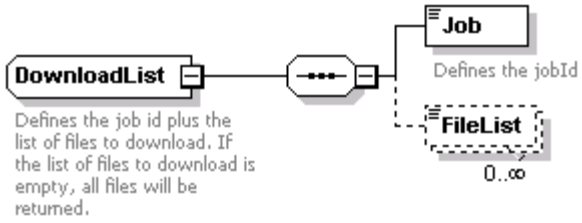
diagram	
namespace	http://edds.egos.esa/model
children	**MissionName RoleName**
used by	element **DefaultFileServerRequest**
annotation	documentation The request for the default file server for the specified mission and role

C.19 complexType DetailedBehaviourState

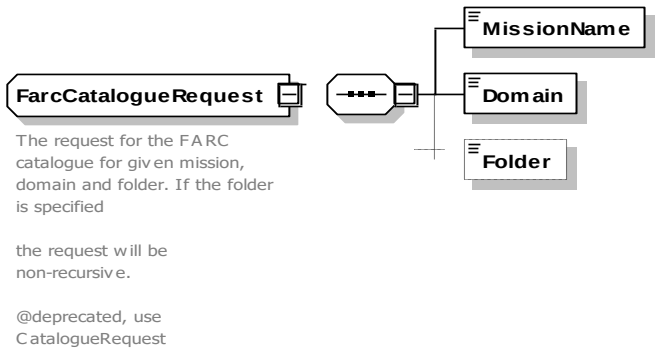
diagram	

namespace	http://edds.egos.esa/model
children	**LimitState StatusState SCCState DeltaState**
used by	element **BehaviourState/DetailedState**
annotation	documentation Detailed state

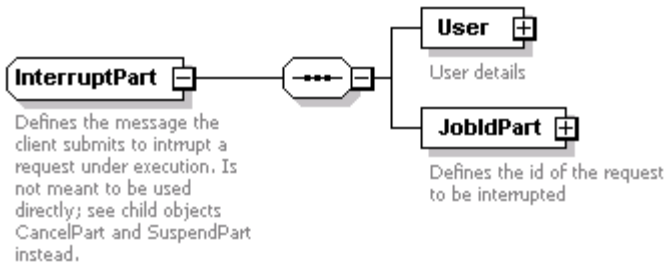
C.20 complexType DownloadList

diagram	
namespace	http://edds.egos.esa/model
children	**Job FileList**
used by	element **DownloadList**

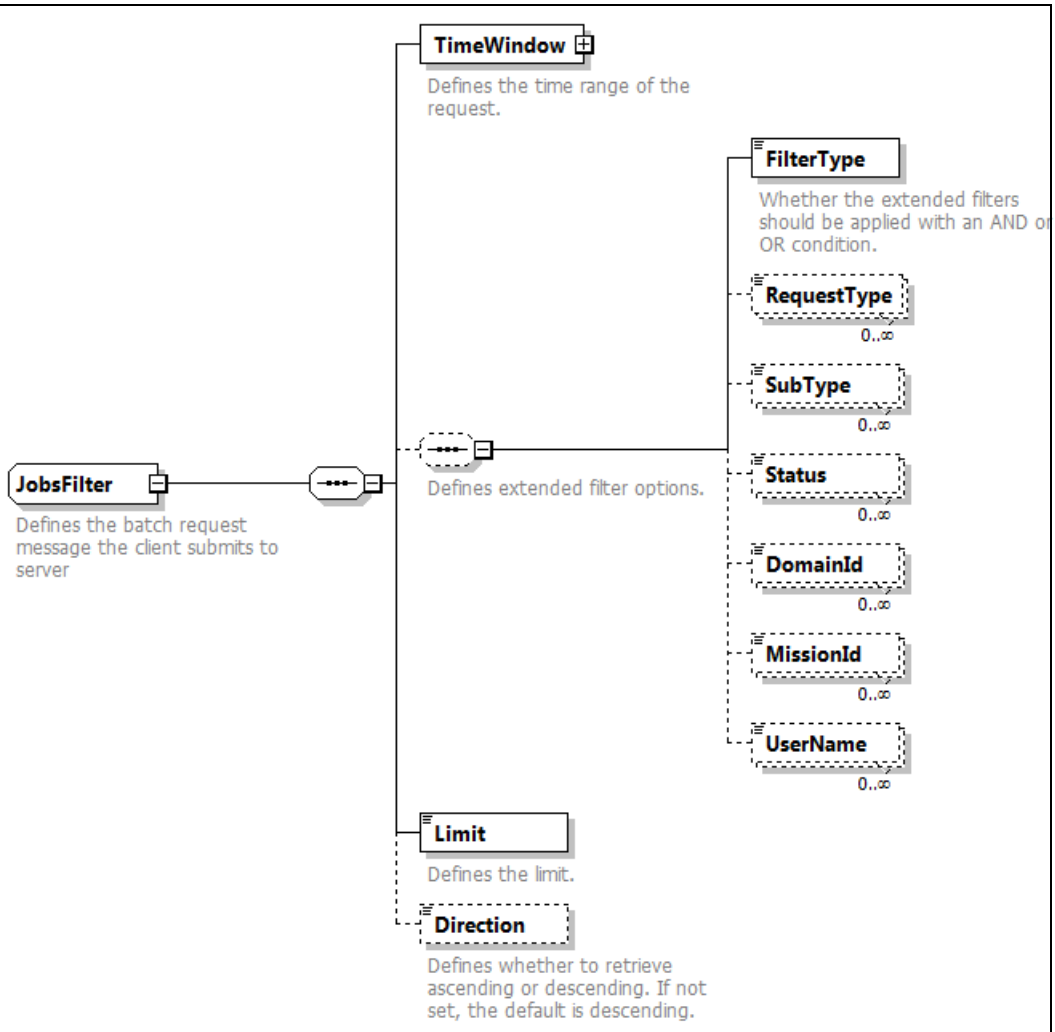
C.21 complexType FarcCatalogueRequest

diagram	
namespace	http://edds.egos.esa/model
children	**MissionName Domain Folder**
used by	element **FarcCatalogueRequest**
annotation	documentation The request for the FARC catalogue for given mission, domain and folder. If the folder is specified the request will be non-recursive. @deprecated, use CatalogueRequest

C.22 complexType InterruptPart

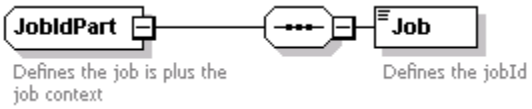
diagram	
namespace	http://edds.egos.esa/model
children	**User JobIdPart**
used by	complexTypes **CancelPart ResumePart SuspendPart**
annotation	documentation Defines the message the client submits to intrrupt a request under execution. Is not meant to be used directly; see child objects CancelPart and SuspendPart instead.

C.23 complexType JobsFilter

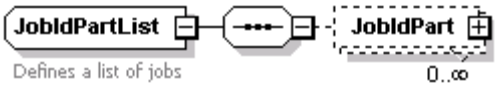
diagram	
namespace	http://edds.egos.esa/model
children	**TimeWindow Limit Direction**

used by	
annotation	documentation Defines the batch request message the client submits to server

C.24 complexType JobIdPart

diagram	
namespace	http://edds.egos.esa/model
children	**Job**
used by	elements **DeleteDataPart DeleteRequestPart JobIdPart CancelPart/JobIdPart**
annotation	documentation Defines the job is plus the job context

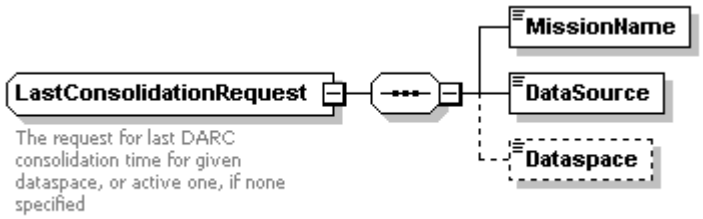
C.25 complexType JobIdPartList

diagram	
namespace	http://edds.egos.esa/model
children	**JobIdPart**
used by	elements **DeleteDataPart DeleteRequestPart JobIdPartList**
annotation	documentation Defines a list of jobs

C.26 complexType JobListIdPart

diagram	
namespace	http://edds.egos.esa/model
children	**JobListIdPartElement**
used by	element **JobListIdPart**
annotation	documentation JobId list

C.27 complexType LastConsolidationRequest

diagram	
namespace	http://edds.egos.esa/model
children	MissionName DataSource Dataspace
used by	element LastConsolidationRequest
annotation	documentation The request for last DARC consolidation time for given dataspace, or active one, if none specified

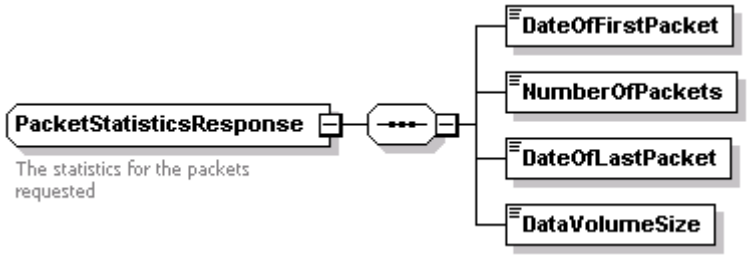
C.28 complexType ListAliasInfo

diagram	
namespace	http://edds.egos.esa/model
children	**AliasType**
used by	
annotation	documentation A list of AliasInfo

C.29 complexType ListString

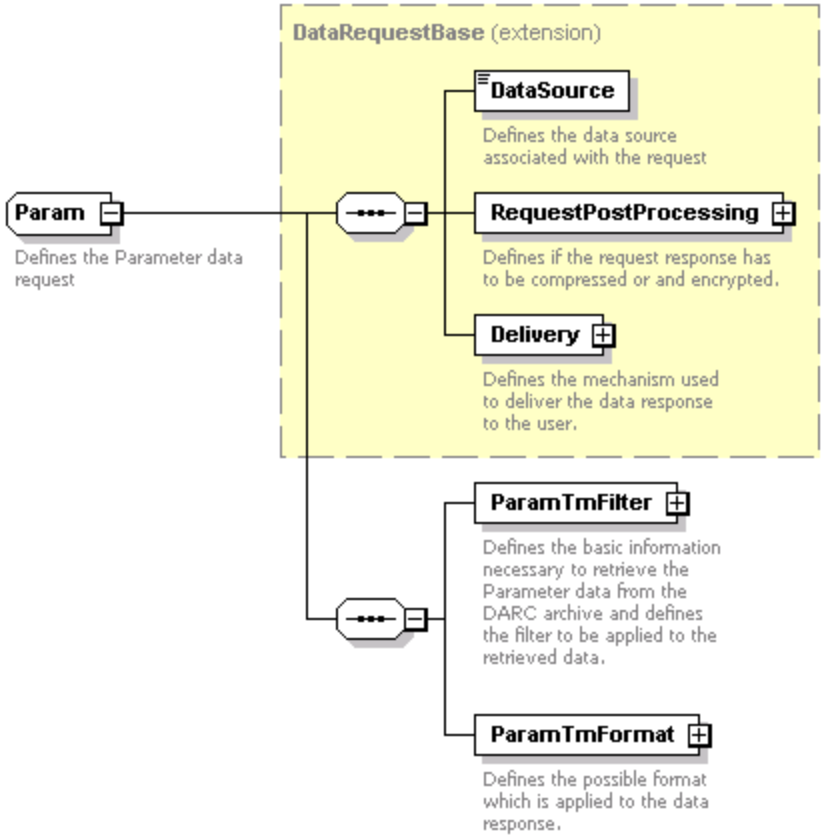
diagram	
namespace	http://edds.egos.esa/model
children	ListElement
used by	element ListString
annotation	documentation A list of String values

C.30 complexType PacketStatisticsResponse

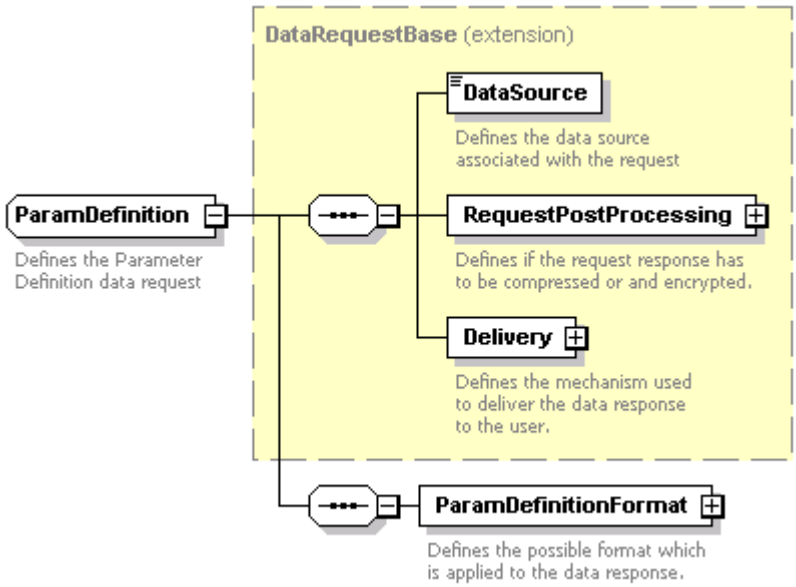
diagram	
namespace	http://edds.egos.esa/model
children	**DateOfFirstPacket NumberOfPackets DateOfLastPacket DataVolumeSize**

used by	element **Response/PacketStatisticsResponse**
annotation	documentation The statistics for the packets requested

C.31 complexType Param

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**RequestPostProcessing Delivery ParamTmFilter ParamTmFormat**
used by	elements **Param DataRequest/Param**
annotation	documentation Defines the Parameter data request

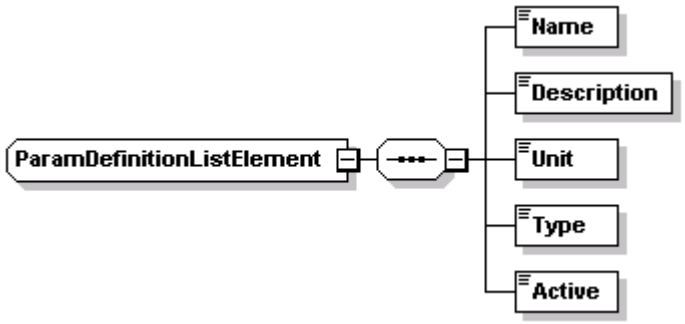
C.32 complexType ParamDefinition

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**RequestPostProcessing Delivery ParamDefinitionFormat**
used by	elements **ParamDefinition DataRequest/ParamDefinition**
annotation	documentation Defines the Parameter Definition data request

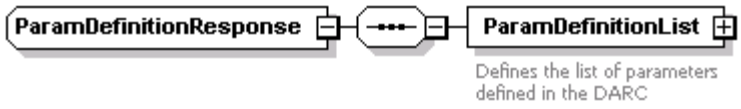
C.33 complexType ParamDefinitionList

diagram	
namespace	http://edds.egos.esa/model
children	**ParamDefinitionListElement**
used by	element **ParamDefinitionResponse/ParamDefinitionList**

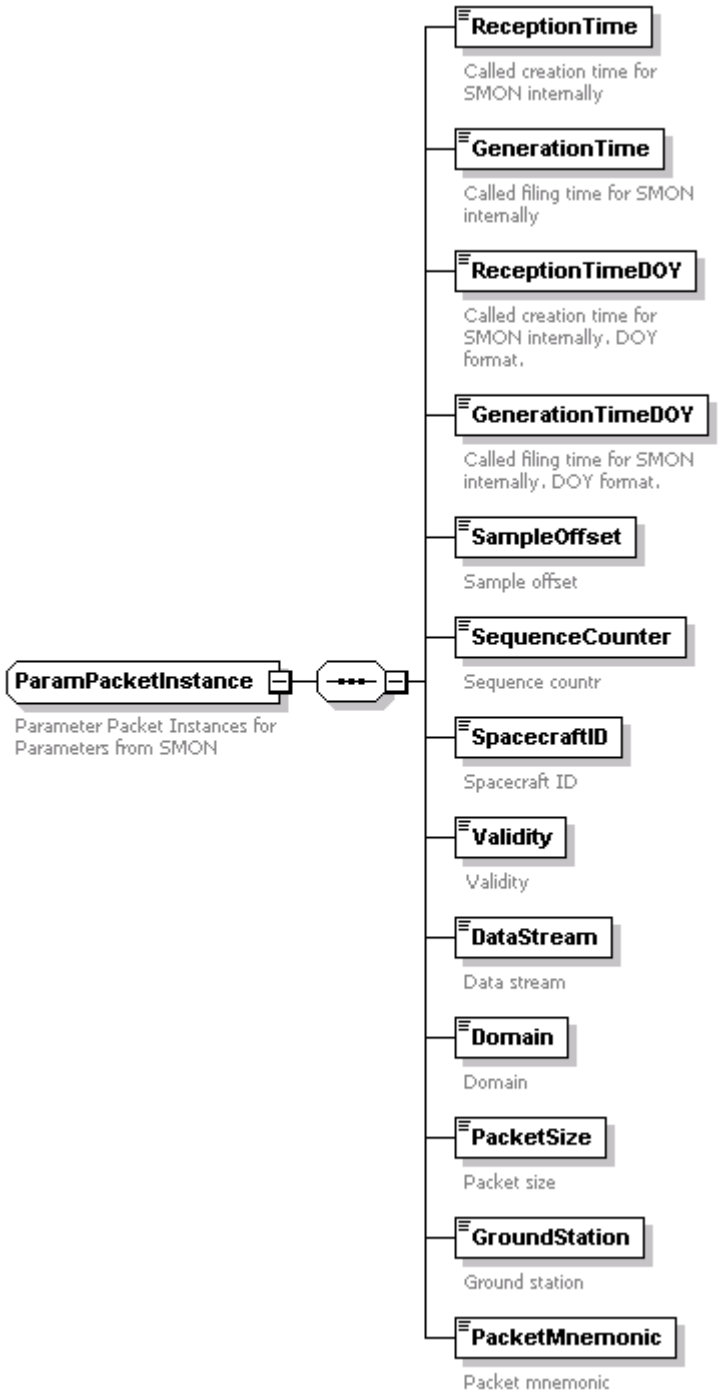
C.34 complexType ParamDefinitionListElement

diagram	
namespace	http://edds.egos.esa/model
children	**Name Description Unit Type Active**
used by	elements **ParamDefinitionListElement ParamDefinitionList/ParamDefinitionListElement**

C.35 complexType ParamDefinitionResponse

diagram	
namespace	http://edds.egos.esa/model
children	**ParamDefinitionList**
used by	element **Response/ParamDefinitionResponse**

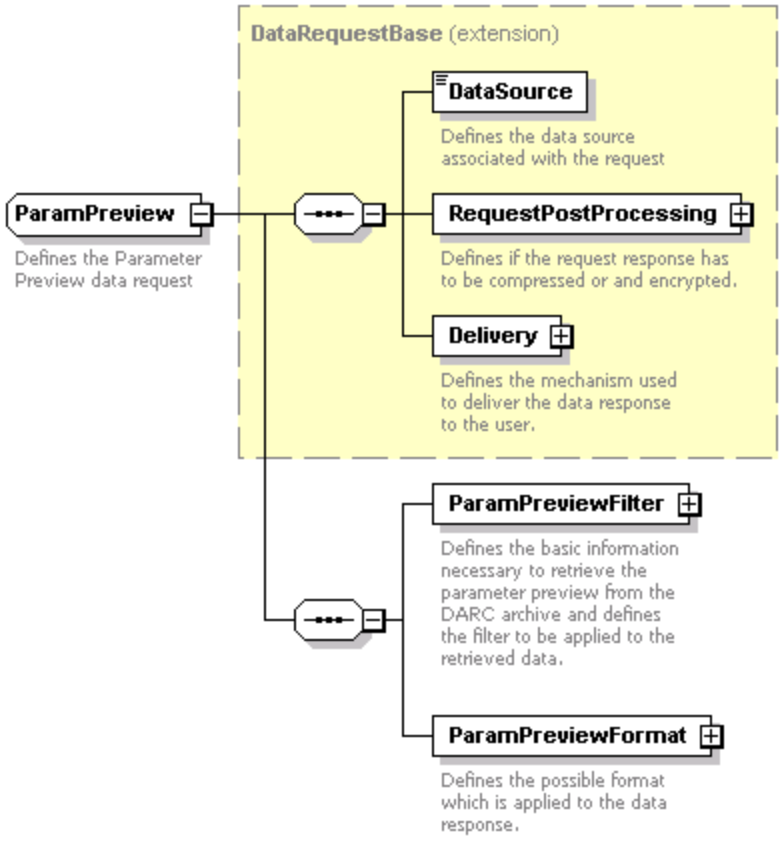
C.36 complexType ParamPacketInstance

<table>
<tr><td>diagram</td><td> 
</td></tr>
<tr><td>namespace</td><td>http://edds.egos.esa/model</td></tr>
<tr><td>children</td><td>ReceptionTime GenerationTime ReceptionTimeDOY GenerationTimeDOY SampleOffset SequenceCounter SpacecraftID Validity DataStream Domain PacketSize GroundStation PacketMnemonic</td></tr>
<tr><td>used by</td><td>elements ParamPacketInstances/PacketInstance ParamPacketInstance</td></tr>
<tr><td>annotation</td><td>documentation
Parameter Packet Instances for Parameters from SMON</td></tr>
</table>

C.37 complexType ParamPacketInstances

diagram	
namespace	http://edds.egos.esa/model
children	**PacketInstance**
used by	element **SMONParamSampleListElement/PacketInstances**
annotation	documentation Parameter Packet Instances for Parameters from SMON

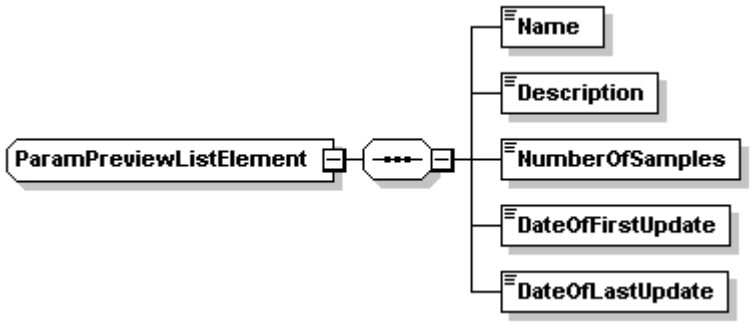
C.38 complexType ParamPreview

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**RequestPostProcessing Delivery ParamPreviewFilter ParamPreviewFormat**
used by	elements **ParamPreview DataRequest/ParamPreview**
annotation	documentation Defines the Parameter Preview data request

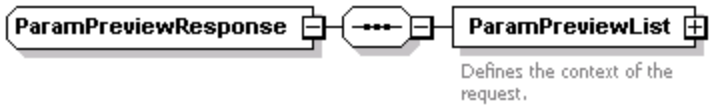
C.39 complexType ParamPreviewList

diagram	
namespace	http://edds.egos.esa/model
children	**ParamPreviewListElement**
used by	element **ParamPreviewResponse/ParamPreviewList**

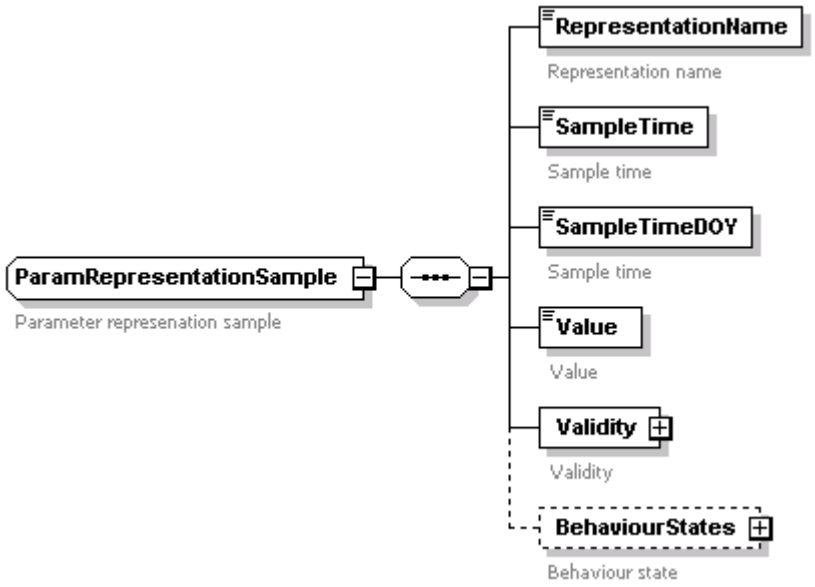
C.40 complexType ParamPreviewListElement

diagram	
namespace	http://edds.egos.esa/model
children	**Name Description NumberOfSamples DateOfFirstUpdate DateOfLastUpdate**
used by	elements **ParamPreviewListElement ParamPreviewList/ParamPreviewListElement**

C.41 complexType ParamPreviewResponse

diagram	
namespace	http://edds.egos.esa/model
children	**ParamPreviewList**
used by	element **Response/ParamPreviewResponse**

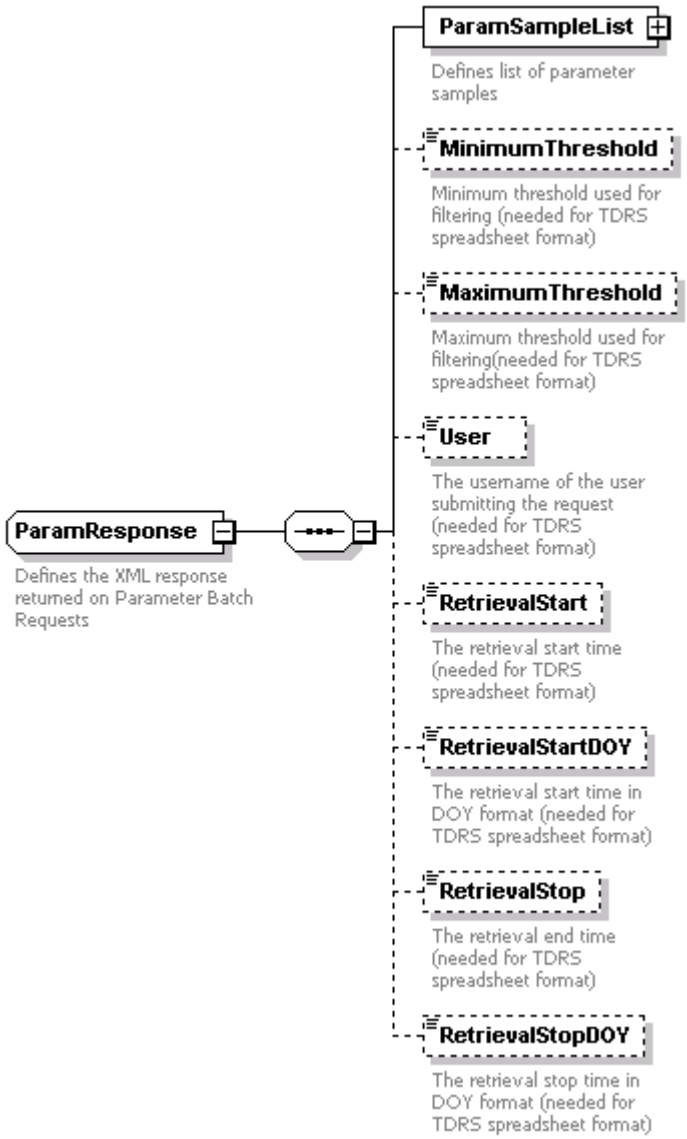
C.42 complexType ParamRepresentationSample

diagram	
namespace	http://edds.egos.esa/model
children	**RepresentationName SampleTime SampleTimeDOY Value Validity BehaviourStates**
used by	elements **ParamRepresentationSample ParamRepresentationSamples/ParamRepresentationSample**
annotation	documentation Parameter represenation sample

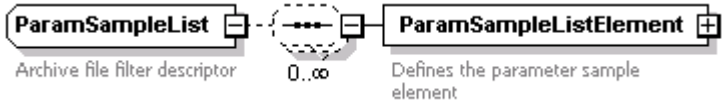
C.43 complexType ParamRepresentationSamples

diagram	
namespace	http://edds.egos.esa/model
children	**ParamRepresentationSample**
used by	element **SMONParamSampleListElement/ParamRepresentationSamples**
annotation	documentation Parameter representation samples from SMON

C.44 complexType ParamResponse

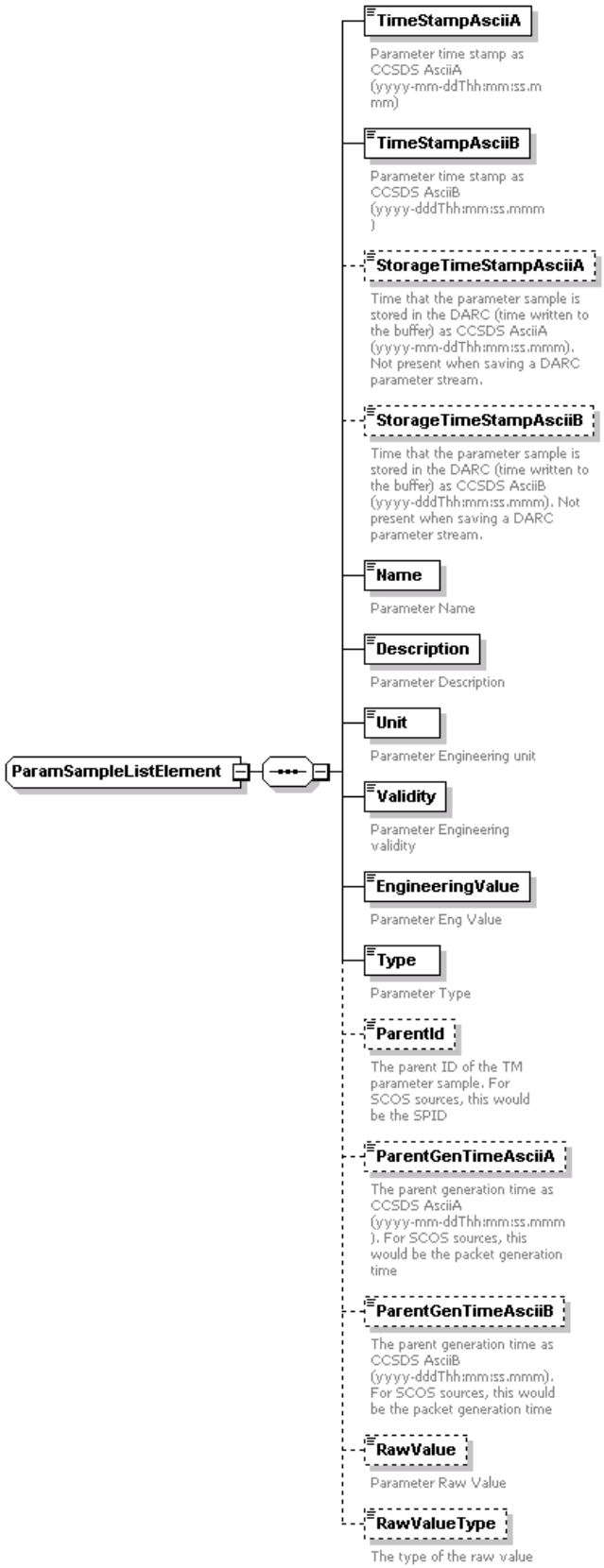
diagram	
namespace	http://edds.egos.esa/model
children	**ParamSampleList MinimumThreshold MaximumThreshold User RetrievalStart RetrievalStartDOY RetrievalStop RetrievalStopDOY**
used by	element **Response/ParamResponse**
annotation	documentation Defines the XML response returned on Parameter Batch Requests

C.45 complexType ParamSampleList

diagram	
namespace	http://edds.egos.esa/model
children	**ParamSampleListElement**

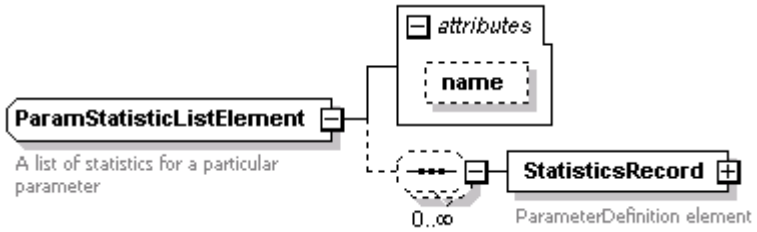
used by	element **ParamResponse/ParamSampleList**
annotation	documentation Defines the list of parameter samples

C.46 complexType ParamSampleListElement

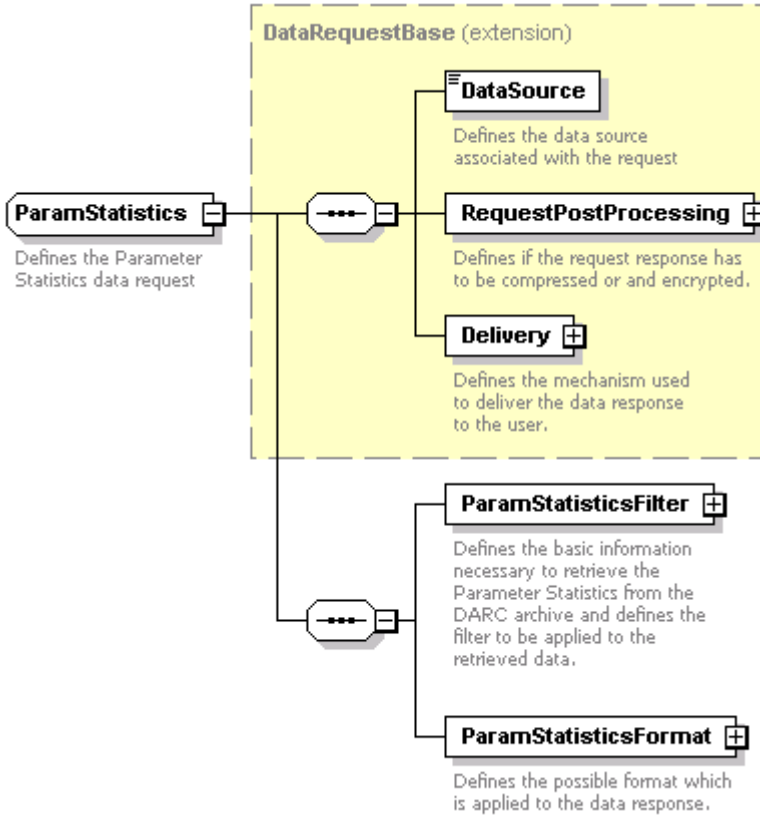
diagram	
namespace	http://edds.egos.esa/model
children	**TimeStampAsciiA TimeStampAsciiB StorageTimeStampAsciiA StorageTimeStampAsciiB Name Description Unit Validity EngineeringValue Type ParentId ParentGenTimeAsciiA ParentGenTimeAsciiB RawValue**

	RawValueType
used by	elements **ParamSampleListElement ParamSampleList/ParamSampleListElement**

C.47 complexType ParamStatisticListElement

diagram	
namespace	http://edds.egos.esa/model
children	**StatisticsRecord**
used by	element **ParamStatisticsList/ParamStatisticListElement**
attributes	Name Type Use Default Fixed annotation name **xs:string**
annotation	documentation A list of statistics for a particular parameter

C.48 complexType ParamStatistics

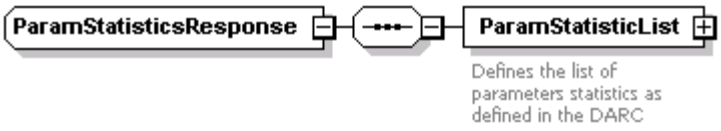
diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase

children	**RequestPostProcessing Delivery ParamStatisticsFilter ParamStatisticsFormat**
used by	elements **ParamStatistics DataRequest/ParamStatistics**
annotation	documentation Defines the Parameter Statistics data request

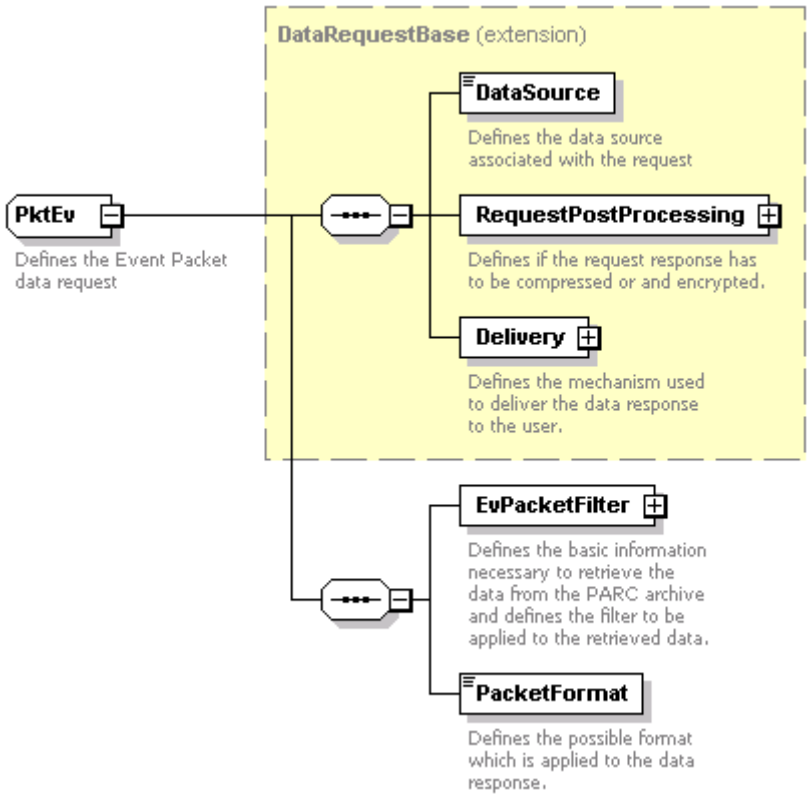
C.49 complexType ParamStatisticsList

diagram	
namespace	http://edds.egos.esa/model
children	**ParamStatisticListElement**
used by	element **ParamStatisticsResponse/ParamStatisticList**
annotation	documentation A list of parameters with all the associated statistics for the time period requested

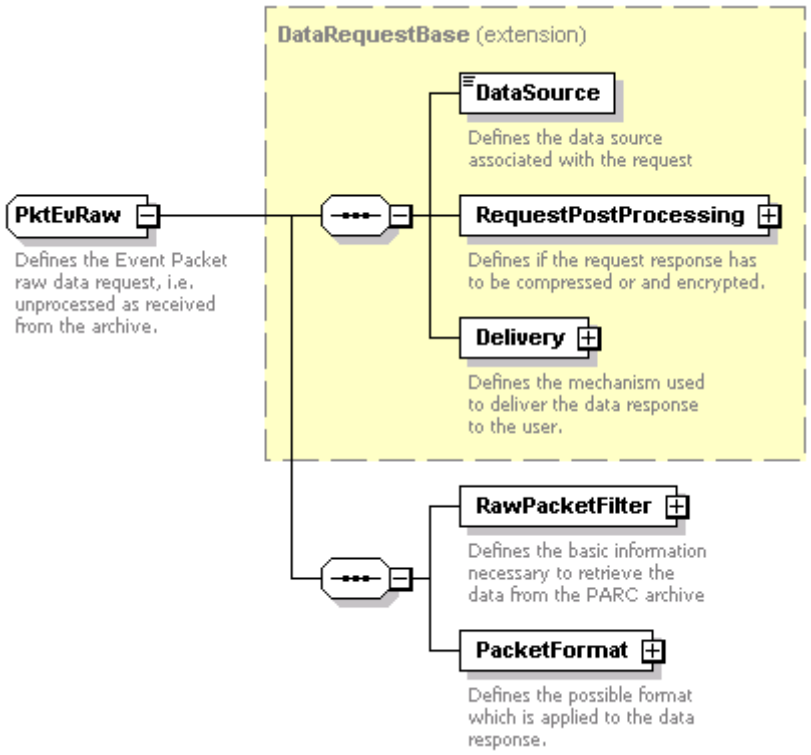
C.50 complexType ParamStatisticsResponse

diagram	
namespace	http://edds.egos.esa/model
children	**ParamStatisticList**
used by	element **Response/ParamStatisticsResponse**

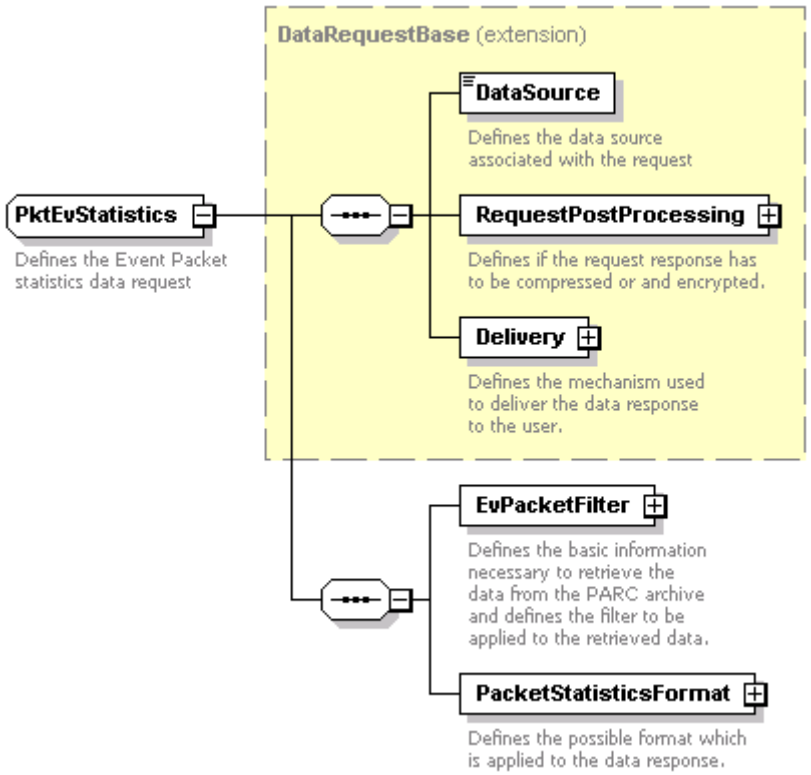
C.51 complexType PktEv

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**RequestPostProcessing Delivery EvPacketFilter PacketFormat**
used by	elements **PktEv DataRequest/PktEv**
annotation	documentation Defines the Event Packet data request

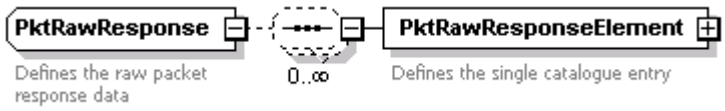
C.52 complexType PktEvRaw

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**RequestPostProcessing Delivery RawPacketFilter PacketFormat**
used by	elements **PktEvRaw DataRequest/PktEvRaw**
annotation	documentation Defines the Event Packet raw data request, i.e. unprocessed as received from the archive.

