

Date: August 2005

XML Telemetric and Command Exchange (XTCE)

version 1.0
formal/05-08-01



OBJECT MANAGEMENT GROUP

Copyright © 2003, Lockheed Martin
Copyright © 2005, Object Management Group
Copyright © 2003, The Boeing Company
Copyright © 2003, The European Space Agency

USE OF SPECIFICATION - TERMS, CONDITIONS & NOTICES

The material in this document details an Object Management Group specification in accordance with the terms, conditions and notices set forth below. This document does not represent a commitment to implement any portion of this specification in any company's products. The information contained in this document is subject to change without notice.

LICENSES

The companies listed above have granted to the Object Management Group, Inc. (OMG) a nonexclusive, royalty-free, paid up, worldwide license to copy and distribute this document and to modify this document and distribute copies of the modified version. Each of the copyright holders listed above has agreed that no person shall be deemed to have infringed the copyright in the included material of any such copyright holder by reason of having used the specification set forth herein or having conformed any computer software to the specification.

Subject to all of the terms and conditions below, the owners of the copyright in this specification hereby grant you a fully-paid up, non-exclusive, nontransferable, perpetual, worldwide license (without the right to sublicense), to use this specification to create and distribute software and special purpose specifications that are based upon this specification, and to use, copy, and distribute this specification as provided under the Copyright Act; provided that: (1) both the copyright notice identified above and this permission notice appear on any copies of this specification; (2) the use of the specifications is for informational purposes and will not be copied or posted on any network computer or broadcast in any media and will not be otherwise resold or transferred for commercial purposes; and (3) no modifications are made to this specification. This limited permission automatically terminates without notice if you breach any of these terms or conditions. Upon termination, you will destroy immediately any copies of the specifications in your possession or control.

PATENTS

The attention of adopters is directed to the possibility that compliance with or adoption of OMG specifications may require use of an invention covered by patent rights. OMG shall not be responsible for identifying patents for which a license may be required by any OMG specification, or for conducting legal inquiries into the legal validity or scope of those patents that are brought to its attention. OMG specifications are prospective and advisory only. Prospective users are responsible for protecting themselves against liability for infringement of patents.

GENERAL USE RESTRICTIONS

Any unauthorized use of this specification may violate copyright laws, trademark laws, and communications regulations and statutes. This document contains information which is protected by copyright. All Rights Reserved. No part of this work covered by copyright herein may be reproduced or used in any form or by any means--graphic, electronic, or mechanical, including photocopying, recording, taping, or information storage and retrieval systems--without permission of the copyright owner.

DISCLAIMER OF WARRANTY

WHILE THIS PUBLICATION IS BELIEVED TO BE ACCURATE, IT IS PROVIDED "AS IS" AND MAY CONTAIN ERRORS OR MISPRINTS. THE OBJECT MANAGEMENT GROUP AND THE COMPANIES LISTED ABOVE MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS

PUBLICATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF TITLE OR OWNERSHIP, IMPLIED WARRANTY OF MERCHANTABILITY OR WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE OR USE.

IN NO EVENT SHALL THE OBJECT MANAGEMENT GROUP OR ANY OF THE COMPANIES LISTED ABOVE BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, RELIANCE OR COVER DAMAGES, INCLUDING LOSS OF PROFITS, REVENUE, DATA OR USE, INCURRED BY ANY USER OR ANY THIRD PARTY IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

The entire risk as to the quality and performance of software developed using this specification is borne by you. This disclaimer of warranty constitutes an essential part of the license granted to you to use this specification.

RESTRICTED RIGHTS LEGEND

Use, duplication or disclosure by the U.S. Government is subject to the restrictions set forth in subparagraph (c) (1) (ii) of The Rights in Technical Data and Computer Software Clause at DFARS 252.227-7013 or in subparagraph (c)(1) and (2) of the Commercial Computer Software - Restricted Rights clauses at 48 C.F.R. 52.227-19 or as specified in 48 C.F.R. 227-7202-2 of the DoD F.A.R. Supplement and its successors, or as specified in 48 C.F.R. 12.212 of the Federal Acquisition Regulations and its successors, as applicable. The specification copyright owners are as indicated above and may be contacted through the Object Management Group, 250 First Avenue, Needham, MA 02494, U.S.A.

TRADEMARKS

The OMG Object Management Group Logo®, CORBA®, CORBA Academy®, The Information Brokerage®, XMI® and IOP® are registered trademarks of the Object Management Group. OMG™, Object Management Group™, CORBA logos™, OMG Interface Definition Language (IDL)™, The Architecture of Choice for a Changing World™, CORBA services™, CORBA facilities™, CORBA med™, CORBA net™, Integrate 2002™, Middleware That's Everywhere™, UML™, Unified Modeling Language™, The UML Cube logo™, MOF™, CWM™, The CWM Logo™, Model Driven Architecture™, Model Driven Architecture Logos™, MDA™, OMG Model Driven Architecture™, OMG MDA™ and the XMI Logo™ are trademarks of the Object Management Group. All other products or company names mentioned are used for identification purposes only, and may be trademarks of their respective owners.

COMPLIANCE

The copyright holders listed above acknowledge that the Object Management Group (acting itself or through its designees) is and shall at all times be the sole entity that may authorize developers, suppliers and sellers of computer software to use certification marks, trademarks or other special designations to indicate compliance with these materials.

Software developed under the terms of this license may claim compliance or conformance with this specification if and only if the software compliance is of a nature fully matching the applicable compliance points as stated in the specification. Software developed only partially matching the applicable compliance points may claim only that the software was based on this specification, but may not claim compliance or conformance with this specification. In the event that testing suites are implemented or approved by Object Management Group, Inc., software developed using this specification may claim compliance or conformance with the specification only if the software satisfactorily completes the testing suites.

OMG's Issue Reporting Procedure

All OMG specifications are subject to continuous review and improvement. As part of this process we encourage readers to report any ambiguities, inconsistencies, or inaccuracies they may find by completing the Issue Reporting Form listed on the main web page <http://www.omg.org>, under Documents, Report a Bug/Issue (<http://www.omg.org/technology/agreement.htm>).

Table of Contents

Foreword	iii
Introduction	v
1 Scope	1
2 Conformance	1
3 Normative References	2
4 Terms and Definitions	2
5 Symbols	2
6 Additional Information	3
6.1 Acknowledgements	3
7 The Specification	5
7.1 The Root Object – The SpaceSystem	5
7.1.1 The Header Record	6
7.1.2 TelemetryMetaData	6
7.1.3 CommandMetaData	11
7.1.4 ServiceSet	14
7.1.5 Defaults	15
7.2 The Schema	15
Annex A - The SpaceSystem Schema	17
Annex B - Schema Style Notes	75
Bibliography	77

Foreword

This XML Telemetric and Command Exchange (XTCE) data specification answers the need for an information model and data exchange format for telemetry and commanding in all phases of a spacecraft, payload, and ground segment lifecycle: system design, development, test, validation, and mission operations.

This specification addresses a compelling need for a standard exchange format recognized independently by each of its authors and contributors. Lockheed Martin, ESA, Boeing, NASA GSFC, USAF SMC, Harris, Raytheon, SciSys, CSC and GST have all made significant contributions representing a wide and varied sampling of the space industry.

Space mission implementations face a very dynamic environment with fast-paced information technology advancement and shrinking space budgets. A more focused use of decreasing public investments in space requires a cost reduction over their entire lifecycle, from development up to the end of the useful life of a spacecraft. The use of standards specifications from the early stages of satellite development through mission operation will reduce life-cycle cost.

Satellite design and development is performed today through the use of a number of disparate tools and techniques. Interface design to satellite systems and to the payloads the satellites are housing is still a manual and time-consuming effort. Data design, both telemetry and commanding, is still performed multiple times by multiple contractors during the lifecycle of the satellite, well before the satellite is ever deployed for mission operations. The standardization of satellite telemetry and command data for spacecraft health and safety, as well as payload interfaces will reduce the cost of these implementations as well as decrease the schedule of development, integration, and test of the satellite and its component systems. This specification can also be used to support multiple, heterogeneous missions, facilitating interoperability between ground control systems, simulators, testing facilities, etc.

At the heart of the specification is a very robust information model for telemetry and commanding that will support all phases of the satellite, payload, and ground segment lifecycle: system design, development, test, validation, and mission operations.

Introduction

Purpose

This specification is an information model for spacecraft telemetry and commanding data. For a given mission there are a number of lifecycle phases that are supported by a variety of systems and organizations. Additionally, many of these organizations support multiple heterogeneous missions using a common ground segment infrastructure. Telemetry and command definitions must be exchanged among all of these phases, systems, and organizations. This is made difficult and costly because there is no standard method for exchanging this information. The lack of standardization currently requires custom ingestion of the telemetry and commanding information. This customization is inherently error-prone, resulting in the need to revalidate at each step in the lifecycle.

A typical example of this process is between the spacecraft manufacturer and spacecraft-operating agency. The spacecraft manufacturer defines the telemetry and command data in a format that is much different than the one used in the ground segment. This creates the need for database translation, increased testing, software customization, and increased probability of error. Standardization of the command and telemetry data definition format will streamline the process allowing dissimilar systems to communicate without the need for the development of mission specific database import/export tools.

Ideally, a spacecraft operator should be able to transition from one ground system to another by simply moving an already existing command and telemetry database compliant with this command and telemetry database specification.

In addition, standardization will enable space or ground segment simulators to more easily support multiple heterogeneous missions.

XTCE provides a standard format for defining the Telemetric and Telecommand (TM/TC) data required to perform the processing shown in Figure 1.

Overview

The normative portion of this specification is presented as a single XML schema compliant with the W3C recommendation of 02/05/2001. The schema is found in Annex A or may be obtained as an independent convenience document.

The schema has an object-oriented structure where all the elements of the specification belong to a single root object – the SpaceSystem.

Philosophy

The space industry is fragmented between Packet telemetry and commanding and Time Division Multiplexing (TDM) telemetry and commanding. While the basic construction of either TDM or packet telemetry is fundamentally not all that different, nomenclature differences between the two give the appearance of a larger divide. The XTCE specification avoids using nomenclature from either the TDM or packet worlds to avoid any possible confusion; terms like ‘minor frame,’ ‘major frame,’ or packet are nowhere in this specification other than in examples. Furthermore, the XTCE specification does not itself use any existing packet or TDM standards (such as CCSDS packet formats, or IRIG-106 minor frame standards), but it does provide a mechanism to use XTCE to build libraries of available containers that represent these standards.

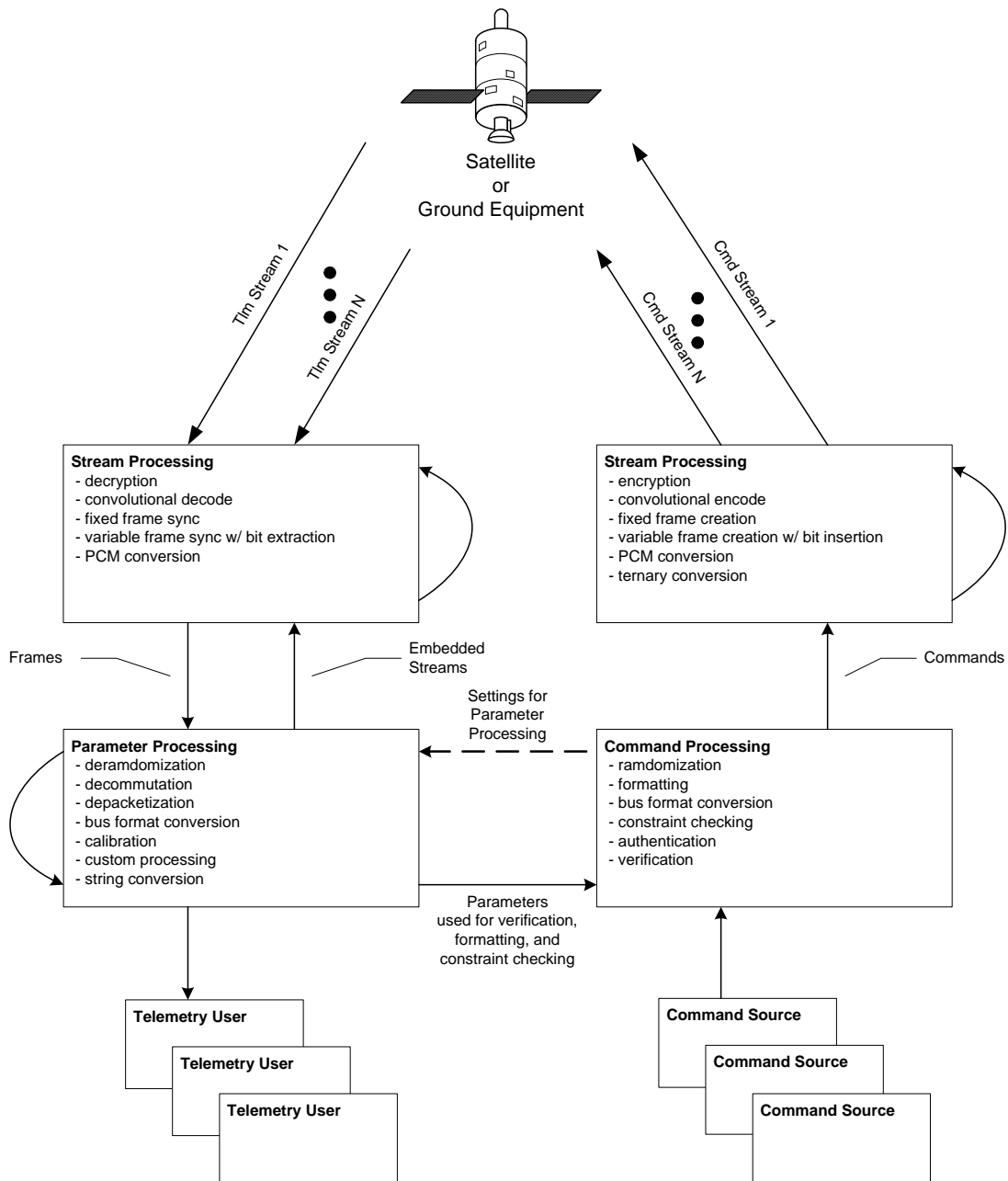


Figure 1.

1 Scope

The specification addresses the need for a standardized information model capable of supporting TM/TC definitions across the broadest possible range of space domain activities. The goal is to allow TM/TC definitions to be exchanged between different organizations and systems, often at the boundaries of mission phases, without the need for customized import/export, re-validation, or even re-implementation of mission databases.

The scope of this specification is limited to satellite telemetry and commanding data constructs necessary to support satellite and payload data design:

- Telemetry data definitions including support for CCSDS packets as well as TDM frames.
- Data manipulation algorithms to support packaging and unpacking of individual data items.
- Commanding data definitions including command identification, argument specification, and validation criteria.
- Data representation definitions.
- Data properties including such things as its default value, validity criteria, and data dependencies.
- The definition of extensible formats such that blocks of information (whether frames of data that are not decommutated or object references or object method calls) can be portrayed in this architecture.

The scope of this specification does not extend to:

- Data distribution mechanisms.
- Command and data protocol specifications.
- RF or analog stream characterization.
- Data groupings including aggregation and coherent data sets.
- Data Representation (visualization properties).
- Scheduling configuration properties.
- Orbital properties.

The specification addresses only the definition of TM/TC data, and not the transfer of live or historical TM/TC data.

2 Conformance

The schema (.xsd file) in Annex A is normative. A compliant database is an XML file that complies with this schema. Fully compliant implementing software will interpret and/or generate any databases compliant with this specification. Compliant implementing software will interpret and/or generate all database elements required by the schema.

3 Normative References

The following normative documents contain provisions which, through reference in this text, constitute provisions of this specification. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply.

http://www.w3.org/TR/REC-xml	W3C Recommendation - Extensible Markup Language (XML) 1.0 (Second Edition, 6 October 2000)
http://www.w3.org/TR/xmlschema-0/	W3C Recommendation - XML Schema Part 0: Primer (2 May 2001)
http://www.w3.org/TR/xmlschema-1/	W3C Recommendation - XML Schema Part 1: Structures (2 May 2001)
http://www.w3.org/TR/xmlschema-2/	W3C Recommendation - XML Schema Part 2: Datatypes (2 May 2001)

4 Terms and Definitions

For the purposes of this specification, the following terms and definitions apply.

Telemetry

(from IEEE Std 1000 [1972]) “Measurement with the aid of intermediate means that permit the measurement to be interpreted at a distance from the primary detector.” All measurements on board the spacecraft are transmitted to the ground system in a telemetry stream. Telemetry as used here refers to these measurements whether on-board the spacecraft or transmitted to the ground system. Most telemetry measurements will require engineering unit conversion and measurements will have associated validation ranges or lists of acceptable values.

Commands

Messages that instruct an action on a remote system. Spacecraft commanding usually implies coding and packaging of the command information, validation and verification, as well as authorization to perform. Telemetry and Commanding data are necessarily related to one another, with some command information originating from telemetry and commands relating to particular telemetry measurements. Therefore, the ability to relate individual telemetry with one another and to commands is a very important part of this specification. Packaging of both telemetry and commands can be performed in a number of ways. The most common way to package data for transmission is to use the CCSDS Telemetry and Commanding Packaging format.

5 Symbols

List of symbols/abbreviations

In general, the XTCE specification favors expressive, fully spelled out terms over abbreviated notation. The exceptions are modifiers used as prefixes or postfixes to objects used within the schema, and of course ‘XTCE’ the name of the standard itself. These terms are listed below.

Abbreviations

Parm	An abbreviation sometimes used for Parameter.
XTCE	XML Telemetric and Command Exchange format.

Prefixes and Postfixes

Meta	A description. For example a MetaCommand is a command description.
Set	An unordered collection. For example a MetaCommandSet is an unordered collection of command descriptions.
List	An ordered collection. For example an ArgumentList is an ordered collection of arguments.
Ref	A reference (by name) to an object defined elsewhere in the XML document. For example an ArgumentRef is a named reference to an Argument defined elsewhere.

6 Additional Information

6.1 Acknowledgements

The following companies submitted and/or supported parts of this specification:

- Lockheed Martin
- The Boeing Company
- The European Space Agency

7 The Specification

7.1 The Root Object – The SpaceSystem

Recognizing that spacecraft operations involve much more than simply controlling the spacecraft, the top-level object is not ‘Spacecraft’ but the more generic term ‘SpaceSystem.’ This name provides deference to the fact that a spacecraft operations center must control antennas, recorders, ground processing equipment, RF hardware, and many other assets that may use this data specification; each of these objects is a ‘SpaceSystem.’ A SpaceSystem, like all of the major objects in an XTCE database, may have a short description, a long description (that may contain HTML markup documentation) and a list of alias names. A SpaceSystem may have a Header, zero or more sub-SpaceSystems, CommandMetaData, and TelemetryMetaData. The CommandMetaData and TelemetryMetaData components contains the bulk of the Telemetric and Command data. The sub-SpaceSystems give the data a hierarchical structure.

Note – on the sub-SpaceSystem and the hierarchical structure:

Because a SpaceSystem may itself contain other SpaceSystems, the organization of the data may be organized hierarchical structure – similar to the structure of a real space system. The hierachical organization offers several important advantages over a flat entity list:

- Fewer name space collisions – almost every spacecraft contains redundant components for reliability or to accomplish the mission. A communications spacecraft may have a dozen transponders each with the same set of telemetry points and commands. In a flat namespace each of those telemetry points needs to be mapped into a unique name. Using a hierarchical namespace, those identical telemetry points can be simply placed into separate sub-SpaceSystems.
- Better organization – modern spacecraft typically have thousands of commands and tens of thousands of telemetry parameters; this number is trending upward. The directory structure provided by this specification provides an improved way to manage this large volume of data. Each subsystem developer can deliver SpaceSystems representing their subsystem without integration issues.
- Defaults at the SpaceSystem level – many of the attributes needed to define spacecraft parameters (e.g., bit order, byte order) are common to most of the parameters in the spacecraft or spacecraft sub-system. This specification allows these attributes to be assigned at the directory level, thereby avoiding their repetition in each parameter.
- Spacecraft that are normally thought of as a SpaceSystem, may actually be sub-SpaceSystems for a constellation of spacecraft SpaceSystems.
- Natural hierarchy – spacecraft designs are increasing in complexity and are normally comprised of systems of systems. The hierarchical organization allowed by a directory structure reflects this.

Note – on Names:

Parameter, and MetaCommand and other major entity names within this database may be any length but are prohibited from containing the ‘/’, ‘.’, and ‘:’ characters as these are reserved. The ‘/’ is used as the SpaceSystem separator (Unix and HTTP style). The ‘:’ is reserved for future use as a selector for data from other SpaceSystems. The ‘.’ is reserved as an attribute selector.

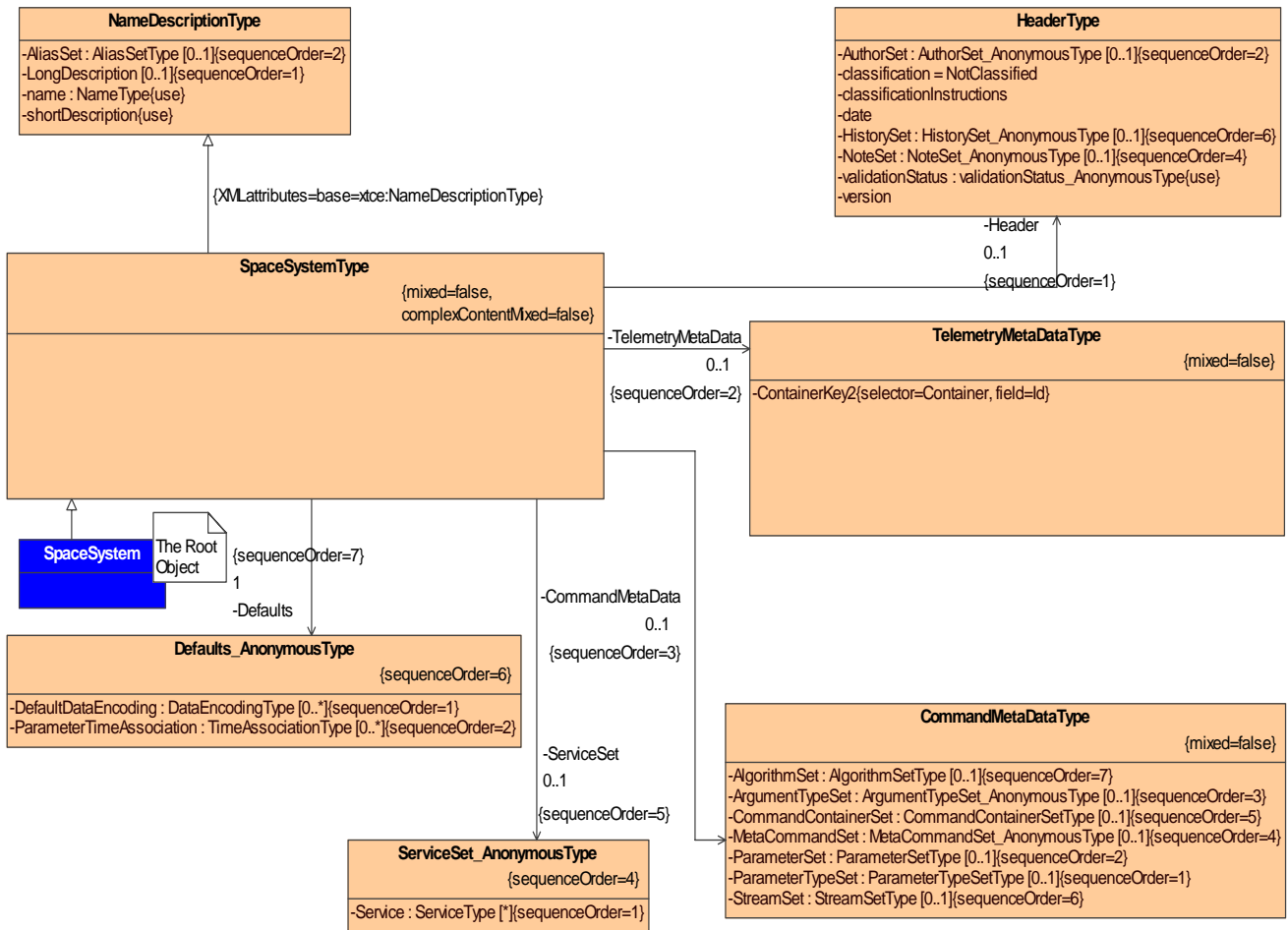


Figure 7.1 - SpaceSystem

7.1.1 The Header Record

A SpaceSystem may contain an optional header record. This record contains some basic context on the data itself (e.g., source, version, revision history, notes, and classification).

7.1.2 TelemetryMetaData

Because Telemetry and Command databases are frequently developed and maintained independently, the XTCE format divides TelemetryMetaData and CommandMetaData into separate, but similar sections. TelemetryMetaData is really nothing more than a grouping for data about Telemetry. TelemetryMetaData has a ParameterTypeSet, a ParameterSet, a ContainerSet, a MessageSet, a StreamSet, and an AlgorithmSet. Following are descriptions of these collection types.