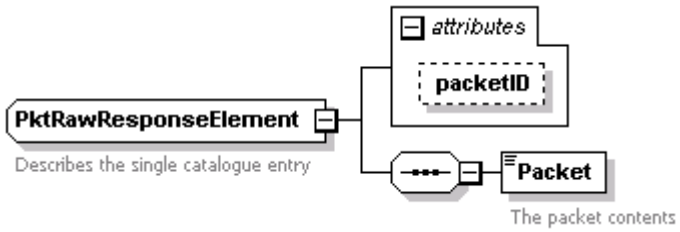
C.53 complexType PktEvStatistics

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**RequestPostProcessing Delivery EvPacketFilter PacketStatisticsFormat**
used by	elements **PktEvStatistics DataRequest/PktEvStatistics**
annotation	documentation Defines the Event Packet statistics data request

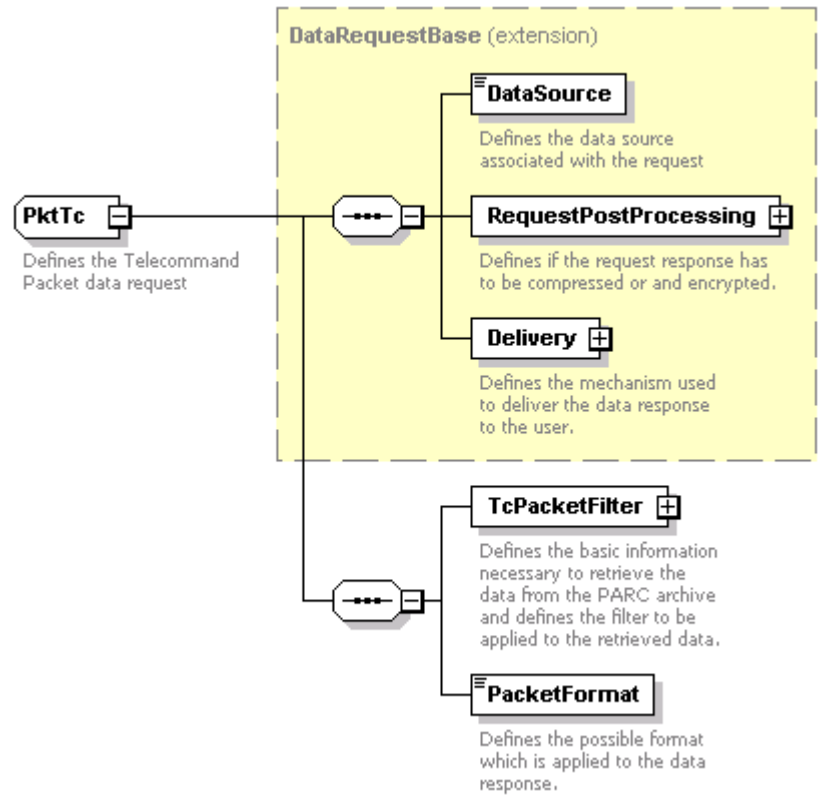
C.54 complexType PktRawResponse

diagram	
namespace	http://edds.egos.esa/model
children	**PktRawResponseElement**
used by	element **Response/PktRawResponse**
annotation	documentation Defines the raw packet response data

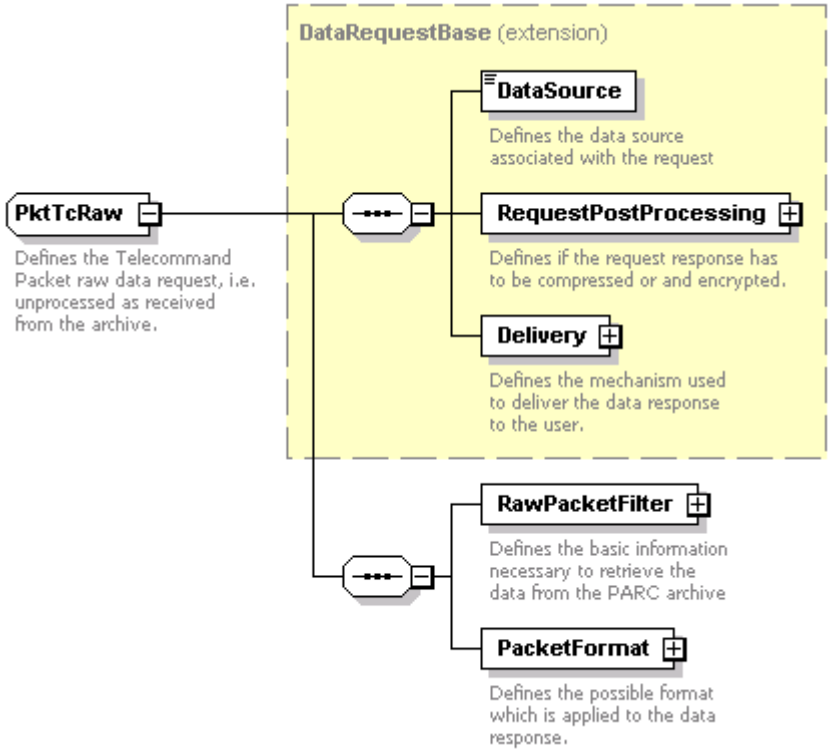
C.55 complexType PktRawResponseElement

diagram	
namespace	http://edds.egos.esa/model
children	**Packet**
used by	elements **PktRawResponseElement PktRawResponse/PktRawResponseElement**
attributes	Name Type Use Default Fixed annotation packetID **xs:long**
annotation	documentation Describes the single catalogue entry

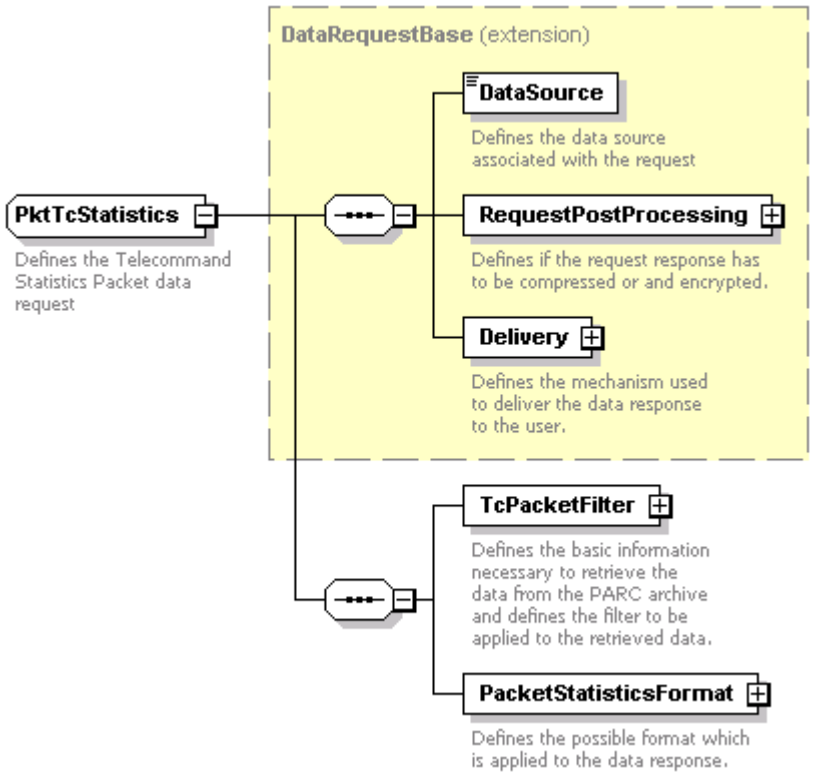
C.56 complexType PktTc

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**RequestPostProcessing Delivery TcPacketFilter PacketFormat**
used by	elements **PktTc DataRequest/PktTc**
annotation	documentation Defines the Telecommand Packet data request

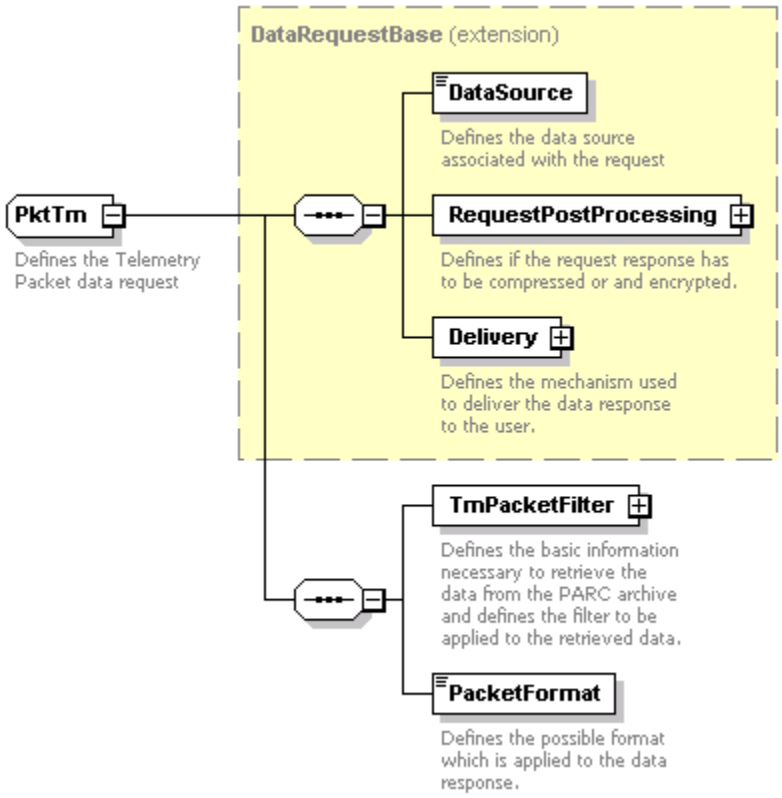
C.57 complexType PktTcRaw

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**RequestPostProcessing Delivery RawPacketFilter PacketFormat**
used by	elements **PktTcRaw DataRequest/PktTcRaw**
annotation	documentation Defines the Telecommand Packet raw data request, i.e. unprocessed as received from the archive.

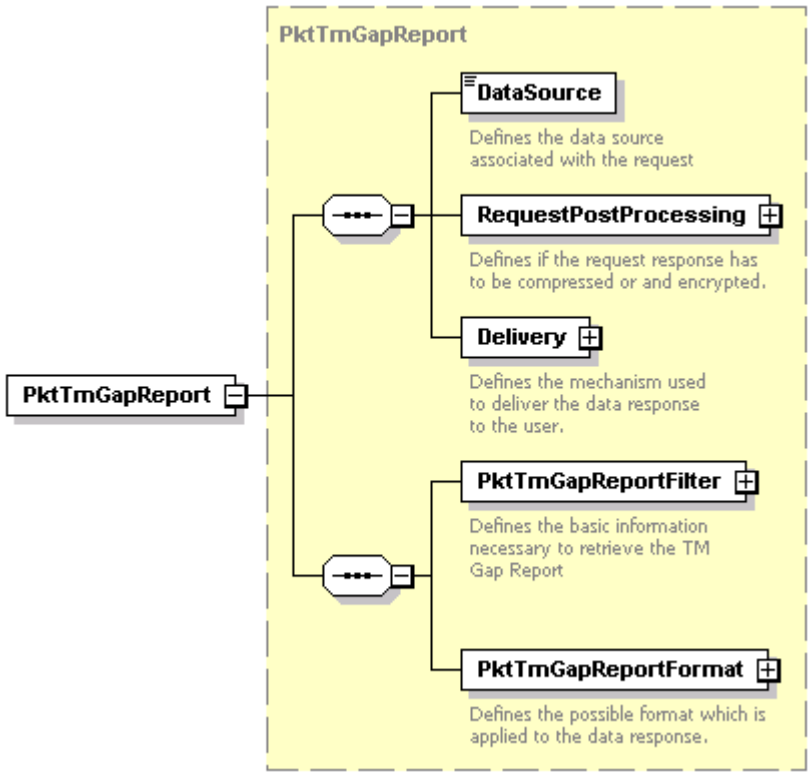
C.58 complexType PktTcStatistics

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**RequestPostProcessing Delivery TcPacketFilter PacketStatisticsFormat**
used by	elements **PktTcStatistics DataRequest/PktTcStatistics**
annotation	documentation Defines the Telecommand Statistics Packet data request

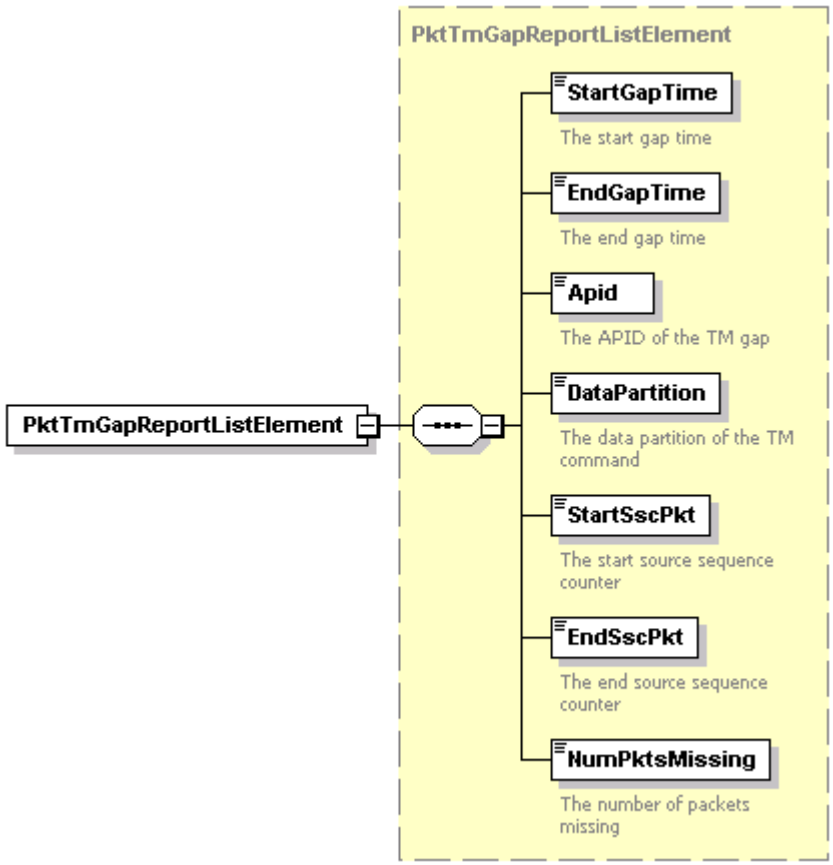
C.59 complexType PktTm

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**RequestPostProcessing Delivery TmPacketFilter PacketFormat**
used by	elements **PktTm DataRequest/PktTm**
annotation	documentation Defines the Telemetry Packet data request

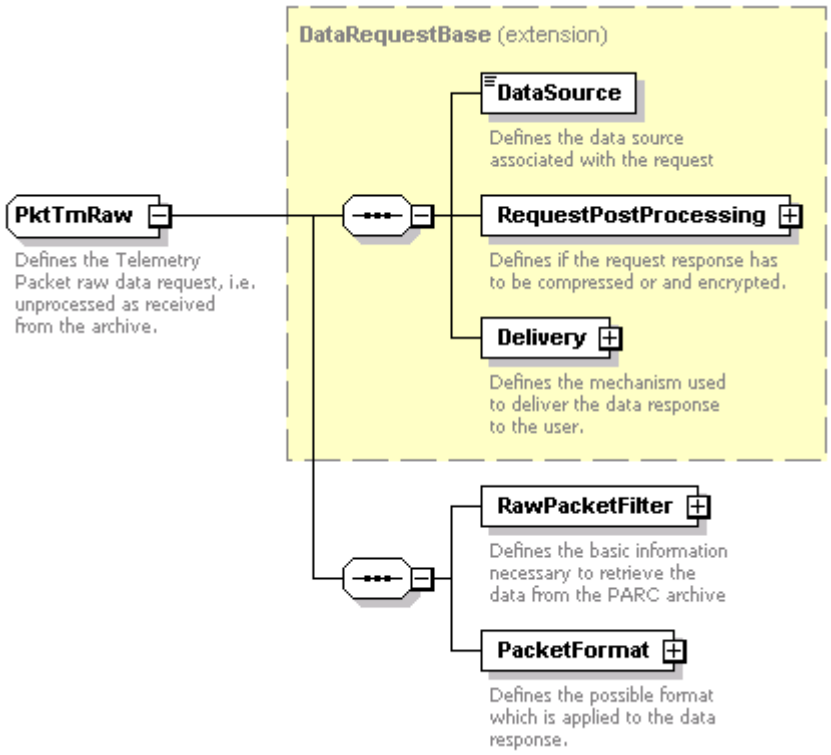
C.60 complexType PktTmGapReport

diagram	
namespace	http://edds.egos.esa/model
type	PktTmGapReport
properties	content complex
children	DataSource RequestPostProcessing Delivery PktTmGapReportFilter PktTmGapReportFormat

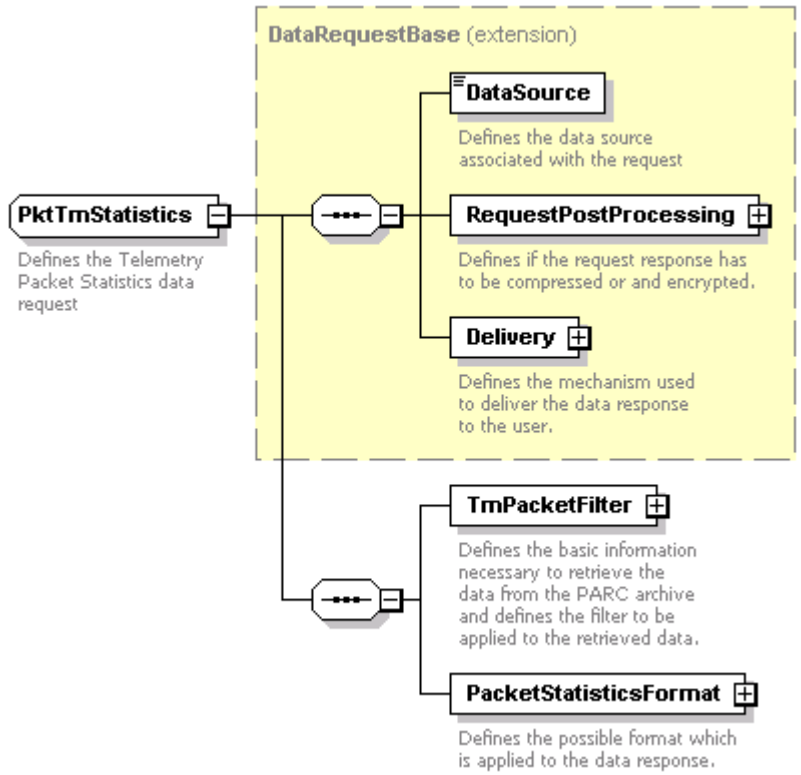
C.61 complexType PktTmGapReportElement

diagram	
namespace	http://edds.egos.esa/model
type	PktTmGapReportListElement
properties	content complex
children	StartGapTime EndGapTime Apid DataPartition StartSscPkt EndSscPkt NumPktsMissing

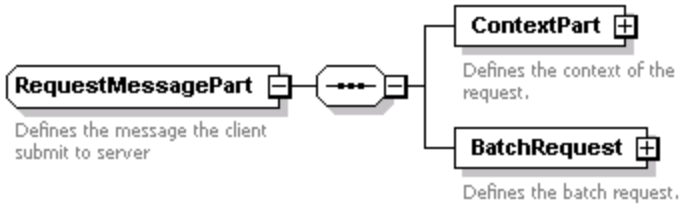
C.62 complexType PktTmRaw

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**RequestPostProcessing Delivery RawPacketFilter PacketFormat**
used by	elements **PktTmRaw DataRequest/PktTmRaw**
annotation	documentation Defines the Telemetry Packet raw data request, i.e. unprocessed as received from the archive.

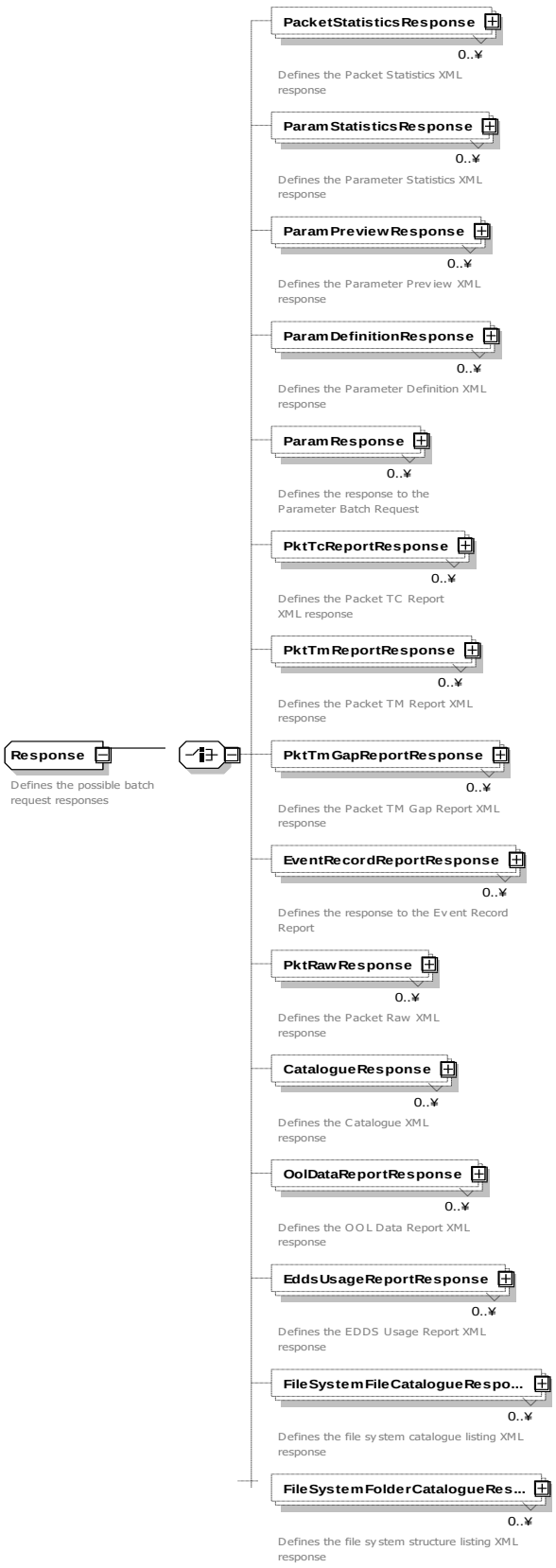
C.63 complexType PktTmStatistics

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**RequestPostProcessing Delivery TmPacketFilter PacketStatisticsFormat**
used by	elements **PktTmStatistics DataRequest/PktTmStatistics**
annotation	documentation Defines the Telemetry Packet Statistics data request

C.64 complexType RequestMessagePart

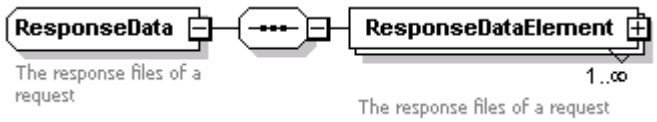
diagram	
namespace	http://edds.egos.esa/model
children	**ContextPart BatchRequest**
used by	element **RequestMessagePart**
annotation	documentation Defines the message the client submit to server

C.65 complexType Response

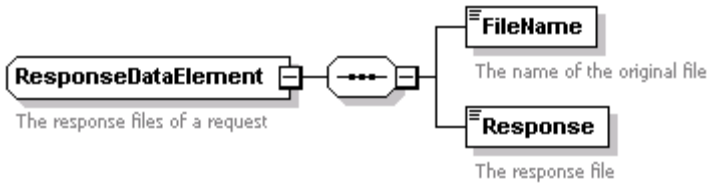
diagram	
namespace	http://edds.egos.esa/model
children	**PacketStatisticsResponse ParamStatisticsResponse ParamPreviewResponse ParamDefinitionResponse ParamResponse PktTcReportResponse PktTmReportResponse PktTmGapReportResponse EventRecordReportResponse PktRawResponse CatalogueResponse**

	OolDataReportResponse EddsUsageReportResponse FileSystemFileCatalogueResponse FileSystemFolderCatalogueResponse
used by	element **ResponsePart/Response**
annotation	documentation Defines the possible batch request responses

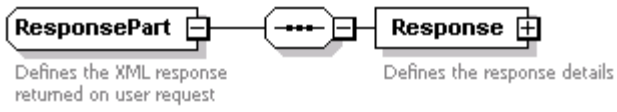
C.66 complexType ResponseData

diagram	
namespace	http://edds.egos.esa/model
children	**ResponseDataElement**
used by	element **ResponseData**
annotation	documentation The response files of a request

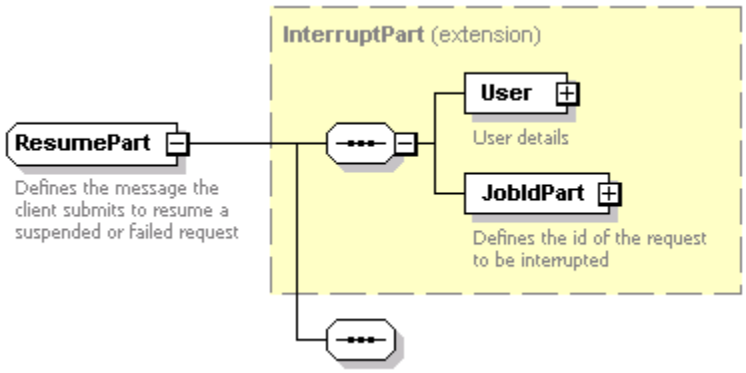
C.67 complexType ResponseDataElement

diagram	
namespace	http://edds.egos.esa/model
children	**FileName Response**
used by	element **ResponseData/ResponseDataElement**
annotation	documentation The response files of a request

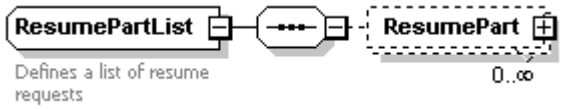
C.68 complexType ResponsePart

diagram	
namespace	http://edds.egos.esa/model
children	**Response**
used by	element **ResponsePart**
annotation	documentation Defines the XML response returned on user request

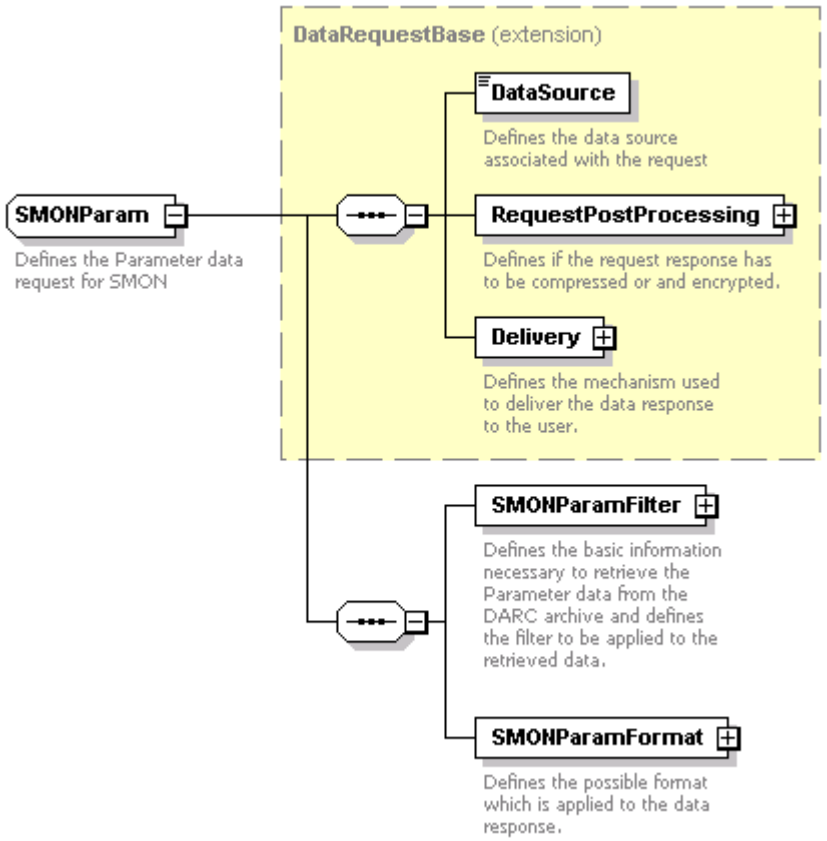
C.69 complexType ResumePart

diagram	
namespace	http://edds.egos.esa/model
type	extension of **InterruptPart**
properties	base InterruptPart
children	**User JobIdPart**
used by	elements **ResumePart ResumePartList/ResumePart**
annotation	documentation Defines the message the client submits to resume a suspended or failed request

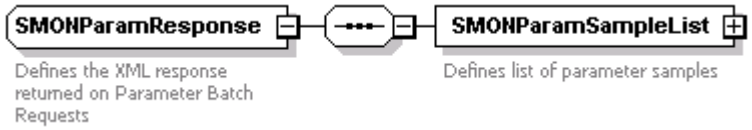
C.70 complexType ResumePartList

diagram	
namespace	http://edds.egos.esa/model
children	**ResumePart**
used by	element **ResumePartList**
annotation	documentation Defines a list of resume requests

C.71 complexType SMONParam

diagram	
namespace	http://edds.egos.esa/model
children	**DataSource RequestPostProcessing Delivery SMONParamFilter SMONParamFormat**
used by	elements **SMONParam DataRequest/SmonParam**
annotation	documentation Defines the Parameter data request for SMON

C.72 complexType SMONParamResponse

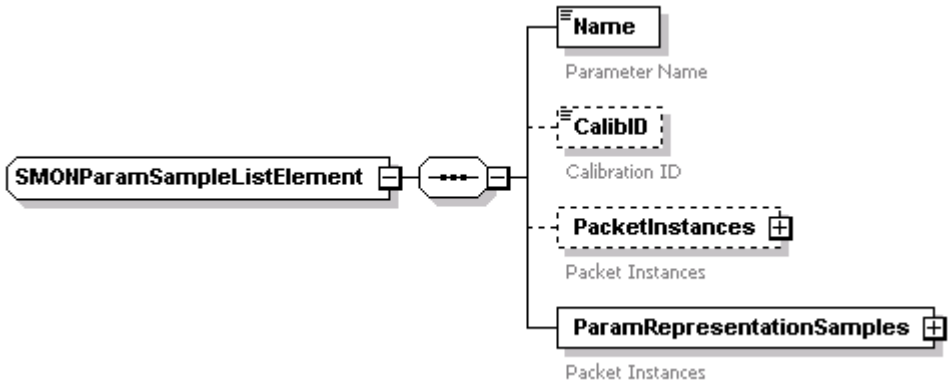
diagram	
namespace	http://edds.egos.esa/model
children	**SMONParamSampleList**
annotation	documentation Defines the XML response returned on Parameter Batch Requests

C.73 complexType SMONParamResponseList

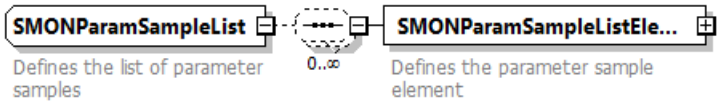
diagram	
namespace	http://edds.egos.esa/model
children	**SMONParamSampleListElement**

used by	element **SMONParamResponse/SMONParamSampleList**
annotation	documentation Defines the list of parameter samples

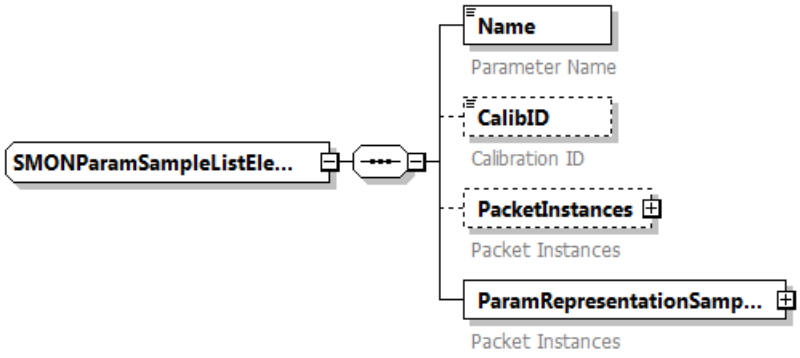
C.74 complexType SMONParamResponseListElement

diagram	
namespace	http://edds.egos.esa/model
children	**Name CalibID PacketInstances ParamRepresentationSamples**
used by	elements **SMONParamSampleListElement SMONParamSampleList/SMONParamSampleListElement**

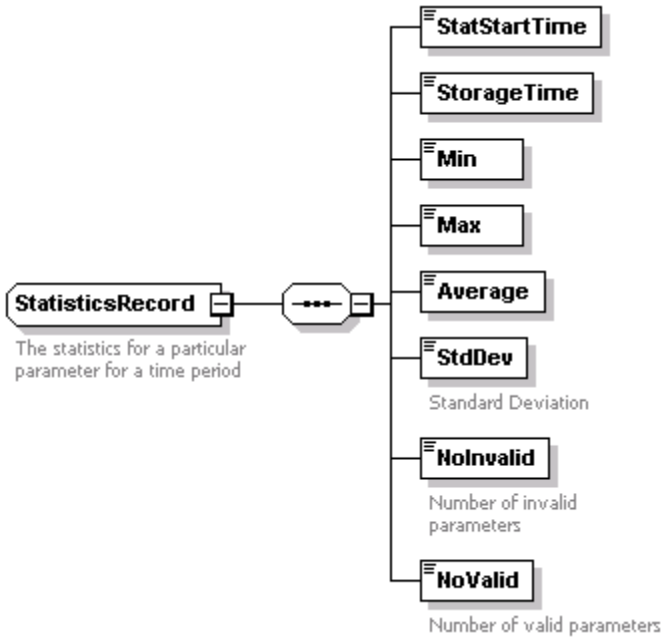
C.75 complexType SMONParamSampleList

diagram	
namespace	http://edds.egos.esa/model
children	**SMONParamSampleListElement**
used by	elements **SMONParamResponse**

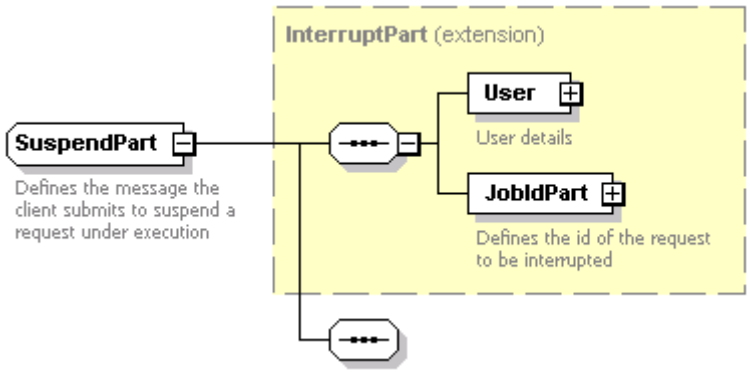
C.76 complexType SMONParamSampleListElement

diagram	
namespace	http://edds.egos.esa/model
children	**Name CalibID PacketInstances ParamRepresentationSamples**
used by	elements **SMONParamSampleList**

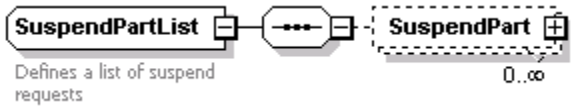
C.77 complexType StatisticsRecord

diagram	
namespace	http://edds.egos.esa/model
children	**StatStartTime StorageTime Min Max Average StdDev NoInvalid NoValid**
used by	element **ParamStatisticListElement/StatisticsRecord**
annotation	documentation The statistics for a particular parameter for a time period

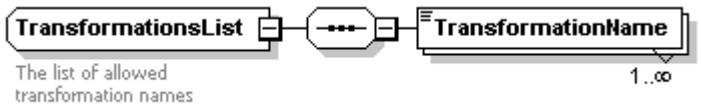
C.78 complexType SuspendPart

diagram	
namespace	http://edds.egos.esa/model
type	extension of **InterruptPart**
properties	base InterruptPart
children	**User JobIdPart**
used by	elements **SuspendPart SuspendPartList/SuspendPart**
annotation	documentation Defines the message the client submits to suspend a request under execution

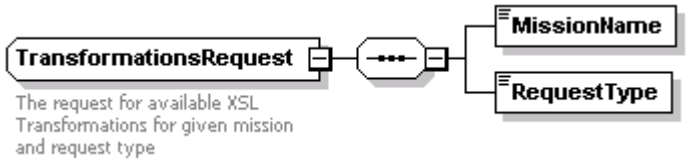
C.79 complexType SuspendPartList

diagram	
namespace	http://edds.egos.esa/model
children	**SuspendPart**
used by	element **SuspendPartList**
annotation	documentation Defines a list of suspend requests

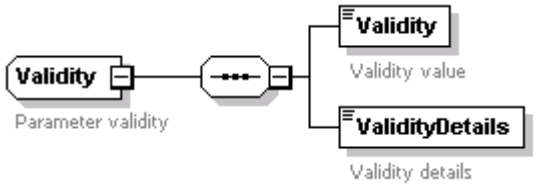
C.80 complexType TransformationsList

diagram	
namespace	http://edds.egos.esa/model
children	**TransformationName**
used by	element **TransformationsList**
annotation	documentation The list of allowed transformation names

C.81 complexType TransformationsRequest

diagram	
namespace	http://edds.egos.esa/model
children	**MissionName RequestType**
used by	element **TransformationsRequest**
annotation	documentation The request for available XSL Transformations for given mission and request type

C.82 complexType Validity

diagram	
namespace	http://edds.egos.esa/model
children	**Validity ValidityDetails**
used by	element **ParamRepresentationSample/Validity**

annotation	documentation Parameter validity

C.83 simpleType AliasInternalType

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value enumeration DATAPROVIDER enumeration BATCH enumeration STREAM

C.84 simpleType DeltaBehaviourState

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value enumeration DELTA_ALLOWED enumeration DELTA_EQUAL enumeration MAX_DELTA_LOW enumeration MAX_DELTA_HIGH enumeration MIN_DELTA_LOW enumeration MIN_DELTA_HIGH

C.85 simpleType Direction

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value enumeration ASC enumeration DESC

C.86 simpleType FilterType

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value annotation enumeration AND documentation The different filter options should all be satisfied. enumeration OR documentation One of the filter options should be satisfied.
annotation	documentation Whether the filters should be applied with an AND or OR condition.

C.87 simpleType LimitBehaviourState

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value enumeration WITHIN_RANGE enumeration LOW enumeration HIGH enumeration LOW_LOW enumeration HIGH_HIGH

C.88 simpleType SCCBehaviourState

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value enumeration SCC_ENABLE enumeration SCC_UNINIT enumeration SCC_DISABLE enumeration SCC_OFF enumeration SCC_FAIL

C.89 simpleType StatusBehaviourState

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value enumeration ALLOWED enumeration UNEXPECTED

C.90 simpleType ValidityValue

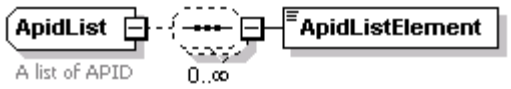
namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value enumeration VAL_VALID enumeration VAL_STATE_OFF enumeration VAL_POWER_OFF enumeration VAL_ROUTE_OFF enumeration VAL_PRESERVE enumeration VAL_TRANSIENT enumeration VAL_TCO_ERROR enumeration VAL_UNUSED_1

	enumeration	VAL_UNUSED_2
	enumeration	VAL_UNUSED_3
	enumeration	VAL_UNUSED_4
	enumeration	VAL_UNUSED_5
	enumeration	VAL_UNUSED_6
	enumeration	VAL_UNUSED_7
	enumeration	VAL_EXPIRED
	enumeration	VAL_UNKNOWN_STATE
	enumeration	VAL_UNKNOWN_CONDITION
	enumeration	VAL_UNKNOWN_CRITERIA
	enumeration	VAL_CALIBRATION
	enumeration	VAL_TOO_EARLY
	enumeration	VAL_UNKNOWN_PKT
	enumeration	VAL_UNINIT
	enumeration	VAL_PKT_RETRV
	enumeration	VAL_MIB_ERROR
	enumeration	VAL_SYSTEM_ERROR
	enumeration	VAL_FIELD_UNKNOWN
	enumeration	VAL_FIELD_ABSENT
	enumeration	VAL_UNKNOWN_TYPE
	enumeration	VAL_UNKNOWN_OP
	enumeration	VAL_UNKNOWN_CONV
	enumeration	VAL_OVERFLOW
	enumeration	VAL_DIVIDE_BY_ZERO
	enumeration	VAL_OL_PARSE

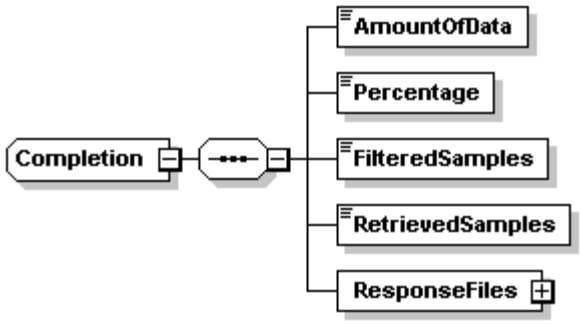
Appendix D Common Data Types

This section of the document describes the common complex types used for all services as defined in the common.xsd file.

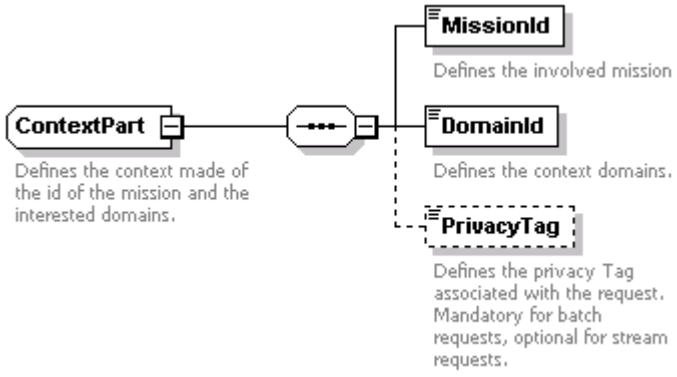
D.1 complexType ApidList

diagram	
namespace	http://edds.egos.esa/model
children	**ApidListElement**
annotation	documentation A list of APID

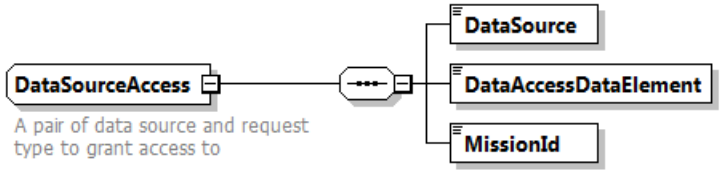
D.2 complexType Completion

diagram	
namespace	http://edds.egos.esa/model
children	**AmountOfData Percentage FilteredSamples RetrievedSamples ResponseFiles**

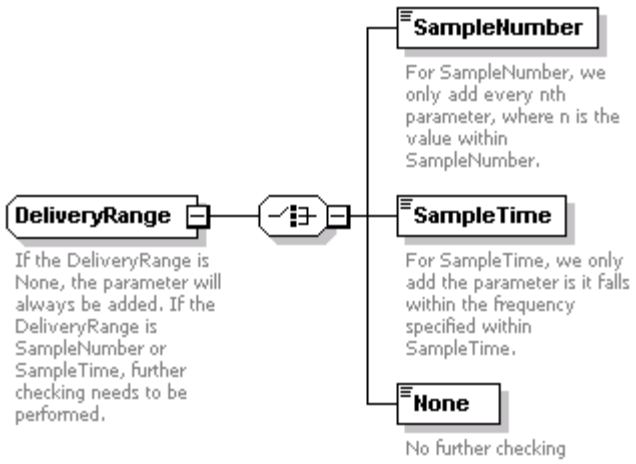
D.3 complexType ContextPart

diagram	
namespace	http://edds.egos.esa/model
children	**MissionId DomainId PrivacyTag**
annotation	documentation Defines the context made of the id of the mission and the interested domains.

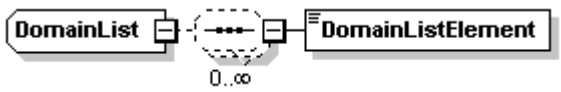
D.4 complexType DataSourceAccess

diagram	
namespace	http://edds.egos.esa/model
children	DataSource DataAccessDataElement **string**
annotation	documentation A pair of data source and request type to grant access to

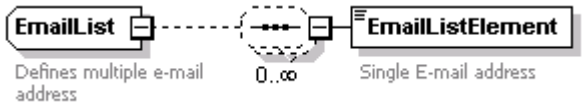
D.5 complexType DeliveryRange

diagram	
namespace	http://edds.egos.esa/model
children	**SampleNumber SampleTime None**
annotation	documentation If the DeliveryRange is None, the parameter will always be added. If the DeliveryRange is SampleNumber or SampleTime, further checking needs to be performed.

D.6 complexType DomainList

diagram	
namespace	http://edds.egos.esa/model
children	**DomainListElement**

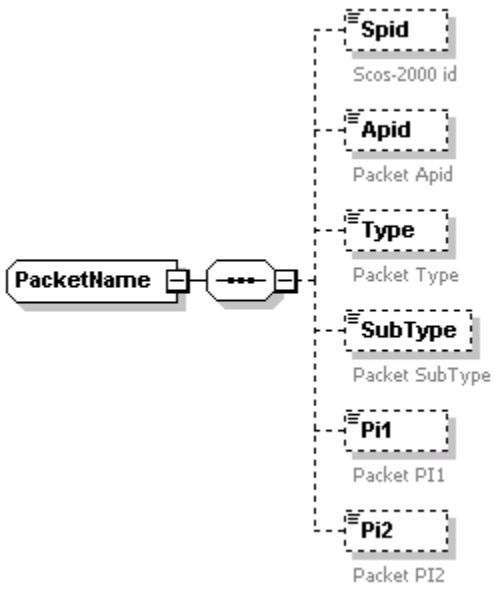
D.7 complexType EmailList

diagram	
namespace	http://edds.egos.esa/model
children	**EmailListElement**
annotation	documentation Defines multiple e-mail address

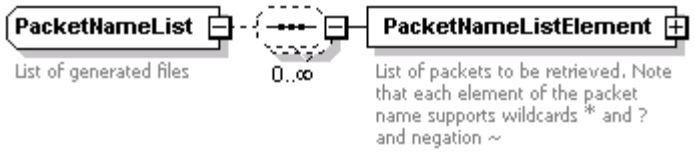
D.8 complexType ParamNameList

diagram	
namespace	http://edds.egos.esa/model
children	**ParamNameListElement**
annotation	documentation A list of parameter names

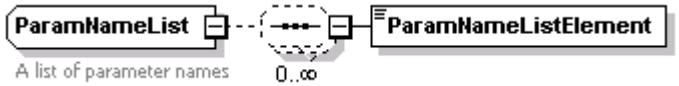
D.9 complexType PacketName

diagram	
namespace	http://edds.egos.esa/model
children	**Spid Apid Type SubType Pi1 Pi2**
used by	element **PacketNameList/PacketNameListElement**

D.10 complexType PacketNameList

diagram	
namespace	http://edds.egos.esa/model
children	**PacketNameListElement**
annotation	documentation List of generated files

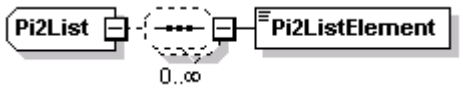
complexType **ParamNameList**

diagram	
namespace	http://edds.egos.esa/model
children	**ParamNameListElement**
annotation	documentation A list of parameter names

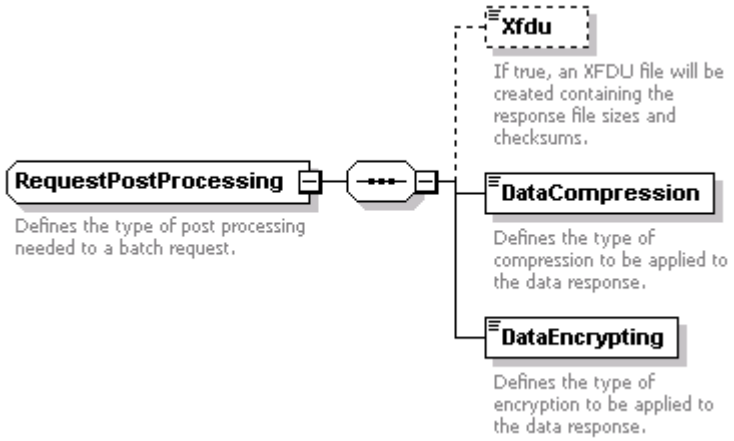
D.11 complexType Pi1List

diagram	
namespace	http://edds.egos.esa/model
children	**Pi1ListElement**

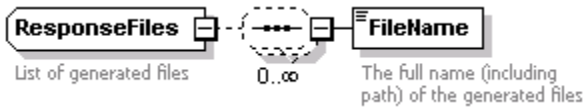
D.12 complexType Pi2List

diagram	
namespace	http://edds.egos.esa/model
children	**Pi2ListElement**

D.13 complexType RequestPostProcessing

diagram	
namespace	http://edds.egos.esa/model
children	**DataCompression DataEncrypting**
annotation	documentation Defines the type of post processing needed to a batch request.

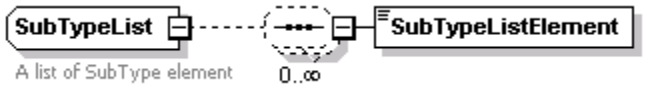
D.14 complexType ResponseFiles

diagram	
namespace	http://edds.egos.esa/model
children	**FileName**
used by	element **Completion/ResponseFiles**
annotation	documentation List of generated files

D.15 complexType SpidList

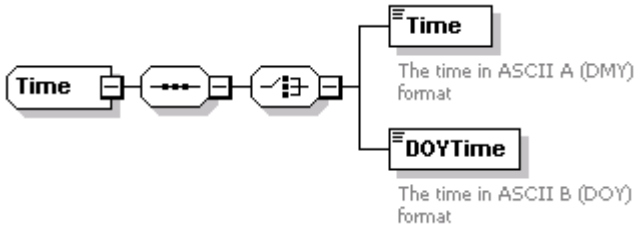
diagram	
namespace	http://edds.egos.esa/model
children	**SpidListElement**
annotation	documentation Not supported by the DARC

D.16 complexType SubTypeList

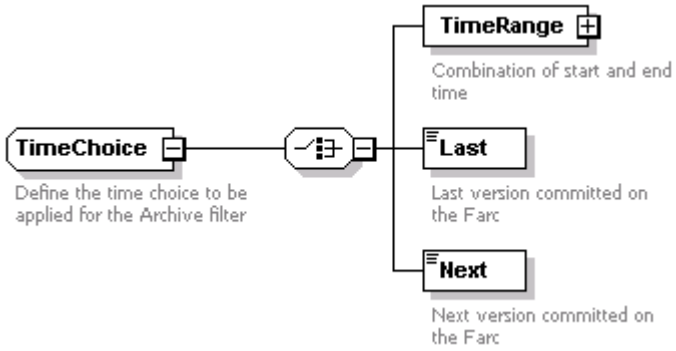
diagram	

namespace	http://edds.egos.esa/model
children	**SubTypeListElement**
annotation	documentation A list of SubType element

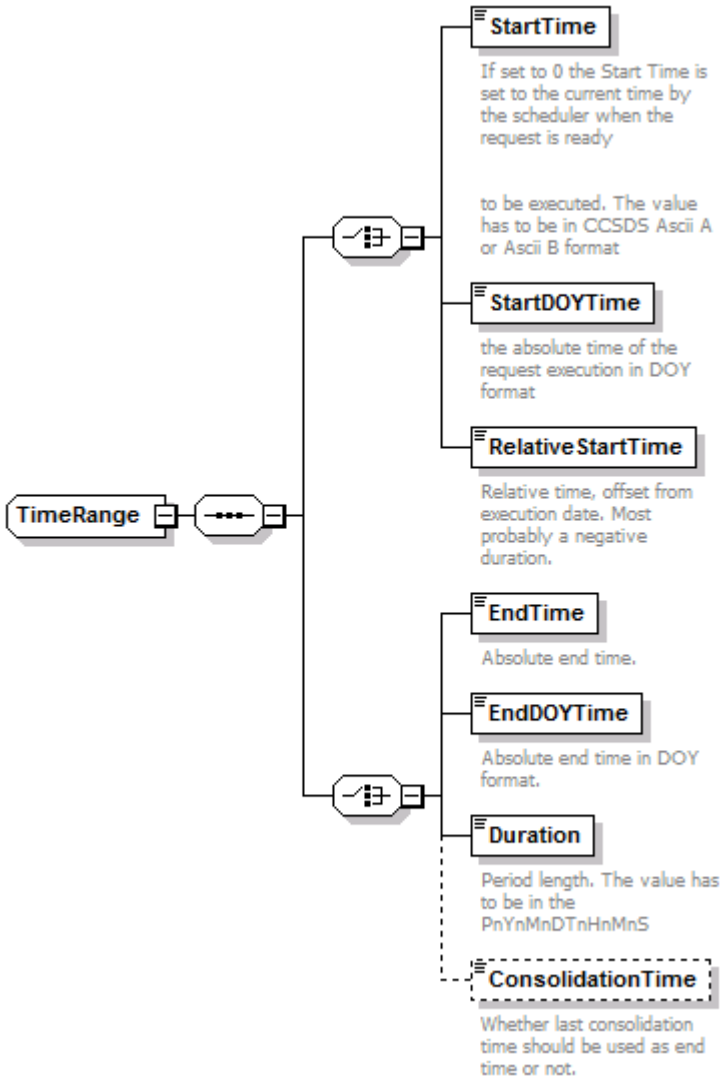
D.17 complexType Time

diagram	
namespace	http://edds.egos.esa/model
children	**Time DOYTime**

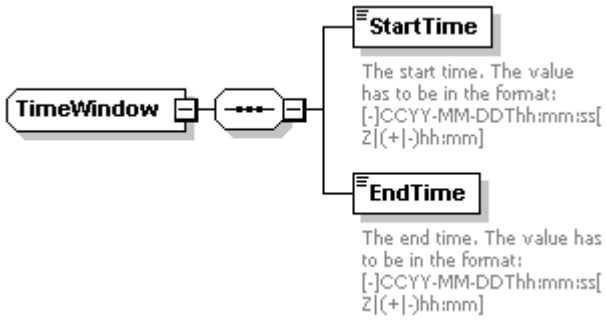
D.18 complexType TimeChoice

diagram	
namespace	http://edds.egos.esa/model
children	**TimeRange Last Next**
annotation	documentation Define the time choice to be applied for the Archive filter

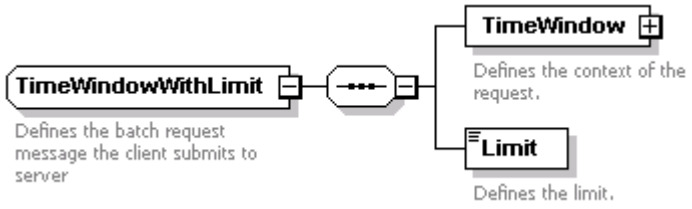
D.19 complexType TimeRange

diagram	
namespace	http://edds.egos.esa/model
children	**StartTime StartDOYTime RelativeStartTime EndTime EndDOYTime Duration ConsolidationTime**
used by	element **EddsUsageReportFilter/TimeRange EvPacketFilter/TimeRange ParamPreviewFilter/TimeRange ParamStatisticsFilter/TimeRange ParamTmFilter/TimeRange OolDataReportFilter/TimeRange TcPacketFilter/TimeRange TmPacketFilter/TimeRange RawPacketFilter/TimeRange TimeChoice/TimeRange**

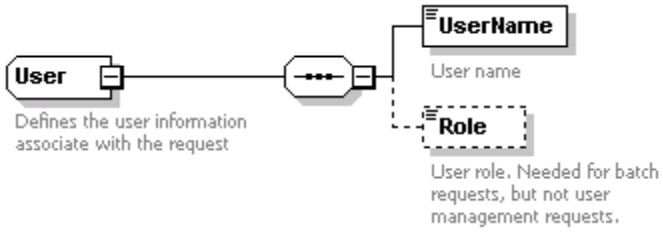
D.20 complexType TimeWindow

diagram	
namespace	http://edds.egos.esa/model
children	**StartTime EndTime**
used by	element **TimeWindow**

D.21 complexType TimeWindowWithLimit

diagram	
namespace	http://edds.egos.esa/model
children	**TimeWindow Limit**
used by	element **TimeWindowWithLimit**
annotation	documentation Defines the batch request message the client submits to server

D.22 complexType User

diagram	
namespace	http://edds.egos.esa/model
children	**UserName Role**
annotation	documentation Defines the user information associate with the request

D.23 simpleType AccountRequestSubType

namespace	http://edds.egos.esa/model

type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value annotation enumeration Group enumeration Role enumeration OperationSet enumeration QuotaSet enumeration DataAccessSet enumeration Mission enumeration UserAccount

D.24 simpleType DataAccessDataElement

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value annotation enumeration Param enumeration ParamStatistics enumeration ParamDefinition enumeration ParamPreview enumeration SmonParam enumeration PktEv enumeration PktTm enumeration PktTc enumeration PktTmStatistics enumeration PktTcStatistics enumeration PktEvStatistics enumeration PktEvRaw enumeration PktTmRaw enumeration PktTcRaw enumeration PktTmReport enumeration PktTcReport enumeration PktTmGapReport enumeration EventRecordReport enumeration ArchiveCatalogue enumeration ArchiveFile enumeration ArchiveSubscription enumeration OolRecordReport enumeration EddsUsageReport enumeration ParamStream enumeration PktEvStream enumeration PktTmStream enumeration PktTcStream enumeration OolStream enumeration FileSystemFileCatalogue enumeration FileSystemFolderCatalogue enumeration FileSystemFile enumeration FileSystemSubscription

D.25 simpleType DataCompression

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	element **RequestPostProcessing/DataCompression**
facets	Kind Value annotation enumeration ZIP enumeration TAR enumeration TARGZ enumeration NONE
annotation	documentation Defines the possible compression algorithm which can be applied to the data responses.

D.26 simpleType DataEncrypting

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	element **RequestPostProcessing/DataEncrypting**
facets	Kind Value annotation enumeration NONE enumeration AES
annotation	documentation Defines the data encryption to be applied to the data responses. The implementation of the encryption algorithm is mission specific.

D.27 simpleType DataSource

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value annotation enumeration FARC enumeration DARC enumeration PARC enumeration EDDS enumeration SMON enumeration FileSystem enumeration DataProvisionService
annotation	documentation Defines data source accociated with the request

D.28 simpleType Direction

namespace	http://edds.egos.esa/model

type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value annotation enumeration ASC enumeration DESC
annotation	documentation Defines whether to retrieve ascending or descending. If not set, the default is descending.

D.29 simpleType DOYDateTime

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	elements **EddsUsageReportFilterKeyword/CompletionDOYTime Time/DOYTime TimeRange/EndDOYTime EddsUsageReportFilterKeyword/ExecutionDOYTime PktTcFilterKeyword/ExecutionDOYTime PktTmFilterKeyword/ExtractionDOYTime PktTmStreamFilterKeyword/ExtractionDOYTime PktEvFilterKeyword/GenerationDOYTime PktTmFilterKeyword/GenerationDOYTime ParamFilterKeyword/OnBoardDOYTime PktTcFilterKeyword/ReleaseDOYTime TimeRange/StartDOYTime EddsUsageReportFilterKeyword/SubmissionDOYTime**
facets	Kind Value annotation pattern \d{4}-(00[1-9]|0[1-9]\d|[12]\d{2}|3([0-5]\d|6[0-6]))T([01]\d|2[0-3]):[0-5]\d:[0-5]\d(\.\d{1,6})?(Z)?
annotation	documentation Defines the YYYY-DDD'T'HH:MM:SS[.SSSSSSZ] format with rules for days: 001-366, hours:00-23, minutes: 00-59, seconds: 00-59, optional fraction of seconds .0-.999999 and optional Zulu time indicator

D.30 simpleType NamePrefixSuffixElement

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value annotation enumeration default enumeration filename enumeration prefix enumeration suffix
annotation	documentation Defines the possible ways to use the file text.

D.31 simpleType BehaviourState

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value annotation enumeration Param enumeration LIMIT_HIGH enumeration LIMIT_HIGH_HIGH enumeration LIMIT_LOW

	enumeration	LIMIT_LOW_LOW
	enumeration	LIMIT_WITHIN_RANGE

D.32 simpleType OolBehaviourDeltaState

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind / Value enumeration LIMIT_HIGH enumeration LIMIT_HIGH_HIGH enumeration OUT_OF_LIMIT_LOW enumeration LIMIT_LOW_LOW enumeration LIMIT_WITHIN_RANGE

D.33 simpleType OolBehaviourLimitState

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind / Value enumeration LIMIT_HIGH enumeration LIMIT_HIGH_HIGH enumeration LIMIT_LOW enumeration LIMIT_LOW_LOW enumeration LIMIT_WITHIN_RANGE

D.34 simpleType OolBehaviourSccState

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind / Value enumeration SCC_ENABLE enumeration SCC_DISABLE enumeration SCC_FAIL enumeration SCC_OFF enumeration SCC_UNINIT

D.35 simpleType OolBehaviourStatusState

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind / Value enumeration STATUS_ALLOWED enumeration STATUS_UNEXPECTED

D.36 simpleType OolParameterState

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value annotation enumeration NOMINAL enumeration WARNING enumeration OUT_OF_LIMITS enumeration SCC enumeration SCC_DISABLE enumeration SCC_OFF enumeration SCC_UNINIT enumeration VIOLATION enumeration NO_VALUE documentation: NO_VALUE is used if a value is returned from the archive that doesn't match a value from this enumerated list

D.37 simpleType OolRecordType

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value annotation enumeration LIMIT enumeration STATE enumeration STATUS_CONSISTENCY enumeration SOFT_HARD_LIMIT enumeration LIMIT_MO enumeration DELTA enumeration NO_VALUE documentation: NO_VALUE is used if a value is returned from the archive that doesn't match a value from this enumerated list

D.38 simpleType OolState

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value annotation enumeration INCONSISTENT_STATUS enumeration NOMINAL enumeration OUT_OF_LIMIT enumeration UNCHECKED enumeration UNDEFINED enumeration VIOLATED enumeration WARNING

D.39 simpleType Operation

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value annotation enumeration OP_EQ enumeration OP_NEQ enumeration OP_GT enumeration OP_GTE enumeration OP_LT enumeration OP_LTE enumeration OP_NOT enumeration OP_MATCH enumeration OP_CONTAINS documentation Used for container objects; matches lists containing the specified filter item, but the container is allowed to contain other items not specified.

D.40 simpleType OrderBy

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value annotation enumeration creation_time ASC enumeration creation_time DESC enumeration scheduled_execution_time ASC enumeration scheduled_execution_time DESC enumeration username ASC enumeration username DESC
annotation	documentation Defines the possible compression algorithm which can be applied to the data responses.

D.41 simpleType ParamValidityStatus

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value annotation enumeration VALID enumeration INVALID enumeration UNKNOWN enumeration EXPIRED
annotation	documentation The possible validity states of a parameter

D.42 simpleType PrivacyTag

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	element **ContextPart/PrivacyTag**
facets	Kind Value annotation enumeration PRIVATE enumeration ROLE enumeration MISSION enumeration PUBLIC
annotation	documentation Defines privacy tag values which can be associated to a request

D.43 simpleType SpidsSpecifier

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	elements **PacketName/Apid PacketName/Pi1 PacketName/Pi2 PacketName/Spid PacketName/SubType PacketName/Type**
facets	Kind Value annotation pattern (\s*((\d+\s*-\s*\d+)|[^-,\s]+)\s*(,\s*((\d+\s*-\s*\d+)|[^-,\s]+)\s*)*)|\s*
annotation	documentation A string with regex restriction to allow for a comma separated list of both string values and numeric value ranges.

D.44 simpleType State

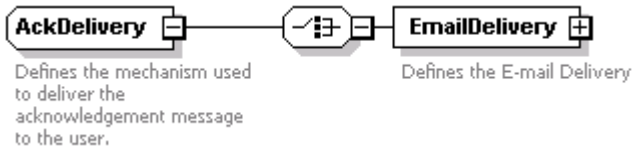
namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value annotation enumeration SUBMITTED documentation The request reached the web server and has been saved into the DB enumeration CANCELED documentation The request has been cancelled enumeration SUSPENDED documentation The request has been suspended enumeration QUEUED documentation A submitted request is taken from the DB and queued for execution in the edds-server enumeration COMPLETED_NO_RESULTS documentation The edds-server completed the elaboration of a request and no results were received from the archive enumeration SERVER_COMPLETED documentation The edds-server completed the elaboration of a request and dispatch the result to the delivery-manager enumeration DELIVERED documentation The request execution is completed enumeration DELIVERED_RESP_DELETED documentation The request execution is completed and the data on EDDS file server is deleted enumeration DELIVERED_PARTIAL_RESULTS documentation The request execution is completed, but not all of the requested data could be retrieved enumeration ERROR_LOCALLY_DELIVERED documentation The request execution is completed, but the delivery manager was not able to dispatch the result through

			FTP. THe file is available in the EDDS Server FTP directory
	enumeration	ERROR_ACCESS_DENIED	documentation The user does not have the right privileges to execute the request
	enumeration	ERROR_INVALID_REQUEST	documentation The request is not valid
	enumeration	ERROR_LIMIT_EXCEEDED	documentation The quota associated with the user has been exceeded
	enumeration	ERROR_SERVER_NOT_AVAILABLE	documentation The needed data provider is not available
	enumeration	ERROR_DELIVERY	documentation The delivery manager was not able to deliver the file
	enumeration	ERROR_UNKNOWN	documentation Not predictable error in the request processing
annotation	documentation Request execution statuses		

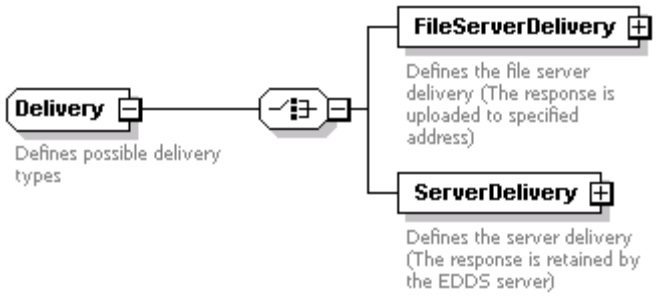
Appendix E Delivery Data Type

This section of the document describes the complex types used for delivery as defined in the delivery.xsd file.

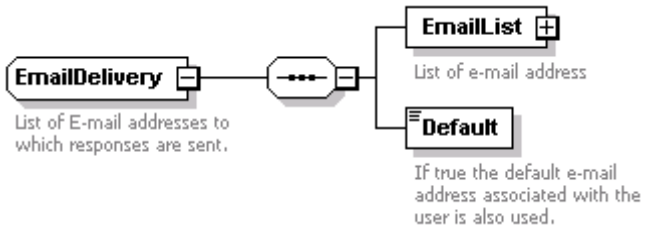
E.1 complexType AckDelivery

diagram	
namespace	http://edds.egos.esa/model
children	**EmailDelivery**
annotation	documentation Defines the mechanism used to deliver the acknowledgement message to the user.

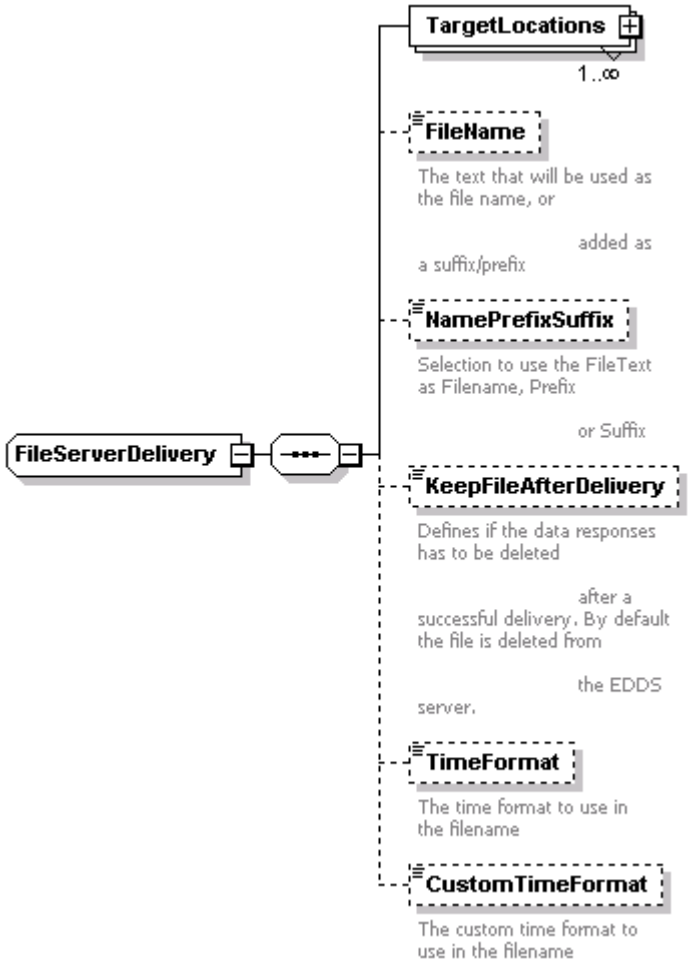
E.2 complexType Delivery

diagram	
namespace	http://edds.egos.esa/model
children	**FileServerDelivery ServerDelivery**
annotation	documentation Defines possible delivery types

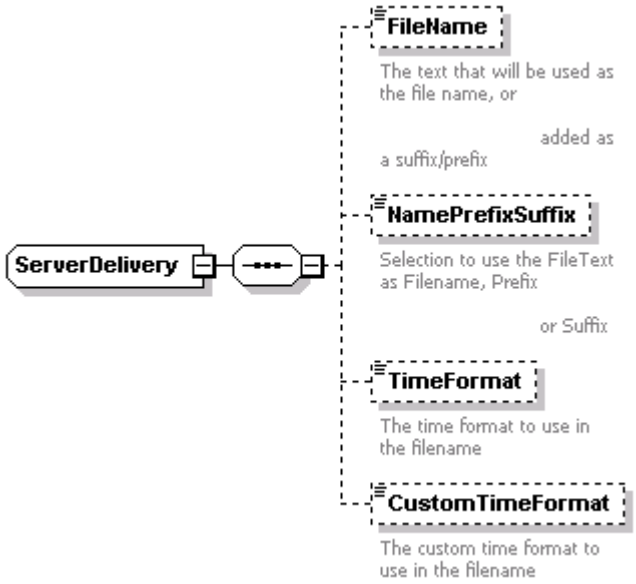
E.3 complexType EmailDelivery

diagram	
namespace	http://edds.egos.esa/model
children	**EmailList Default**
used by	element **AckDelivery/EmailDelivery**
annotation	documentation List of E-mail addresses to which responses are sent.

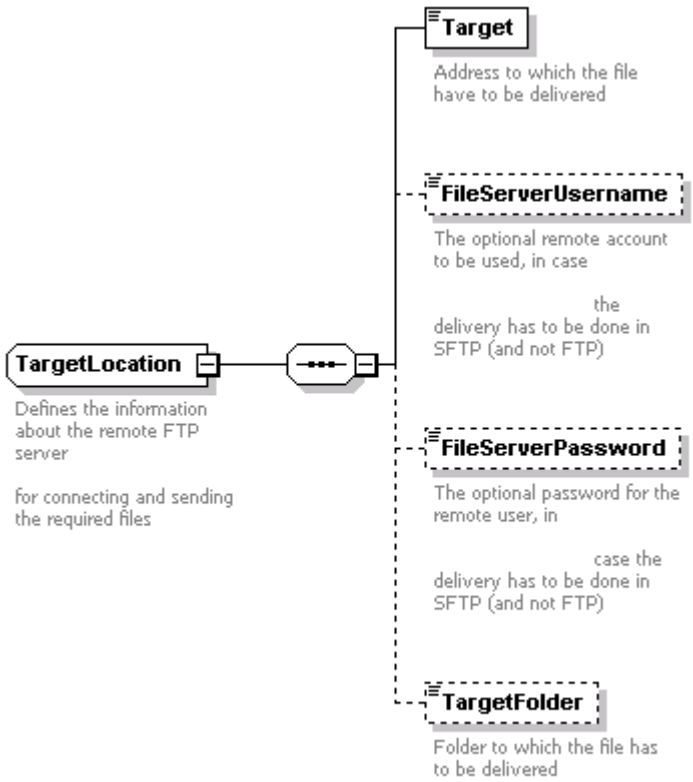
E.4 complexType FileServerDelivery

diagram	
namespace	http://edds.egos.esa/model
children	**TargetLocations FileName NamePrefixSuffix KeepFileAfterDelivery TimeFormat CustomTimeFormat**
used by	element **Delivery/FileServerDelivery**

E.5 complexType ServerDelivery

diagram	
namespace	http://edds.egos.esa/model
children	**FileName NamePrefixSuffix TimeFormat CustomTimeFormat**
used by	element **Delivery/ServerDelivery**

E.6 complexType TargetLocation

diagram	
namespace	http://edds.egos.esa/model
children	**Target FileServerUsername FileServerPassword TargetFolder**

used by	element **FileServerDelivery/TargetLocations**
annotation	documentation Defines the information about the remote FTP server for connecting and sending the required files

E.7 simpleType FileName

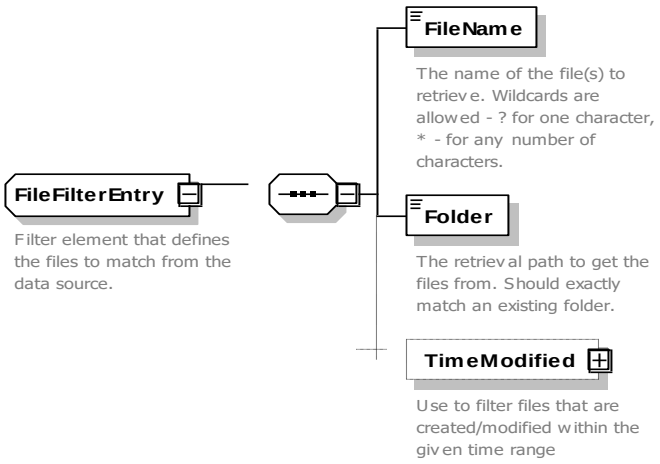
namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value pattern [^\n\r\t`?\\<>|:]*

E.8 simpleType TimeFormat

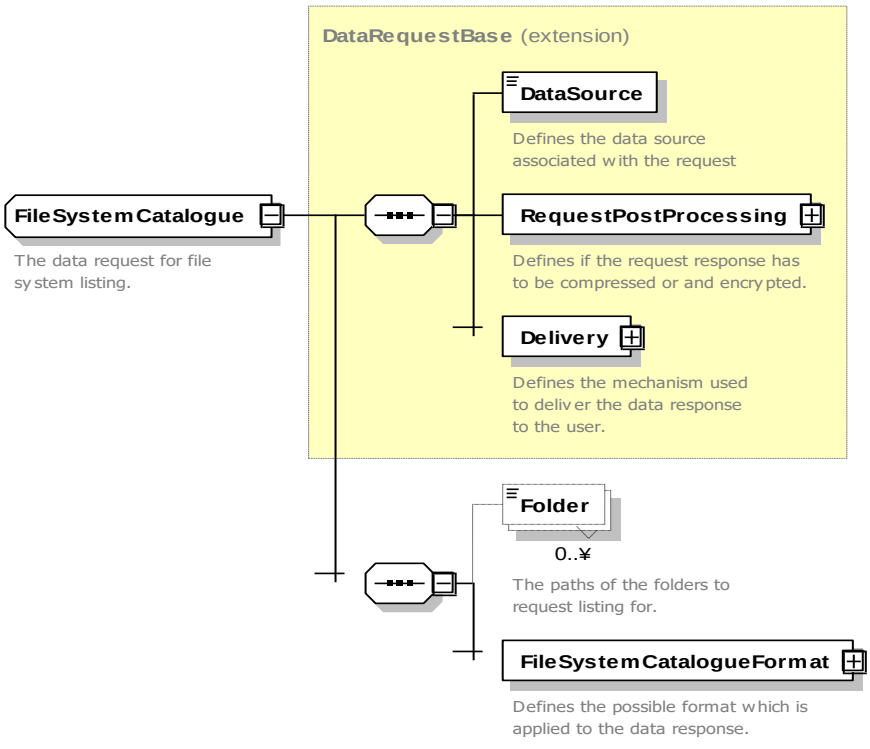
namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value enumeration CCSDS enumeration DOY enumeration CUSTOM

Appendix FFile System Data Types

F.1 complexType FileFilterEntry

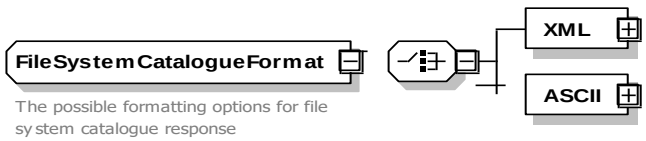
diagram	
namespace	http://edds.egos.esa/model
children	**FileName Folder TimeModified**
used by	element **FileSystemFile/FileFilterEntry**
annotation	documentation Filter element that defines the files to match from the data source.