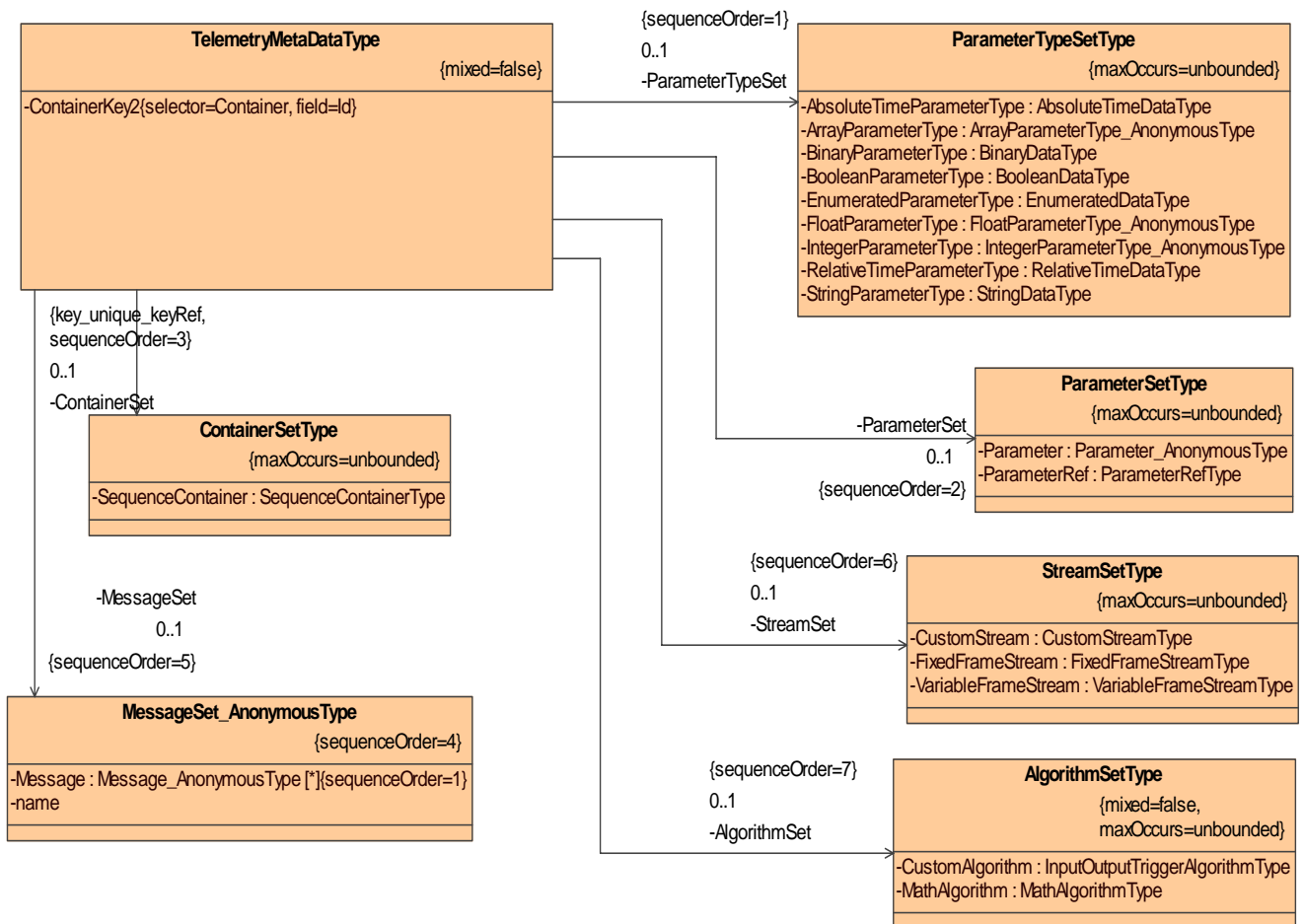


Figure 7.2 - Telemetry MetaData

ParameterTypeSet

A ParameterTypeSet is an unordered collection of ParameterTypes. ParameterTypes are the MetaData for Parameters; ParameterTypes are instantiated to create Parameters. ParameterType is the description of something that can have a value (a Parameter). Information contained in ParameterType includes the data type, description, alarm limits, engineering units, and string conversion specifications and calibrations. Most Parameters are telemetered parameters (a.k.a measurands) and must also include information about how the Parameter value is encoded for transmission. This information includes size in bits, byte order, data type, and parity checks. All of the encoding information is in the Encoding area.

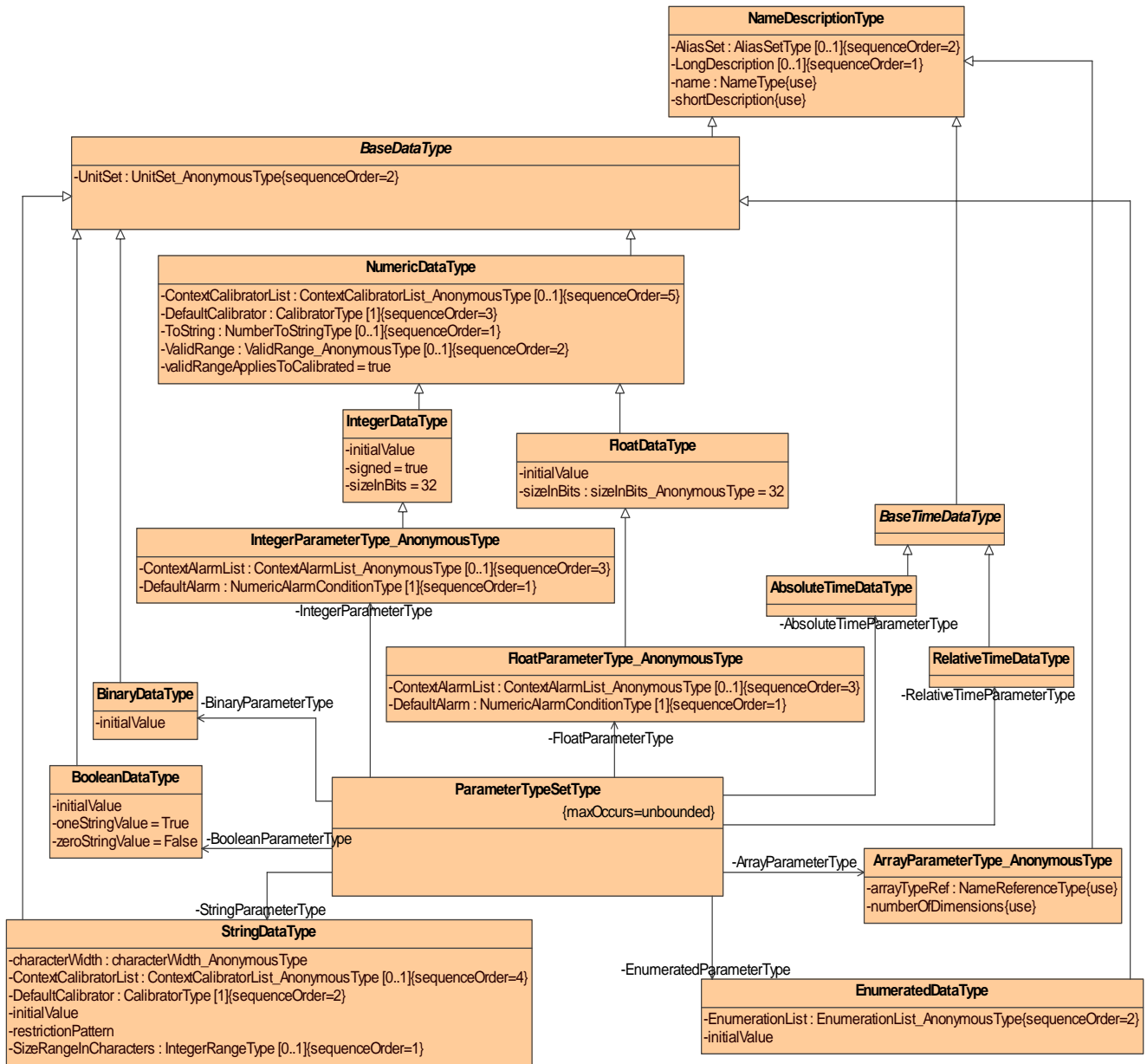


Figure 7.3 - ParameterTypeSet

ParameterSet

A ParameterSet is an unordered collection of Parameters. Parameters are instantiations of ParameterTypes. Parameters are normally a very simple name and reference to a ParameterType. Parameters may also have alias names and may have properties unique to that instantiation. At any point in time (instance) a Parameter has a value; a Parameter is not the value itself. Parameter names are case sensitive, may be any length, cannot contain spaces ‘.’, ‘/’, ‘[’ or ‘]’ as these are reserved characters. The aliases have no restrictions.

ContainerSet

A ContainerSet is an unordered collection of SequenceContainers. SequenceContainers are defined in the packaging section. The packaging section contains the information required to assemble an uplink from its component parts and disassemble a downlink from its component parts. The packaging section has been created to be extremely generic so that it may be used to define TDM telemetry streams, packetized streams, or any other package format. A SequenceContainer contains (in the EntryList) an ordered list of raw parameters, parameter segments, stream segments, containers, or container segments. SequenceContainers may inherit from other sequence containers. The inheritance aspect of SequenceContainers is useful not only for minimizing the effort required to describe a family of SequenceContainers, but is also a very powerful and expressive means of container identification – the process of distinguishing one container from others (e.g., minorFrame 20 is a type of minorFrame where the minor frame counter equals 8). SequenceContainer inheritance may be arbitrarily deep.

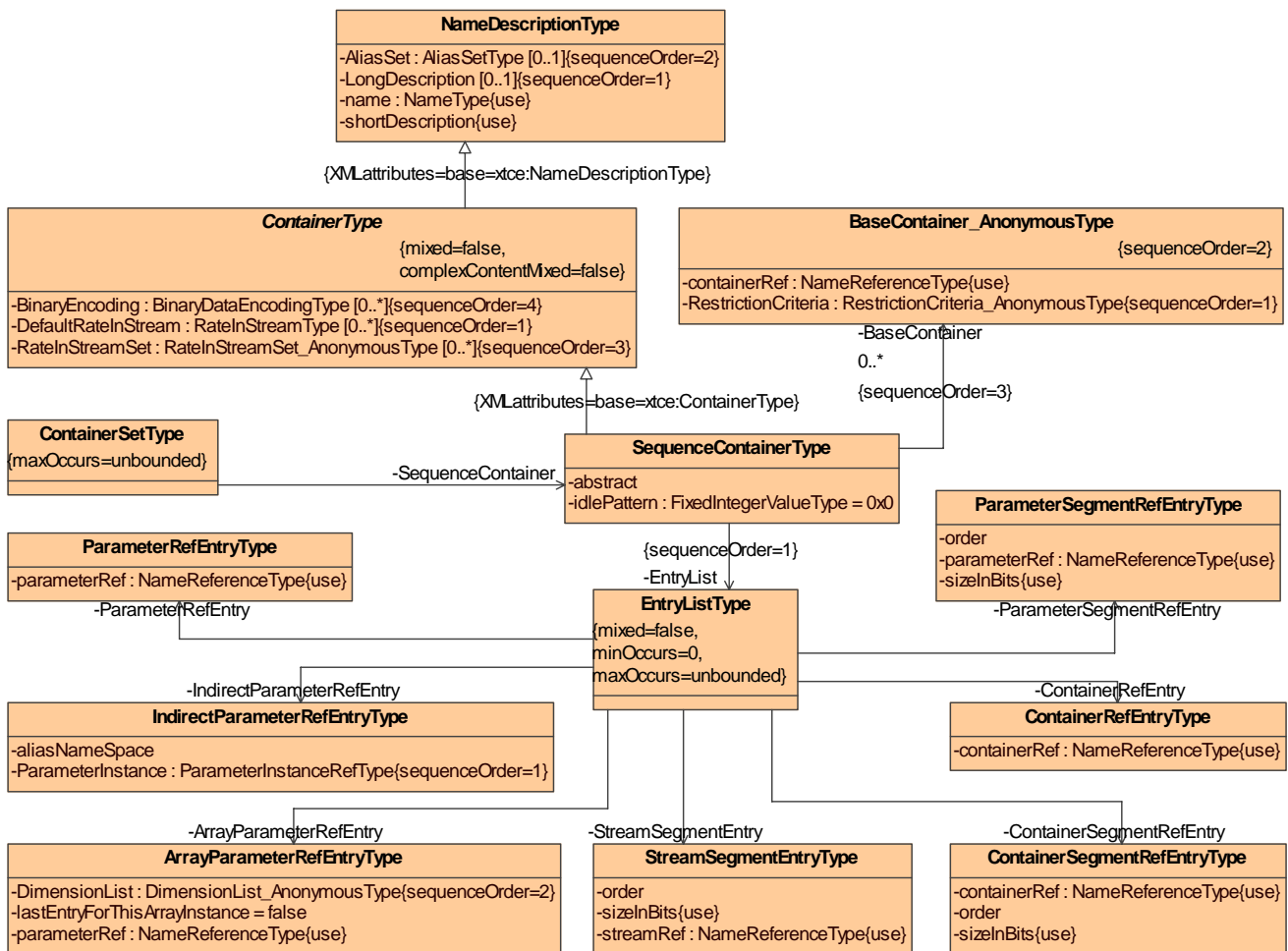


Figure 7.4 - ContainerSet

A SequenceContainer may represent a packet, a frame, a sub-frame, or any other grouping/structure of data items. The simple form of a Sequence element is an ordered set of Parameter References or other Container References.

MessageSet

A MessageSet is an unordered collection of Messages. Messages are an alternative method of uniquely identifying containers within a Service. A message provides a test in the form of MatchCriteria to match to a container. A simple example might be: When minorframeID=21, the message is the 21st minorframe container. The collection of messages to search through will be bound by a Service.

StreamSet

A StreamSet is an unordered collection of Streams. Spacecraft uplinks and spacecraft downlinks are digital streams of data and there are a number of processing functions that are done on the stream level. The stream section contains the knowledge for how to assemble, disassemble, and process spacecraft uplink and downlink streams.