F.2 complexType FileSystemCatalogue

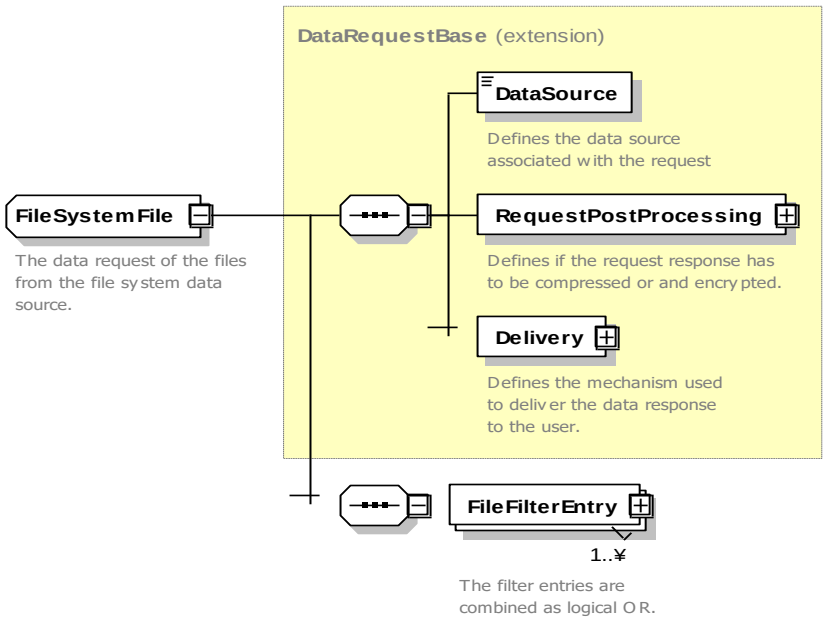
diagram	

namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase abstract true
children	**DataSource RequestPostProcessing Delivery Folder FileSystemCatalogueFormat**
used by	complexTypes FileSystemFileCatalogue FileSystemFolderCatalogue
annotation	documentation The data request for file system listing.

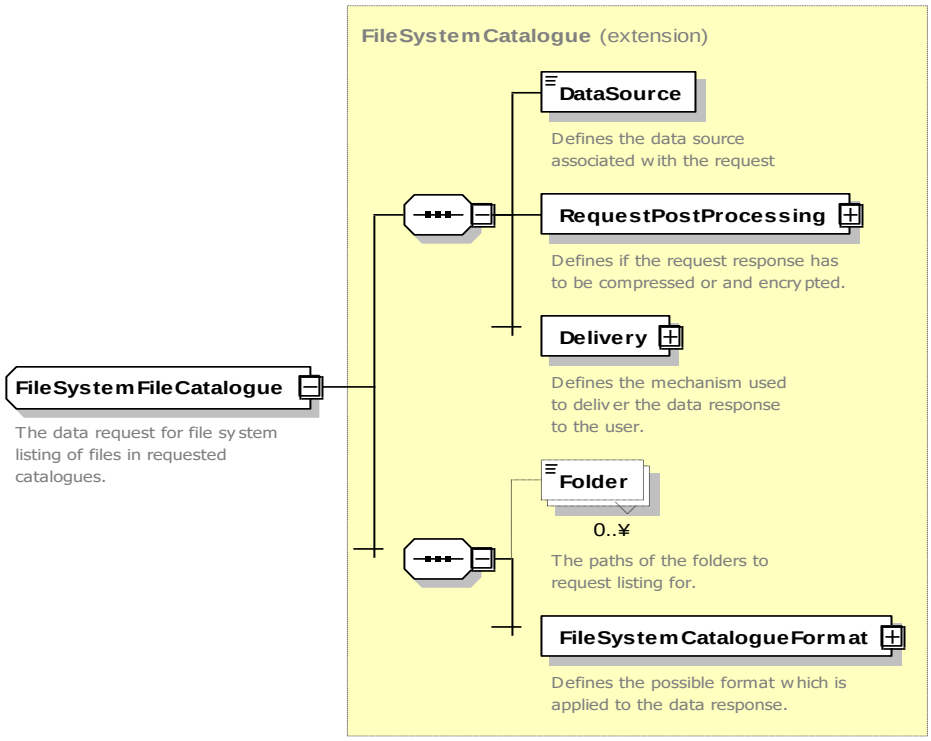
F.3 complexType FileSystemCatalogueFormat

diagram	
namespace	http://edds.egos.esa/model
children	**XML ASCII**
used by	element **FileSystemCatalogue/FileSystemCatalogueFormat**
annotation	documentation The possible formatting options for file system catalogue response

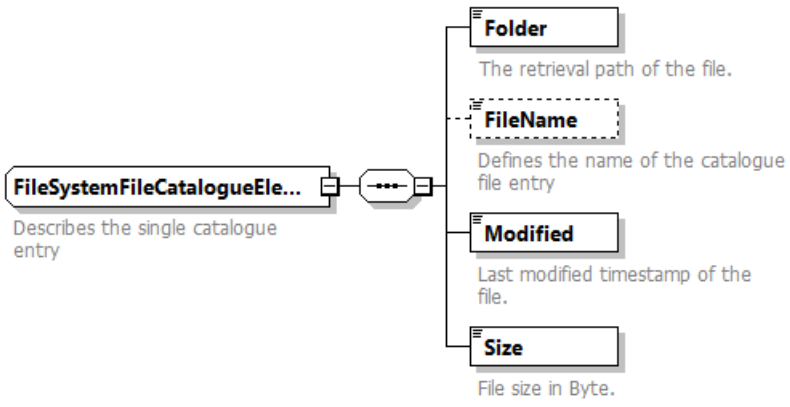
F.4 complexType FileSystemFile

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**DataSource RequestPostProcessing Delivery FileFilterEntry**
annotation	documentation The data request of the files from the file system data source.

F.5 complexType FileSystemFileCatalogue

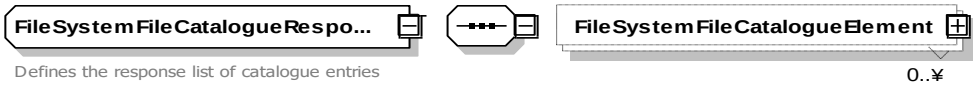
diagram	
namespace	http://edds.egos.esa/model
type	extension of FileSystemCatalogue
properties	base FileSystemCatalogue
children	**DataSource RequestPostProcessing Delivery Folder FileSystemCatalogueFormat**
annotation	documentation The data request for file system listing of files in requested catalogues.

F.6 complexType FileSystemFileCatalogueElement

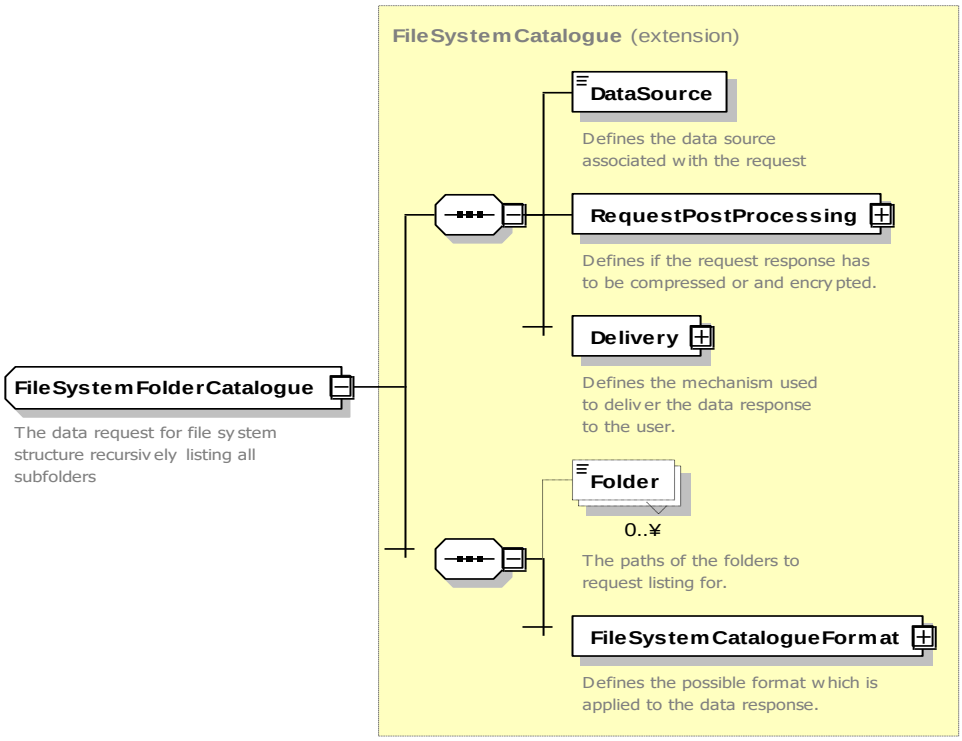
diagram	
namespace	http://edds.egos.esa/model
children	**Folder FileName Modified Size**

used by	elements **FileSystemFileCatalogueElement** **FileSystemFileCatalogueResponse/FileSystemFileCatalogueElement**
annotation	documentation Describes the single catalogue entry

F.7 complexType FileSystemFileCatalogueResponse

diagram	
namespace	http://edds.egos.esa/model
children	**FileSystemFileCatalogueElement**
annotation	documentation Defines the response list of catalogue entries

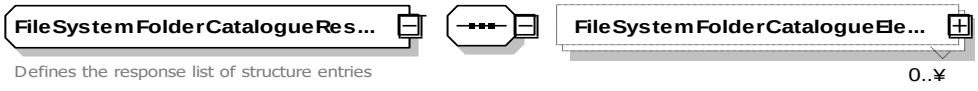
F.8 complexType FileSystemFolderCatalogue

diagram	
namespace	http://edds.egos.esa/model
type	extension of **FileSystemCatalogue**
properties	base FileSystemCatalogue
children	**DataSource RequestPostProcessing Delivery Folder FileSystemCatalogueFormat**
annotation	documentation The data request for file system structure recursively listing all subfolders

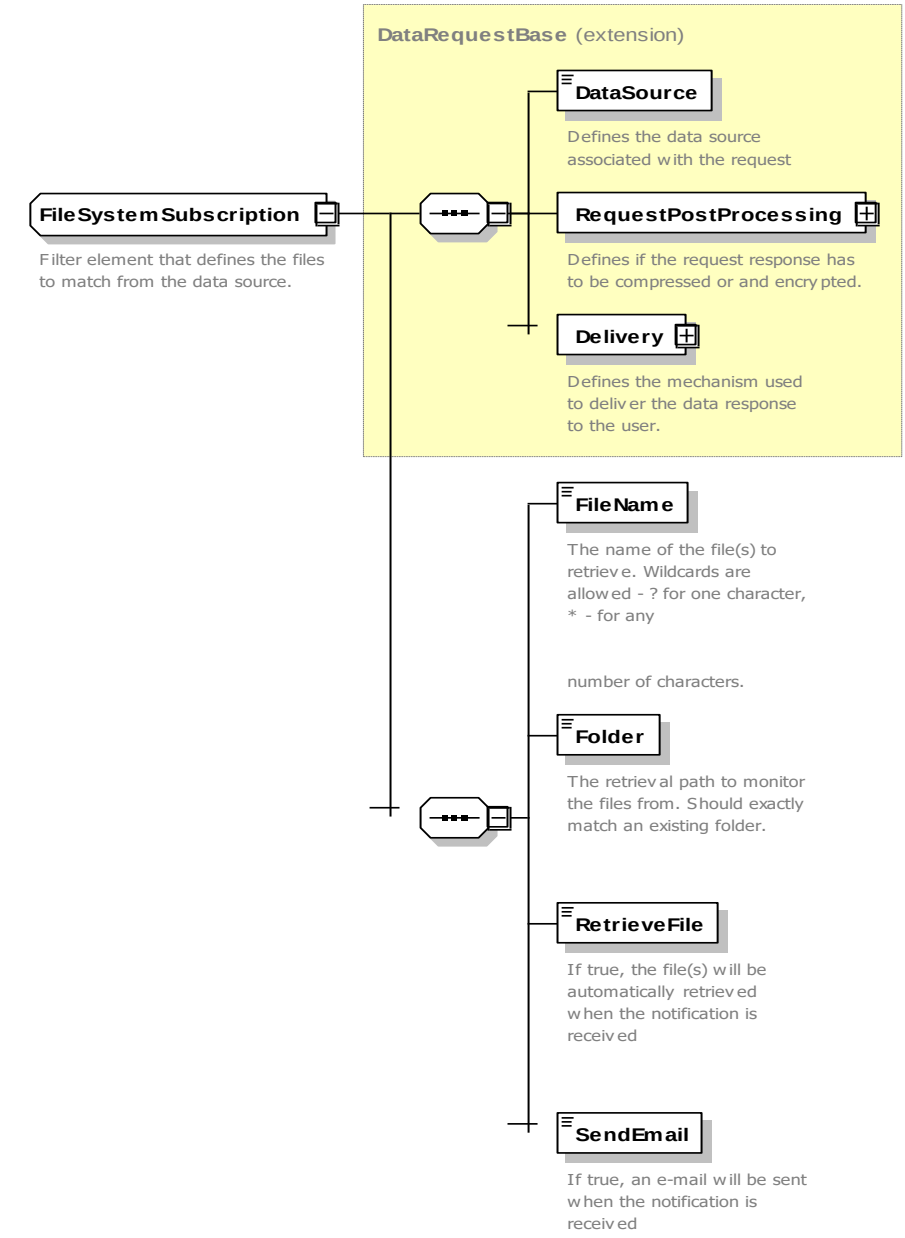
F.9 complexType FileSystemFolderCatalogueElement

diagram	
namespace	http://edds.egos.esa/model
children	**Folder**
used by	elements **FileSystemFolderCatalogueElement** **FileSystemFolderCatalogueResponse/FileSystemFolderCatalogueElement**
annotation	documentation Describes the single structure response entry

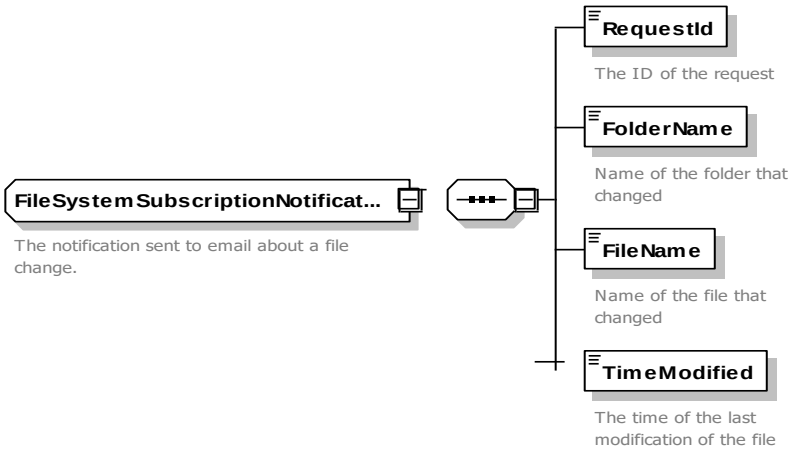
F.10 complexType FileSystemFolderCatalogueResponse

diagram	
namespace	http://edds.egos.esa/model
children	**FileSystemFolderCatalogueElement**
annotation	documentation Defines the response list of structure entries

F.11 complexType FileSystemSubscription

<table>
<tr><td>diagram</td><td> 
</td></tr>
<tr><td>namespace</td><td>http://edds.egos.esa/model</td></tr>
<tr><td>type</td><td>extension of DataRequestBase</td></tr>
<tr><td>properties</td><td>base DataRequestBase</td></tr>
<tr><td>children</td><td>DataSource RequestPostProcessing Delivery FileName Folder RetrieveFile SendEmail</td></tr>
<tr><td>annotation</td><td>documentation
Filter element that defines the files to match from the data source.</td></tr>
</table>

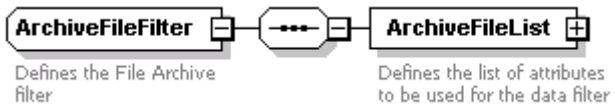
F.12 complexType FileSystemSubscriptionNotification

diagram	
namespace	http://edds.egos.esa/model
children	**RequestId FolderName FileName TimeModified**
used by	element **FileSystemSubscriptionNotification**
annotation	documentation The notification sent to email about a file change.

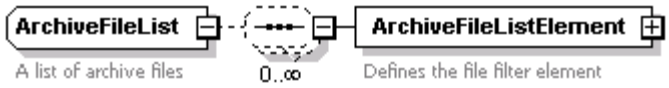
Appendix G Filter Data Type

This section of the document describes the filter complex types as defined in the filter.xsd file.

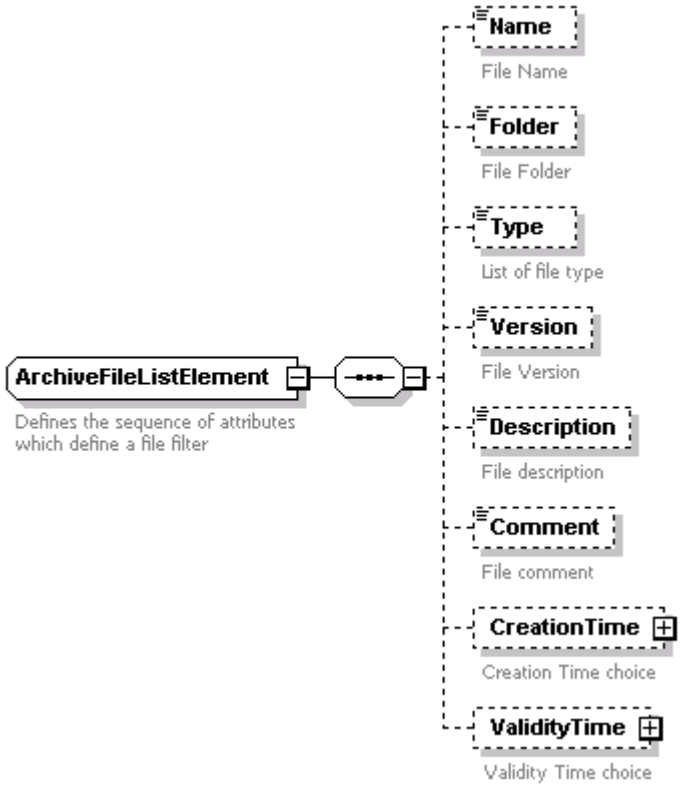
G.1 complexType ArchiveFileFilter

diagram	
namespace	http://edds.egos.esa/model
children	**ArchiveFileList**
annotation	documentation Defines the File Archive filter

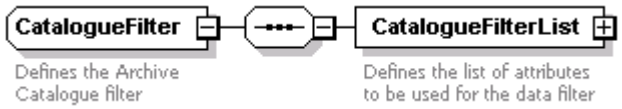
G.2 complexType ArchiveFileList

diagram	
namespace	http://edds.egos.esa/model
children	**ArchiveFileListElement**
used by	element **ArchiveFileFilter/ArchiveFileList**
annotation	documentation A list of archive files

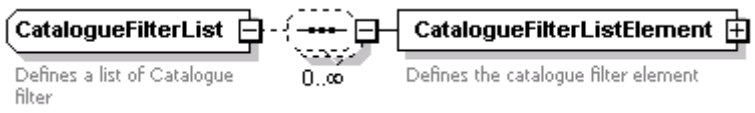
G.3 complexType ArchiveFileListElement

diagram	
namespace	http://edds.egos.esa/model
children	**Name Folder Type Version Description Comment CreationTime ValidityTime**
used by	element **ArchiveFileList/ArchiveFileListElement**
annotation	documentation Defines the sequence of attributes which define a file filter

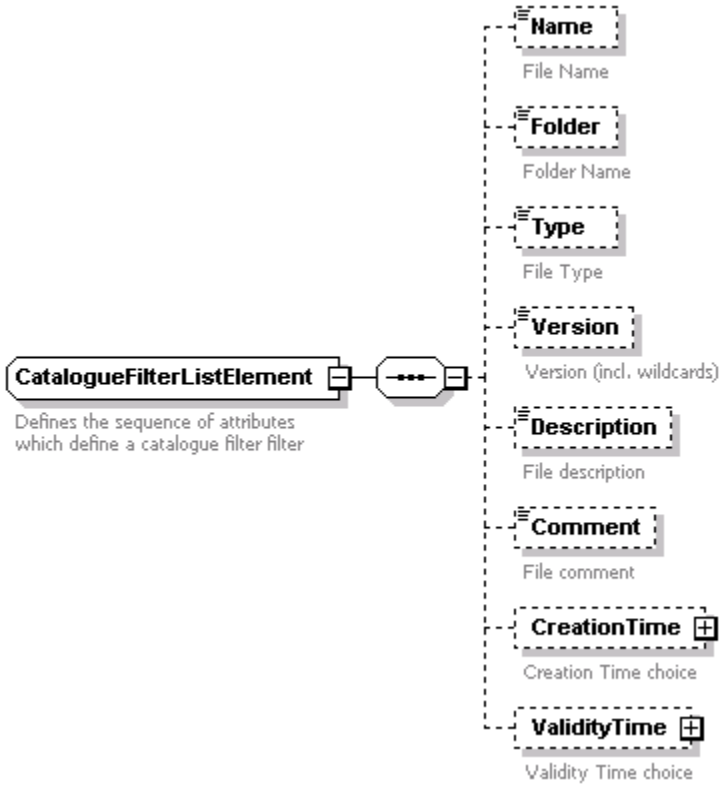
G.4 complexType CatalogueFilter

diagram	
namespace	http://edds.egos.esa/model
children	**CatalogueFilterList**
annotation	documentation Defines the Archive Catalogue filter

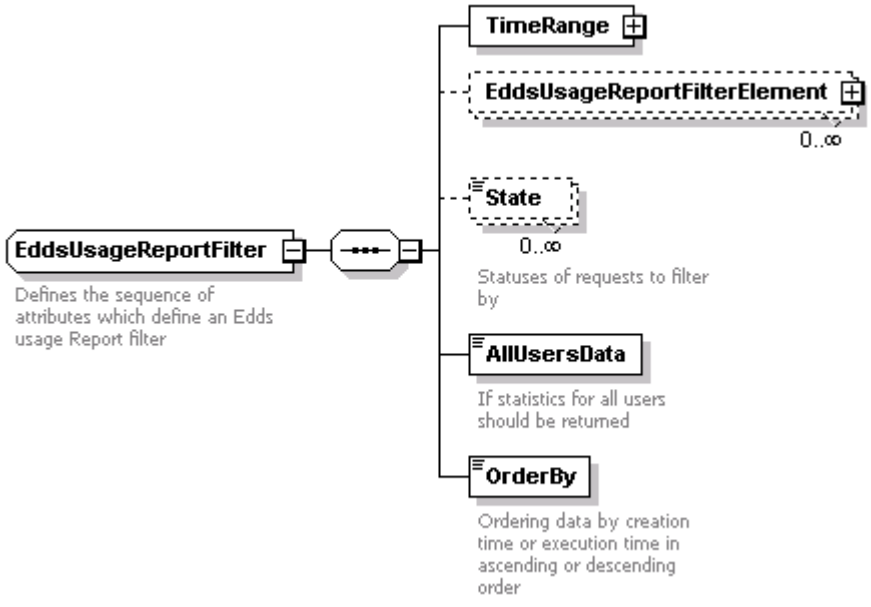
G.5 complexType CatalogueFilterList

diagram	
namespace	http://edds.egos.esa/model
children	**CatalogueFilterListElement**
used by	element **CatalogueFilter/CatalogueFilterList**
annotation	documentation Defines a list of Catalogue filter

G.6 complexType CatalogueFilterListElement

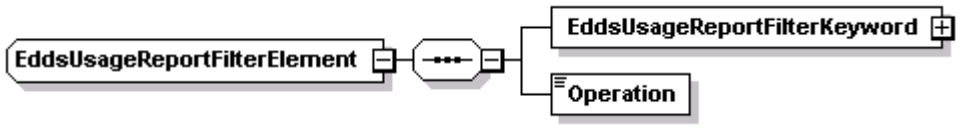
diagram	
namespace	http://edds.egos.esa/model
children	**Name Folder Type Version Description Comment CreationTime ValidityTime**
used by	element **CatalogueFilterList/CatalogueFilterListElement**
annotation	documentation Defines the sequence of attributes which define a catalogue filter filter

G.7 complexType EddsUsageReportFilter

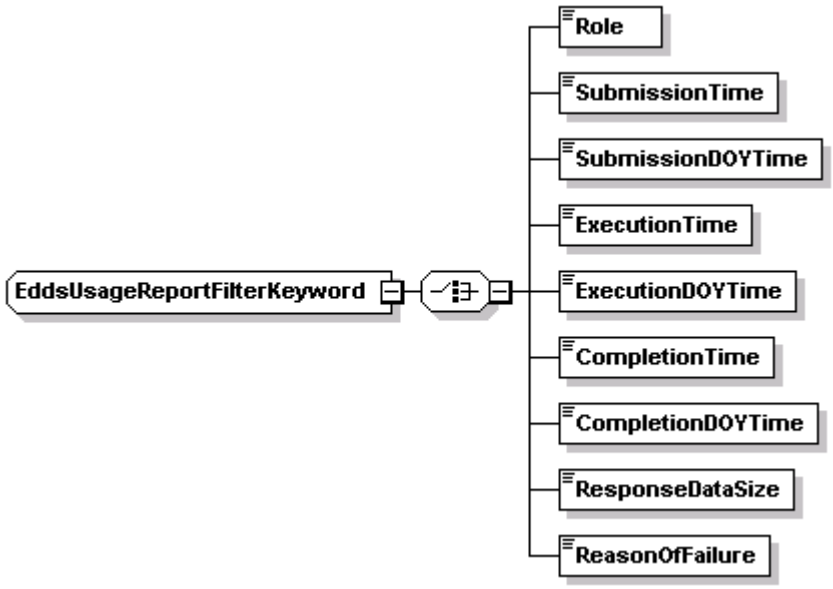
diagram	

namespace	http://edds.egos.esa/model
children	**TimeRange EddsUsageReportFilterElement State AllUsersData OrderBy**
annotation	documentation Defines the sequence of attributes which define an Edds usage Report filter

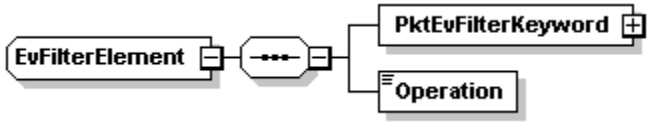
G.8 complexType EddsUsageReportFilterElement

diagram	
namespace	http://edds.egos.esa/model
children	**EddsUsageReportFilterKeyword Operation**
used by	element **EddsUsageReportFilter/EddsUsageReportFilterElement**

G.9 complexType EddsUsageReportFilterKeyword

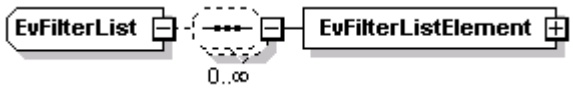
diagram	
namespace	http://edds.egos.esa/model
children	**Role SubmissionTime SubmissionDOYTime ExecutionTime ExecutionDOYTime CompletionTime CompletionDOYTime ResponseDataSize ReasonOfFailure**
used by	element **EddsUsageReportFilterElement/EddsUsageReportFilterKeyword**

G.10 complexType EvFilterElement

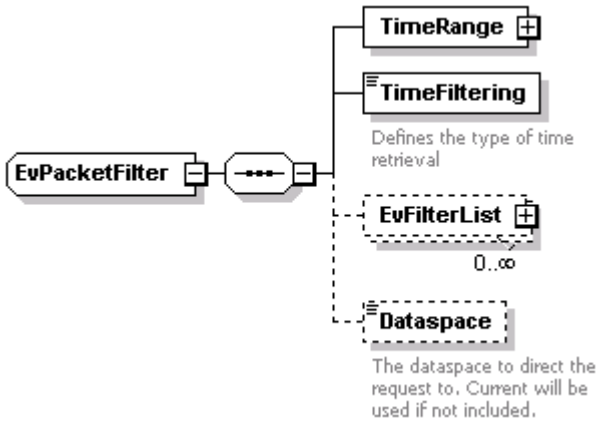
diagram	

namespace	http://edds.egos.esa/model
children	**PktEvFilterKeyword Operation**
used by	element **EvFilterList/EvFilterListElement**

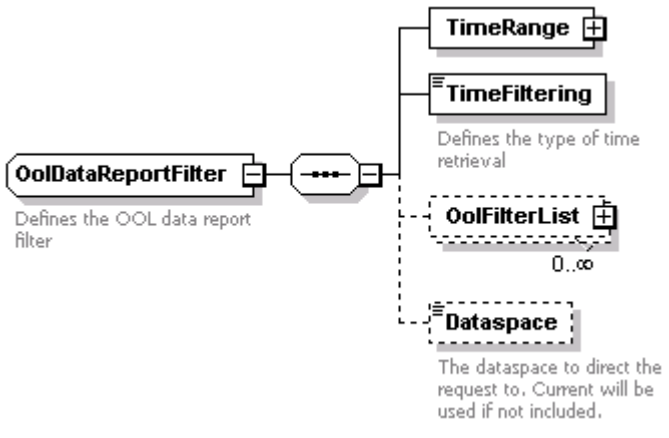
G.11 complexType EvFilterList

diagram	
namespace	http://edds.egos.esa/model
children	**EvFilterListElement**
used by	element **EvPacketFilter/EvFilterList**

G.12 complexType EvPacketFilter

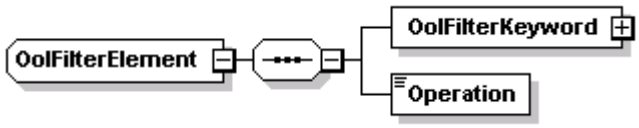
diagram	
namespace	http://edds.egos.esa/model
children	**TimeRange TimeFiltering EvFilterList Dataspace**

G.13 complexType OolDataReportFilter

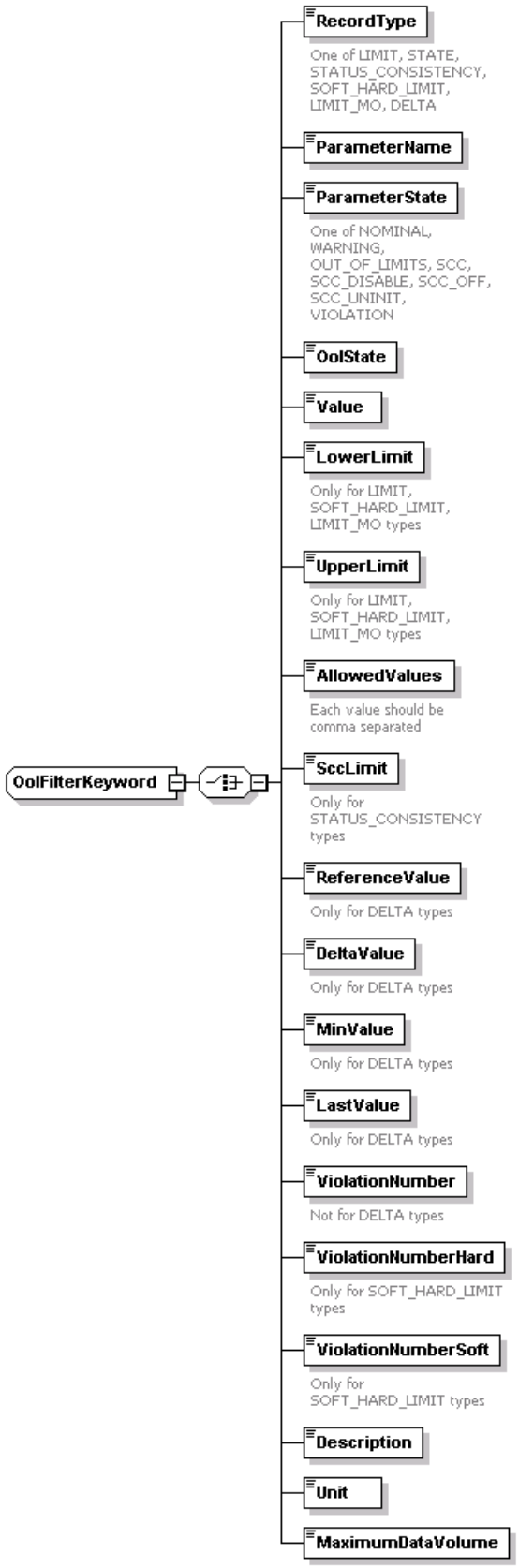
diagram	

namespace	http://edds.egos.esa/model
children	**TimeRange TimeFiltering OolFilterList Dataspace**
annotation	documentation Defines the OOL data report filter

G.14 complexType OolFilterElement

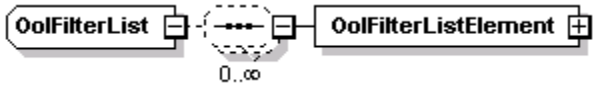
diagram	
namespace	http://edds.egos.esa/model
children	**OolFilterKeyword Operation**
used by	element **OolFilterList/OolFilterListElement**

G.15 complexType OolFilterKeyword

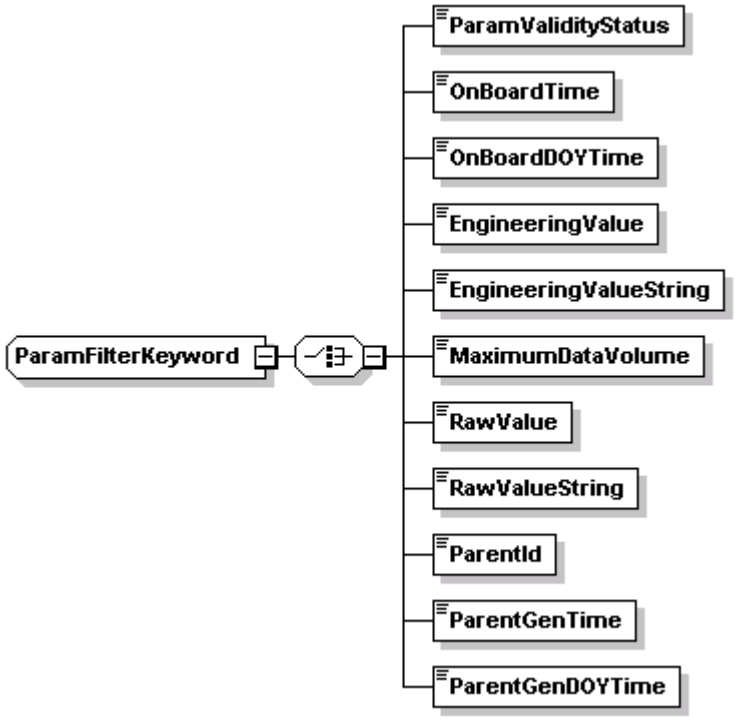
diagram	
namespace	http://edds.egos.esa/model
children	**RecordType ParameterName ParameterState OolState Value LowerLimit UpperLimit AllowedValues SccLimit ReferenceValue DeltaValue MinValue LastValue ViolationNumber ViolationNumberHard ViolationNumberSoft**

	Description Unit MaximumDataVolume
used by	element **OolFilterElement/OolFilterKeyword**

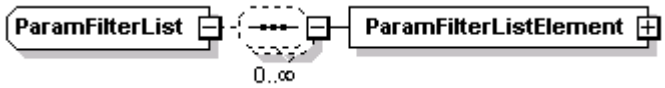
G.16 complexType OolFilterList

diagram	
namespace	http://edds.egos.esa/model
children	**OolFilterListElement**
used by	element **OolDataReportFilter/OolFilterList**

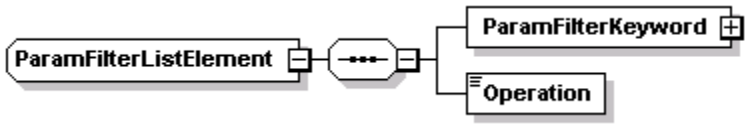
G.17 complexType ParamFilterKeyword

diagram	
namespace	http://edds.egos.esa/model
children	**ParamValidityStatus OnBoardTime OnBoardDOYTime EngineeringValue EngineeringValueString MaximumDataVolume RawValue RawValueString ParentId ParentGenTime ParentGenDOYTime**
used by	element **ParamFilterListElement/ParamFilterKeyword**

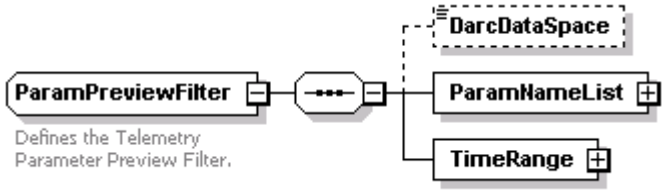
G.18 complexType ParamFilterList

diagram	
namespace	http://edds.egos.esa/model
children	**ParamFilterListElement**
used by	element **ParamTmFilter/ParamFilterList**

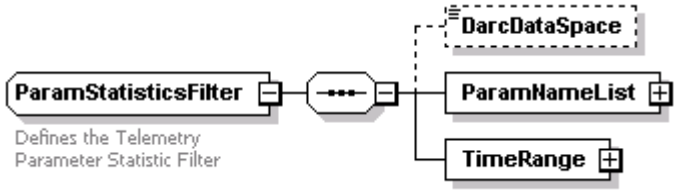
G.19 complexType ParamFilterListElement

diagram	
namespace	http://edds.egos.esa/model
children	**ParamFilterKeyword Operation**
used by	element **ParamFilterList/ParamFilterListElement**

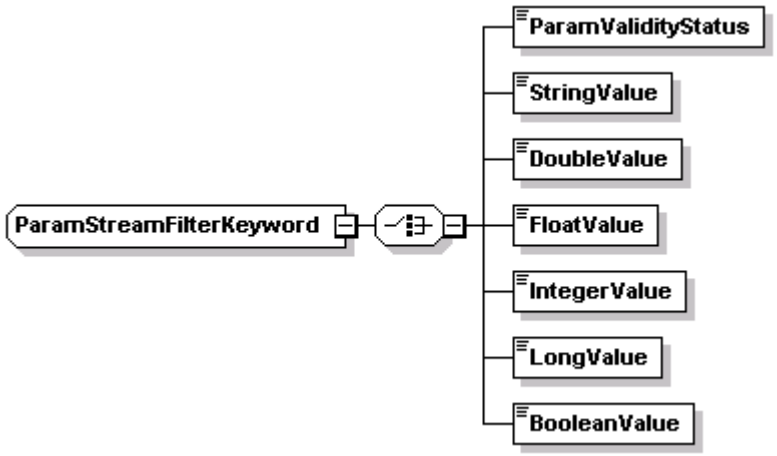
G.20 complexType ParamPreviewFilter

diagram	
namespace	http://edds.egos.esa/model
children	**DarcDataSpace ParamNameList TimeRange**
annotation	documentation Defines the Telemetry Parameter Preview Filter.