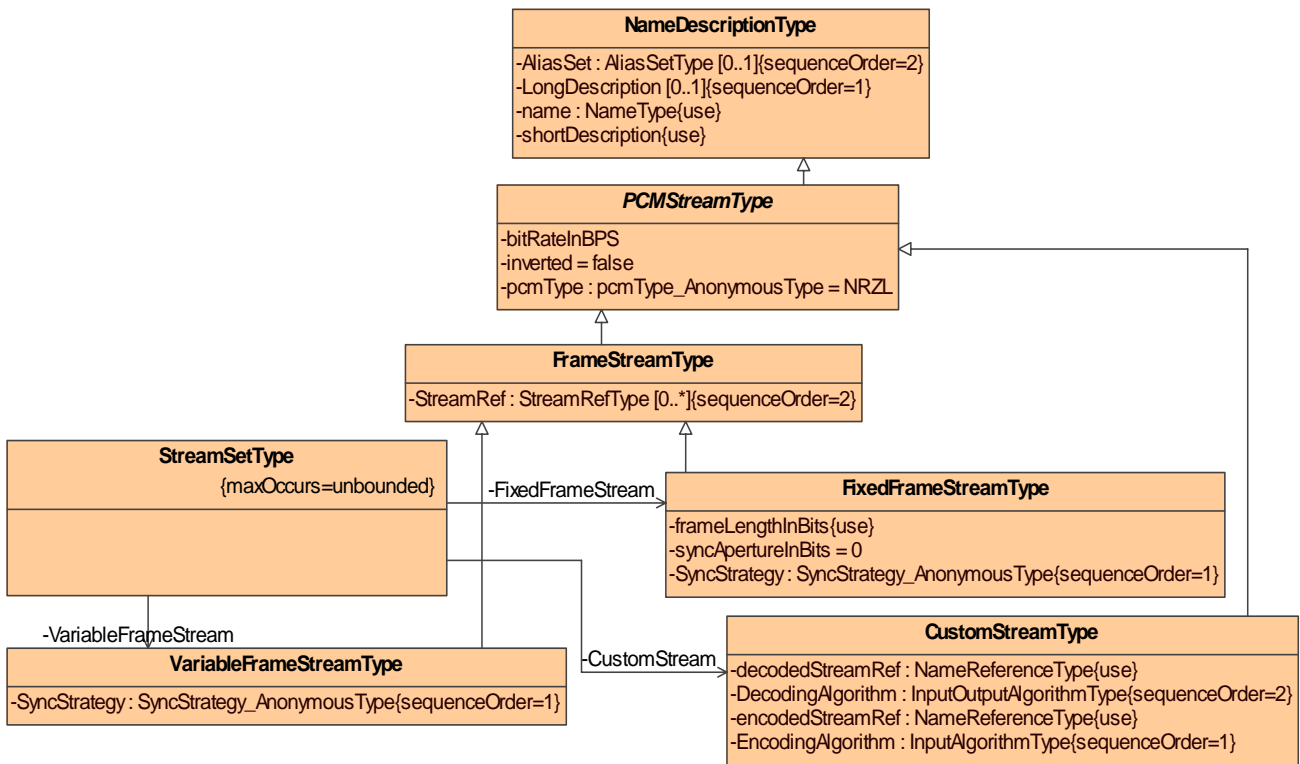


Figure 7.5 - StreamSet

AlgorithmSet

An AlgorithmSet is an unordered collection of Algorithms. In spacecraft ground systems, it is necessary to perform some specialized processing to process the telemetry, and preprocess commands. There are a number of predefined algorithms and the algorithm section makes it possible to reference externally defined algorithms for arbitrarily sophisticated data processing.

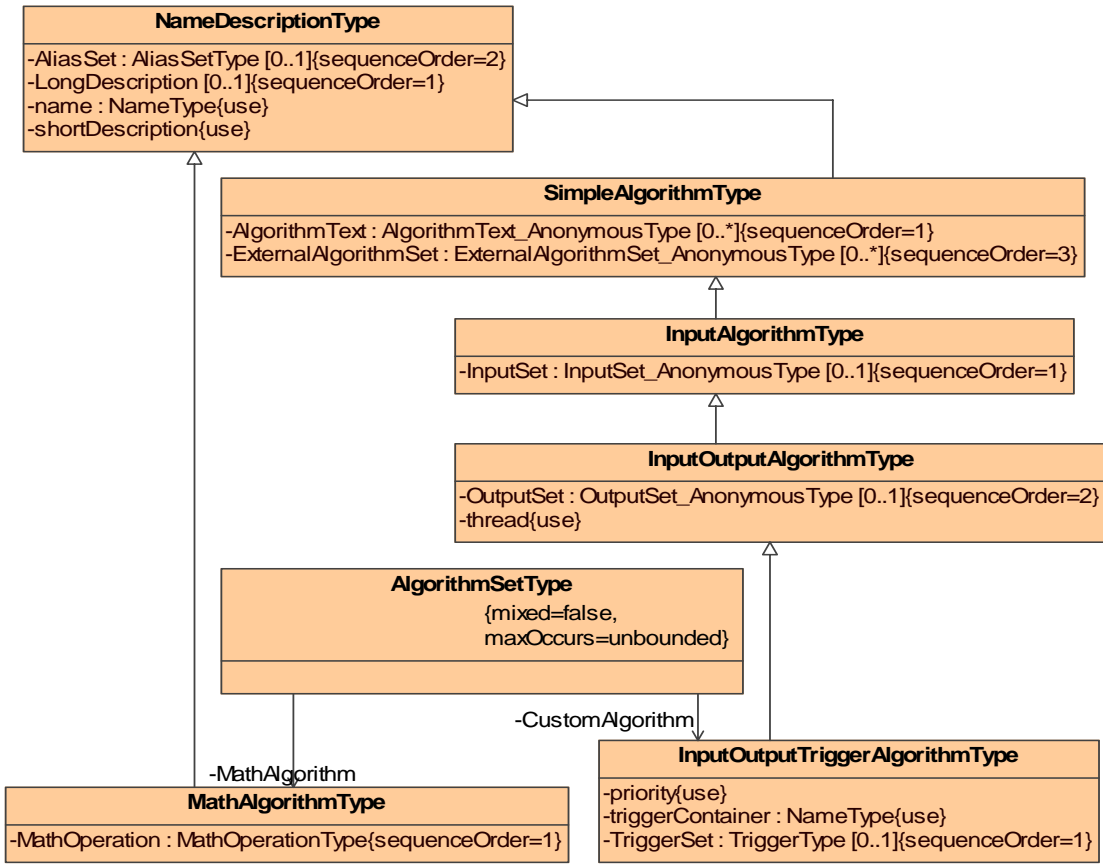


Figure 7.6 - AlgorithmSet

7.1.3 CommandMetaData

The CommandMetaData element is very similar to TelemetryMetaData, but also contains information that is specific only to commanding. CommandMetaData has a ParameterTypeSet, a ParameterSet, a ContainerSet, a MessageSet, a StreamSet, and an AlgorithmSet – exactly like TelemetryMetaData. CommandMetaData, however, also has an ArgumentTypeSet and a MetaCommandSet.

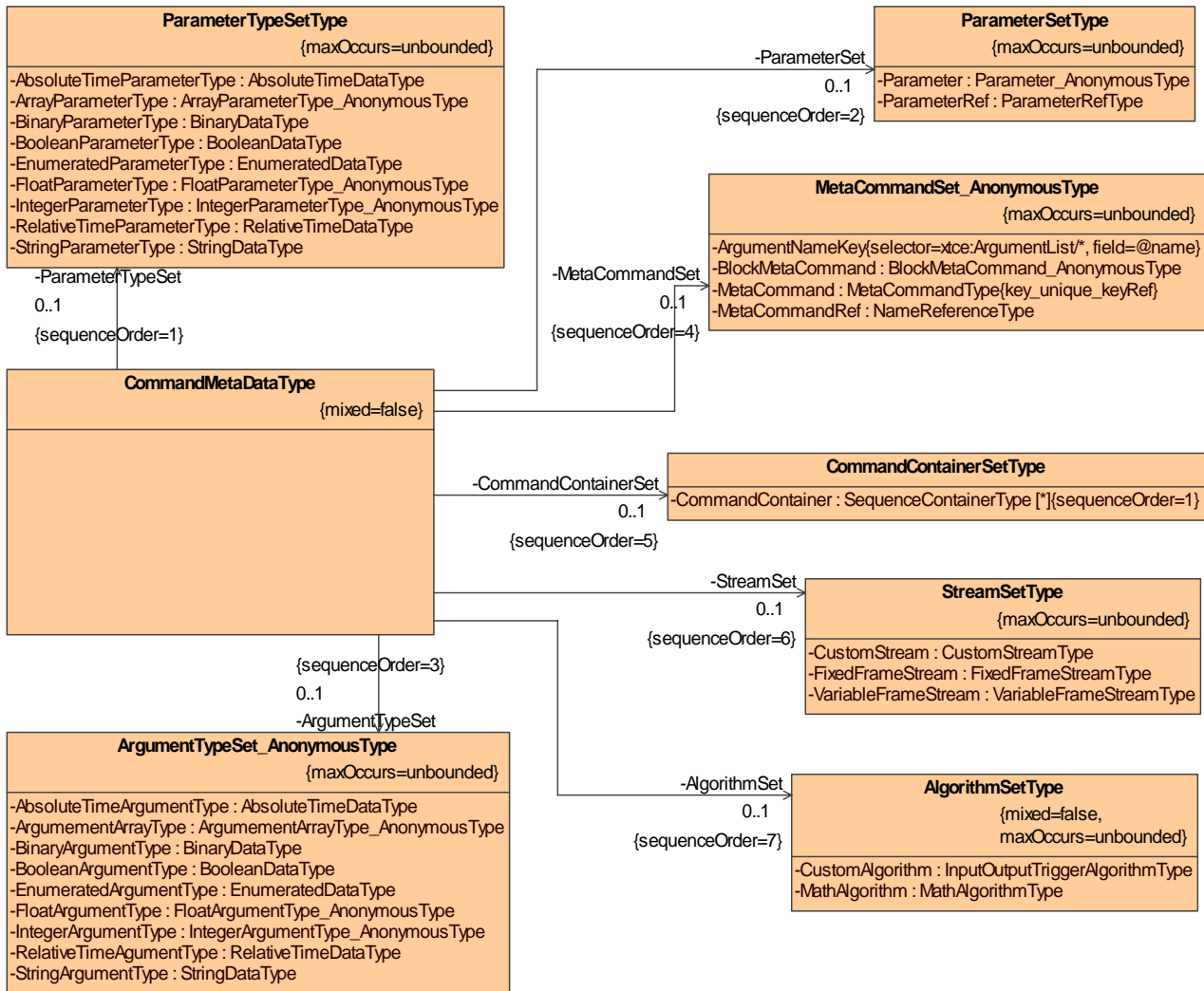


Figure 7.7 - CommandMetaData

ArgumentTypeSet

An ArgumentTypeSet is an unordered collection of ArgumentTypes. ArgumentTypes (very similar to ParameterTypes) are the MetaData for Command Arguments; ArgumentTypes are instantiated to create Arguments. ArgumentType contains the description of something that can have a value and is used as an operator supplied option to a Command (Command Argument). Information contained in ArgumentType includes the data type, description, valid range, engineering units, and string conversion specifications and calibrations. Most Arguments, are sent via a data link and must also include information about how the value is encoded for transmission. This information includes size in bits, byte order, data type, and parity checks. All of the encoding information is in the Encoding area.

MetaCommandSet

A MetaCommandSet contains an unordered collection of MetaCommands. MetaCommands are descriptions of commands. MetaCommands have a name, a BaseMetaCommand, an ArgumentList, a CommandContainer, a TransmissionConstraintList, a DefaultSignificance, a ContextSignificanceList, an Interlock, Verifiers, and a ParameterToSetList.

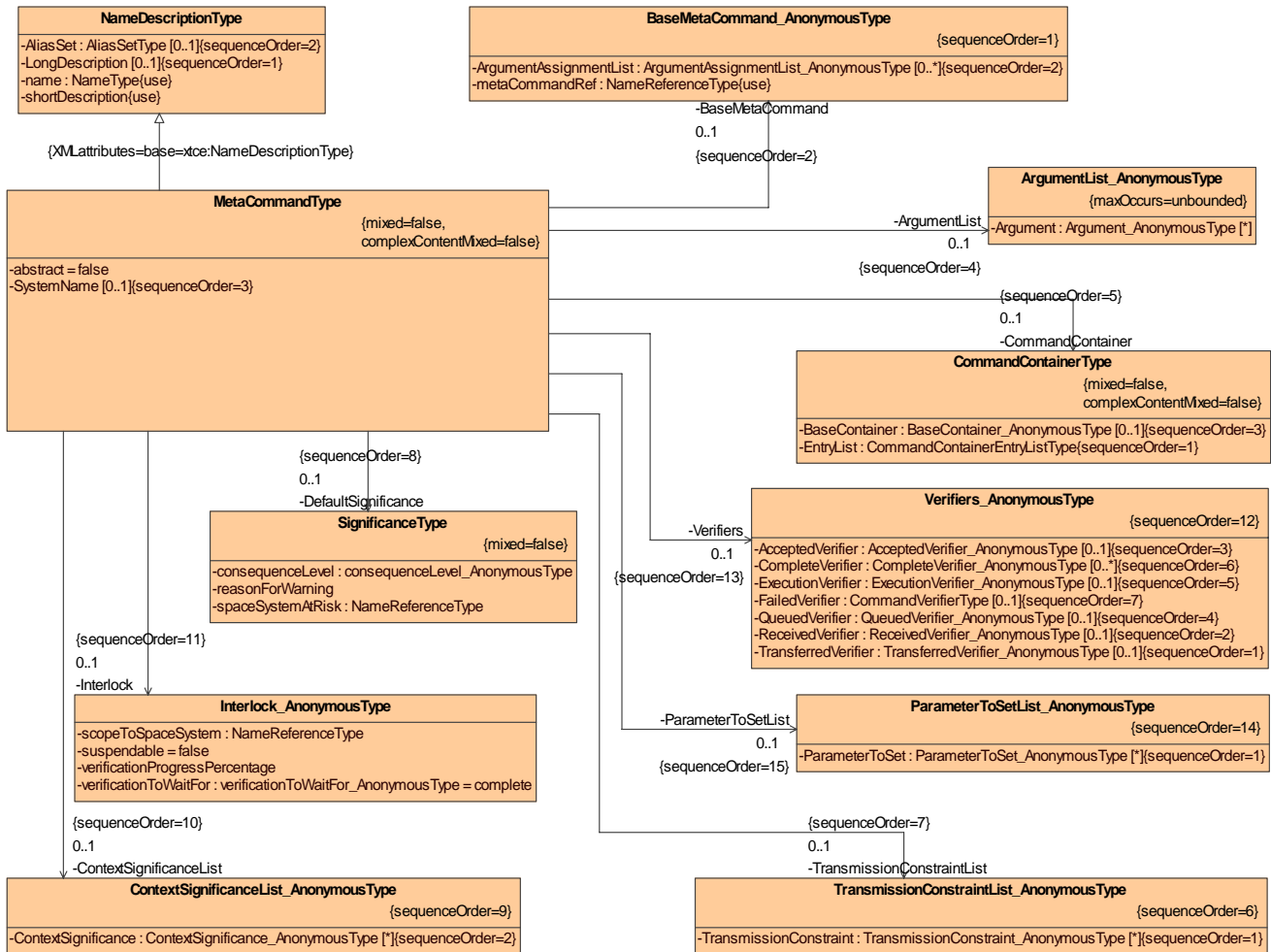


Figure 7.8 - MetaCommandType

BaseMetaCommand

The MetaCommand is derived from this Base. Arguments of the base MetaCommand are further specified in this MetaCommand.