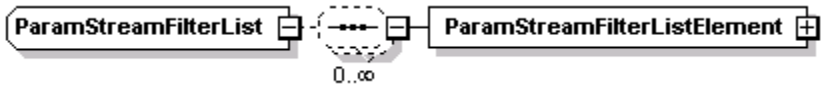
G.21 complexType ParamStatisticsFilter

diagram	
namespace	http://edds.egos.esa/model
children	**DarcDataSpace ParamNameList TimeRange**
annotation	documentation Defines the Telemetry Parameter Statistic Filter

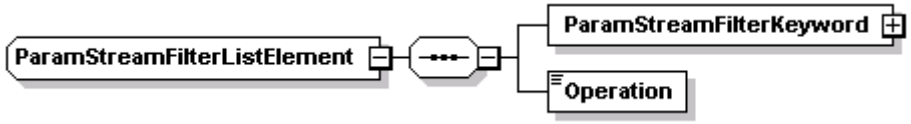
G.22 complexType ParamStreamFilterKeyword

diagram	
namespace	http://edds.egos.esa/model
children	**ParamValidityStatus StringValue DoubleValue FloatValue IntegerValue LongValue BooleanValue**
used by	element **ParamStreamFilterListElement/ParamStreamFilterKeyword**

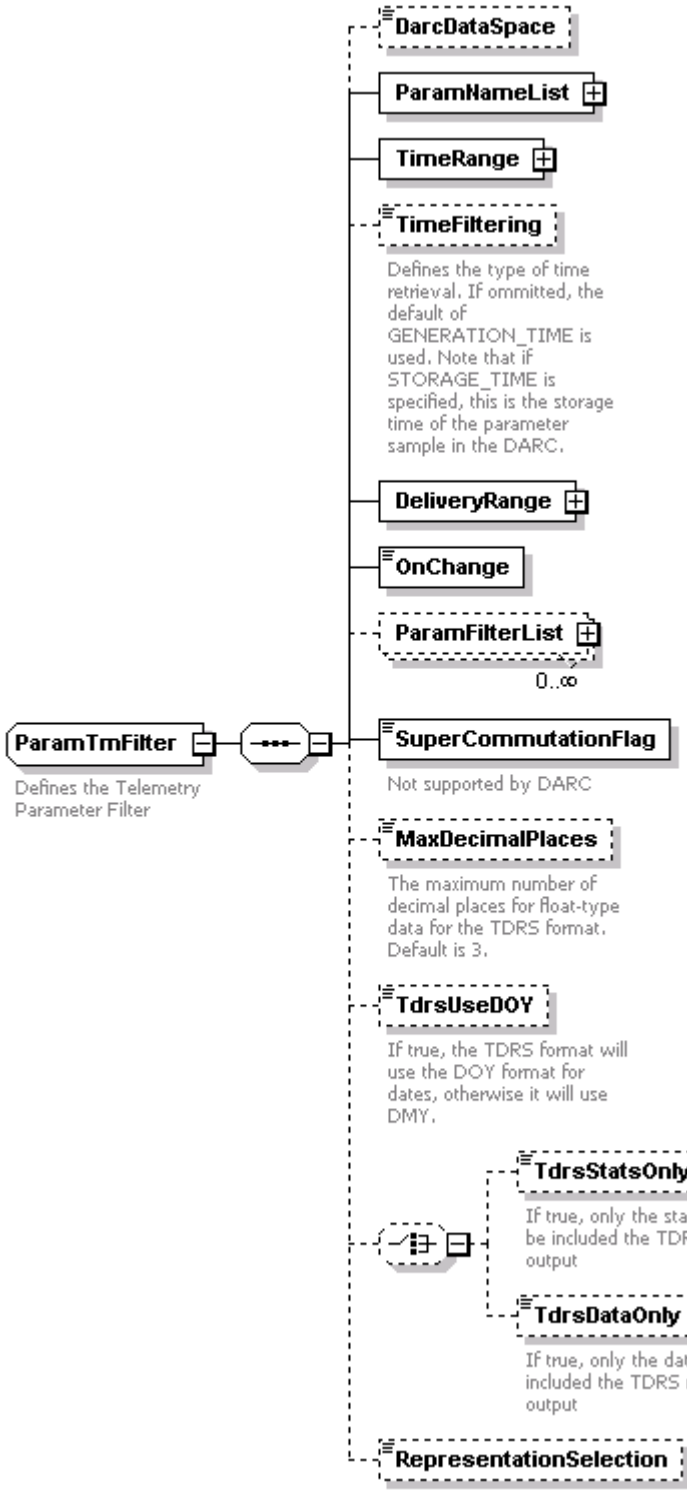
G.23 complexType ParamStreamFilterList

diagram	
namespace	http://edds.egos.esa/model
children	**ParamStreamFilterListElement**

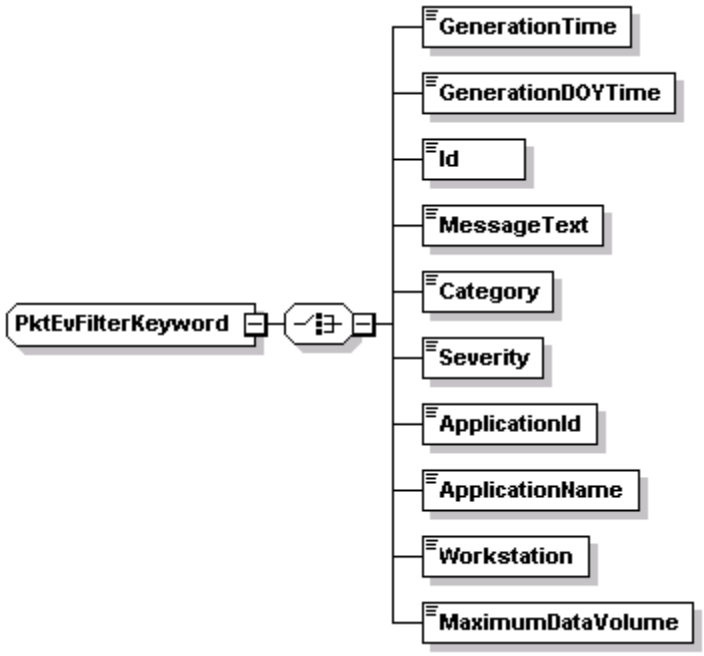
G.24 complexType ParamStreamFilterListElement

diagram	
namespace	http://edds.egos.esa/model
children	**ParamStreamFilterKeyword Operation**
used by	element **ParamStreamFilterList/ParamStreamFilterListElement**

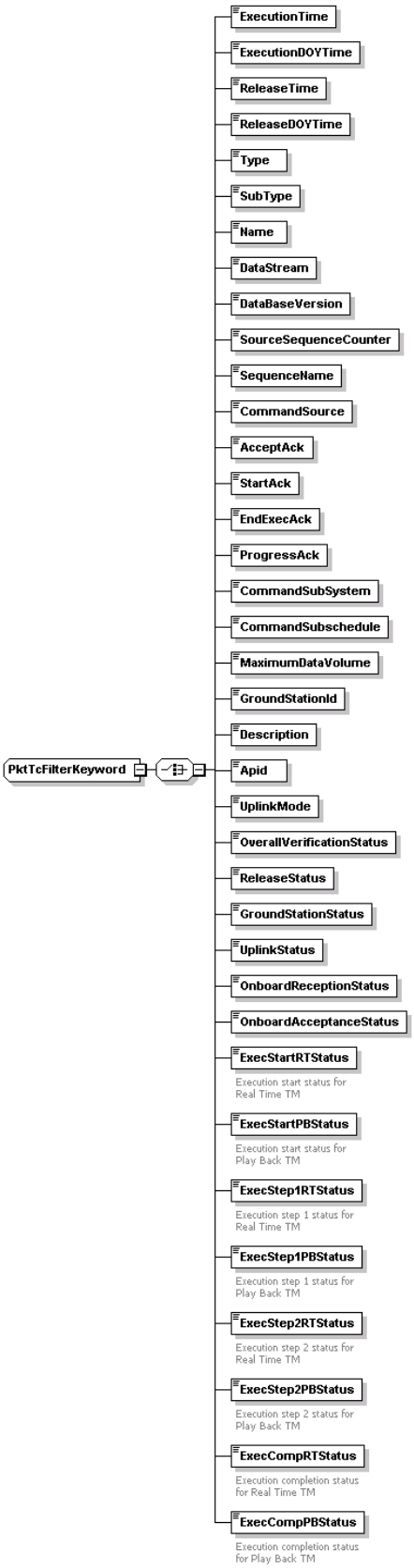
G.25 complexType ParamTmFilter

diagram	
namespace	http://edds.egos.esa/model
children	**DarcDataSpace ParamNameList TimeRange TimeFiltering DeliveryRange OnChange ParamFilterList SuperCommutationFlag MaxDecimalPlaces TdrsUseDOY TdrsStatsOnly TdrsDataOnly RepresentationSelection**
annotation	documentation Defines the Telemetry Parameter Filter

G.26 complexType PktEvFilterKeyword

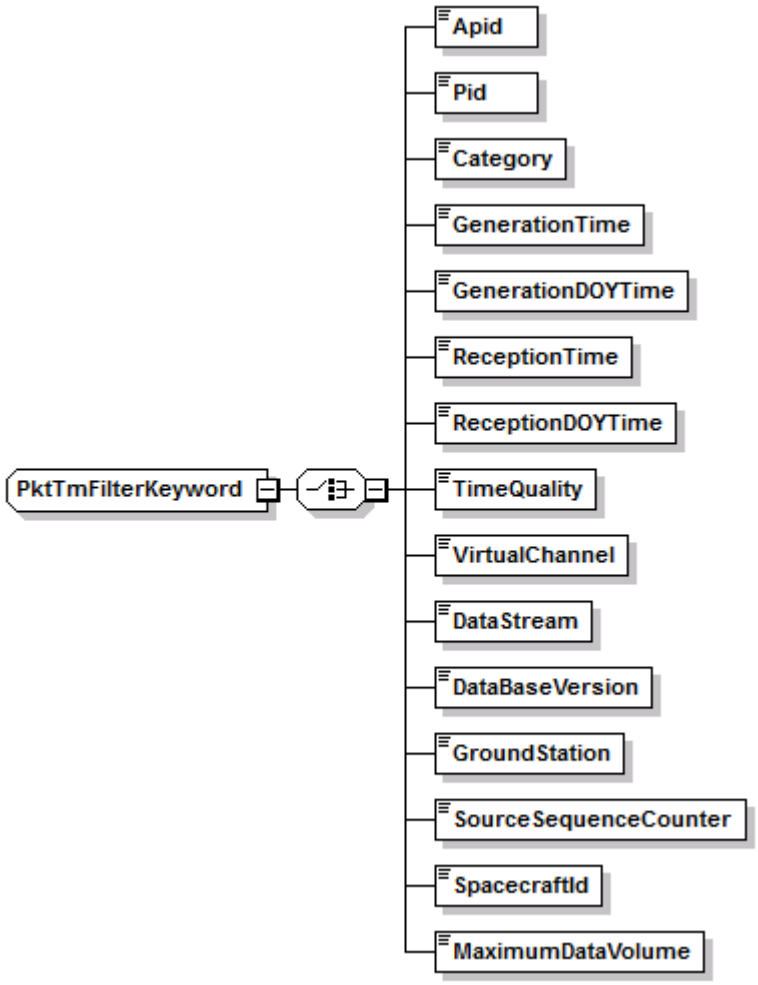
diagram	
namespace	http://edds.egos.esa/model
children	GenerationTime GenerationDOYTime Id MessageText Category Severity ApplicationId ApplicationName Workstation MaximumDataVolume
used by	element **EvFilterElement/PktEvFilterKeyword**

G.27 complexType PktTcFilterKeyword

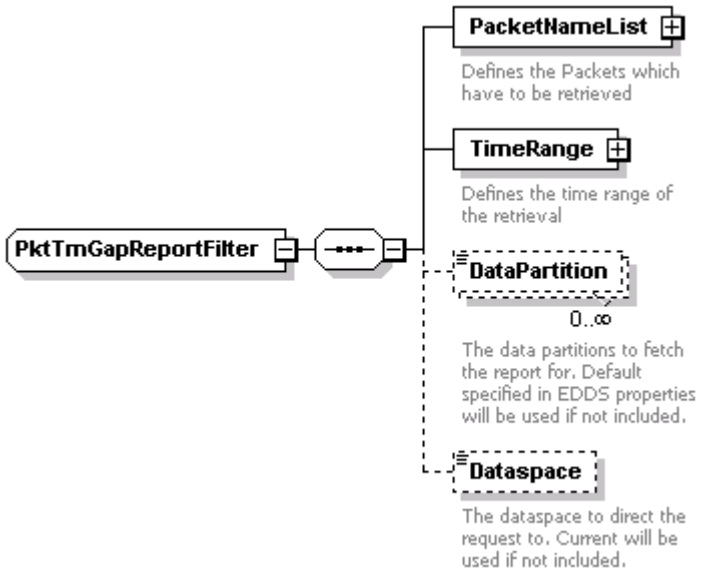
diagram	
namespace	http://edds.egos.esa/model

children	ExecutionTime ExecutionDOYTime ReleaseTime ReleaseDOYTime Type SubType Name DataStream DataBaseVersion SourceSequenceCounter SequenceName CommandSource AcceptAck StartAck EndExecAck ProgressAck CommandSubSystem CommandSubschedule MaximumDataVolume GroundStationId Description Apid UplinkMode OverallVerificationStatus ReleaseStatus GroundStationStatus UplinkStatus OnboardReceptionStatus OnboardAcceptanceStatus ExecStartRTStatus ExecStartPBStatus ExecStep1RTStatus ExecStep1PBStatus ExecStep2RTStatus ExecStep2PBStatus ExecCompRTStatus ExecCompPBStatus
used by	element **TcFilterElement/PktTcFilterKeyword**

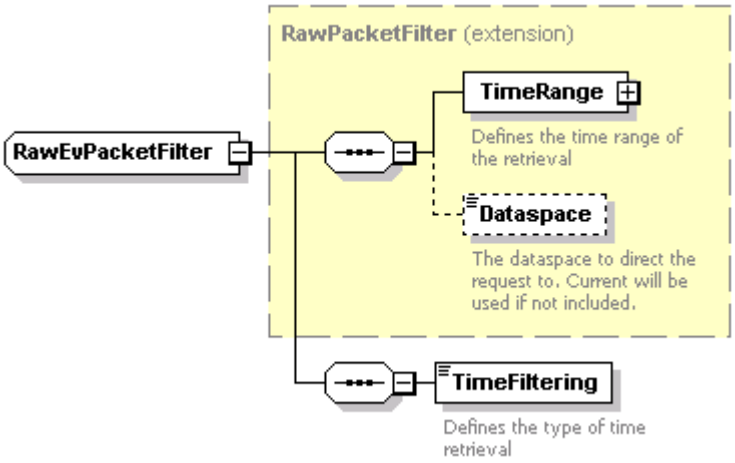
G.28 complexType PktTmFilterKeyword

diagram	
namespace	http://edds.egos.esa/model
children	**Apid Pid Category GenerationTime GenerationDOYTime ReceptionTime ReceptionDOYTime TimeQuality VirtualChannel DataStream DataBaseVersion GroundStation SourceSequenceCounter SpacecraftId MaximumDataVolume**
used by	element **TmFilterElement/PktTmFilterKeyword**

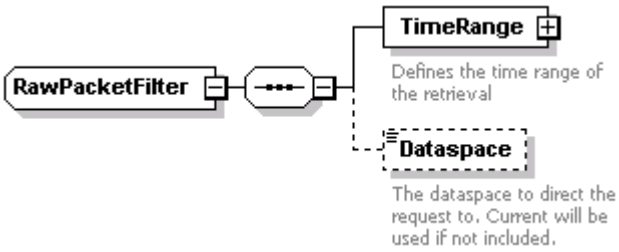
G.29 complexType PktTmGapReportFilter

diagram	
namespace	http://edds.egos.esa/model
children	PacketNameList TimeRange DataPartition Dataspace

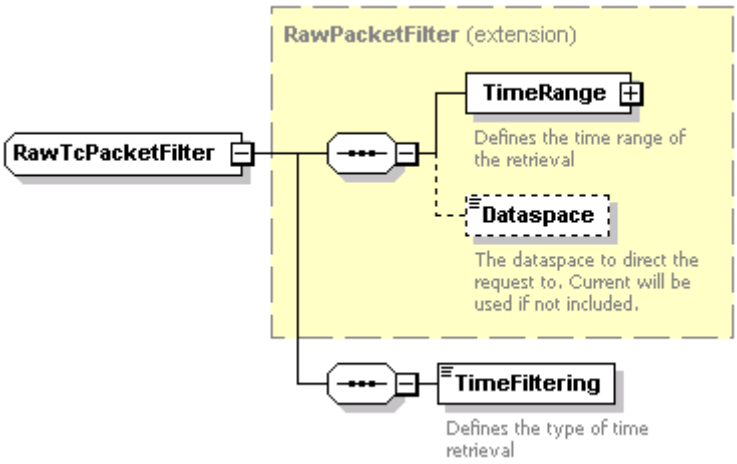
G.30 complexType RawEvPacketFilter

diagram	
namespace	http://edds.egos.esa/model
type	extension of **RawPacketFilter**
properties	base RawPacketFilter
children	**TimeRange Dataspace TimeFiltering**
used by	complexType **RawTmPacketFilter**

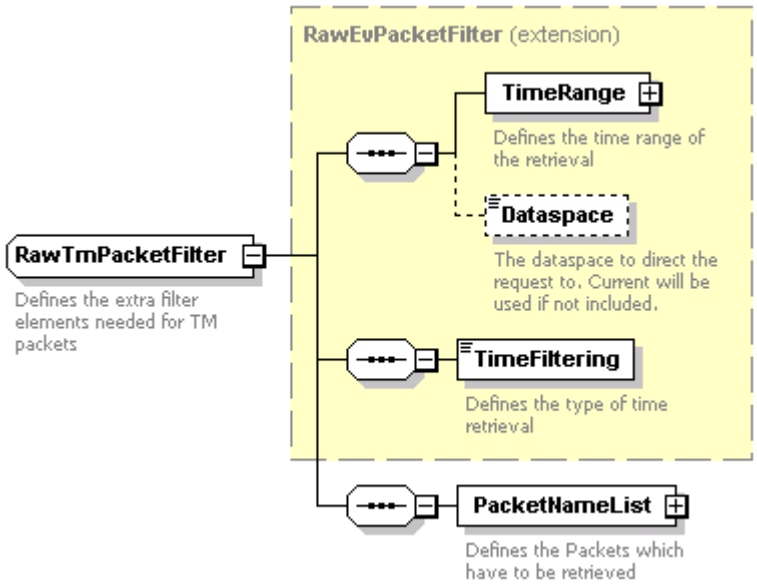
G.31 complexType RawPacketFilter

diagram	
namespace	http://edds.egos.esa/model
properties	abstract true
children	**TimeRange Dataspace**
used by	complexTypes **RawEvPacketFilter RawTcPacketFilter**

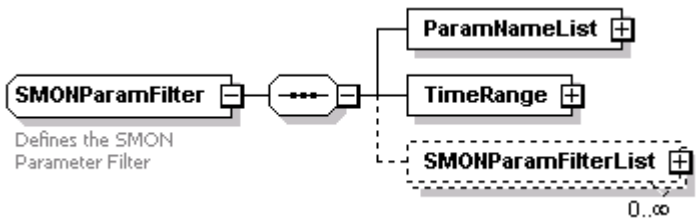
G.32 complexType RawTcPacketFilter

diagram	
namespace	http://edds.egos.esa/model
type	extension of **RawPacketFilter**
properties	base RawPacketFilter
children	**TimeRange Dataspace TimeFiltering**

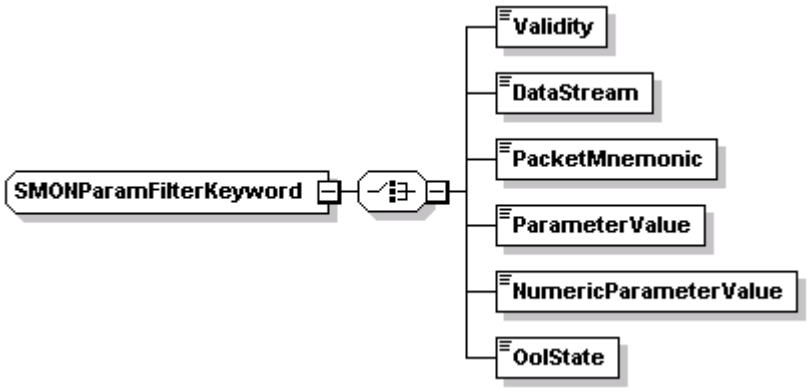
G.33 complexType RawTmPacketFilter

diagram	
namespace	http://edds.egos.esa/model
type	extension of **RawEvPacketFilter**
properties	base RawEvPacketFilter
children	**TimeRange Dataspace TimeFiltering PacketNameList**
annotation	documentation Defines the extra filter elements needed for TM packets

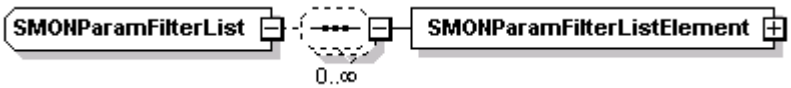
G.34 complexType SMONParamFilter

diagram	
namespace	http://edds.egos.esa/model
children	**ParamNameList TimeRange SMONParamFilterList**
used by	element **SMONParam/SMONParamFilter**
annotation	documentation Defines the SMON Parameter Filter

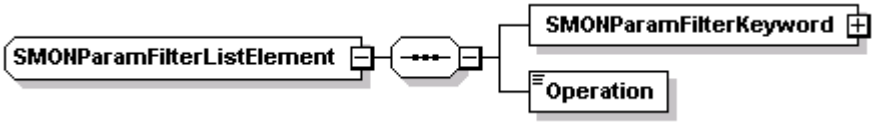
G.35 complexType SMONParamFilterKeyword

diagram	
namespace	http://edds.egos.esa/model
children	**Validity DataStream PacketMnemonic ParameterValue NumericParameterValue OolState**
used by	element **SMONParamFilterListElement/SMONParamFilterKeyword**

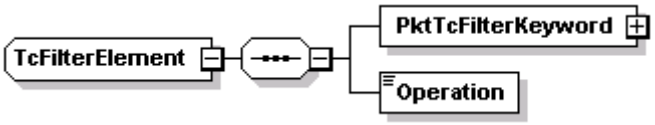
G.36 complexType SMONParamFilterList

diagram	
namespace	http://edds.egos.esa/model
children	**SMONParamFilterListElement**
used by	element **SMONParamFilter/SMONParamFilterList**

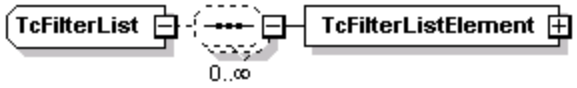
G.37 complexType SMONParamFilterListElement

diagram	
namespace	http://edds.egos.esa/model
children	**SMONParamFilterKeyword Operation**
used by	element **SMONParamFilterList/SMONParamFilterListElement**

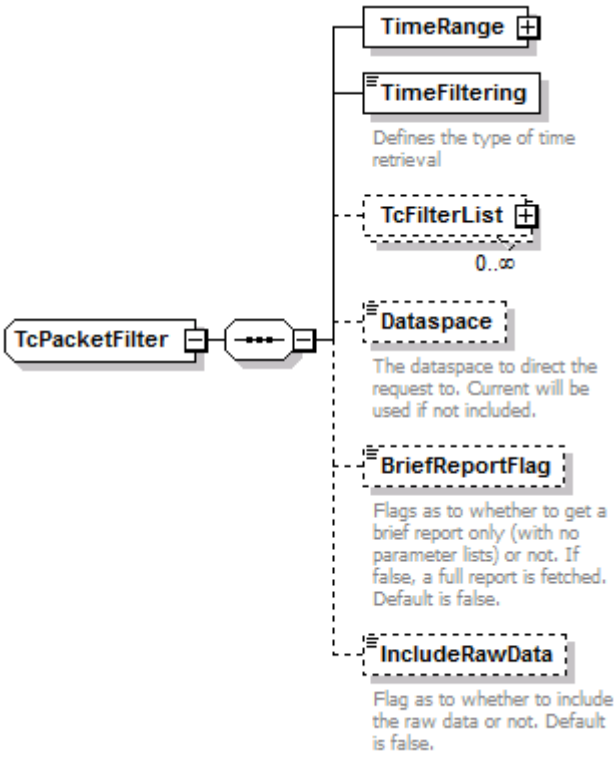
G.38 complexType TcFilterElement

diagram	
namespace	http://edds.egos.esa/model
children	**PktTcFilterKeyword Operation**
used by	element **TcFilterList/TcFilterListElement**

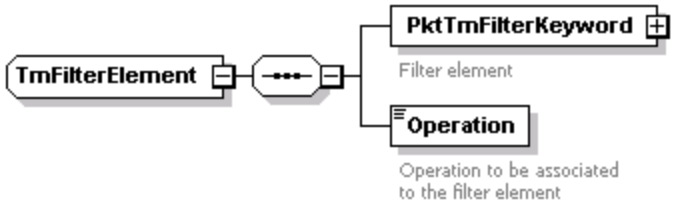
G.39 complexType TcFilterList

diagram	
namespace	http://edds.egos.esa/model
children	**TcFilterListElement**
used by	element **TcPacketFilter/TcFilterList**

G.40 complexType TcPacketFilter

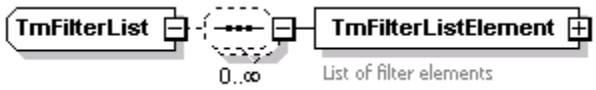
diagram	
namespace	**http://edds.egos.esa/model**
children	**TimeRange TimeFiltering TcFilterList Dataspace BriefReportFlag IncludeRawData**

G.41 complexType TmFilterElement

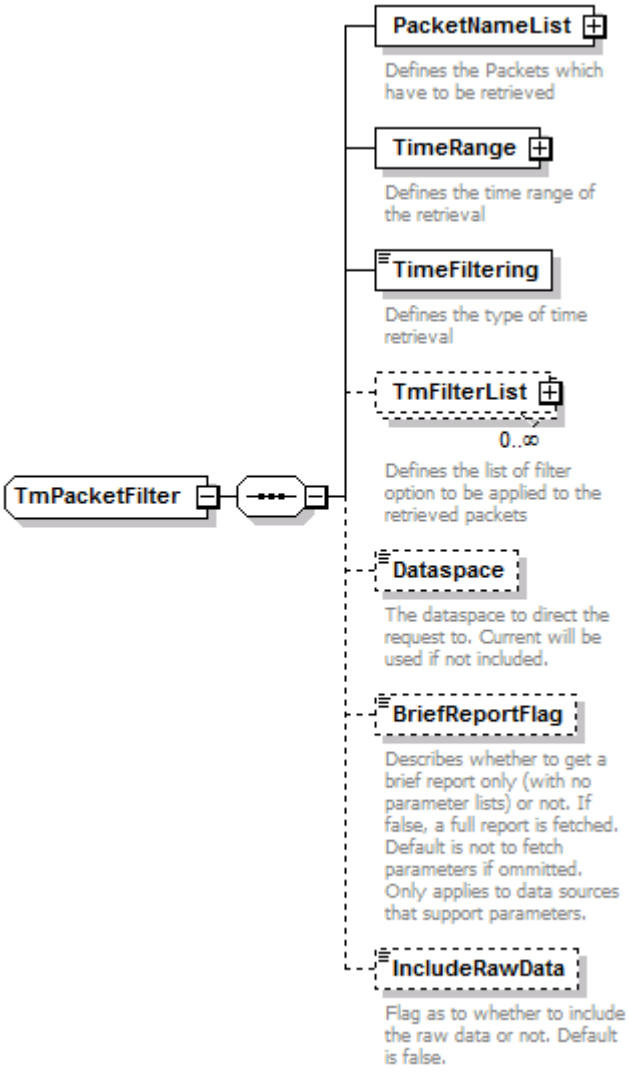
diagram	

namespace	http://edds.egos.esa/model
children	**PktTmFilterKeyword Operation**
used by	element **TmFilterList/TmFilterListElement**

G.42 complexType TmFilterList

diagram	
namespace	http://edds.egos.esa/model
children	**TmFilterListElement**
used by	element **TmPacketFilter/TmFilterList**

G.43 complexType TmPacketFilter

diagram	
namespace	http://edds.egos.esa/model

children	**PacketNameList TimeRange TimeFiltering TmFilterList Dataspace BriefReportFlag IncludeRawData**

G.44 simpleType CommandSource

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	element **PktTcFilterKeyword/CommandSource**
facets	Kind Value enumeration MANUAL_STACK enumeration AUTO_STACK enumeration EXT_SOURCE enumeration TC_SPACON enumeration OBQM_DISP enumeration SMF_SOURCE enumeration NAVPRS_STACK enumeration MKMF_STACK enumeration PKMF_STACK enumeration SKMF_STACK

G.45 simpleType RepresentationSelection

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	element **SMONParamFilter**
facets	Kind Value enumeration ENG enumeration RAW enumeration ALL

G.46 simpleType DarcRepresentationSelection

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	element **ParamTmFilter**
facets	Kind Value enumeration ENG enumeration RAW

G.47 simpleType StageStatus

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**

properties	base xs:string
used by	elements **PktTcFilterKeyword/ExecCompPBStatus PktTcFilterKeyword/ExecCompRTStatus PktTcFilterKeyword/ExecStartPBStatus PktTcFilterKeyword/ExecStartRTStatus PktTcFilterKeyword/ExecStep1PBStatus PktTcFilterKeyword/ExecStep1RTStatus PktTcFilterKeyword/ExecStep2PBStatus PktTcFilterKeyword/ExecStep2RTStatus PktTcFilterKeyword/GroundStationStatus PktTcFilterKeyword/OnboardAcceptanceStatus PktTcFilterKeyword/OnboardReceptionStatus PktTcFilterKeyword/OverallVerificationStatus PktTcFilterKeyword/ReleaseStatus PktTcFilterKeyword/UplinkStatus**
facets	Kind Value annotation enumeration documentation Not applicable enumeration I documentation I for idle enumeration P documentation P for pending enumeration S documentation S for success enumeration F documentation F for failed enumeration U documentation U for unverified enumeration X documentation X for unknown enumeration T documentation T for timeout enumeration E documentation E for superseded enumeration N documentation N for uncertain failed enumeration V documentation V for uncertain successful enumeration C documentation C for affected enumeration D documentation D for disabled enumeration A documentation A for assumed passed enumeration SCC documentation SCC for Status Consistency Check

G.48 simpleType TcTimeFiltering

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	elements **TcPacketFilter/TimeFiltering RawTcPacketFilter/TimeFiltering**
facets	Kind Value annotation enumeration RELEASE_TIME documentation Command release time of the packet (Primary Key) enumeration EXECUTION_TIME documentation Predicted or actual execution time of the packet (Secondary Key)

G.49 simpleType TimeFiltering

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	elements **EvPacketFilter/TimeFiltering OolDataReportFilter/TimeFiltering TmPacketFilter/TimeFiltering RawEvPacketFilter/TimeFiltering**
facets	Kind Value annotation enumeration GENERATION_TIME documentation On board Time of the packet (Primary Key)

	enumeration	STORAGE_TIME	documentation Reception time of the packet into the archive (Secondary Key)

G.50 simpleType UplinkMode

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	elements **PktTcFilterKeyword/UplinkMode**
facets	Kind Value annotation enumeration A documentation AD transmission mode enumeration B documentation BD transmission mode

G.51 simpleType Severity

namespace	http://edds.egos.esa/model
type	**Severity**
type	restriction of **xs:string**
properties	base xs:string
used by	elements **PktEvFilterKeyword/Severity**
facets	Kind Value annotation enumeration Warning enumeration Error enumeration Information enumeration Fatal

G.52 simpleType Category

namespace	http://edds.egos.esa/model
type	**Category**
type	restriction of **xs:string**
properties	base xs:string
used by	elements **PktEvFilterKeyword/Category**
facets	Kind Value annotation enumeration Log enumeration Software enumeration System enumeration MIB

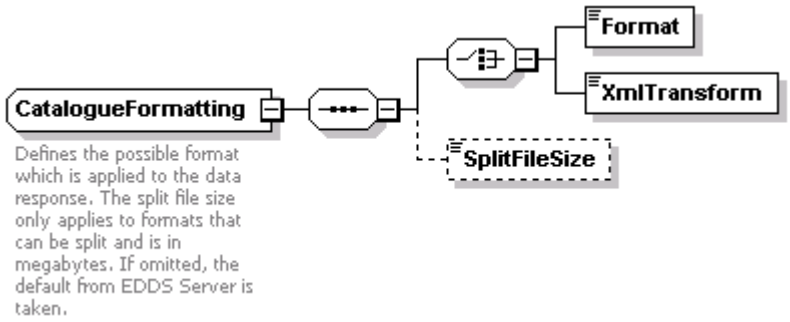
Appendix H Format Data Type

H.1 group SplittableFormat

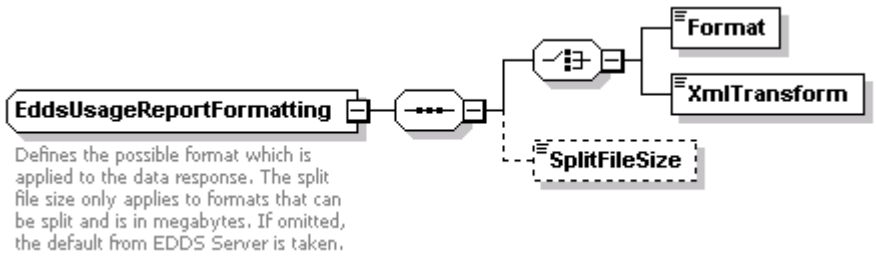
diagram	
namespace	http://edds.egos.esa/model
children	**SplitFileSize**
annotation	documentation Defines the base properties for splittable formats.

This section of the document describes the format types as defined in the format.xsd file.

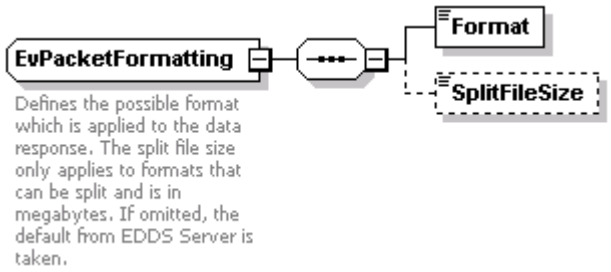
H.2 complexType CatalogueFormatting

diagram	
namespace	http://edds.egos.esa/model
children	**Format XmlTransform SplitFileSize**
annotation	documentation Defines the possible format which is applied to the data response.

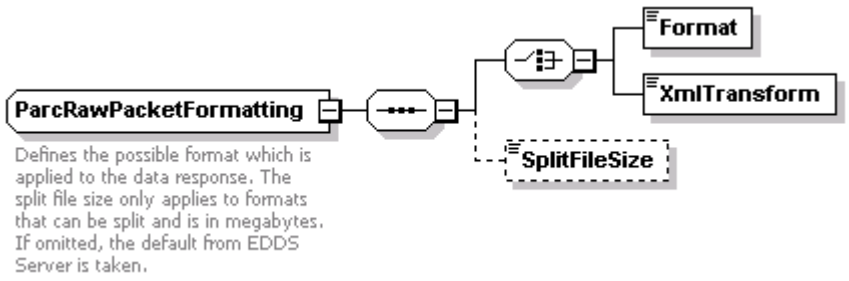
H.3 complexType EddsUsageReportFormatting

diagram	
namespace	http://edds.egos.esa/model
children	**Format XmlTransform SplitFileSize**
annotation	documentation Defines the possible format which is applied to the data response.

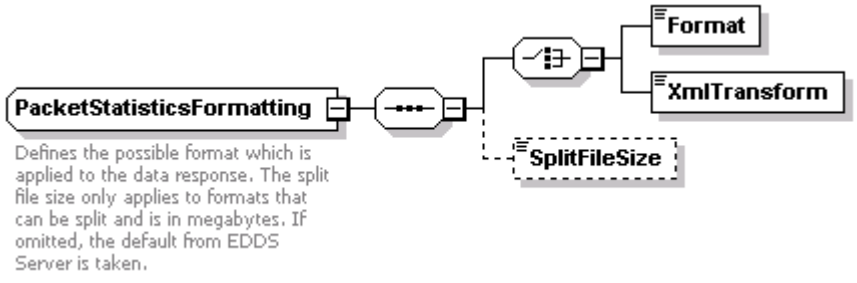
H.4 complexType EvPacketFormatting

diagram	
namespace	http://edds.egos.esa/model
children	**Format SplitFileSize**
annotation	documentation Defines the possible format which is applied to the data response. The split file size only applies to formats that can be split and is in megabytes. If omitted, the default from EDDS Server is taken.

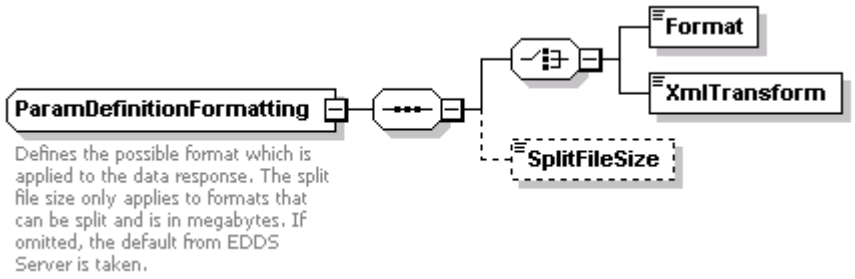
H.5 complexType PacketFormatting

diagram	
namespace	http://edds.egos.esa/model
children	**Format XmlTransform SplitFileSize**
annotation	documentation Defines the possible format which is applied to the data response. The split file size only applies to formats that can be split and is in megabytes. If omitted, the default from EDDS Server is taken.

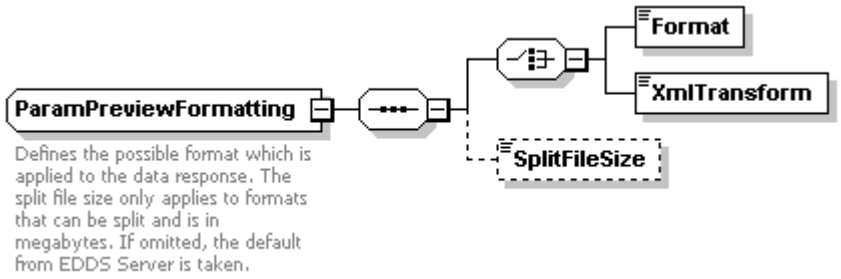
H.6 complexType PacketStatisticsFormatting

diagram	
namespace	http://edds.egos.esa/model
children	**Format XmlTransform SplitFileSize**
annotation	documentation Defines the possible format which is applied to the data response.

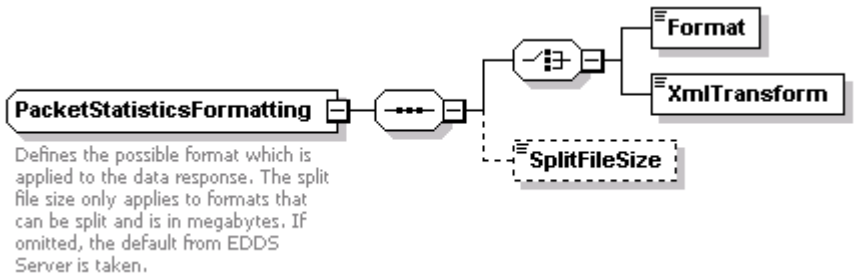
H.7 complexType ParamDefinitionFormatting

diagram	
namespace	http://edds.egos.esa/model
children	**Format XmlTransform SplitFileSize**
annotation	documentation Defines the possible format which is applied to the data response.

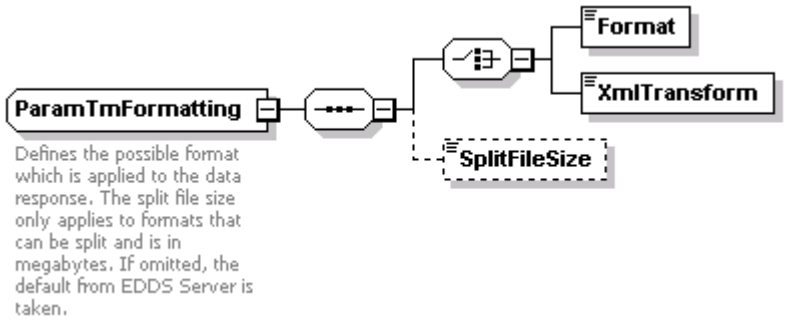
H.8 complexType ParamPreviewFormatting

diagram	
namespace	http://edds.egos.esa/model
children	**Format XmlTransform SplitFileSize**
annotation	documentation Defines the possible format which is applied to the data response.

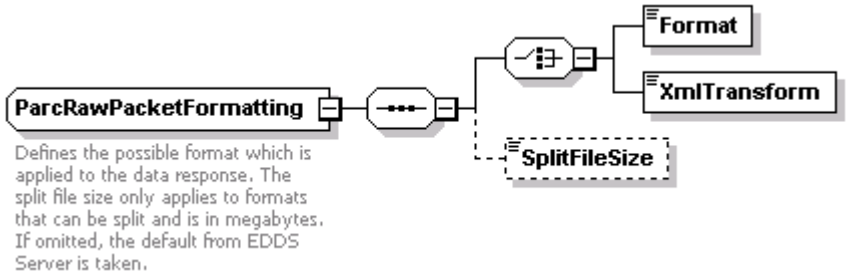
H.9 complexType ParamStatisticsFormatting

diagram	
namespace	http://edds.egos.esa/model
children	**Format XmlTransform SplitFileSize**
annotation	documentation Defines the possible format which is applied to the data response.

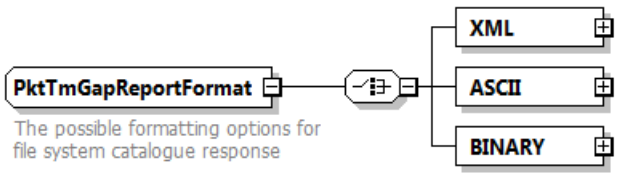
H.10 complexType ParamTmFormatting

diagram	
namespace	http://edds.egos.esa/model
children	**Format XmlTransform SplitFileSize**
annotation	documentation Defines the possible format which is applied to the data response.

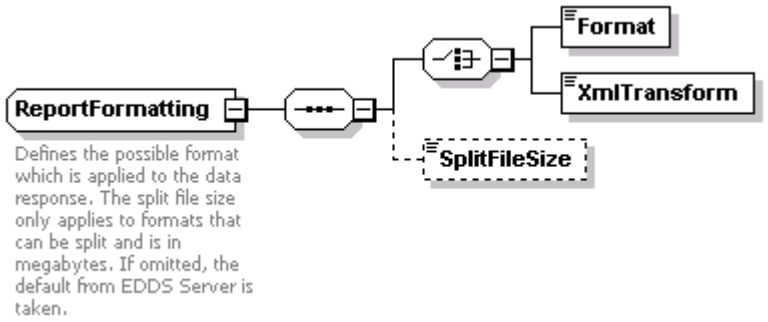
H.11 complexType ParcRawPacketFormatting

diagram	
namespace	http://edds.egos.esa/model
children	**Format XmlTransform SplitFileSize**
annotation	documentation Defines the possible format which is applied to the data response.

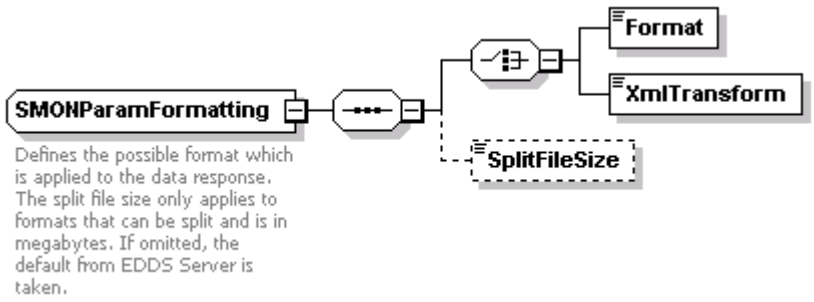
H.12 complexType PktTmGapReportFormat

diagram	
namespace	http://edds.egos.esa/model
children	**XML ASCII BINARY**
annotation	documentation The possible formatting options for file system catalogue response

H.13 complexType ReportFormatting

diagram	
namespace	http://edds.egos.esa/model
children	**Format XmlTransform SplitFileSize**
annotation	documentation Defines the possible format which is applied to the data response.

H.14 complexType SMONParamFormatting

diagram	
namespace	http://edds.egos.esa/model
children	**Format XmlTransform SplitFileSize**
annotation	documentation Defines the possible format which is applied to the data response.

H.15 simpleType ArchiveFileFormat

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value annotation enumeration BINARY
annotation	documentation Defines the list of supported format of the data response.

H.16 simpleType CatalogueFormat

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	element **CatalogueFormatting/Format**

facets	Kind Value annotation enumeration XML enumeration ASCII
annotation	documentation Defines the list of supported format of the data response.

H.17 simpleType EddsUsageReportFormat

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	element **EddsUsageReportFormatting/Format**
facets	Kind Value annotation enumeration XML
annotation	documentation Defines the list of supported format of the data response.

H.18 simpleType EvPacketFormat

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value annotation enumeration EDDS_RAW
annotation	documentation Defines the list of supported format of the data response.

H.19 simpleType PacketFormat

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
facets	Kind Value annotation enumeration EDDS_RAW enumeration SFDU enumeration GDDS_BINARY
annotation	documentation Defines the list of supported format of the data response.

H.20 simpleType PacketStatisticsFormat

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	element **PacketStatisticsFormatting/Format**

facets	Kind Value annotation enumeration XML
annotation	documentation Defines the list of supported format of the data response.

H.21 simpleType ParamDefinitionFormat

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	element **ParamDefinitionFormatting/Format**
facets	Kind Value annotation enumeration XML enumeration BINARY
annotation	documentation Defines the list of supported format of the data response.

H.22 simpleType ParamPreviewFormat

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	element **ParamPreviewFormatting/Format**
facets	Kind Value annotation enumeration XML enumeration ASCII
annotation	documentation Defines the list of supported format of the data response.

H.23 simpleType ParamStatisticsFormat

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	element **ParamStatisticsFormatting/Format**
facets	Kind Value annotation enumeration XML enumeration ASCII
annotation	documentation Defines the list of supported format of the data response.

H.24 simpleType ParamTmFormat

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**

properties	base xs:string
used by	element **ParamTmFormatting/Format**
facets	Kind Value annotation enumeration XML documentation XML format enumeration TDRS documentation TDRS spreadsheet format
annotation	documentation Defines the list of supported format of the data response.

H.25 simpleType ParcRawPacketFormat

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	element **ParcRawPacketFormatting/Format**
facets	Kind Value annotation enumeration XML enumeration BINARY
annotation	documentation Defines the list of supported format of the data response.

H.26 simpleType ReportFormat

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	element **ReportFormatting/Format**
facets	Kind Value annotation enumeration XML enumeration ASCII enumeration BINARY
annotation	documentation Defines the list of supported format of the data response.

H.27 simpleType SMONParamFormat

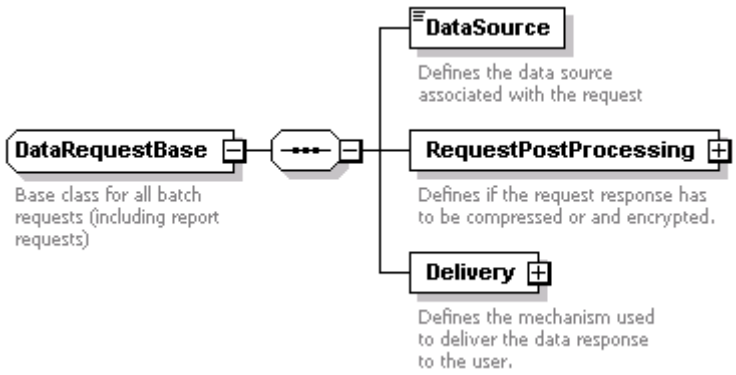
namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	element **SMONParamFormatting/Format**
facets	Kind Value annotation enumeration XML documentation XML format enumeration DARC_XML documentation DARC XML format enumeration DARC_BINARY documentation DARC binary format enumeration TDRS documentation TDRS spreadsheet format enumeration BINARY documentation Binary format

annotation	documentation Defines the list of supported format of the data response.

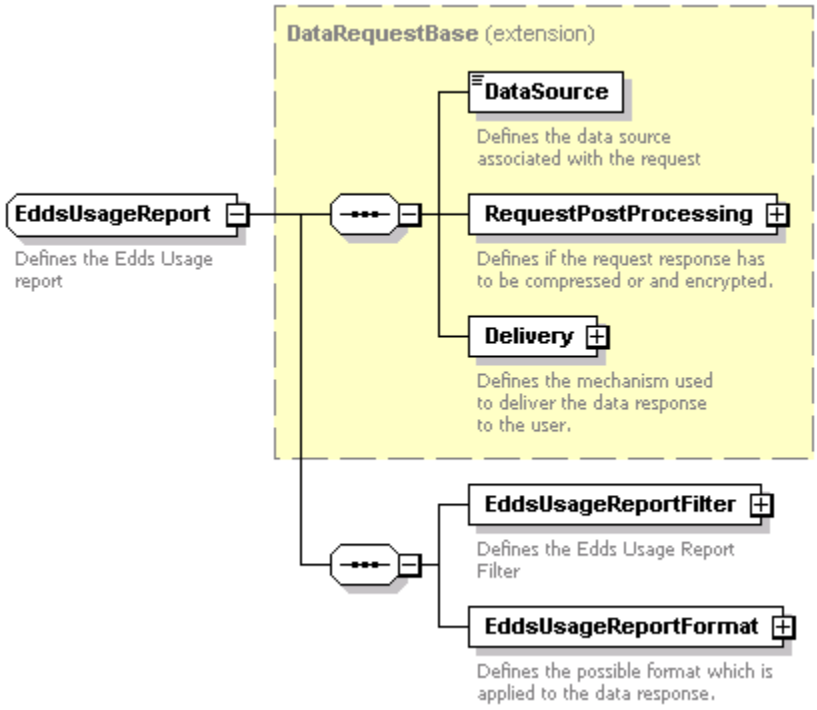
Appendix I Report Data Type

This section of the document describes the report complex types as defined in the report.xsd file.

I.1 complexType DataRequestBase

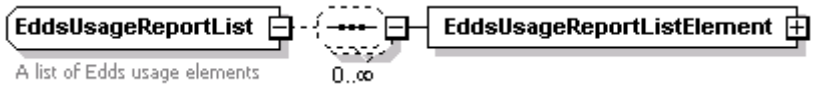
diagram	
namespace	http://edds.egos.esa/model
children	**DataSource RequestPostProcessing Delivery**
used by	complexTypes **EddsUsageReport EventRecordReport OolRecordReport PktTcReport PktTmReport**
annotation	documentation Base class for all batch requests (including report requests)

I.2 complexType EddsUsageReport

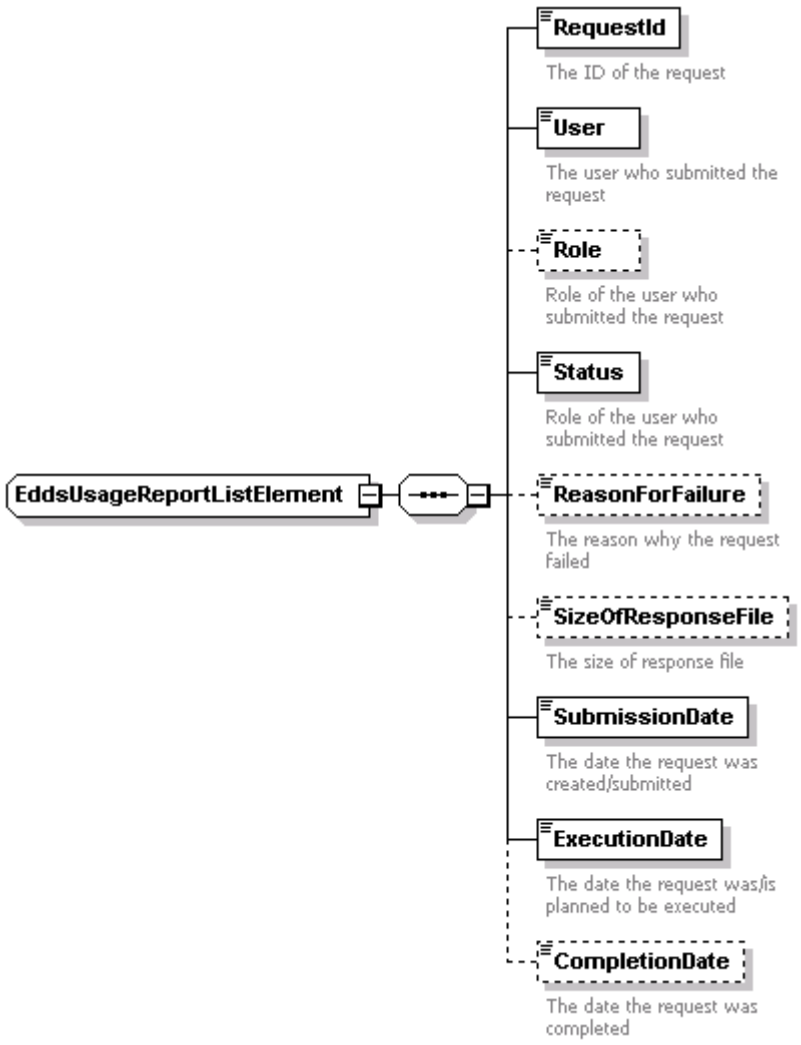
diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase

children	**DataSource RequestPostProcessing Delivery EddsUsageReportFilter EddsUsageReportFormat**
annotation	documentation Defines the Edds Usage report

I.3 complexType EddsUsageReportList

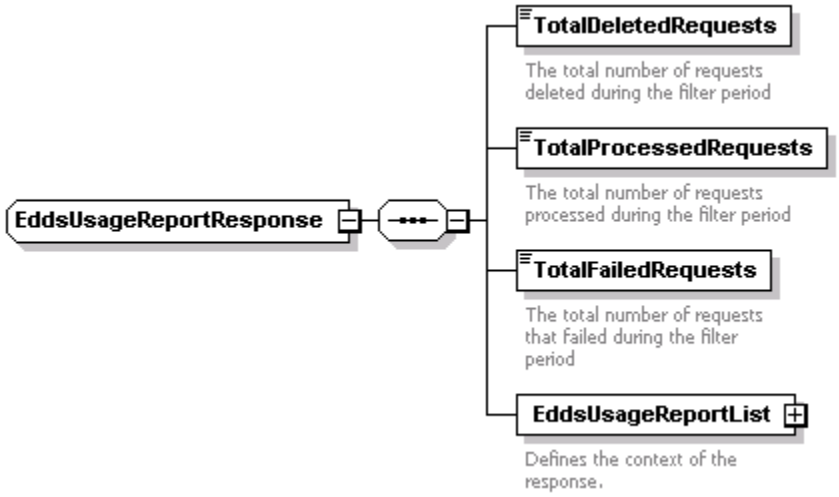
diagram	
namespace	http://edds.egos.esa/model
children	**EddsUsageReportListElement**
used by	element **EddsUsageReportResponse/EddsUsageReportList**
annotation	documentation A list of Edds usage elements

I.4 complexType EddsUsageReportListElement

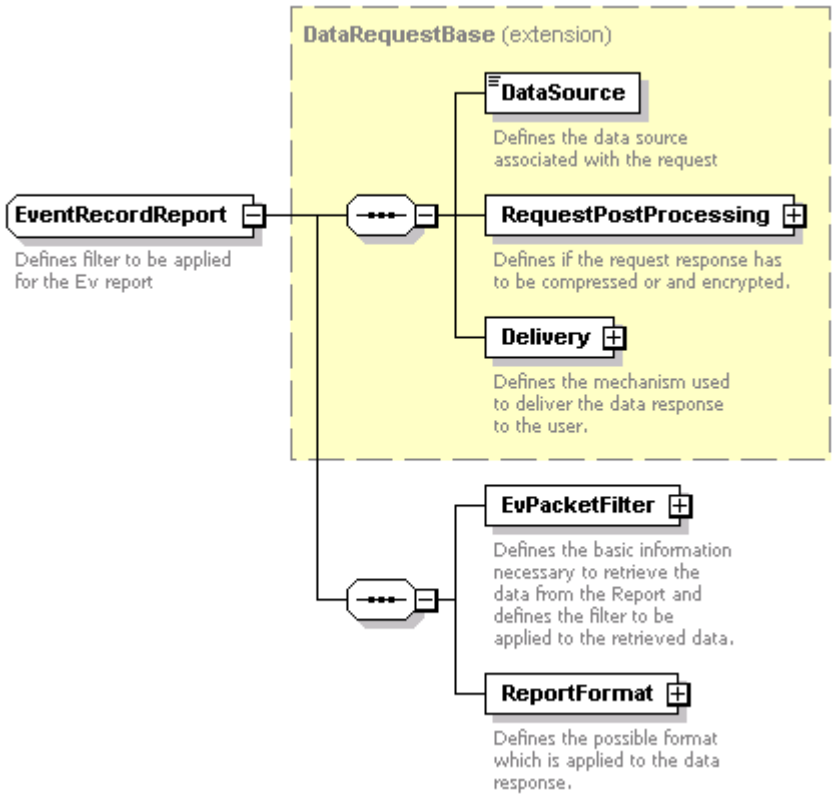
diagram	
namespace	http://edds.egos.esa/model

children	**RequestId User Role Status ReasonForFailure SizeOfResponseFile SubmissionDate ExecutionDate CompletionDate**
used by	element **EddsUsageReportList/EddsUsageReportListElement**

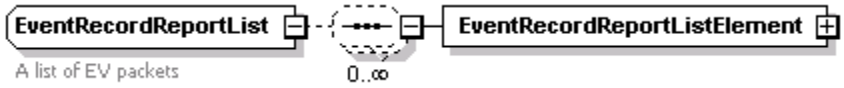
I.5 complexType EddsUsageReportResponse

diagram	
namespace	http://edds.egos.esa/model
children	**TotalDeletedRequests TotalProcessedRequests TotalFailedRequests EddsUsageReportList**

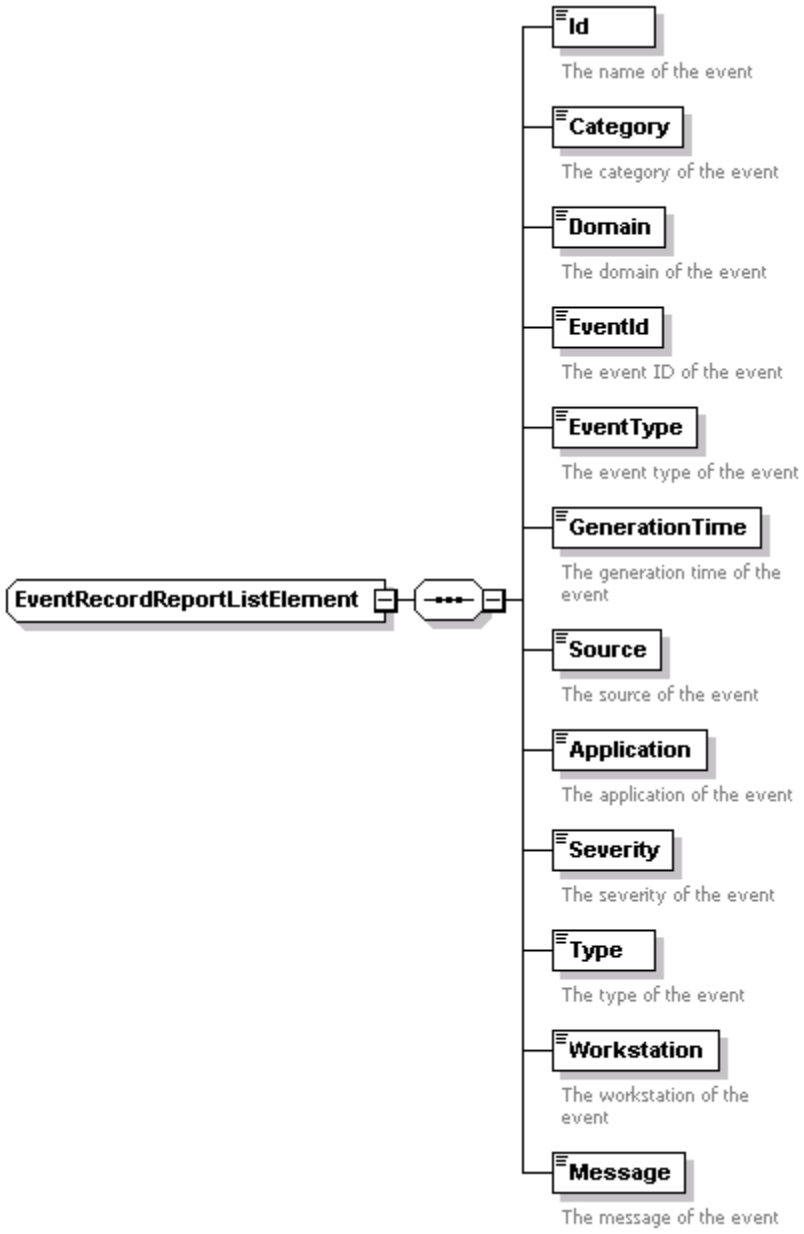
I.6 complexType EventRecordReport

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**DataSource RequestPostProcessing Delivery EvPacketFilter ReportFormat**
annotation	documentation Defines filter to be applied for the Ev report

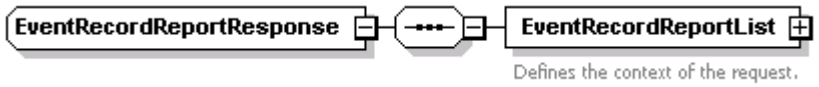
I.7 complexType EventRecordReportList

diagram	
namespace	http://edds.egos.esa/model
children	**EventRecordReportListElement**
used by	element **EventRecordReportResponse/EventRecordReportList**
annotation	documentation A list of EV packets

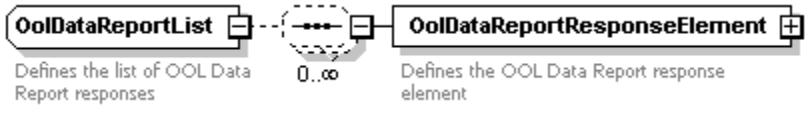
I.8 complexType EventRecordReportListElement

diagram	
namespace	http://edds.egos.esa/model
children	Id Category Domain EventId EventType GenerationTime Source Application Severity Type Workstation Message
used by	element **EventRecordReportList/EventRecordReportListElement**

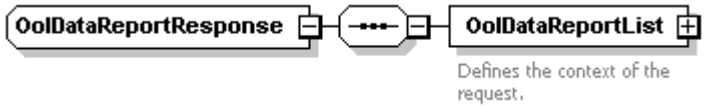
I.9 complexType EventRecordReportResponse

diagram	
namespace	http://edds.egos.esa/model
children	**EventRecordReportList**

I.10 complexType OolDataReportList

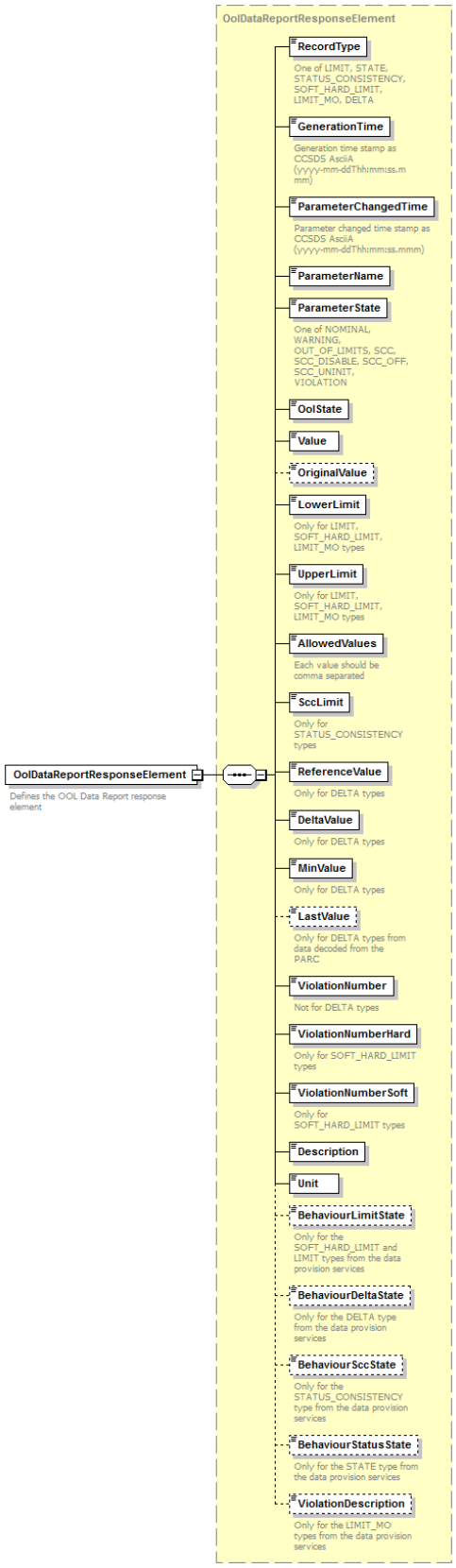
diagram	
namespace	http://edds.egos.esa/model
children	**OolDataReportResponseElement**
used by	element **OolDataReportResponse/OolDataReportList**
annotation	documentation Defines the list of OOL Data Report responses

I.11 complexType OolDataReportResponse

diagram	
namespace	http://edds.egos.esa/model
children	**OolDataReportList**

I.12 complexType OolDataReportResponseElement

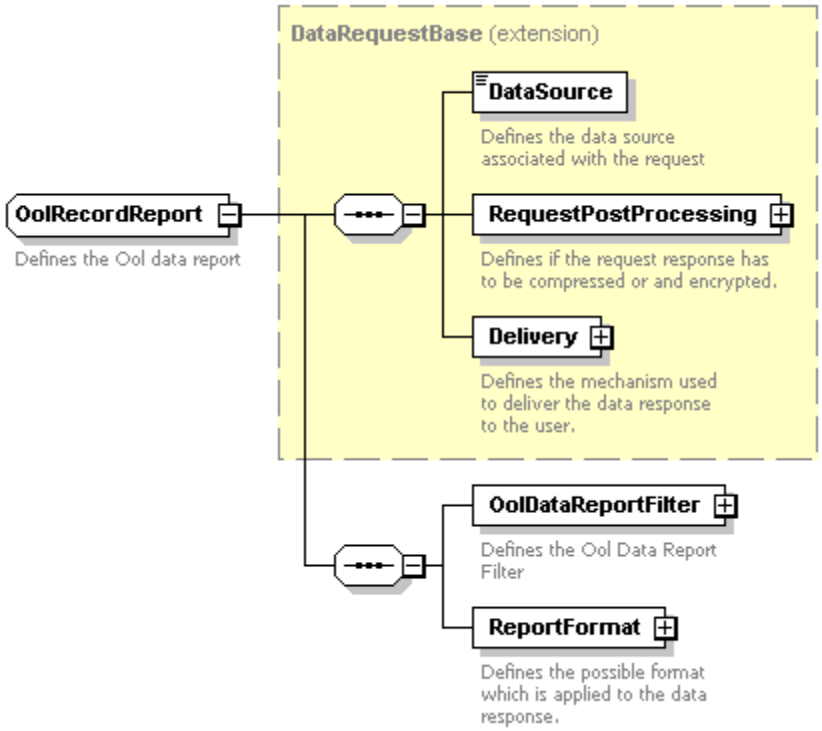
diagram



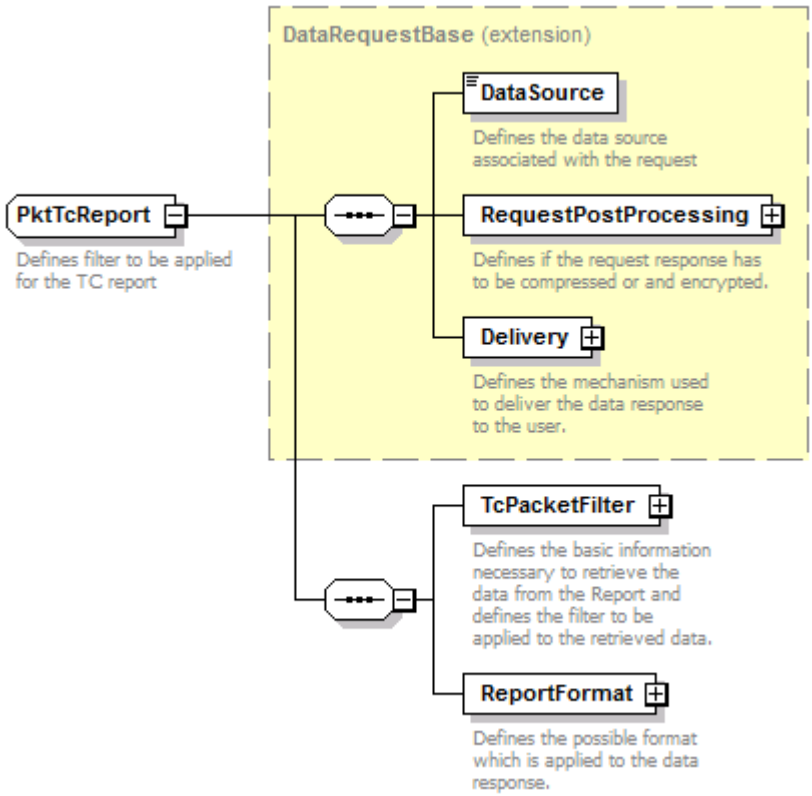
namespace	http://edds.egos.esa/model
children	**RecordType GenerationTime ParameterChangedTime ParameterName ParameterState OolState Value OriginalValue LowerLimit UpperLimit AllowedValues SccLimit ReferenceValue DeltaValue MinValue**

	LastValue ViolationNumber ViolationNumberHard ViolationNumberSoft Description Unit BehaviourLimitState BehaviourDeltaState BehaviourSccState BehaviourStatusState ViolationDescription
used by	element **OolDataReportList/OolDataReportResponseElement**

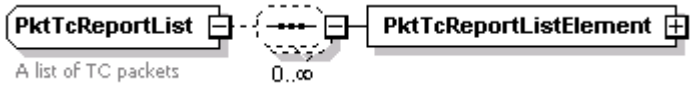
I.13 complexType OolRecordReport

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**DataSource RequestPostProcessing Delivery OolDataReportFilter ReportFormat**
annotation	documentation Defines the Ool data report

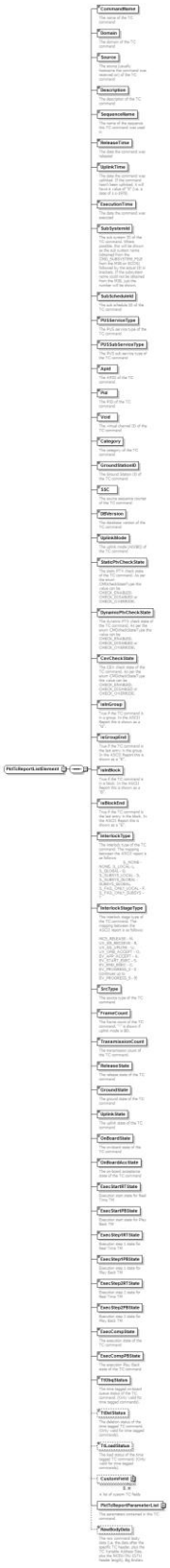
I.14 complexType PktTcReport

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**DataSource RequestPostProcessing Delivery TcPacketFilter ReportFormat**
annotation	documentation Defines filter to be applied for the TC report

I.15 complexType PktTcReportList

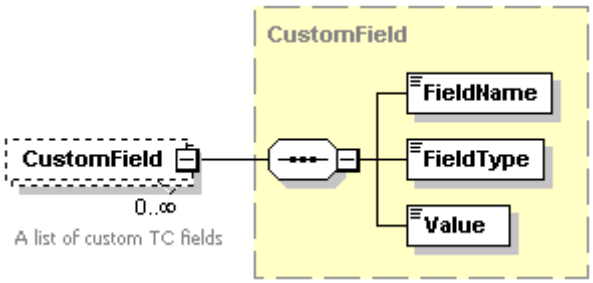
diagram	
namespace	http://edds.egos.esa/model
children	**PktTcReportListElement**
used by	element **PktTcReportResponse/PktTcReportList**
annotation	Documentation A list of TC packets

I.16 complexType PktTcReportListElement

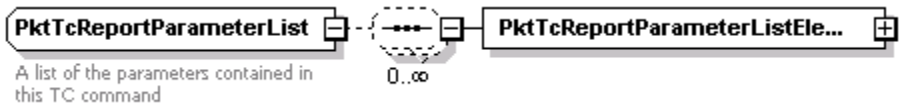
diagram	
namespace	http://edds.egos.esa/model
children	**CommandName Domain Source Description SequenceName ReleaseTime UplinkTime ExecutionTime SubSystemId SubScheduleId PUSServiceType PUSSubServiceType Apid Pid Vcid Category GroundStationID SSC DBVersion UplinkMode StaticPtvCheckState DynamicPtvCheckState CevCheckState**

	isInGroup isGroupEnd isInBlock isBlockEnd InterlockType InterlockStageType SrcType FrameCount TransmissionCount ReleaseState GroundState UplinkState OnBoardState OnBoardAccState ExecStartRTState ExecStartPBState ExecStep1RTState ExecStep1PBState ExecStep2RTState ExecStep2PBState ExecCompState ExecCompPBState TtObqStatus **TtDelStatus TtLoadStatus** CustomField PktTcReportParameterList RawBodyData
used by	element **PktTcReportList/PktTcReportListElement**

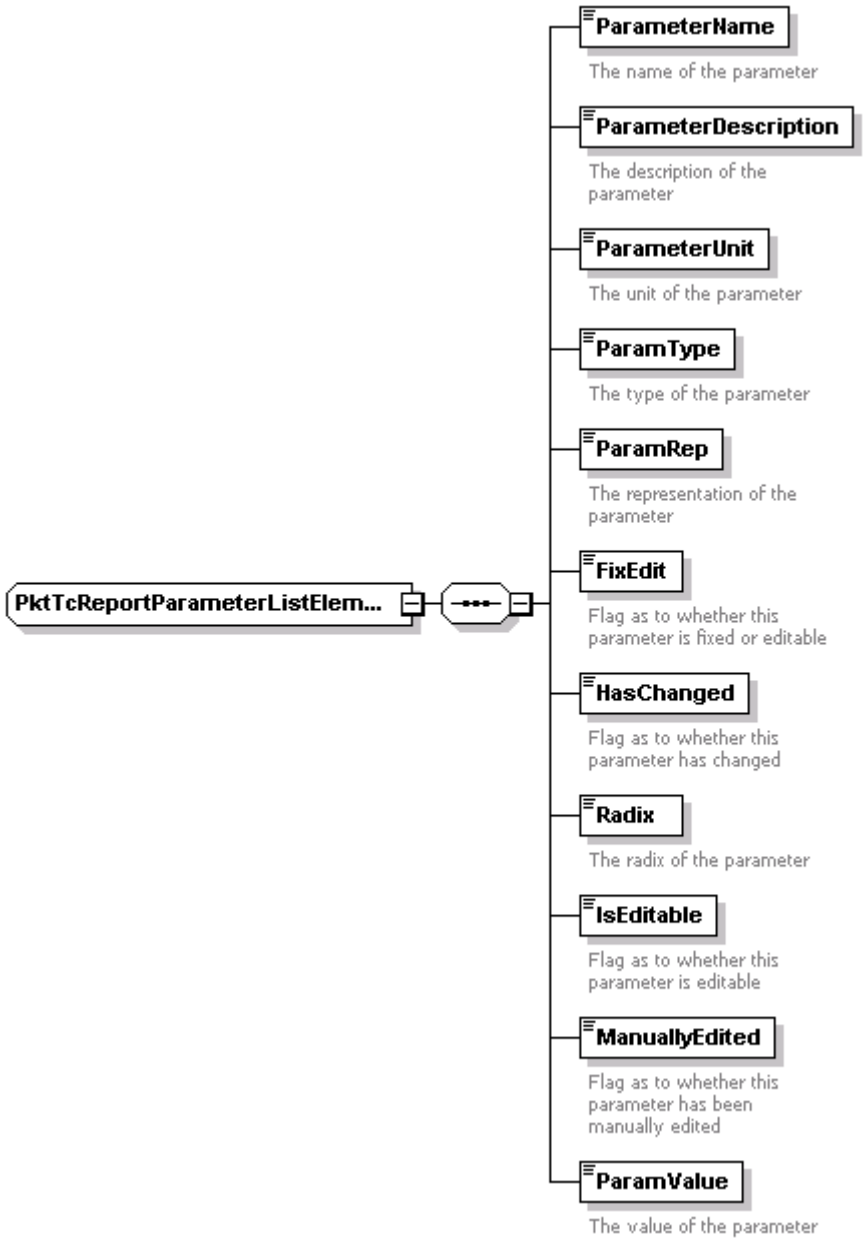
I.17 complexType CustomField

diagram	
type	CustomField
properties	isRef 0 minOcc 0 maxOcc unbounded content complex
children	FieldName FieldType Value
annotation	documentation A list of custom TC fields

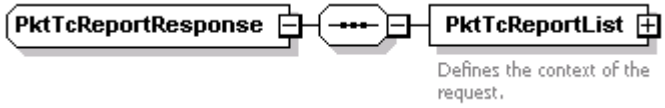
I.18 complexType PktTcReportParameterList

diagram	
namespace	http://edds.egos.esa/model
children	**PktTcReportParameterListElement**
used by	element **PktTcReportListElement/PktTcReportParameterList**
annotation	documentation A list of the parameters contained in this TC command