ArgumentList

An ArgumentList is an ordered collection of Arguments. Many commands have one or more options. These are called command arguments. Command arguments may be of any of the standard data types. MetaCommand arguments are local to the MetaCommand.

CommandContainer

A Command Container tells how to package this command – very similar to SequenceContainers, but CommandContainers may also have arguments and fixed values in the sequence. Each MetaCommand may have one CommandContainer.

TransmissionConstraintList

TransmissionConstraintList is an ordered list of TransmissionConstraints. A CommandTransmission constraint is used to check that the command can be run in the current operating mode and may block the transmission of the command if the constraint condition is true.

DefaultSignificance and ContextSignificanceList

Some Command and Control Systems may require special user access to our confirmations before transmitting commands with certain levels. The Significance includes the name of the SpaceSystem at risk, and a significance level. MetaCommands will also inherit any Significance defined in the Base MetaCommand. Significance levels are: none, watch, warning, distress, critical, and severe. Additionally, it is possible to change or have different significance levels set as driven by the operating context of the SpaceSystem.

Interlock

An Interlock is a type of Constraint, but not on Command instances of this MetaCommand; Interlocks apply instead to the next command. An Interlock will block successive commands until this command has reached a certain stage (through verifications). Interlocks are scoped to a SpaceSystem basis.

Verifiers

A Command Verifier is a conditional check on the telemetry from a SpaceSystem that provides positive indication on the successful execution of a command. There are eight different verifiers, each associated with difference stages in command completion: TransferredToRange, TransferredFromRange, Received, Accepted, Queued, Execution, Complete, and Failed. There may be multiple completion verifiers. Completed verifiers are added to the Base MetaCommand verifiers. All others will replace a verifier defined in a Base MetaCommand.

ParameterToSetList

The ParameterToSetList is an ordered collection of ParametersToSet. A ParameterToSet is a Parameter whose value will be set after the Command has reached a certain stage. New Parameters to Set are appended to the Base Command list.

7.1.4 ServiceSet

ServiceSet is an unordered collection of Services. A service is a logical grouping of containers and/or messages.

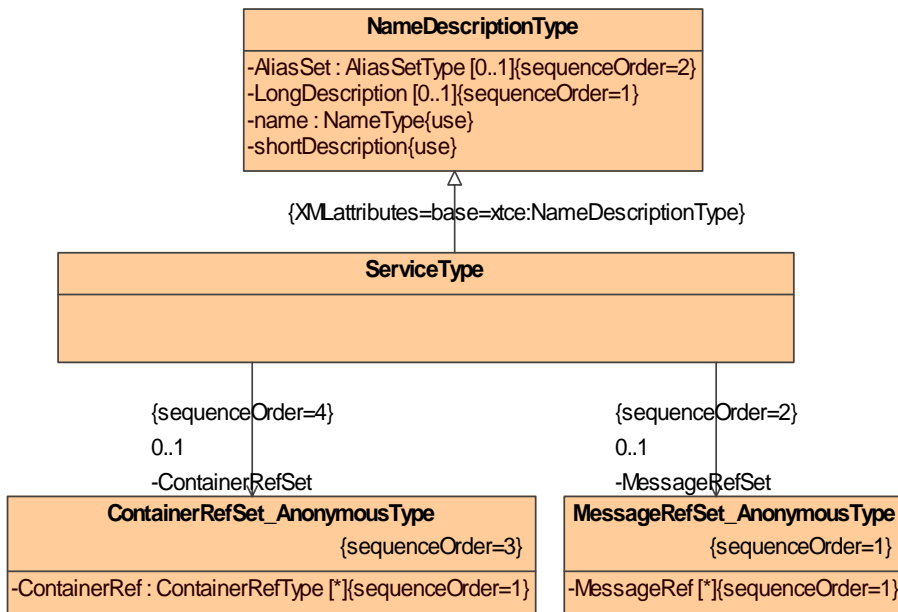


Figure 7.9 - ServiceType

7.1.5 Defaults

Defaults has default data encoding for ParameterTypes and ArgumentTypes and default parameter time association that will be applied to all Parameters within this SpaceSystem. These defaults may be overridden by sub-SpaceSystems or by the Types or Parameters themselves. Defaults simply provides a means to avoid repeating attributes such as 'bit order' for every Type definition.

7.2 The Schema

The W3C XML schema is the normative specification. The schema is provided in Annex A. Any XML document compliant with the specification must validate with the schema and any other rules noted in the 'appinfo' annotation. Style notes used within the schema are provided in Annex B.

Annex A (normative)

The SpaceSystem Schema

A.1 Introduction

The XTCE normative specification is contained entirely as a W3C XML Schema. This schema is available as a standalone document at <http://www.omg.org/space/xtce/SpaceSystem1.0.xsd>

A.2 Schema Text

```
<?xml version="1.0" encoding="UTF-8"?>  
<!--
```

Style Notes, used throughout the schema:

- Element and Type names begin with a capitol letter.
- Type names end with the word "Type".
- Attribute names begin with a lowercase letter.
- Usually, when the UML class diagram references classes, W3C Elements are used, and whenever the UML referencees simple types (strings, ints), W3C Attributes are used. In general, attributes are preferred over elements because they're easier to deal with in SAX and DOM, but whenever the Element/Attribute may one day carry metadata, elements should be used. One exception, is enumerated classes, because enumerations may be defined for attributes but not for elements.
- Bias toward self-describing names over short, bandwidth conserving ones.
- Use mixed case in names rather than underscores to combine multiple words (camelCase).
- A documentation annotation is included in every element and type definition. Annotations for a type are included with the type definition, use of the type is annotated in the element definition.
- Hints on units (for values with units) are provided in the names of attributes and elements (e.g. "dataRateInBPS" is preferred over "dataRate" OR "frameLengthInBits" is preferred over "frameLength").
- Major elements or any elements used multiple times are first defined with a complexType definition
- All collections are put inside either a "List" element or a "Set" Element depending on whether the collection is ordered or unordered.
- Simplicity in the XML files is favored over simplicity in the Schema
- Whenever an additional validity check must be performed that is not describable in the schema language, an appinfo annotation describes that validity check.

-->

```
<schema targetNamespace="http://www.omg.org/space/xtce" xmlns:xtce="http://www.omg.org/space/xtce" xmlns="http://www.w3.org/2001/  
XMLSchema" elementFormDefault="qualified" attributeFormDefault="unqualified">  
  <annotation>  
    <documentation>OMG Document Number: dtc/2005-01-05</documentation>  
    <documentation>$Id: SpaceSystemV1.0.xsd,v 1.15 2005/01/24 05:02:50 gerry Exp $</documentation>  
    <documentation xml:lang="en">This is the master schema for the OMG Space Domain Task Force XML Telemetric and Command  
data Exchange (XTCE) format.</documentation>  
  </annotation>  
  <!-- ***** SpaceSystem -->  
  <element name="SpaceSystem" type="xtce:SpaceSystemType" nillable="true">
```

```

<annotation>
  <documentation>The ROOT Element</documentation>
</annotation>
<key name="parameterNameKey">
  <annotation>
    <documentation>ensure unique parameter name at the system level</documentation>
  </annotation>
  <selector xpath="xtce:TelemetryMetaData/ParameterSet/*"/>
  <field xpath="@name"/>
</key>
<key name="metaCommandNameKey">
  <annotation>
    <documentation>ensure unique metaCommand name at the system level</documentation>
  </annotation>
  <selector xpath="xtce:MetaCommandData/MetaCommandSet/*"/>
  <field xpath="@name"/>
</key>
<key name="algorithmNameKey">
  <annotation>
    <documentation>ensure unique algorithm name at the system level</documentation>
  </annotation>
  <selector xpath="xtce:AlgorithmSet/*"/>
  <field xpath="@name"/>
</key>
<key name="serviceNameKey">
  <annotation>
    <documentation>ensure unique service name at the system level</documentation>
  </annotation>
  <selector xpath="xtce:ServiceSet/*"/>
  <field xpath="@name"/>
</key>
</element>
<complexType name="SpaceSystemType" mixed="false">
  <annotation>
    <documentation>SpaceSystem is a collection of SpaceSystem(s) including space assets, ground assets, multi-satellite systems and sub-systems. A SpaceSystem is the root element for the set of data necessary to monitor and command an arbitrary space device - this includes the binary decomposition the data streams going into and out of a device.</documentation>
  </annotation>
  <complexContent mixed="false">
    <extension base="xtce:NameDescriptionType">
      <sequence>
        <element name="Header" type="xtce:HeaderType" minOccurs="0"/>
        <element name="TelemetryMetaData" type="xtce:TelemetryMetaDataType" minOccurs="0"/>
        <element name="CommandMetaData" type="xtce:CommandMetaDataType" minOccurs="0"/>
        <element name="ServiceSet" minOccurs="0">
          <annotation>
            <documentation>A service is a logical grouping of container and/or messages.</documentation>
          </annotation>
          <complexType>
            <sequence>
              <element name="Service" type="xtce:ServiceType" maxOccurs="unbounded"/>
            </sequence>
          </complexType>
        </element>
        <element name="Defaults" minOccurs="0">
          <annotation>

```

`<documentation>`Defaults has default data encoding for ParameterTypes and ArgumentTypes and default parameter time association that will be applied to all Parameters within this SpaceSystem. These defaults may be overridden by sub-SpaceSystems or by the Types or Parameters themselves. Defaults simply provides a means to avoid repeating attributes such as 'bit order' for every Type definition.`</documentation>`

```

</annotation>
<complexType>
  <sequence>
    <element name="DefaultDataEncoding" type="xtce:DataEncodingType" minOccurs="0">
      <annotation>
        <documentation>Since the data encoding (bit order and byte order) is normally the same throughout
a spacesystem, using this element allows that data encoding to be specified as a default.</documentation>
      </annotation>
    </element>
    <element name="ParameterTimeAssociation" type="xtce:TimeAssociationType" minOccurs="0">
      <annotation>
        <documentation>Default time to associate each ParameterInstance with.</documentation>
      </annotation>
    </element>
  </sequence>
</complexType>
</element>
<element ref="xtce:SpaceSystem" minOccurs="0" maxOccurs="unbounded"/>
</sequence>
</extension>
</complexContent>
</complexType>
<complexType name="CommandMetaData" mixed="false">
  <annotation>
    <documentation xml:lang="en">Command Meta Data contains information about commands</documentation>
  </annotation>
  <sequence>
    <element name="ParameterTypeSet" type="xtce:ParameterTypeSetType" minOccurs="0">
      <annotation>
        <documentation>A list of parameter types</documentation>
      </annotation>
    </element>
    <element name="ParameterSet" type="xtce:ParameterSetType" minOccurs="0">
      <annotation>
        <documentation>Parameters referenced by MetaCommands. This Parameter Set is located here so that MetaCommand
data can be built independantly of TelemetryMetaData.</documentation>
      </annotation>
    </element>
    <element name="ArgumentTypeSet" minOccurs="0">
      <complexType>
        <choice maxOccurs="unbounded">
          <element name="StringArgumentType" type="xtce:StringDataType"/>
          <element name="EnumeratedArgumentType" type="xtce:EnumeratedDataType"/>
          <element name="IntegerArgumentType">
            <complexType>
              <complexContent>
                <extension base="xtce:IntegerDataType">
                  <sequence>
                    <element name="DefaultAlarm" type="xtce:NumericAlarmConditionType" minOccurs="0">
                    <element name="ContextAlarmList" minOccurs="0">
                      <complexType>
                        <sequence>

```

```

<element name="ContextAlarm" type="xtce:ContextAlarmType"
maxOccurs="unbounded"/>
    </sequence>
  </complexType>
</element>
</sequence>
</extension>
</complexContent>
</complexType>
</element>
<element name="BinaryArgumentType" type="xtce:BinaryDataType"/>
<element name="FloatArgumentType">
  <complexType>
    <complexContent>
      <extension base="xtce:FloatDataType">
        <sequence>
          <element name="DefaultAlarm" type="xtce:NumericAlarmConditionType" minOccurs="0"/>
          <element name="ContextAlarmList" minOccurs="0">
            <complexType>
              <sequence>
                <element name="ContextAlarm" type="xtce:ContextAlarmType"
maxOccurs="unbounded"/>
                </sequence>
              </complexType>
            </element>
          </sequence>
        </extension>
      </complexContent>
    </complexType>
  </element>
</choice>
</complexType>
</element>
<element name="BooleanArgumentType" type="xtce:BooleanDataType"/>
<element name="RelativeTimeArgumentType" type="xtce:RelativeTimeDataType"/>
<element name="AbsoluteTimeArgumentType" type="xtce:AbsoluteTimeDataType"/>
<element name="ArgumentArrayType">
  <complexType>
    <attribute name="arrayType" type="xtce:NameReferenceType" use="required"/>
    <attribute name="numberOfDimensions" type="positiveInteger" use="required"/>
  </complexType>
</element>
</choice>
</complexType>
</element>
<element name="MetaCommandSet">
  <annotation>
    <documentation>A set of Command Definitions</documentation>
  </annotation>
  <complexType>
    <choice maxOccurs="unbounded">
      <element name="MetaCommand" type="xtce:MetaCommandType">
        <annotation>
          <documentation>All commands to be sent on this mission are listed here. In addition this area has
verification and validation information</documentation>
        </annotation>
        <key name="ArgumentNameKey">
          <selector xpath="xtce:ArgumentList/*"/>
          <field xpath="@name"/>
        </key>
      </element>
    </choice>
  </complexType>