I.19 complexType PktTcReportParameterListElement

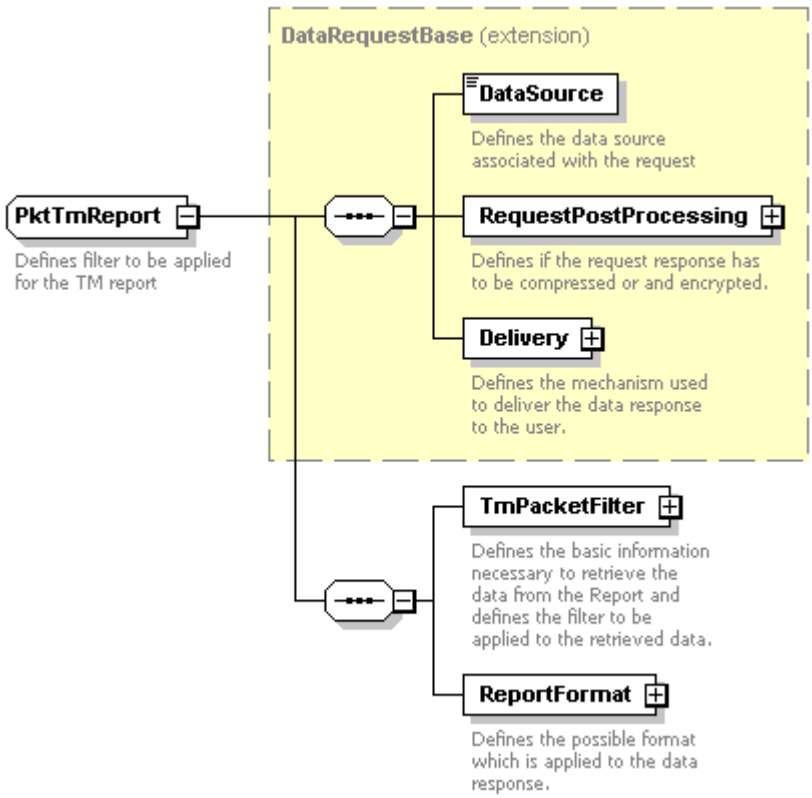
diagram	
namespace	http://edds.egos.esa/model
children	**ParameterName ParameterDescription ParameterUnit ParamType ParamRep FixEdit HasChanged Radix IsEditable ManuallyEdited ParamValue**
used by	element **PktTcReportParameterList/PktTcReportParameterListElement**

I.20 complexType PktTcReportResponse

diagram	
namespace	http://edds.egos.esa/model

children	**PktTcReportList**

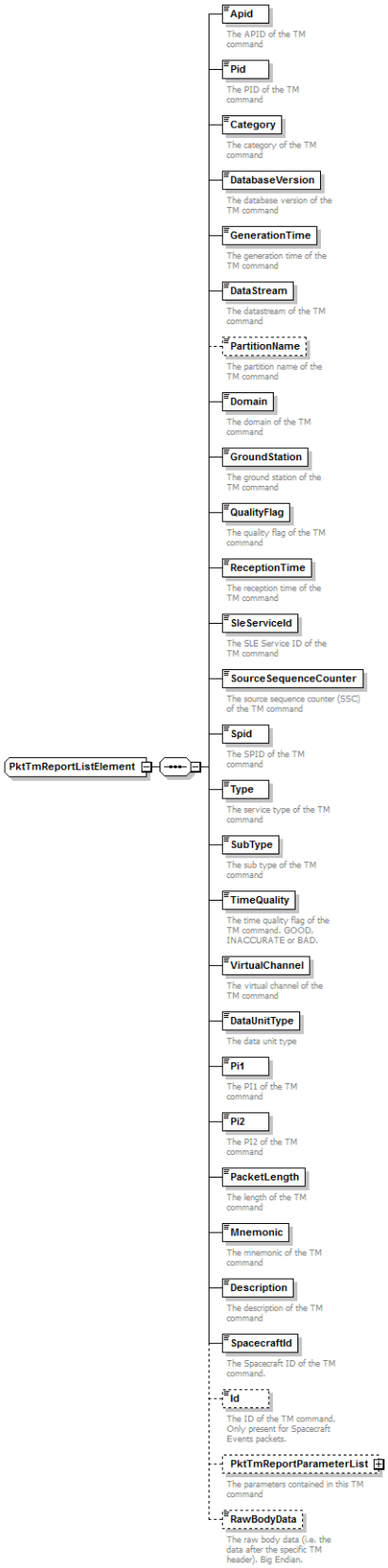
I.21 complexType PktTmReport

diagram	
namespace	http://edds.egos.esa/model
type	extension of **DataRequestBase**
properties	base DataRequestBase
children	**DataSource RequestPostProcessing Delivery TmPacketFilter ReportFormat**
annotation	documentation Defines filter to be applied for the TM report

I.22 complexType PktTmReportList

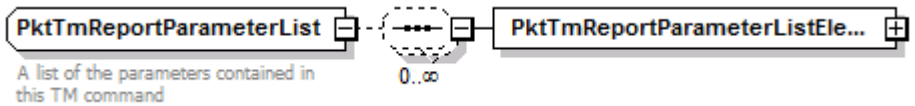
diagram	
namespace	http://edds.egos.esa/model
children	**PktTmReportListElement**
used by	element **PktTmReportResponse/PktTmReportList**
annotation	documentation A list of TM packets

I.23 complexType PktTmReportListElement

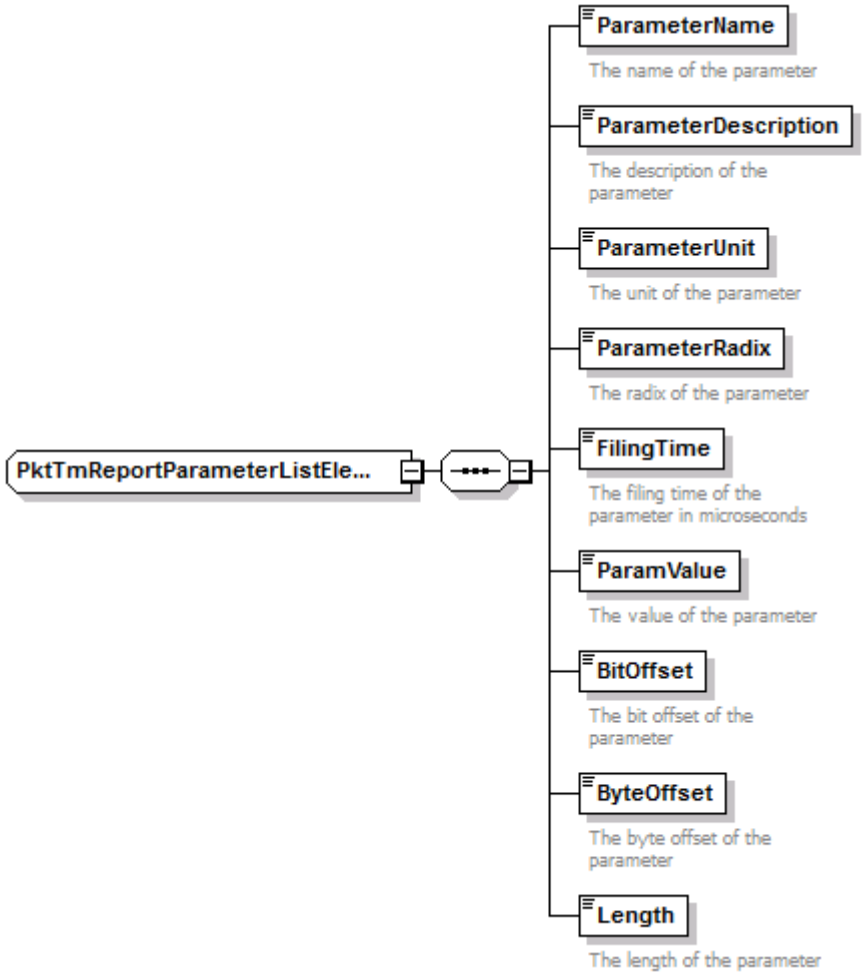
diagram	
namespace	http://edds.egos.esa/model
children	**Apid Pid Category DatabaseVersion GenerationTime DataStream PartitionName Domain GroundStation QualityFlag ReceptionTime SleServiceId SourceSequenceCounter Spid Type SubType TimeQuality**

	VirtualChannel DataUnitType Pi1 Pi2 PacketLength Mnemonic Description SpacecraftId Id PktTmReportParameterList RawBodyData
used by	element **PktTmReportList/PktTmReportListElement**

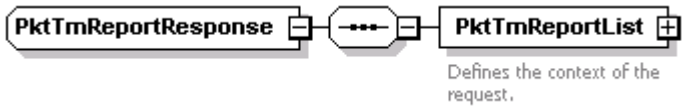
I.24 complexType PktTmReportParameterList

diagram	
namespace	http://edds.egos.esa/model
children	**PktTmReportParameterListElement**
used by	element **PktTmReportListElement/PktTmReportParameterList**

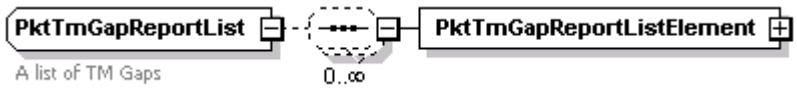
I.25 complexType PktTmReportParameterListElement

diagram	
namespace	http://edds.egos.esa/model
children	**ParameterName ParameterDescription ParameterUnit ParameterRadix FilingTime ParamValue BitOffset ByteOffset Length**
used by	element **PktTmReportParameterList/PktTmReportParameterListElement**

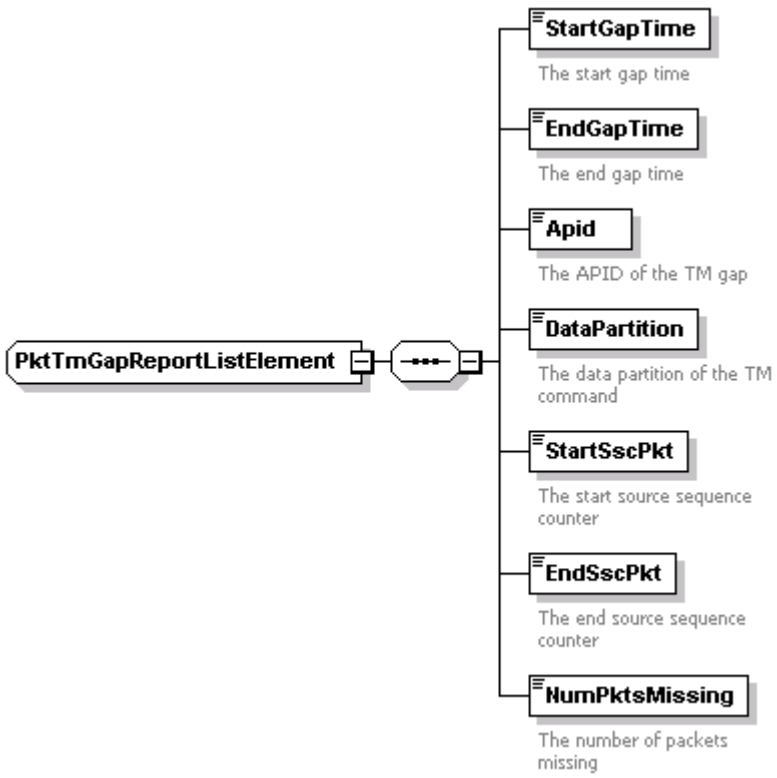
I.26 complexType PktTmReportResponse

diagram	
namespace	http://edds.egos.esa/model
children	**PktTmReportList**

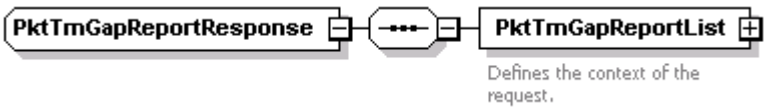
I.27 complexType PktTmGapReportList

diagram	
namespace	http://edds.egos.esa/model
children	PktTmGapReportListElement
used by	element PktTmGapReportResponse/PktTmGapReportList

I.28 complexType PktTmGapReportListElement

diagram	
namespace	http://edds.egos.esa/model
children	StartGapTime EndGapTime Apid DataPartition StartSscPkt EndSscPkt NumPktsMissing
used by	element PktTmGapReportList/PktTmGapReportListElement

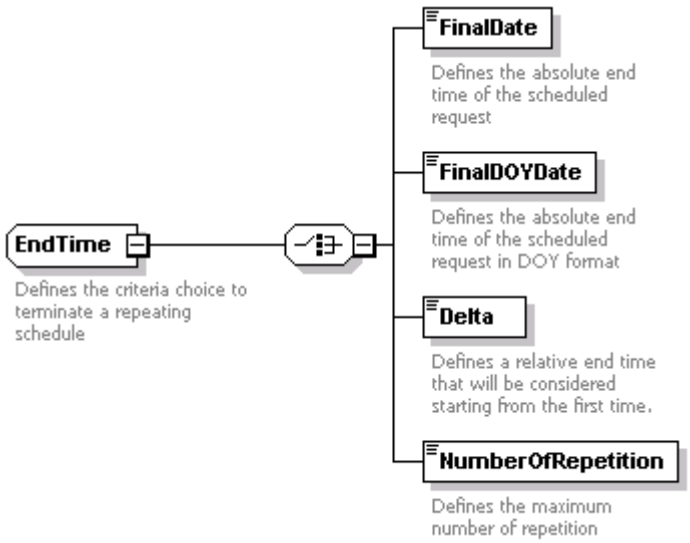
I.29 complexType PktTmGapReportResponse

diagram	
namespace	http://edds.egos.esa/model
children	PktTmGapReportList

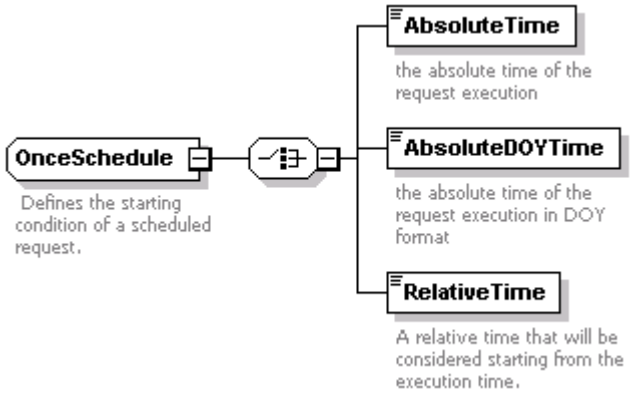
Appendix J Schedule Data Type

This section of the document describes the complex types used for scheduling as defined in the schedule.xsd file.

J.1 complexType EndTime

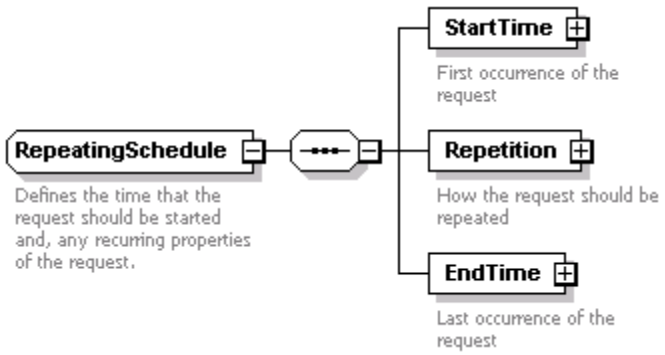
diagram	
namespace	http://edds.egos.esa/model
children	**FinalDate FinalDOYDate Delta NumberOfRepetition**
used by	element **RepeatingSchedule/EndTime**
annotation	documentation Defines the criteria choice to terminate a repeating schedule

J.2 complexType OnceSchedule

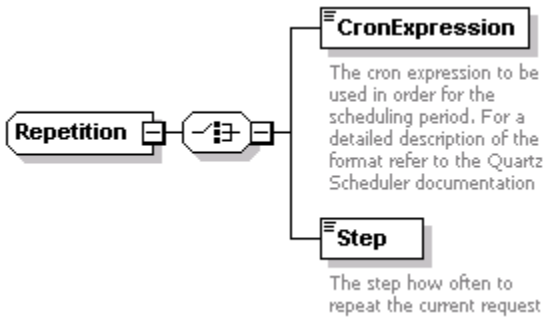
diagram	
namespace	http://edds.egos.esa/model
children	**AbsoluteTime AbsoluteDOYTime RelativeTime**
used by	elements **ScheduleOnce/Once ScheduleWithExpiry/RequestExpiryDate RepeatingSchedule/StartTime**

annotation	documentation Defines the starting condition of a scheduled request.

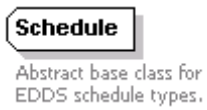
J.3 complexType RepeatingSchedule

diagram	
namespace	http://edds.egos.esa/model
children	**StartTime Repetition EndTime**
used by	element **ScheduleRepeating/RepeatingSchedule**
annotation	documentation Defines the time that the request should be started and, any recurring properties of the request.

J.4 complexType Repetition

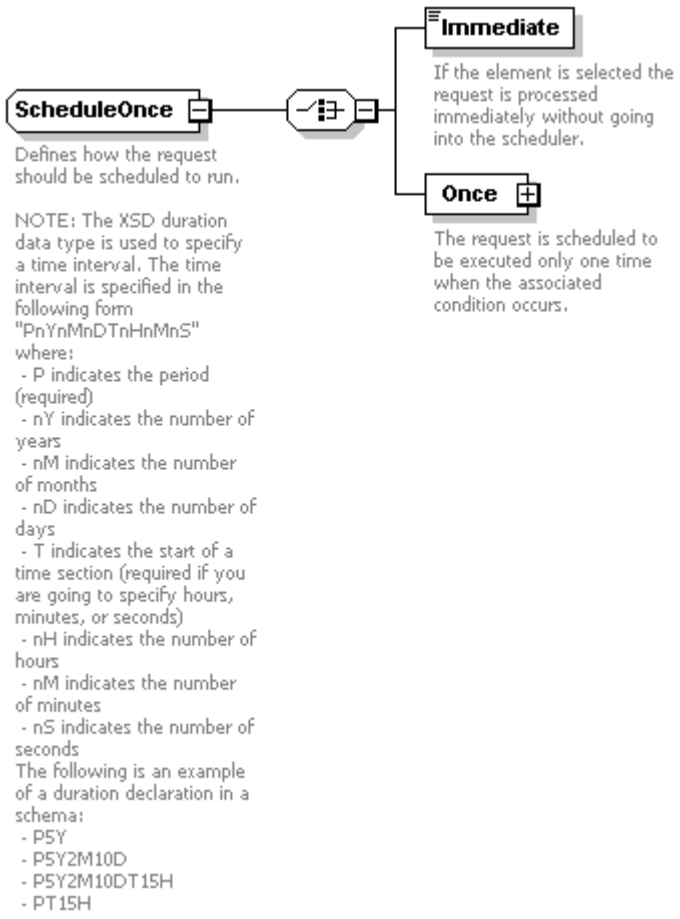
diagram	
namespace	http://edds.egos.esa/model
children	**CronExpression Step**
used by	element **RepeatingSchedule/Repetition**

J.5 complexType Schedule

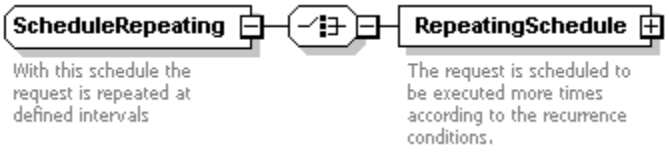
diagram	
namespace	http://edds.egos.esa/model
properties	abstract true
used by	complexTypes **ScheduleOnce ScheduleRepeating**

annotation	documentation Abstract base class for EDDS schedule types.

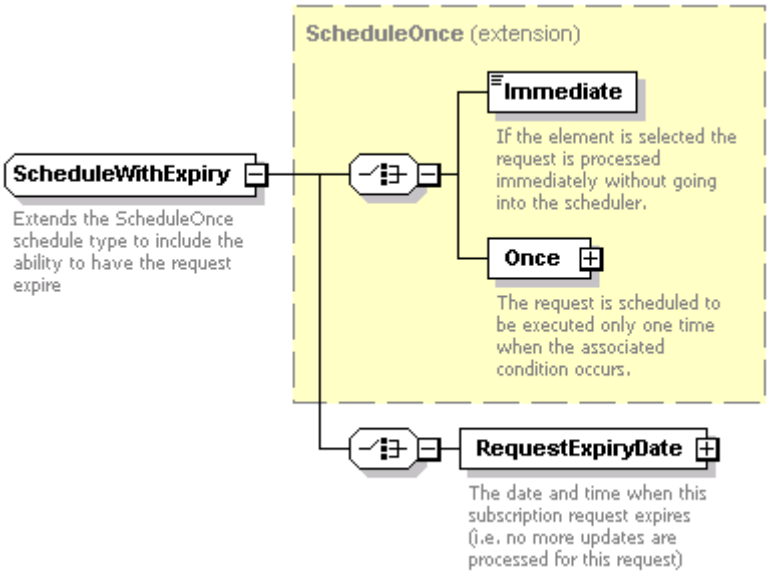
J.6 complexType ScheduleOnce

diagram	
namespace	http://edds.egos.esa/model
type	extension of **Schedule**
properties	base Schedule
children	**Immediate Once**
used by	complexType **ScheduleWithExpiry**
annotation	documentation Defines how the request should be scheduled to run. NOTE: The XSD duration data type is used to specify a time interval. The time interval is specified in the following form "PnYnMnDTnHnMnS" where: - P indicates the period (required) - nY indicates the number of years - nM indicates the number of months - nD indicates the number of days - T indicates the start of a time section (required if you are going to specify hours, minutes, or seconds) - nH indicates the number of hours - nM indicates the number of minutes - nS indicates the number of seconds The following is an example of a duration declaration in a schema: - P5Y - P5Y2M10D - P5Y2M10DT15H - PT15H

J.7 complexType ScheduleRepeating

diagram	
namespace	http://edds.egos.esa/model
type	extension of **Schedule**
properties	base Schedule
children	**RepeatingSchedule**
annotation	documentation With this schedule the request is repeated at defined intervals

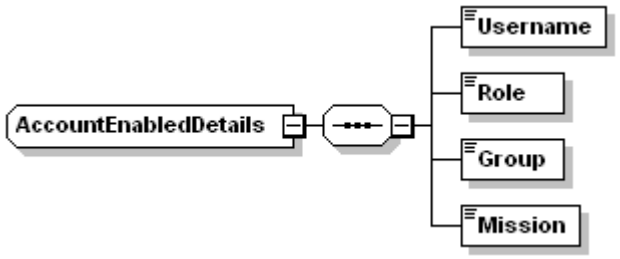
J.8 complexType ScheduleWithExpiry

diagram	
namespace	http://edds.egos.esa/model
type	extension of **ScheduleOnce**
properties	base ScheduleOnce
children	**Immediate Once RequestExpiryDate**
annotation	documentation Extends the ScheduleOnce schedule type to include the ability to have the request expire

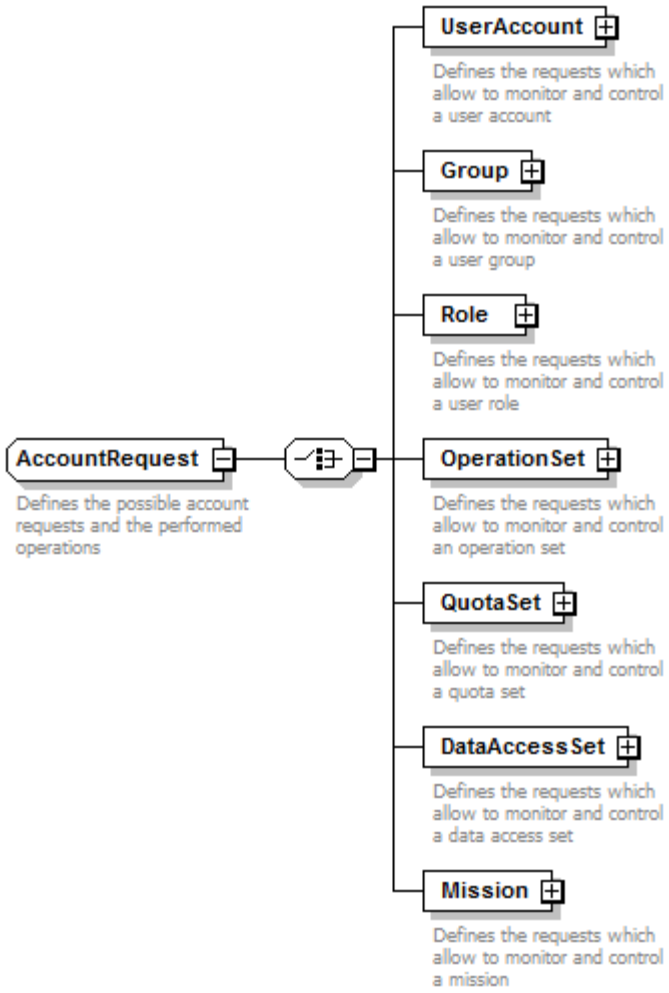
Appendix K User Management Data Type

This section of the document describes the complex types used for user mangement as defined in the userrequest.xsd file.

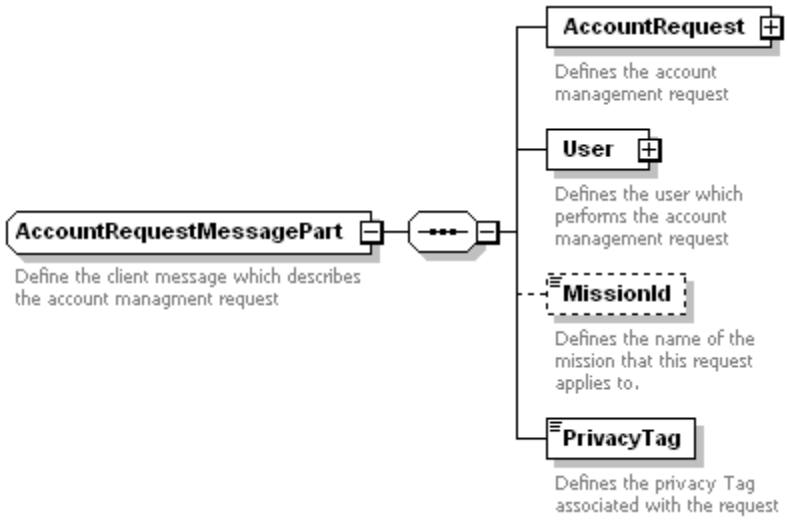
K.1 complexType AccountEnabledDetails

diagram	
namespace	http://edds.egos.esa/model
children	**Username Role Group Mission**
used by	element **AccountEnabledDetails**

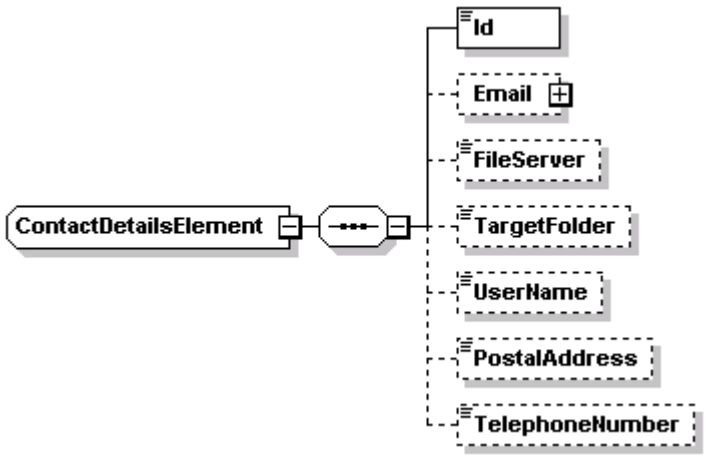
K.2 complexType AccountRequest

diagram	
namespace	http://edds.egos.esa/model
children	**UserAccount Group Role OperationSet QuotaSet DataAccessSet Mission**
used by	element **AccountRequestMessagePart/AccountRequest**
annotation	documentation Defines the possible account requests and the performed operations

K.3 complexType AccountRequestMessagePart

diagram	
namespace	http://edds.egos.esa/model
children	**AccountRequest User MissionId PrivacyTag**
used by	element **AccountRequestMessagePart**
annotation	documentation Define the client message which describes the account management request

K.4 complexType ContactDetailsElement

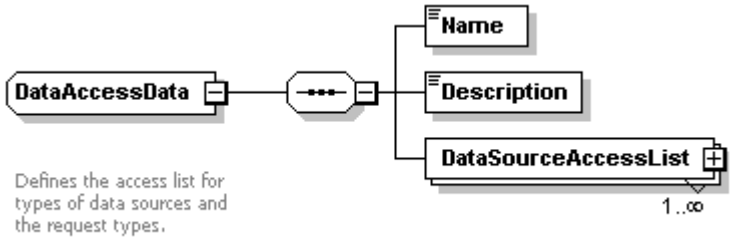
diagram	
namespace	http://edds.egos.esa/model
children	**Id Email FileServer TargetFolder UserName PostalAddress TelephoneNumber**
used by	elements **UserAccountBase/ContactDetailsList UserAttributes/ContactDetailsList**

K.5 complexType ContactDetailsList

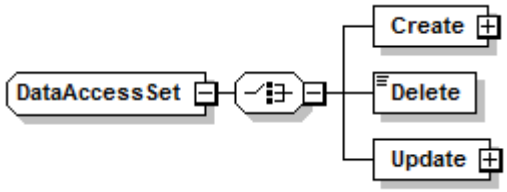
diagram	

namespace	http://edds.egos.esa/model
children	**ContactDetailsListElement**
used by	elements **UserAccountData/ContactDetailsList UserAttributes/ContactDetailsList**

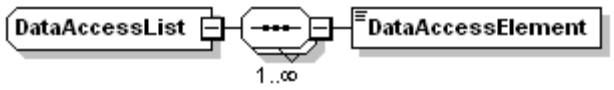
K.6 complexType DataAccessData

diagram	
namespace	http://edds.egos.esa/model
children	**Name Description DataSourceAccessList**
used by	elements **DataAccessSet/Create MissionDetailsList/DataAccessDataList DataAccessSet/Update**
annotation	documentation Defines the access list for types of data sources and the request types.

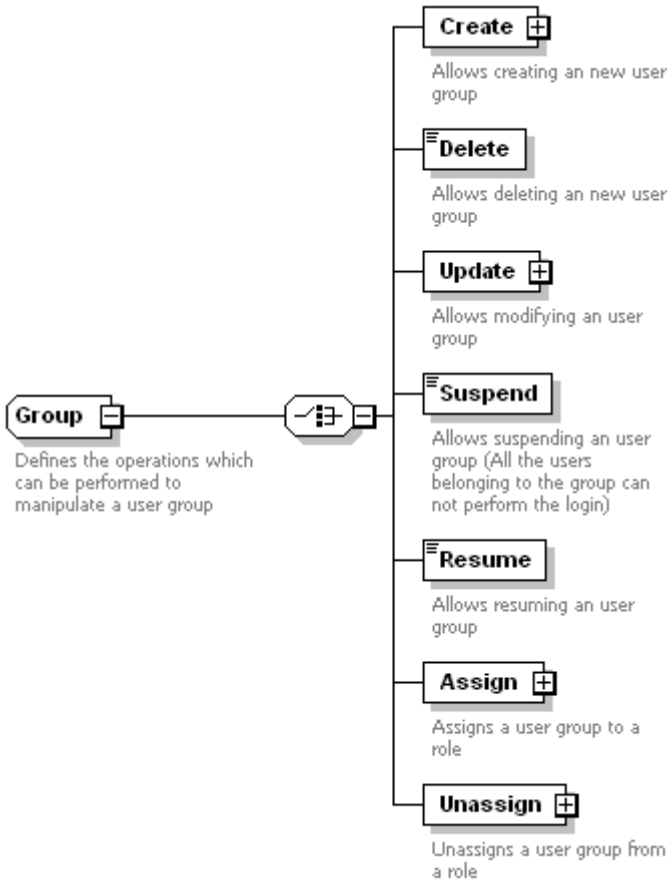
K.7 complexType DataAccessSet

diagram	
namespace	http://edds.egos.esa/model
children	**Create Delete Update**
used by	element **AccountRequest/DataAccessSet**

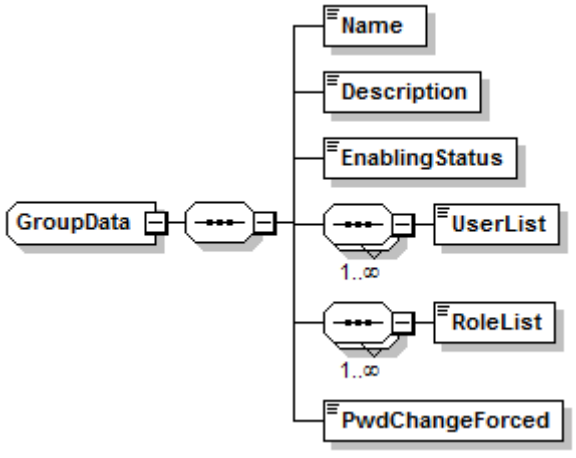
K.8 complexType DataAccessList

diagram	
namespace	http://edds.egos.esa/model
children	**DataAccessElement**
used by	element **DataAccessData/DataAccessList**

K.9 complexType Group

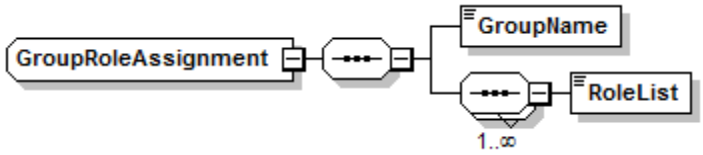
diagram	
namespace	http://edds.egos.esa/model
children	**Create Delete Update Suspend Resume Assign Unassign**
used by	element **AccountRequest/Group**
annotation	documentation Defines the operations which can be performed to manipulate a user group

K.10 complexType GroupData

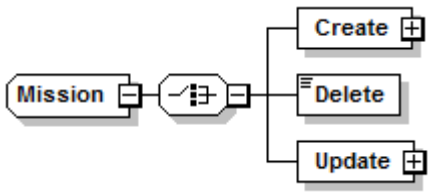
diagram	
namespace	http://edds.egos.esa/model
children	**Name Description EnablingStatus UserList RoleList PwdChangeForced**

used by	elements **Group/Create MissionDetailsList/GroupDataList Group/Update**

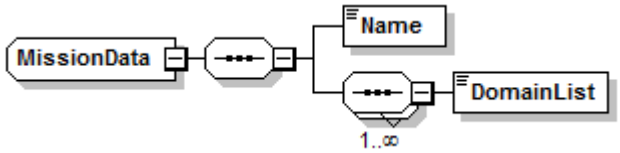
K.11 complexType GroupRoleAssignment

diagram	
namespace	http://edds.egos.esa/model
children	**GroupName RoleList**
used by	elements **Group/Assign Group/Unassign**

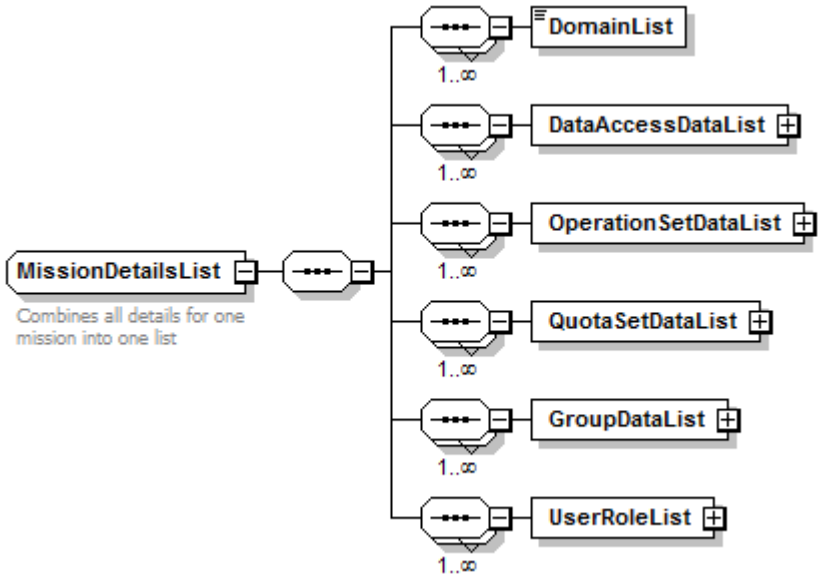
K.12 complexType Mission

diagram	
namespace	http://edds.egos.esa/model
children	**Create Delete Update**
used by	element **AccountRequest/Mission**

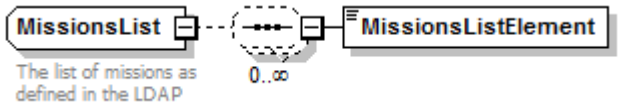
K.13 complexType MissionData

diagram	
namespace	http://edds.egos.esa/model
children	**Name DomainList**
used by	elements **Mission/Create Mission/Update**

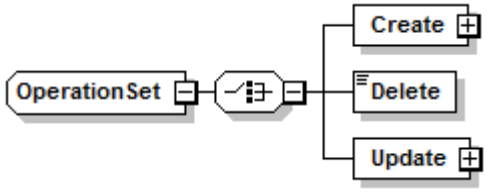
K.14 complexType MissionDetailsList

diagram	
namespace	http://edds.egos.esa/model
children	**DomainList DataAccessDataList OperationSetDataList QuotaSetDataList GroupDataList UserRoleList**
used by	element **MissionDetails**
annotation	documentation Combines all details for one mission into one list

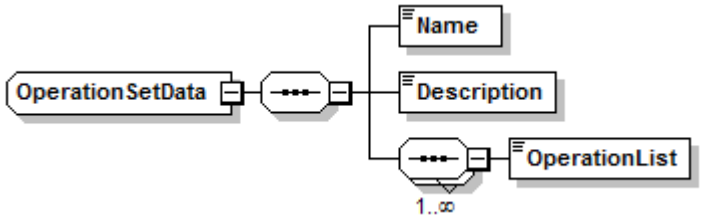
K.15 complexType MissionsList

diagram	
namespace	http://edds.egos.esa/model
children	**MissionsListElement**
used by	element **MissionsList**
annotation	documentation The list of missions as defined in the LDAP

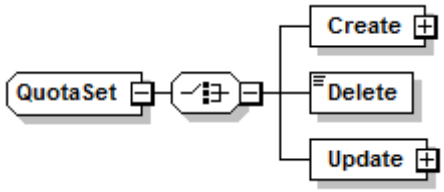
K.16 complexType OperationSet

diagram	
namespace	http://edds.egos.esa/model
children	**Create Delete Update**
used by	element **AccountRequest/OperationSet**

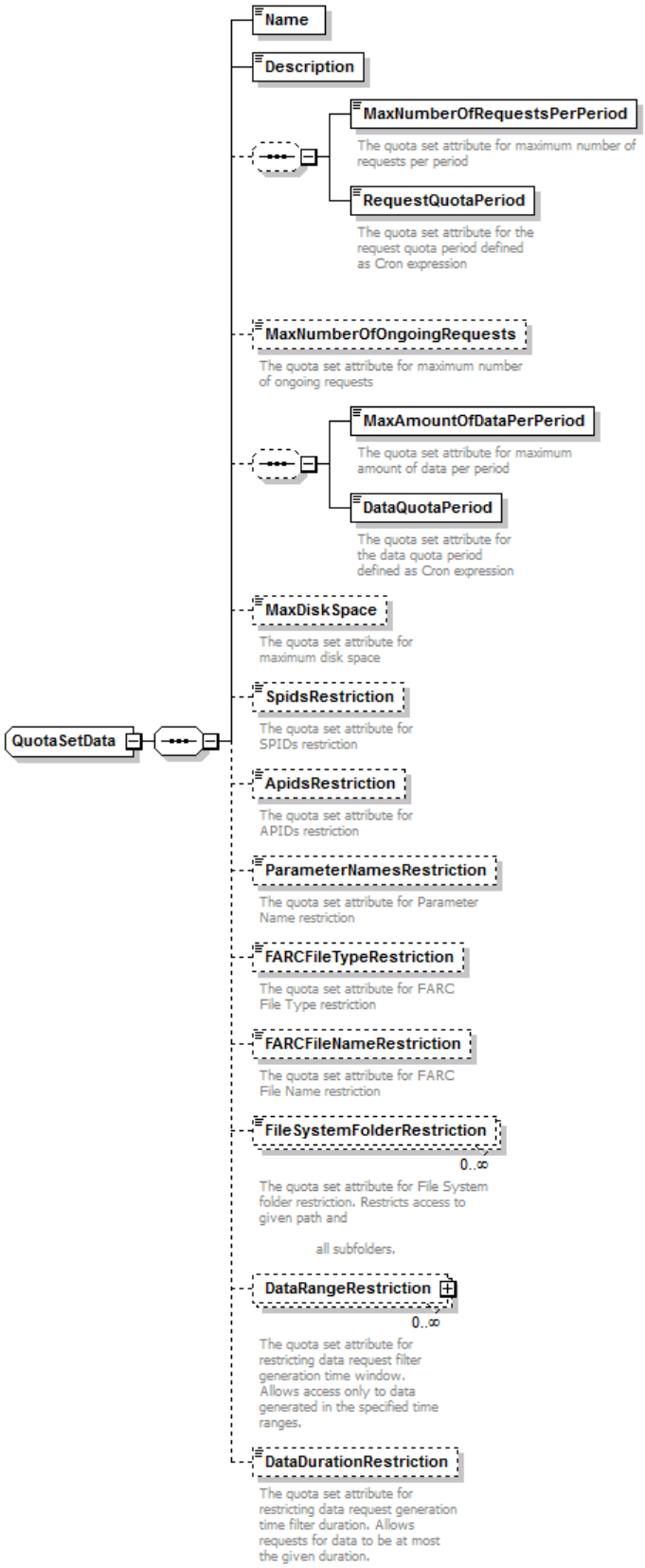
K.17 complexType OperationSetData

diagram	
namespace	http://edds.egos.esa/model
children	**Name Description OperationList**
used by	elements **OperationSet/Create MissionDetailsList/OperationSetDataList OperationSet/Update**

K.18 complexType QuotaSet

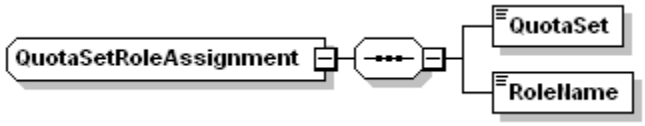
diagram	
namespace	http://edds.egos.esa/model
children	**Create Delete Update**
used by	element **AccountRequest/QuotaSet**

K.19 complexType QuotaSetData

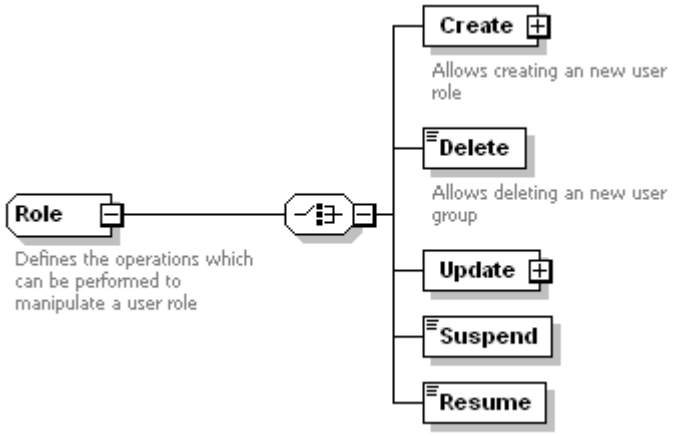
diagram	
namespace	http://edds.egos.esa/model
children	**Name Description MaxNumberOfRequestsPerPeriod RequestQuotaPeriod MaxNumberOfOngoingRequests MaxAmountOfDataPerPeriod DataQuotaPeriod MaxDiskSpace**

	SpidsRestriction ApidsRestriction ParameterNamesRestriction FARCFileTypeRestriction FARCFileNameRestriction FileSystemFolderRestriction DataRangeRestriction DataDurationRestriction
used by	elements **QuotaSet/Create UserQuotaDetails/QuotaLimits MissionDetailsList/QuotaSetDataList QuotaSet/Update**

K.20 complexType QuotaSetRoleAssignment

diagram	
namespace	http://edds.egos.esa/model
children	**QuotaSet RoleName**
used by	elements **QuotaSet/Assign QuotaSet/Unassign**

K.21 complexType Role

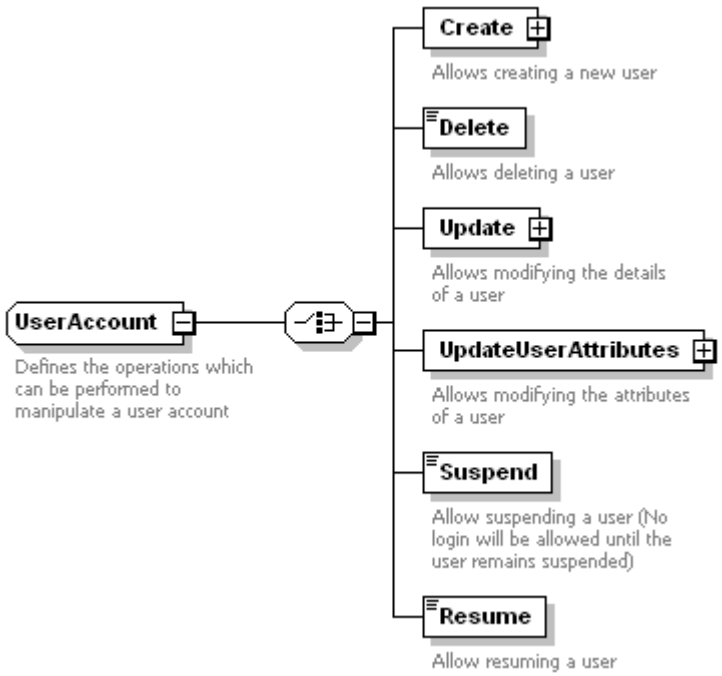
diagram	
namespace	http://edds.egos.esa/model
children	**Create Delete Update Suspend Resume**
used by	element **AccountRequest/Role**
annotation	documentation Defines the operations which can be performed to manipulate a user role

K.22 complexType RolesList

diagram	
namespace	http://edds.egos.esa/model
children	**RoleElement**
used by	element **RolesList**
annotation	documentation

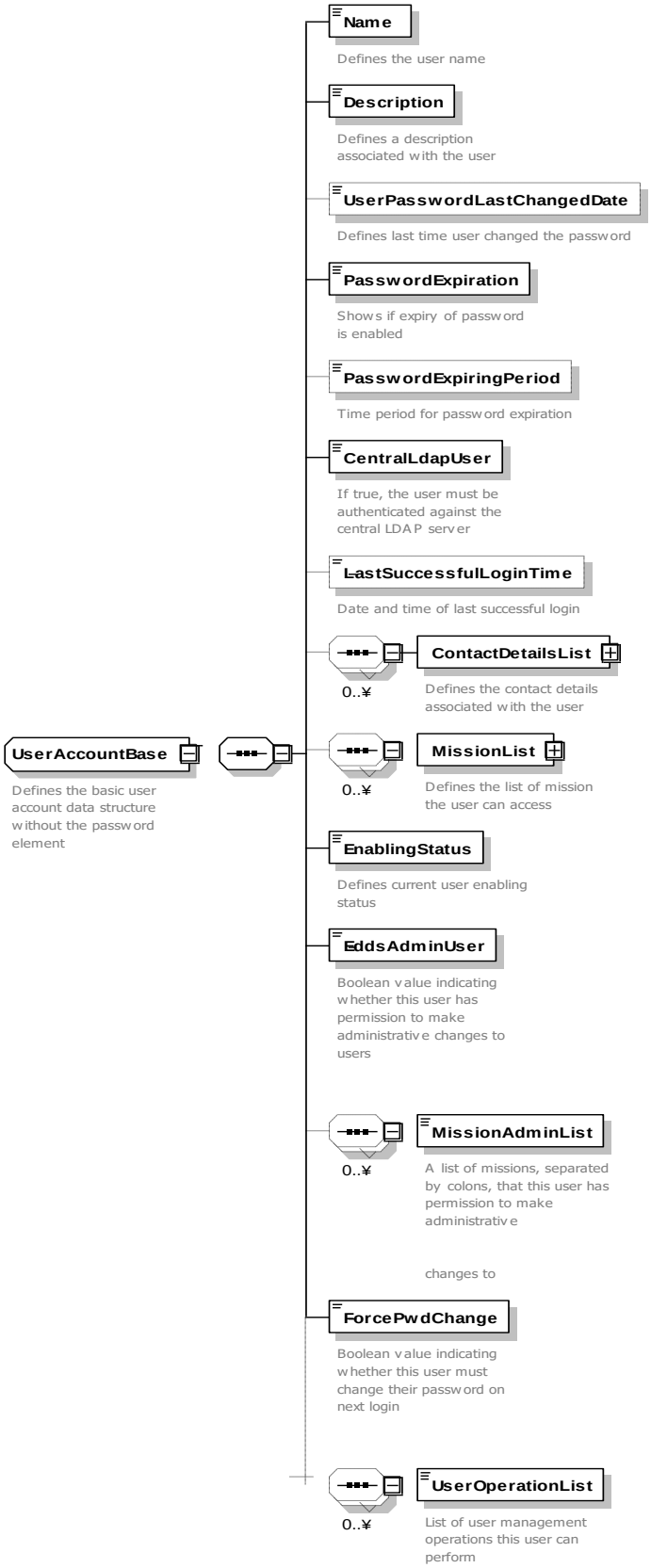
	The list of roles a user has permission to use for the specified mission

K.23 complexType UserAccount

diagram	
namespace	http://edds.egos.esa/model
children	**Create Delete Update UpdateUserAttributes Suspend Resume**
used by	element **AccountRequest/UserAccount**
annotation	documentation Defines the operations which can be performed to manipulate a user account

K.24 complexType UserAccountBase

diagram

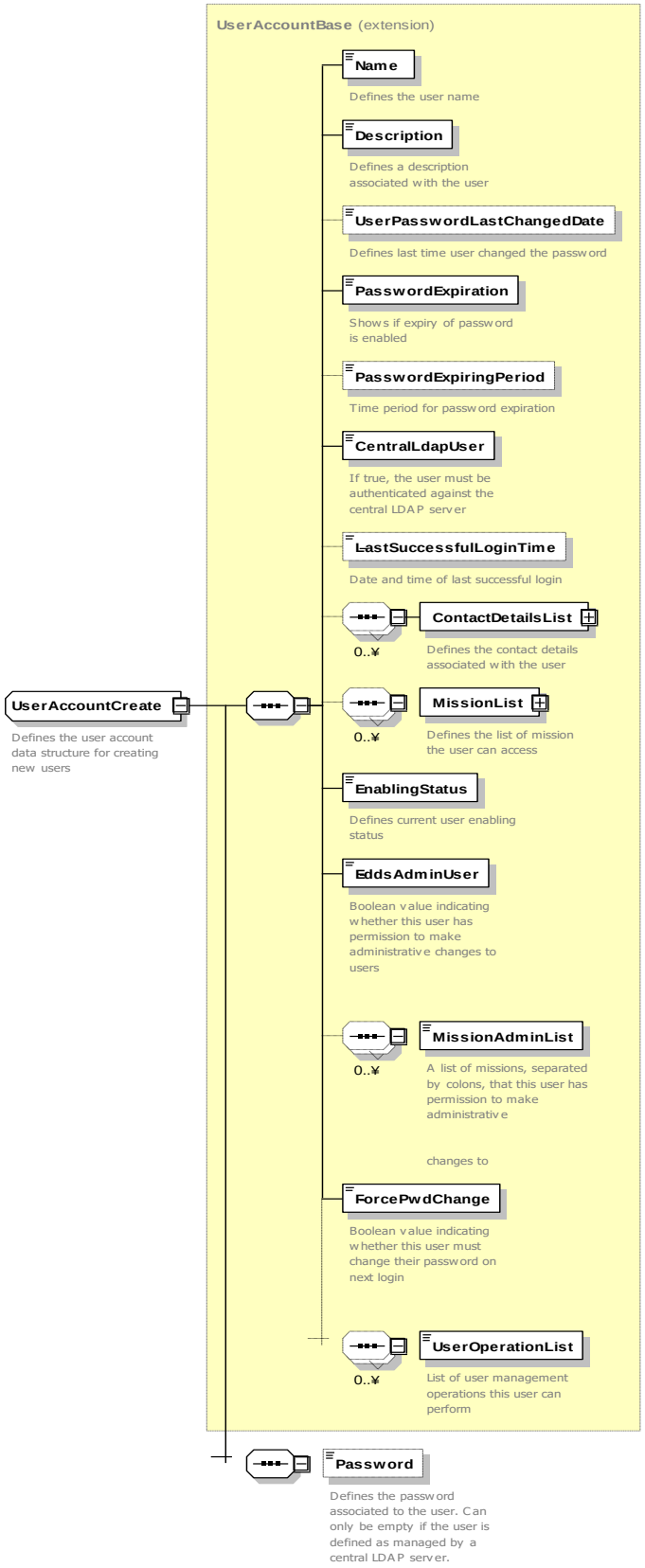


namespace	http://edds.egos.esa/model
children	**Name Description UserPasswordLastChangedDate PasswordExpiration PasswordExpiringPeriod CentralLdapUser LastSuccessfulLoginTime ContactDetailsList MissionList EnablingStatus**

	EddsAdminUser MissionAdminList ForcePwdChange UserOperationList
used by	complexTypes **UserAccountCreate UserAccountDetails**
annotation	documentation Defines the basic user account data structure without the password element

K.25 complexType UserAccountCreate

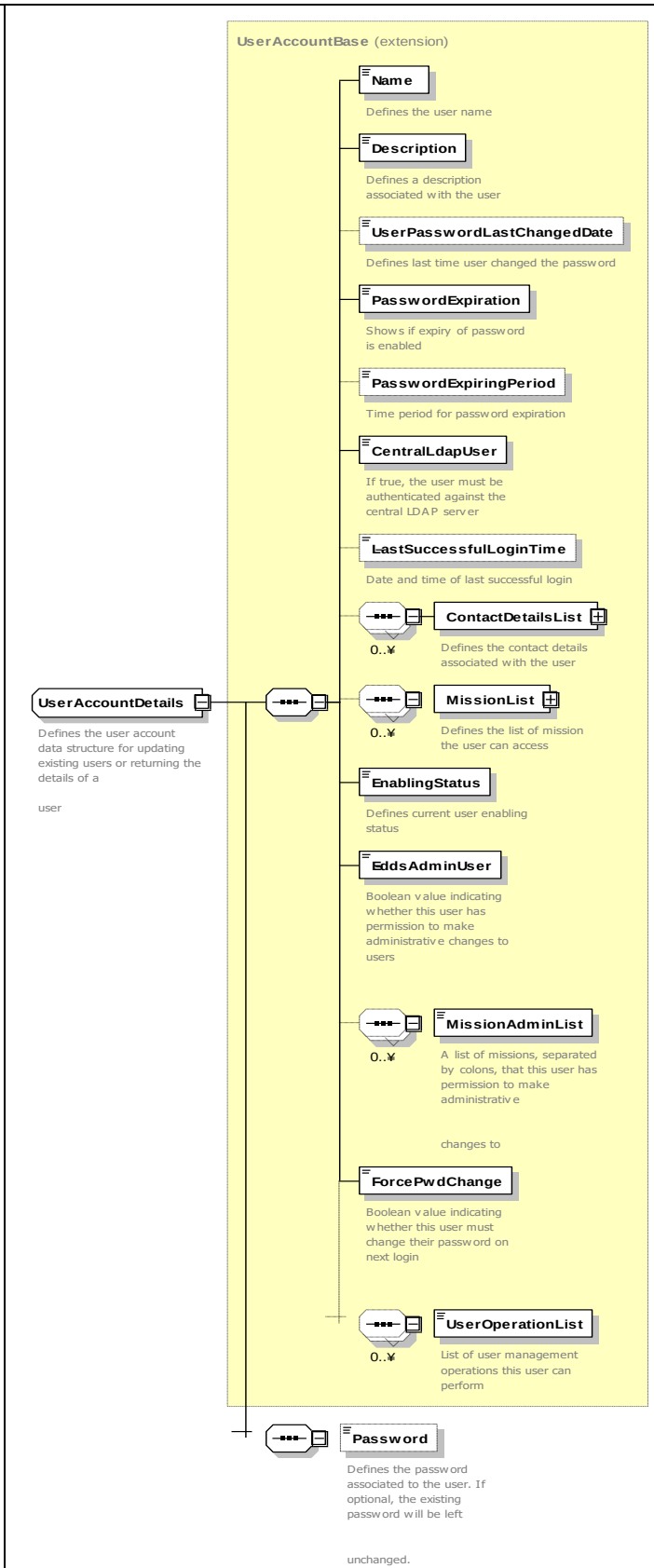
diagram



namespace	http://edds.egos.esa/model
type	extension of UserAccountBase

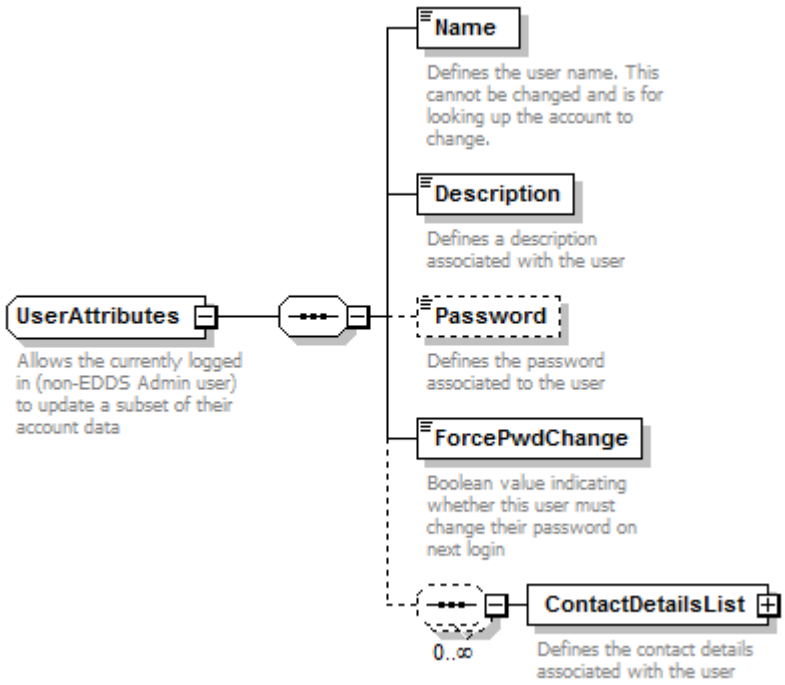
properties	base UserAccountBase
children	**Name Description UserPasswordLastChangedDate PasswordExpiration PasswordExpiringPeriod CentralLdapUser LastSuccessfulLoginTime ContactDetailsList MissionList EnablingStatus EddsAdminUser MissionAdminList ForcePwdChange UserOperationList Password**
used by	element **UserAccount/Create**
annotation	documentation Defines the user account data structure for creating new users

K.26 complexType UserAccountDetails

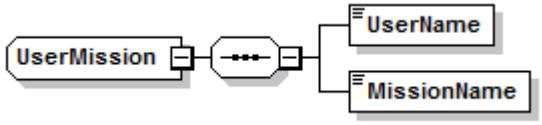
diagram	
namespace	http://edds.egos.esa/model
type	extension of **UserAccountBase**

properties	base UserAccountBase
children	**Name Description UserPasswordLastChangedDate PasswordExpiration PasswordExpiringPeriod CentralLdapUser LastSuccessfulLoginTime ContactDetailsList MissionList EnablingStatus EddsAdminUser MissionAdminList ForcePwdChange UserOperationList Password**
used by	elements **UserAccount/Update UserAccountDetails**
annotation	documentation Defines the user account data structure for updating existing users or returning the details of a user

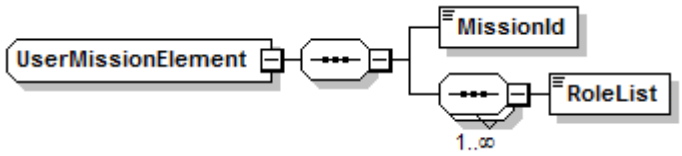
K.27 complexType UserAttributes

diagram	
namespace	http://edds.egos.esa/model
children	**Name Description Password ForcePwdChange ContactDetailsList**
used by	element **UserAccount/UpdateUserAttributes**
annotation	documentation Allows the currently logged in (non-EDDS Admin user) to update a subset of their account data

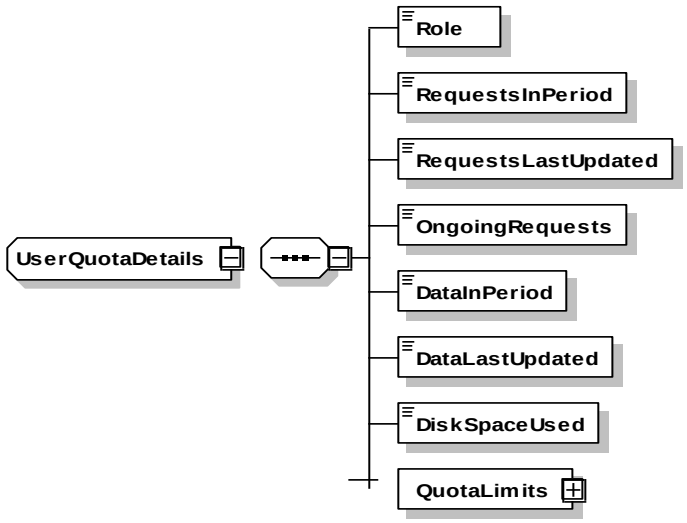
K.28 complexType UserMission

diagram	
namespace	http://edds.egos.esa/model
children	UserName MissionName
used by	element UserMission

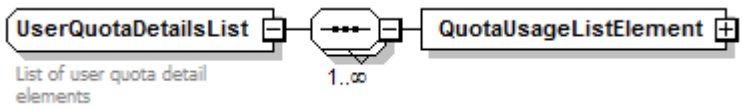
K.29 complexType UserMissionElement

diagram	
namespace	http://edds.egos.esa/model
children	**MissionId RoleList**
used by	element **UserAccountBase/MissionList**

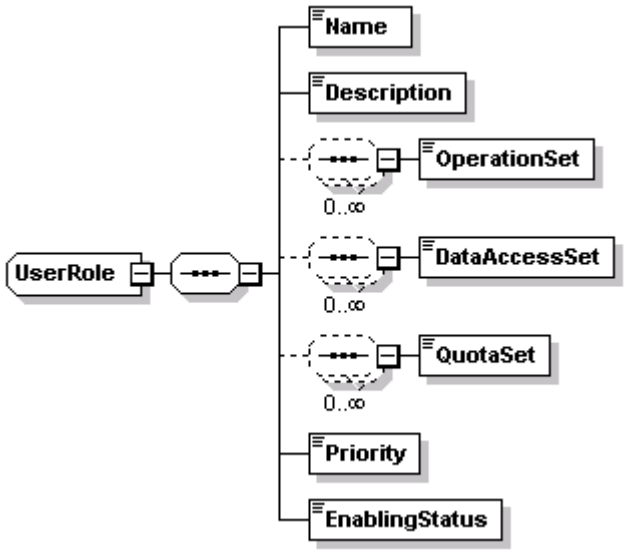
K.30 complexType UserQuotaDetails

diagram	
namespace	http://edds.egos.esa/model
children	**Role RequestsInPeriod RequestsLastUpdated OngoingRequests DataInPeriod DataLastUpdated DiskSpaceUsed QuotaLimits**
used by	element **UserQuotaDetailsList/QuotaUsageListElement**

K.31 complexType UserQuotaDetailsList

diagram	
namespace	http://edds.egos.esa/model
children	**QuotaUsageListElement**
used by	element **UserQuotaDetails**
annotation	documentation List of user quota detail elements

K.32 complexType UserRole

diagram	
namespace	http://edds.egos.esa/model
children	**Name Description OperationSet DataAccessSet QuotaSet Priority EnablingStatus**
used by	elements **Role/Create Role/Update MissionDetailsList/UserRoleList**

K.33 complexType UsersList

diagram	
namespace	http://edds.egos.esa/model
children	**UsersListElement**
used by	element **UsersList**
annotation	documentation The list of users as defined in the LDAP

K.34 simpleType OperationElement

namespace	http://edds.egos.esa/model
type	restriction of **xs:string**
properties	base xs:string
used by	element **OperationSetData/OperationList**
facets	Kind / Value / annotation enumeration CREATE_GROUP enumeration DELETE_GROUP enumeration UPDATE_GROUP enumeration ASSIGN_ROLE_TO_GROUP enumeration UNASSIGN_ROLE_FROM_GROUP enumeration SUSPEND_GROUP

	enumeration	RESUME_GROUP
	enumeration	CREATE_ROLE
	enumeration	DELETE_ROLE
	enumeration	UPDATE_ROLE
	enumeration	SUSPEND_ROLE
	enumeration	RESUME_ROLE
	enumeration	UPDATE_ROLE_DEFAULT_CONTACT_DETAILS
	enumeration	CREATE_OPERATIONS_SET
	enumeration	DELETE_OPERATIONS_SET
	enumeration	UPDATE_OPERATIONS_SET
	enumeration	ASSIGN_OPERATIONS_SET_TO_ROLE
	enumeration	UNASSIGN_OPERATIONS_SET_FROM_ROLE
	enumeration	CREATE_DATA_ACCESS_SET
	enumeration	DELETE_DATA_ACCESS_SET
	enumeration	UPDATE_DATA_ACCESS_SET
	enumeration	ASSIGN_DATA_ACCESS_SET_TO_ROLE
	enumeration	UNASSIGN_DATA_ACCESS_SET_FROM_ROLE
	enumeration	CREATE_QUOTA_SET
	enumeration	DELETE_QUOTA_SET
	enumeration	UPDATE_QUOTA_SET
	enumeration	ASSIGN_QUOTA_SET_TO_ROLE
	enumeration	UNASSIGN_QUOTA_SET_FROM_ROLE
	enumeration	SCHEDULE_REQUEST
	enumeration	VIEW_LOG_FILE

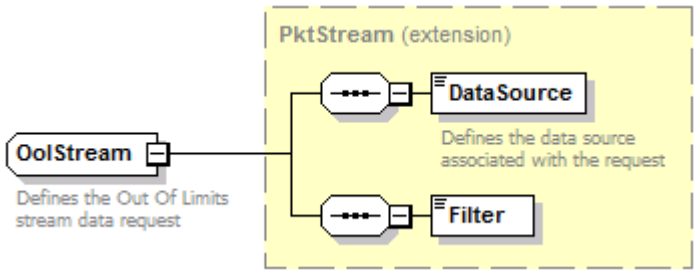
K.35 simpleType UserOperationElement

namespace	http://edds.egos.esa/model		
type	restriction of **xs:string**		
properties	base xs:string		
used by	element **UserAccountBase/UserOperationList**		
facets	Kind	Value	annotation
	enumeration	CREATE_USER_ACCOUNT	
	enumeration	DELETE_USER_ACCOUNT	
	enumeration	UPDATE_USER_ACCOUNT	
	enumeration	UPDATE_USER_PERSONAL_ATTRIBUTES	
	enumeration	SUSPEND_USER_ACCOUNT	
	enumeration	RESUME_USER_ACCOUNT	

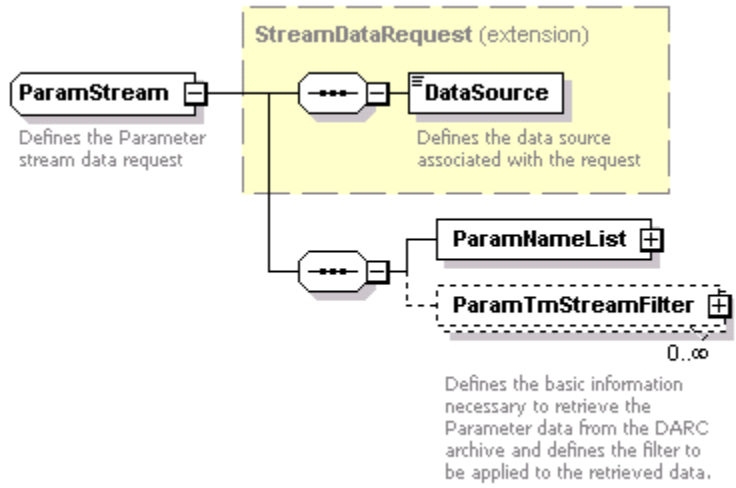
Appendix LStream Request Data Type

This section of the document describes the complex types used for stream requests as defined in the streamrequest.xsd file.

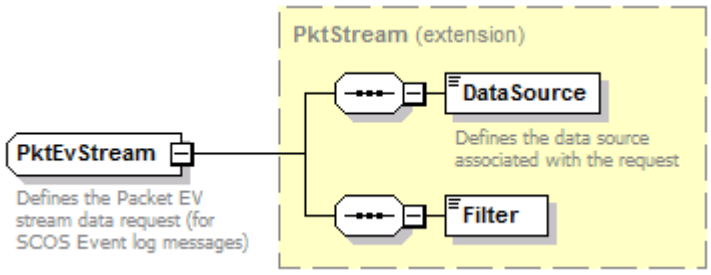
L.1 complexType OolStream

diagram	
namespace	http://edds.egos.esa/model
type	extension of PktStream
properties	base PktStream
children	**DataSource Filter**
annotation	documentation Defines the Out Of Limits stream data request

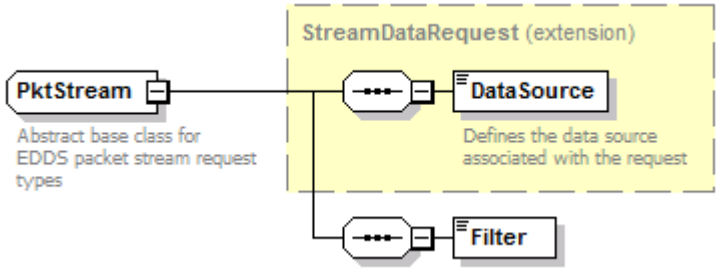
L.2 complexType ParamStream

diagram	
namespace	http://edds.egos.esa/model
type	extension of StreamDataRequest
properties	base StreamDataRequest
children	**ParamNameList ParamTmStreamFilter**
annotation	documentation Defines the Parameter stream data request

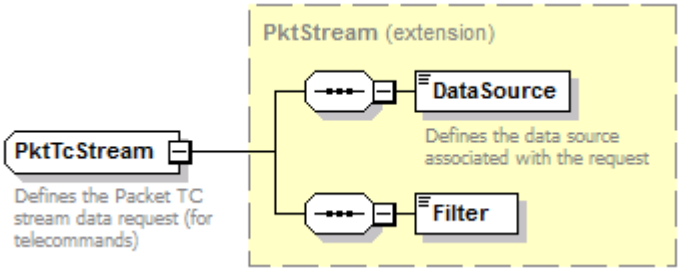
L.3 complexType PktEvStream

diagram	
namespace	http://edds.egos.esa/model
type	extension of PktStream
properties	base PktStream
children	**DataSource Filter**
annotation	documentation Defines the Packet EV stream data request (for SCOS Event log messages)

L.4 complexType PktStream

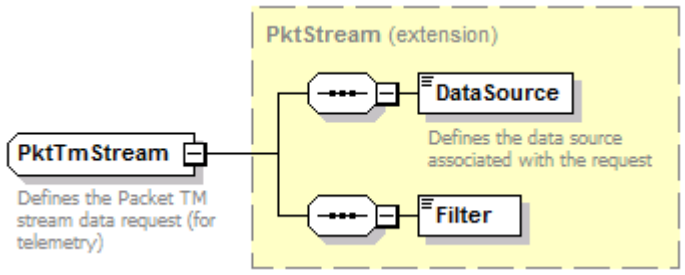
diagram	
namespace	http://edds.egos.esa/model
type	extension of StreamDataRequest
properties	base StreamDataRequest abstract true
used by	complexTypes OolStream PktEvStream PktTcStream PktTmStream
annotation	documentation Abstract base class for EDDS packet stream request types

L.5 complexType PktTcStream

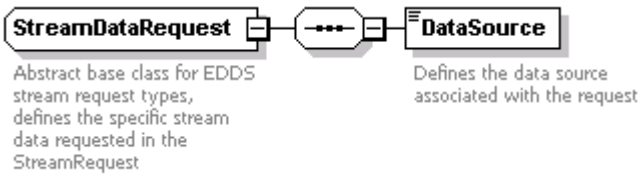
diagram	
namespace	http://edds.egos.esa/model
type	extension of PktStream
properties	base PktStream

children	**DataSource Filter**
annotation	documentation Defines the Packet TC stream data request (for telecommands)

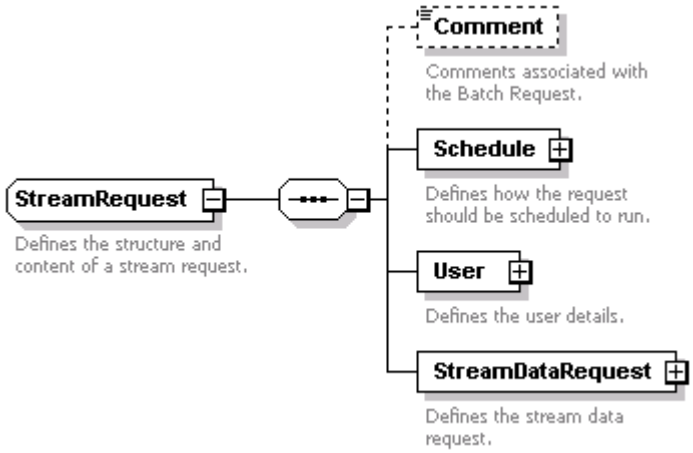
L.6 complexType PktTmStream

diagram	
namespace	http://edds.egos.esa/model
type	extension of **PktStream**
properties	base PktStream
children	**DataSource Filter**
annotation	documentation Defines the Packet TM stream data request (for telemetry)

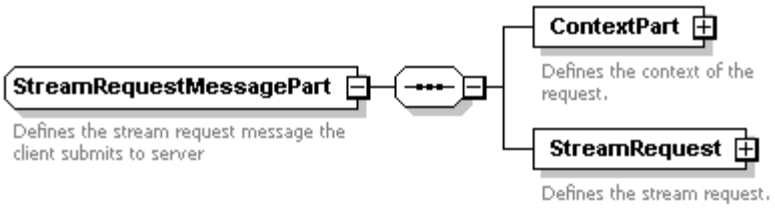
L.7 complexType StreamDataRequest

diagram	
namespace	http://edds.egos.esa/model
properties	abstract true
children	**DataSource**
used by	element **StreamRequest/StreamDataRequest** complexTypes **ParamStream PktStream**
annotation	documentation Abstract base class for EDDS stream request types, defines the specific stream data requested in the StreamRequest

L.8 complexType StreamRequest

diagram	
namespace	http://edds.egos.esa/model
children	**Comment Schedule User StreamDataRequest**
used by	element **StreamRequestMessagePart/StreamRequest**
annotation	documentation Defines the structure and content of a stream request.

L.9 complexType StreamRequestMessagePart

diagram	
namespace	http://edds.egos.esa/model
children	**ContextPart StreamRequest**
used by	element **StreamRequestMessagePart**
annotation	documentation Defines the stream request message the client submits to server

Appendix M EDDS Parameter Spreadsheet

M.1 TDRS like SpreadSheet

EDDS provides a sample XSLT file for the Telemetry Parameter batch request so that the TDRS spreadsheet format can be created from the retrieved XML. The file is called “Param.xsl” and can be modified to each mission’s needs. The file is deployed by default into the EDDS Server runtime directory within “config/XSLT”. The filename must match the report type. More details on the format of the generated TDRS spreadsheet file can be found in the file “TDRS External Interfaces Control Document” [RD-14]

The following fields are included in the output of the TDRS spreadsheet file, but are only filled with dummy values as the information cannot be obtained yet these fields are mandatory in the TDRS ICD:

- Retrieval ID (always set to 0);
- Execution ID (always set to 0);
- Expiration day (always set to 0).

The values can be changed by editing the XSLT file.

As the XSLT file must load the XML data into memory to be able to create the TDRS output file, EDDS also provides a native TDRS format that can work with much larger amounts of data. This format can be selected on the Parameter TM Format request and optionally the number of decimal places can be limited under the TDRS Options section. By default, EDDS will not limit the number of decimal places and so the data will be as received from the DARC/SMON. The restrictions above regarding the mandatory fields that cannot be populated remain in place, except the Retrieval ID is set to the EDDS Request ID.

M.2 Presto Plot Stylesheet

The style sheet for transforming XML results are mission specific and provided by the mission, and so considered out of the scope of this document.

Appendix N Google Protocol Buffers

N.1 Introduction

EDDS uses a software library called Google Protocol Buffers internally for use in passing messages between applications as well as for stream data. Google Protocol Buffers offers several advantages over the use of XML; in particular it is considerably faster and smaller than an XML message – an essential requirement for real-time streaming. More information on Google Protocol Buffers can be found here: http://code.google.com/apis/protocolbuffers/

EDDS uses the Google Protocol Buffers implementation found within the ActiveMQ library. However, the proto files are also compatible with the Protocol Buffers implementation from Googlei Inc. The proto files (which describe the data structure) are compiled into Java classes when compiling the software from source. The proto files can be found in the project edds-ws-common within src/main/protoFiles. The Maven plugin `org.apache.activemq.protobuf:activemq-protobuf` is used to compile the proto files.

EDDS uses the `toFramedByteArray()` and `fromFramedByteArray()` methods (of ActiveMQ protobuf implementation) in the compiled proto code for converting the data to a byte array for sending on the payload of a JMS message and reading the data back again on the receiver's end. Several helper classes exist for reading the encoded byte arrays sent from the server to the client. The Java client API also uses these classes for making it easier for users of the API to access the information returned by the web server. The helper classes can be found in esa.egos.edds.types within edds-ws-common.

There is helper class DarcParameterBinarySupport available to simplify reading data from DARC parameter binary response. This class can be found from project edds-ws-client.

Here is a list of the proto files within EDDS and their purpose:

Proto file	Description
darcparameter.proto	A record containing information about a DARC parameter as placed on the DARC stream topic
farccatalogue.proto	Contains records containing information about files stored in the FARC
Catalogue.proto	Used for EDDS notification messaging
FsCatalogue.proto	Used for EDDS notification messaging about file changse on filesystem
paramdef.proto	A record containing information about a DARC parameter definition
requestinfo.proto	A record containing information about a batch, stream or account request
systemlog.proto	A record containing information about a log message from EDDS
DarcParameterBinary.proto	A record containing information about DARC parameters in Parameter TM request response files in BINARY format
SMONParameter.proto	A record containing information about TM parameters in SMON Parameter request response files in BINARY format
EventRecordReport.proto	A record containing information about EV data in Event Record Report request response files in BINARY format
PacketRaw.proto	A record containing information about raw data in Packet TC Raw, Packet TM Raw and Packet EV Raw requests response files in BINARY format
PacketTCReport.proto	A record containing information about TC data in Packet TC Report request response files in BINARY format
PacketTMReport.proto	A record containing information about TM data in Packet TM Report request response files in BINARY format
PacketTMGapReport.proto	A record containing information about TM gap data in Packet TM Gap Report request response files in BINARY format
ParamValidity.proto	Definition of parameter validity as a separate proto file to be reused by multiple proto file definitions.
OOLReport.proto	A record containing information about out of limit data in Out of Limits Report request response files in BINARY format