```



```

        </key>
      </element>
      <element name="MetaCommandRef" type="xtce:NameReferenceType">
        <annotation>
          <documentation>Used to include a MetaCommand defined in another sub-system in this sub-system.</
documentation>
        </annotation>
      </element>
      <element name="BlockMetaCommand">
        <annotation>
          <documentation>BlockMetaCommands are simply a list of individual MetaCommands that can be packaged
up in a single BlockMetaCommand.</documentation>
        </annotation>
        <complexType>
          <complexContent>
            <extension base="xtce:NameDescriptionType">
              <sequence>
                <element name="MetaCommandStepList">
                  <complexType>
                    <sequence>
                      <element name="MetaCommandStep" maxOccurs="unbounded">
                        <complexType>
                          <sequence>
                            <element name="ArgumentList" minOccurs="0">
                              <complexType>
                                <sequence>
                                  <element name="Argument" maxOccurs="unbounded">
                                    <complexType>
                                      <attribute name="name" type="string" use="required"/>
                                      <attribute name="value" type="string" use="required"/>
                                    </complexType>
                                  </element>
                                </sequence>
                              </complexType>
                            </element>
                          </sequence>
                        </complexType>
                      </element>
                    </sequence>
                  </complexType>
                </element>
                <attribute name="metaCommandRef" type="xtce:NameReferenceType"
use="required"/>
              </sequence>
            </complexType>
          </element>
        </sequence>
      </complexType>
    </element>
  </sequence>
</complexType>
</element>
</sequence>
</extension>
</complexContent>
</complexType>
</element>
</choice>
</complexType>
</element>
<element name="CommandContainerSet" type="xtce:CommandContainerSetType" minOccurs="0">
  <annotation>
    <documentation>The Command Container defines the construction of a Command.</documentation>
  </annotation>
</element>

```

```

        <element name="StreamSet" type="xtce:StreamSetType" minOccurs="0"/>
        <element name="AlgorithmSet" type="xtce:AlgorithmSetType" minOccurs="0"/>
    </sequence>
</complexType>
<complexType name="TelemetryMetaDataType" mixed="false">
    <annotation>
        <documentation>All the data about telemetry is contained in TelemetryMetaData</documentation>
    </annotation>
    <sequence>
        <element name="ParameterTypeSet" type="xtce:ParameterTypeSetType" minOccurs="0">
            <annotation>
                <documentation>A list of parameter types</documentation>
            </annotation>
        </element>
        <element name="ParameterSet" type="xtce:ParameterSetType" minOccurs="0">
            <annotation>
                <documentation>A list of Parameters for this Space System. </documentation>
            </annotation>
        </element>
        <element name="ContainerSet" type="xtce:ContainerSetType" minOccurs="0">
            <annotation>
                <documentation xml:lang="en">Holds the list of all potential container definitions for telemetry. Containers may parts of
packets or TDM, and then groups of the containers, and then an entire entity -- such as a packet. In order to maximize re-used for duplication,
the pieces may defined once here, and then assembled as needed into larger structures, also here.</documentation>
            </annotation>
            <key name="ContainerKey2">
                <selector xpath="Container"/>
                <field xpath="Id"/>
            </key>
        </element>
        <element name="MessageSet" minOccurs="0">
            <annotation>
                <documentation>Messages are an alternative method of uniquely identifying containers within a Service. A message
provides a test in the form of MatchCriteria to match to a container. A simple example might be: [When minorframeID=21, the message is the
21st minorframe container. The collection of messages to search thru will be bound by a Service.</documentation>
            </annotation>
            <complexType>
                <sequence>
                    <element name="Message" maxOccurs="unbounded">
                        <complexType>
                            <complexContent>
                                <extension base="xtce:NameDescriptionType">
                                    <sequence>
                                        <element name="MatchCriteria" type="xtce:MatchCriteriaType"/>
                                        <element name="ContainRef" type="xtce:ContainerRefType">
                                            <annotation>
                                                <documentation>The ContainerRef should point to ROOT container that will describe
an entire packet/minor frame or chunk of telemetry.</documentation>
                                            </annotation>
                                        </element>
                                    </sequence>
                                </extension>
                            </complexContent>
                        </complexType>
                    </element>
                </sequence>
            </complexType>
        </element>
    </sequence>

```

```

        <attribute name="name" type="string"/>
    </complexType>
</element>
<element name="StreamSet" type="xtce:StreamSetType" minOccurs="0"/>
<element name="AlgorithmSet" type="xtce:AlgorithmSetType" minOccurs="0"/>
</sequence>
</complexType>
<complexType name="ServiceRefType">
    <annotation>
        <documentation xml:lang="en">A reference to a Service</documentation>
    </annotation>
    <simpleContent>
        <extension base="xtce:NameReferenceType">
            <attribute name="serviceRef" type="xtce:NameReferenceType" use="required"/>
        </extension>
    </simpleContent>
</complexType>
<complexType name="AlgorithmSetType" mixed="false">
    <annotation>
        <documentation>An unordered collection of algorithms</documentation>
    </annotation>
    <choice maxOccurs="unbounded">
        <element name="CustomAlgorithm" type="xtce:InputOutputTriggerAlgorithmType"/>
        <element name="MathAlgorithm" type="xtce:MathAlgorithmType"/>
    </choice>
</complexType>
<!-- ***** End of Top Level SpaceSystem Schema -->
<!-- ***** -->
<!-- ***** Packaging Schema -->
<annotation>
    <documentation xml:lang="en">This schema defines the dictionary for containers, which in turn describe the physical composition of
data in a communication system</documentation>
</annotation>
<complexType name="ContainerType" abstract="true" mixed="false">
    <annotation>
        <documentation xml:lang="en">An abstract block of data; used as the base type for more specific container types</
documentation>
    </annotation>
    <complexContent mixed="false">
        <extension base="xtce:NameDescriptionType">
            <sequence>
                <annotation>
                    <documentation>RateInStream is used to: a) generate alarms when the Container is updated too frequently or too
infrequently, b) provide some 'guidelines' for generating forward link containers, c) provide some guidelines for spacecraft simulators to
generate telemetry containers. If necessary, these rates may be defined on a per stream basis.</documentation>
                    <appinfo>The software should check that any Stream names referenced in the RateInStreamSet actually exist.</
appinfo>
                </annotation>
                <element name="DefaultRateInStream" type="xtce:RateInStreamType" minOccurs="0"/>
                <element name="RateInStreamSet" minOccurs="0">
                    <complexType>
                        <sequence>
                            <element name="RateInStream" maxOccurs="unbounded">
                                <complexType>
                                    <complexContent>
                                        <extension base="xtce:RateInStreamType">

```

```

        <attribute name="streamRef" type="xtce:NameReferenceType" use="required"/>
      </extension>
    </complexContent>
  </complexType>
</element>
</sequence>
</complexType>
</element>
<element name="BinaryEncoding" type="xtce:BinaryDataEncodingType" minOccurs="0">
  <annotation>
    <documentation>May be used to indicate error detection and correction, change byte order, provide the size
    (when it can't be derived), or perform some custom processing.</documentation>
  </annotation>
</element>
</sequence>
</extension>
</complexContent>
</complexType>
<complexType name="SequenceContainerType">
  <annotation>
    <documentation>A list of raw parameters, parameter segments, stream segments, containers, or container segments. Sequence
    containers may inherit from other sequence containers; when they do, the sequence in the parent SequenceContainer is 'inherited' and if the
    location of entries in the child sequence is not specified, it is assumed to start where the parent sequence ended. Parent sequence containers
    may be marked as "abstract". The idle pattern is part of any unallocated space in the Container.</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:ContainerType">
      <sequence>
        <element name="EntryList" type="xtce:EntryListType"/>
        <element name="BaseContainer" minOccurs="0">
          <complexType>
            <sequence>
              <element name="RestrictionCriteria">
                <annotation>
                  <documentation>Given that this Container is the Base container type, RestrictionCriteria lists
                  conditions that must be true for this Container to be 'this' subContainer type. May be a simple Comparison List, a Boolean Expression, and/or
                  in a Graph of containers established by the NextContainer</documentation>
                </annotation>
              <complexType>
                <complexContent>
                  <extension base="xtce:MatchCriteriaType">
                    <choice>
                      <element name="NextContainer" type="xtce:ContainerRefType" minOccurs="0"/>
                    </choice>
                  </extension>
                </complexContent>
              </complexType>
            </element>
          </sequence>
          <attribute name="containerRef" type="xtce:NameReferenceType" use="required"/>
        </complexType>
      </element>
    </sequence>
    <attribute name="abstract" type="boolean"/>
    <attribute name="idlePattern" type="xtce:FixedIntegerValueType" default="0x0"/>
  </extension>

```

```

    </complexContent>
  </complexType>
  <complexType name="SequenceEntryType">
    <annotation>
      <documentation>An abstract type used by sequence containers. An entry contains a location in the container. The location may
be either fixed or dynamic, absolute (to the start or end of the enclosing container, or relative (to either the previous or subsequent entry).
Entries may also repeat.</documentation>
    </annotation>
    <sequence>
      <element name="LocationInContainerInBits" minOccurs="0">
        <annotation>
          <documentation>If no LocationInContainer value is given, the entry is assumed to begin immediately after the previous
entry.</documentation>
        </annotation>
        <complexType>
          <complexContent>
            <extension base="xtce:IntegerValueType">
              <attribute name="referenceLocation" default="previousEntry">
                <annotation>
                  <documentation>The location may be relative to the start of the container (containerStart), relative to
the end of the previous entry (previousEntry), relative to the end of the container (containerEnd), or relative to the entry that follows this one
(nextEntry). If going forward (containerStart and previousEntry) then, then the location refers to the start of the Entry. If going backwards
(containerEnd and nextEntry) then, the location refers to the end of the entry.</documentation>
                </annotation>
                <simpleType>
                  <restriction base="string">
                    <enumeration value="containerStart"/>
                    <enumeration value="containerEnd"/>
                    <enumeration value="previousEntry"/>
                    <enumeration value="nextEntry"/>
                  </restriction>
                </simpleType>
              </attribute>
            </extension>
          </complexContent>
        </complexType>
      </element>
      <element name="RepeatEntry" type="xtce:RepeatType" minOccurs="0">
        <annotation>
          <documentation xml:lang="en">May be used when this entry repeats itself in the sequence container. If not supplied, the
entry does not repeat.</documentation>
        </annotation>
      </element>
      <element name="IncludeCondition" type="xtce:MatchCriteriaType" minOccurs="0">
        <annotation>
          <documentation>This entry will only be included in the sequence when this condition is true. If no IncludeCondition is
given, then it will be included. A parameter that is not included will be treated as if it did not exist in the sequence at all.</documentation>
        </annotation>
      </element>
    </sequence>
  </complexType>
  <complexType name="ContainerRefType">
    <annotation>
      <documentation xml:lang="en">Holds a reference to a container</documentation>
    </annotation>
    <attribute name="containerRef" type="xtce:NameReferenceType" use="required">

```

```

        <annotation>
            <documentation xml:lang="en">name of container</documentation>
        </annotation>
    </attribute>
</complexType>
<complexType name="MessageRefType">
    <annotation>
        <documentation xml:lang="en">Holds a reference to a message</documentation>
    </annotation>
    <attribute name="messageRef" type="xtce:NameReferenceType" use="required">
        <annotation>
            <documentation xml:lang="en">name of container</documentation>
        </annotation>
    </attribute>
</complexType>
<complexType name="ServiceType">
    <annotation>
        <documentation xml:lang="en">Holds a set of services, logical groups of containers OR messages (not both).</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:NameDescriptionType">
            <sequence>
                <element name="MessageRefSet" minOccurs="0">
                    <complexType>
                        <sequence>
                            <element name="MessageRef" maxOccurs="unbounded"/>
                        </sequence>
                    </complexType>
                </element>
                <element name="ContainerRefSet">
                    <complexType>
                        <sequence>
                            <element name="ContainerRef" type="xtce:ContainerRefType" maxOccurs="unbounded"/>
                        </sequence>
                    </complexType>
                </element>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<complexType name="ContainerSetType">
    <annotation>
        <documentation>Unordered Set of Containers</documentation>
    </annotation>
    <choice maxOccurs="unbounded">
        <element name="SequenceContainer" type="xtce:SequenceContainerType">
            <annotation>
                <documentation>SequenceContainers define sequences of parameters or other containers. </documentation>
            </annotation>
        </element>
    </choice>
</complexType>
<complexType name="EntryListType" mixed="false">
    <annotation>
        <documentation>Contains an ordered list of Entries. Used in Sequence Container</documentation>
    </annotation>

```

```

<choice minOccurs="0" maxOccurs="unbounded">
  <element name="ParameterRefEntry" type="xtce:ParameterRefEntryType"/>
  <element name="ParameterSegmentRefEntry" type="xtce:ParameterSegmentRefEntryType"/>
  <element name="ContainerRefEntry" type="xtce:ContainerRefEntryType"/>
  <element name="ContainerSegmentRefEntry" type="xtce:ContainerSegmentRefEntryType"/>
  <element name="StreamSegmentEntry" type="xtce:StreamSegmentEntryType"/>
  <element name="IndirectParameterRefEntry" type="xtce:IndirectParameterRefEntryType"/>
  <element name="ArrayParameterRefEntry" type="xtce:ArrayParameterRefEntryType"/>
</choice>
</complexType>

```

```

<complexType name="ParameterRefEntryType">

```

```

  <annotation>

```

```

    <documentation>An entry that is a single Parameter</documentation>

```

```

  </annotation>

```

```

  <complexContent>

```

```

    <extension base="xtce:SequenceEntryType">

```

```

      <attribute name="parameterRef" type="xtce:NameReferenceType" use="required"/>

```

```

    </extension>

```

```

  </complexContent>

```

```

</complexType>

```

```

<complexType name="ParameterSegmentRefEntryType">

```

```

  <annotation>

```

<documentation>An entry that is only a portion of a parameter value indicating that the entire parameter value must be assembled from other parameter segments. It is assumed that parameter segments happen sequentially in time, that is the first part if a telemetry parameter first, however (and there's always a however), if this is not the case the order of this parameter segment may be supplied with the order attribute where the first segment order="0".</documentation>

```

  </annotation>

```

```

  <complexContent>

```

```

    <extension base="xtce:SequenceEntryType">

```

```

      <attribute name="parameterRef" type="xtce:NameReferenceType" use="required"/>

```

```

      <attribute name="order" type="positiveInteger"/>

```

```

      <attribute name="sizeInBits" type="positiveInteger" use="required"/>

```

```

    </extension>

```

```

  </complexContent>

```

```

</complexType>

```

```

<complexType name="ContainerRefEntryType">

```

```

  <annotation>

```

```

    <documentation>An entry that is simply a reference to another container.</documentation>

```

```

  </annotation>

```

```

  <complexContent>

```

```

    <extension base="xtce:SequenceEntryType">

```

```

      <attribute name="containerRef" type="xtce:NameReferenceType" use="required"/>

```

```

    </extension>

```

```

  </complexContent>

```

```

</complexType>

```

```

<complexType name="ContainerSegmentRefEntryType">

```

```

  <annotation>

```

<documentation>An entry that is only a portion of a parameter value indicating that the entire parameter value must be assembled from other parameter segments. It is assumed that parameter segments happen sequentially in time, that is the first part if a telemetry parameter first, however (and there's always a however), if this is not the case the order of this parameter segment may be supplied with the order attribute where the first segment order="0".</documentation>

```

  </annotation>

```

```

  <complexContent>

```

```

    <extension base="xtce:SequenceEntryType">

```

```

      <attribute name="containerRef" type="xtce:NameReferenceType" use="required"/>

```

```

      <attribute name="order" type="positiveInteger"/>

```

```

        <attribute name="sizeInBits" type="positiveInteger" use="required"/>
    </extension>
</complexContent>
</complexType>
<complexType name="StreamSegmentEntryType">
    <annotation>
        <documentation>An entry that is a portion of a stream (streams are by definition, assumed continuous) It is assumed that stream
segments happen sequentially in time, that is the first part if a steam first, however, if this is not the case the order of the stream segments may
be supplied with the order attribute where the first segment order="0".</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:SequenceEntryType">
            <attribute name="streamRef" type="xtce:NameReferenceType" use="required"/>
            <attribute name="order" type="positiveInteger"/>
            <attribute name="sizeInBits" type="positiveInteger" use="required"/>
        </extension>
    </complexContent>
</complexType>
<complexType name="IndirectParameterRefEntryType">
    <annotation>
        <documentation>An entry whose name is given by the value of a ParamameterInstance. This entry may be used to implement
dwell telemetry streams. The value of the parameter in ParameterInstance must use either the name of the Parameter or its alias. If it's an
alias name, the alias namespace is supplied as an attribute.</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:SequenceEntryType">
            <sequence>
                <element name="ParameterInstance" type="xtce:ParameterInstanceRefType"/>
            </sequence>
            <attribute name="aliasNameSpace" type="string"/>
        </extension>
    </complexContent>
</complexType>
<complexType name="ArrayParameterRefEntryType">
    <annotation>
        <documentation>An entry that is an array parameter. This entry is somewhat special because the entry may represent only a part
of the Array and it's important to decrbe which diminsions of the array come first in the sequence as well as the size of the array.</
documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:SequenceEntryType">
            <sequence>
                <element name="DimensionList">
                    <annotation>
                        <documentation>Where the Dimension list is in this form: Array[1stDim][2ndDim][lastDim]. The last dimension is
assumed to be the least significant - that is this dimension will cycle through it's combination before the next to last dimension changes. The
order MUST ascend or the array will need to be broken out entry by entry. </documentation>
                    </annotation>
                    <complexType>
                        <sequence>
                            <element name="Dimension" maxOccurs="unbounded">
                                <annotation>
                                    <documentation>For partial entries of an array, the starting and ending index for each dimension,
OR the Size must be specified. Indexes are zero based.</documentation>
                                    <appinfo>For an ArrayParameterType of size N, their should be N Dimensions</appinfo>
                                </annotation>
                            </element>
                        </sequence>
                    </complexType>
                </element>
            </sequence>
        </extension>
    </complexContent>
</complexType>

```


continuous.</appinfo>

```

        <appinfo>An array made up by multiple Entries should not have index's that overlap, but should be
    </appinfo>
    </annotation>
    <complexType mixed="false">
        <sequence>
            <element name="StartingIndex" type="xtce:IntegerValueType">
                <annotation>
                    <documentation>zero based index</documentation>
                </annotation>
            </element>
            <element name="EndingIndex" type="xtce:IntegerValueType"/>
        </sequence>
    </complexType>
</element>
</sequence>
</complexType>
</element>
</sequence>
<attribute name="parameterRef" type="xtce:NameReferenceType" use="required"/>
<attribute name="lastEntryForThisArrayInstance" type="boolean" default="false"/>
</extension>
</complexContent>
</complexType>
<complexType name="RateInStreamType">
    <annotation>
        <documentation>Used in packaging to define the expected rate that any individual container will be in a Stream</documentation>
    </annotation>
    <attribute name="basis" default="perSecond">
        <simpleType>
            <restriction base="string">
                <enumeration value="perSecond"/>
                <enumeration value="perContainerUpdate"/>
            </restriction>
        </simpleType>
    </attribute>
    <attribute name="minimumValue" type="double"/>
    <attribute name="maximumValue" type="double"/>
</complexType>
<!-- ***** End of Packaging Schema -->
<!-- ***** Telemetry Schema -->
<complexType name="ParameterPropertiesType" mixed="false">
    <annotation>
        <documentation>A wrapper for those properties that are unique to telemetry parameters.</documentation>
    </annotation>
    <sequence>
        <element name="SystemName" type="string" minOccurs="0">
            <annotation>
                <documentation>Optional. Normally used when the database is built in a flat, non-hierarchical format</documentation>
            </annotation>
        </element>
        <element name="ValidityCondition" type="xtce:MatchCriteriaType" minOccurs="0">
            <annotation>
                <documentation>Optional condition that must be true for this Parameter to be valid</documentation>
            </annotation>
        </element>
    </sequence>
</complexType>

```

```

<element name="PhysicalAddressSet" minOccurs="0">
  <annotation>
    <documentation>One or more physical addresses may be associated with each Parameter. Examples of physical
addresses include a location on the spacecraft or a location on a data collection bus. </documentation>
  </annotation>
  <complexType>
    <sequence>
      <element name="PhysicalAddress" minOccurs="0" maxOccurs="unbounded">
        <annotation>
          <documentation xml:lang="en">Contains the address (e.g., channel information) required to process the
spacecraft telemetry streams. May be an onboard id, a mux address, or a physical location.</documentation>
          <documentation xml:lang="en">Contains the address (channel information) required to process the
spacecraft telemetry streams</documentation>
        </annotation>
        <complexType>
          <complexContent>
            <extension base="xtce:PhysicalAddressType"/>
          </complexContent>
        </complexType>
      </element>
    </sequence>
  </complexType>
</element>
<element name="TimeAssociation" type="xtce:TimeAssociationType" minOccurs="0">
  <annotation>
    <documentation>This time will override any Default value for TimeAssociation. </documentation>
  </annotation>
</element>
</sequence>
<attribute name="dataSource" use="optional">
  <annotation>
    <documentation>A telemetered Parameter is one that will have values in telemetry. A derived Parameter is one that is
calculated, usually be an Algorithm. A constant Parameter is one that is used as a constant in the system (e.g. a vehicle id). A local Parameter
is one that is used purely on the ground (e.g. a ground command counter).</documentation>
  </annotation>
  <simpleType>
    <restriction base="string">
      <enumeration value="telemetered"/>
      <enumeration value="derived"/>
      <enumeration value="constant"/>
      <enumeration value="local"/>
    </restriction>
  </simpleType>
</attribute>
<attribute name="readOnly" type="boolean" use="optional" default="false">
  <annotation>
    <documentation>A Parameter marked as 'readOnly' true is constant and non-settable</documentation>
  </annotation>
</attribute>
</complexType>
<complexType name="TimeAssociationType">
  <annotation>
    <documentation>Telemetry parameter instances are oftentimes "time-tagged" with a timing signal either provided on the ground or
on the space system. This data element allows one to specify which of possibly many AbsoluteTimeParameters to use to "time-tag" parameter
instances with. </documentation>
  </annotation>
  <appinfo>The parameter ref must be to an AbsoluteTime Parameter</appinfo>

```

```

</annotation>
<simpleContent>
  <extension base="xtce:ParameterInstanceRefType">
    <attribute name="interpolateTime" type="boolean" default="true">
      <annotation>
        <documentation xml:lang="en">If true, then the current value of the AbsoluteTime will be projected to current time.
I.E., if the value of the AbsoluteTime parameter was set 10 seconds ago, then 10 seconds will be added to it's value before associating this time
with the parameter.</documentation>
      </annotation>
    </attribute>
    <attribute name="offset" type="date">
      <annotation>
        <documentation>The offset is used to supply a relative time offset from the time association and to this parameter</
documentation>
      </annotation>
    </attribute>
  </extension>
</simpleContent>
</complexType>
<complexType name="ParameterInstanceRefType">
  <annotation>
    <documentation>A reference to an instance of a Parameter. Used when the value of a parameter is required for a calculation or
as an index value. A positive value for instance is forward in time, a negative value for count is backward in time, a 0 value for count means use
the current value of the parameter or the first value in a container.</documentation>
  </annotation>
  <simpleContent>
    <extension base="xtce:ParameterRefType">
      <attribute name="instance" type="integer" default="0"/>
      <attribute name="useCalibratedValue" type="boolean" default="true"/>
    </extension>
  </simpleContent>
</complexType>
<complexType name="ParameterRefType">
  <annotation>
    <documentation xml:lang="en">A reference to a Parameter. Uses Unix 'like' naming across the SpaceSystem Tree (e.g.,
SimpleSat/Bus/EPDS/BatteryOne/Voltage). To reference an individual member of an array use the zero based bracket notation commonly
used in languages like C, C++, and Java.</documentation>
  </annotation>
  <simpleContent>
    <extension base="xtce:NameReferenceType">
      <attribute name="parameterRef" type="xtce:NameReferenceType" use="required"/>
    </extension>
  </simpleContent>
</complexType>
<complexType name="PhysicalAddressType" mixed="false">
  <annotation>
    <documentation>When it's important to know the physical address(s) on the spacecraft that this parameter may be collected from,
use this. </documentation>
  </annotation>
  <sequence>
    <element name="SubAddress" type="xtce:PhysicalAddressType" minOccurs="0"/>
  </sequence>
  <attribute name="sourceName" type="string"/>
  <attribute name="sourceAddress" type="string"/>
</complexType>
<complexType name="ParameterTypeSetType">

```

```

<annotation>
  <documentation xml:lang="en">Holds the list of parameter definitions. A Parameter is a description of something that can have a
value; it is not the value itself. </documentation>
</annotation>
<choice maxOccurs="unbounded">
  <element name="StringParameterType" type="xtce:StringDataType"/>
  <element name="EnumeratedParameterType" type="xtce:EnumeratedDataType"/>
  <element name="IntegerParameterType">
    <complexType>
      <complexContent>
        <extension base="xtce:IntegerDataType">
          <sequence>
            <element name="DefaultAlarm" type="xtce:NumericAlarmConditionType" minOccurs="0"/>
            <element name="ContextAlarmList" minOccurs="0">
              <complexType>
                <sequence>
                  <element name="ContextAlarm" type="xtce:ContextAlarmType" maxOccurs="unbounded"/>
                </sequence>
              </complexType>
            </element>
          </sequence>
        </extension>
      </complexContent>
    </complexType>
  </element>
  <element name="BinaryParameterType" type="xtce:BinaryDataType"/>
  <element name="FloatParameterType">
    <complexType>
      <complexContent>
        <extension base="xtce:FloatDataType">
          <sequence>
            <element name="DefaultAlarm" type="xtce:NumericAlarmConditionType" minOccurs="0"/>
            <element name="ContextAlarmList" minOccurs="0">
              <complexType>
                <sequence>
                  <element name="ContextAlarm" type="xtce:ContextAlarmType" maxOccurs="unbounded"/>
                </sequence>
              </complexType>
            </element>
          </sequence>
        </extension>
      </complexContent>
    </complexType>
  </element>
  <element name="BooleanParameterType" type="xtce:BooleanDataType"/>
  <element name="RelativeTimeParameterType" type="xtce:RelativeTimeDataType"/>
  <element name="AbsoluteTimeParameterType" type="xtce:AbsoluteTimeDataType"/>
  <element name="ArrayParameterType">
    <annotation>
      <documentation>An array type. Will be an array of parameters of the type referenced in 'arrayTypeRef' and have the
number of array dimensions as specified in 'numberOfDimensions' </documentation>
    </annotation>
    <complexType>
      <complexContent>
        <extension base="xtce:NameDescriptionType">
          <attribute name="arrayTypeRef" type="xtce:NameReferenceType" use="required"/>
        </extension>
      </complexContent>
    </complexType>
  </element>

```

```

        <attribute name="numberOfDimensions" type="positiveInteger" use="required"/>
    </extension>
</complexContent>
</complexType>
</element>
</choice>
</complexType>
<!--***** End of Telemetry Schema -->
<!--***** -->
<!--***** Command Schema -->
<!--CommandDefinitionType -->
<complexType name="MetaCommandType" mixed="false">
    <annotation>
        <documentation xml:lang="en">A type definition used as the base type for a CommandDefinition</documentation>
    </annotation>
    <complexContent mixed="false">
        <extension base="xtce:NameDescriptionType">
            <sequence>
                <element name="BaseMetaCommand" minOccurs="0">
                    <annotation>
                        <documentation>The MetaCommand is derived from this Base. Arguments of the base MetaCommand are
further specified.</documentation>
                    </annotation>
                <complexType>
                    <sequence>
                        <element name="ArgumentAssignmentList" minOccurs="0">
                            <complexType>
                                <sequence>
                                    <element name="ArgumentAssignment" maxOccurs="unbounded">
                                        <complexType>
                                            <attribute name="argumentName" type="xtce:NameReferenceType" use="required"/>
                                            <attribute name="argumentValue" type="string" use="required"/>
                                        </complexType>
                                    </element>
                                </sequence>
                            </complexType>
                        </element>
                    </sequence>
                    <attribute name="metaCommandRef" type="xtce:NameReferenceType" use="required"/>
                </complexType>
            </element>
            <element name="SystemName" type="string" minOccurs="0">
                <annotation>
                    <documentation>Optional. Normally used when the database is built in a flat, non-hierarchical format</
documentation>
                </annotation>
            </element>
            <element name="ArgumentList" minOccurs="0">
                <annotation>
                    <documentation>Many commands have one or more options. These are called command arguments.
Command arguments may be of any of the standard data types. MetaCommand arguments are local to the MetaCommand.</documentation>
                </annotation>
                <complexType>
                    <choice maxOccurs="unbounded">
                        <element name="Argument" maxOccurs="unbounded">
                            <annotation>

```

```

        <appinfo>Need to ensure that the named types actually exist</appinfo>
    </annotation>
    <complexType>
        <complexContent>
            <extension base="xtce:NameDescriptionType">
                <attribute name="argumentTypeRef" type="xtce:NameReferenceType" use="required"/>
            </extension>
        </complexContent>
    </complexType>
</element>
</choice>
</complexType>
</element>
<element name="CommandContainer" type="xtce:CommandContainerType" minOccurs="0">
    <annotation>
        <documentation>Tells how to package this command</documentation>
    </annotation>
</element>
<element name="TransmissionConstraintList" minOccurs="0">
    <annotation>
        <documentation>Appended to the TransmissionConstraint List of the base command. Constraints are checked
in order. </documentation>
    </annotation>
    <complexType>
        <sequence>
            <element name="TransmissionConstraint" maxOccurs="unbounded">
                <annotation>
                    <documentation>A CommandTransmission constraint is used to check that the command can be run
in the current operating mode and may block the transmission of the command if the constraint condition is true.</documentation>
                </annotation>
                <complexType>
                    <complexContent>
                        <extension base="xtce:MatchCriteriaType">
                            <attribute name="timeOut" type="xtce:RelativeTimeType">
                                <annotation>
                                    <documentation>Pause during timeOut, fail when the timeout passes</
documentation>
                                </annotation>
                            </attribute>
                            <!-- removed for CASTOR: default="PT0S" -->
                        </extension>
                    </complexContent>
                </complexType>
                <annotation>
                    <documentation>Indicates whether the constraints for a Command may be
suspended.</documentation>
                </annotation>
            </element>
        </sequence>
    </complexType>
</element>
<element name="DefaultSignificance" type="xtce:SignificanceType" minOccurs="0">
    <annotation>

```

```

        <documentation>Some Command and Control Systems may require special user access our confirmations
before transmitting commands with certain levels. Will inherit any level defined in the Base MetaCommand.</documentation>
    </annotation>
</element>
<element name="ContextSignificanceList" minOccurs="0">
    <annotation>
        <documentation>Used when the significance of a command varies by the operating context</documentation>
    </annotation>
    <complexType>
        <sequence>
            <element name="ContextSignificance" maxOccurs="unbounded">
                <complexType>
                    <sequence>
                        <element name="ContextMatch" type="xtce:MatchCriteriaType"/>
                        <element name="Significance" type="xtce:SignificanceType"/>
                    </sequence>
                </complexType>
            </element>
        </sequence>
    </complexType>
</element>
<element name="Interlock" minOccurs="0">
    <annotation>
        <documentation>An Interlock is a type of Constraint, but not on Command instances of this MetaCommand;
Interlocks apply instead to the next command. An Interlock will block successive commands until this command has reached a certain stage
(through verifications). Interlocks are scoped to a SpaceSystem basis.</documentation>
    </annotation>
    <complexType>
        <attribute name="scopeToSpaceSystem" type="xtce:NameReferenceType">
            <annotation>
                <documentation>The name of a SpaceSystem this Interlock applies to. By default, it only applies to the
SpaceSystem that contains this MetaCommand.</documentation>
            </annotation>
        </attribute>
        <attribute name="verificationToWaitFor" default="complete">
            <simpleType>
                <restriction base="string">
                    <enumeration value="transferredToRange"/>
                    <enumeration value="sentFromRange"/>
                    <enumeration value="received"/>
                    <enumeration value="accepted"/>
                    <enumeration value="queued"/>
                    <enumeration value="executing"/>
                    <enumeration value="complete"/>
                </restriction>
            </simpleType>
        </attribute>
        <attribute name="verificationProgressPercentage" type="decimal">
            <annotation>
                <documentation>Only applies when the verificationToWaitFor attribute is 'queued' or 'executing'.</
documentation>
            </annotation>
        </attribute>
        <attribute name="suspendable" type="boolean" default="false">
            <annotation>

```

```

        <documentation>A flag that indicates that under special circumstances, this Interlock can be
suspended.</documentation>
    </annotation>
</attribute>
</complexType>
</element>
<element name="Verifiers" minOccurs="0">
    <annotation>
        <documentation>A Command Verifier is a conditional check on the telemetry from a SpaceSystem that provides
positive indication on the successful execution of a command. Completed verifiers are added to the Base MetaCommand verifiers. All others
will replace a verifier defined in a Base MetaCommand.</documentation>
    </annotation>
    <complexType>
        <sequence>
            <element name="TransferredToRangeVerifier" minOccurs="0">
                <annotation>
                    <documentation>Transferred to range means the command has been received by a the network that
connects the ground system to the spacecraft. Obviously, this verifier must come from something other than the spacecraft. </documentation>
                </annotation>
                <complexType>
                    <complexContent>
                        <extension base="xtce:CommandVerifierType"/>
                    </complexContent>
                </complexType>
            </element>
            <element name="SentFromRangeVerifier" minOccurs="0">
                <annotation>
                    <documentation>Sent from range means the command has been transmitted to the spacecraft by a
the network that connects the ground system to the spacecraft. Obviously, this verifier must come from something other than the spacecraft. </
documentation>
                </annotation>
                <complexType>
                    <complexContent>
                        <extension base="xtce:CommandVerifierType"/>
                    </complexContent>
                </complexType>
            </element>
            <element name="ReceivedVerifier" minOccurs="0">
                <annotation>
                    <documentation>A verifier that simply means the SpaceSystem has received the command.</
documentation>
                </annotation>
                <complexType>
                    <complexContent>
                        <extension base="xtce:CommandVerifierType"/>
                    </complexContent>
                </complexType>
            </element>
            <element name="AcceptedVerifier" minOccurs="0">
                <annotation>
                    <documentation>A verifier that means the SpaceSystem has accepted the command</
documentation>
                </annotation>
                <complexType>
                    <complexContent>
                        <extension base="xtce:CommandVerifierType"/>
                    </complexContent>
                </complexType>
            </element>
        </sequence>
    </complexType>
</element>

```



```

        </complexContent>
      </complexType>
    </element>
    <element name="QueuedVerifier" minOccurs="0">
      <annotation>
        <documentation>A verifier that means the command is scheduled for execution by the
SpaceSystem.</documentation>
      </annotation>
      <complexType>
        <complexContent>
          <extension base="xtce:CommandVerifierType"/>
        </complexContent>
      </complexType>
    </element>
    <element name="ExecutionVerifier" minOccurs="0">
      <annotation>
        <documentation>A verifier that indicates that the command is being executed. An optional Element
indicates how far along the command has progressed either as a fixed value or an (possibly scaled) ParameterInstance value.</
documentation>
      </annotation>
      <complexType>
        <complexContent>
          <extension base="xtce:CommandVerifierType">
            <sequence minOccurs="0">
              <element name="PercentComplete" type="xtce:DecimalValueType"/>
            </sequence>
          </extension>
        </complexContent>
      </complexType>
    </element>
    <element name="CompleteVerifier" minOccurs="0" maxOccurs="unbounded">
      <annotation>
        <documentation>A possible set of verifiers that all must be true for the command be considered
completed. </documentation>
      </annotation>
      <complexType>
        <complexContent>
          <extension base="xtce:CommandVerifierType">
            <sequence minOccurs="0">
              <element name="ReturnParmRef" type="xtce:ParameterRefType"/>
            </sequence>
          </extension>
        </complexContent>
      </complexType>
    </element>
    <element name="FailedVerifier" type="xtce:CommandVerifierType" minOccurs="0">
      <annotation>
        <documentation>When true, indicates that the command failed. timeToWait is how long to wait for
the FailedVerifier to test true.</documentation>
      </annotation>
    </element>
  </sequence>
</complexType>
</element>
<element name="ParameterToSetList" minOccurs="0">
  <annotation>

```

```

        <documentation>Parameters that are set with a new value after the command has been sent. Appended to the
Base Command list</documentation>
    </annotation>
    <complexType>
        <sequence>
            <element name="ParameterToSet" maxOccurs="unbounded">
                <annotation>
                    <documentation>Sets a Parameter to a new value after the command has been verified (all
verifications have passed)</documentation>
                </annotation>
                <complexType>
                    <complexContent>
                        <extension base="xtce:ParameterToSetType">
                            <attribute name="verifierToTriggerOn" default="release">
                                <simpleType>
                                    <restriction base="string">
                                        <enumeration value="release"/>
                                        <enumeration value="transferredToRange"/>
                                        <enumeration value="sentFromRange"/>
                                        <enumeration value="received"/>
                                        <enumeration value="accepted"/>
                                        <enumeration value="queued"/>
                                        <enumeration value="executing"/>
                                        <enumeration value="complete"/>
                                        <enumeration value="failed"/>
                                    </restriction>
                                </simpleType>
                            </attribute>
                        </extension>
                    </complexContent>
                </complexType>
            </element>
        </sequence>
    </complexType>
</element>
</sequence>
<attribute name="abstract" type="boolean" default="false"/>
</extension>
</complexContent>
</complexType>
<complexType name="CommandContainerEntryListType" mixed="false">
    <annotation>
        <documentation>Similar to an EntryList type but also may include command arguments or -as a convenience - fixed value
entries.</documentation>
    </annotation>
    <choice minOccurs="0" maxOccurs="unbounded">
        <element name="ParameterRefEntry" type="xtce:ParameterRefEntryType"/>
        <element name="ParameterSegmentRefEntry" type="xtce:ParameterSegmentRefEntryType"/>
        <element name="ContainerRefEntry" type="xtce:ContainerRefEntryType"/>
        <element name="ContainerSegmentRefEntry" type="xtce:ContainerSegmentRefEntryType"/>
        <element name="StreamSegmentEntry" type="xtce:StreamSegmentEntryType"/>
        <element name="IndirectParameterRefEntry" type="xtce:IndirectParameterRefEntryType"/>
        <element name="ArrayParameterRefEntry" type="xtce:ArrayParameterRefEntryType"/>
        <element name="ArgumentRefEntry">
            <complexType>
                <complexContent>

```

```

        <extension base="xtce:SequenceEntryType">
            <attribute name="argumentRef" type="xtce:NameReferenceType" use="required"/>
        </extension>
    </complexContent>
</complexType>
</element>
<element name="ArrayArgumentRefEntry" type="xtce:ArrayParameterRefEntryType"/>
<element name="FixedValueEntry">
    <complexType>
        <complexContent>
            <extension base="xtce:SequenceEntryType">
                <attribute name="binaryValue" type="hexBinary" use="required"/>
                <attribute name="sizeInBits" type="integer"/>
            </extension>
        </complexContent>
    </complexType>
</element>
</choice>
</complexType>
<complexType name="CommandContainerType" mixed="false">
    <annotation>
        <documentation>The Key = Command Op Code. Each MetaCommand may have one CommandContainer. The sequence may
now contain command fields</documentation>
    </annotation>
    <complexContent mixed="false">
        <extension base="xtce:ContainerType">
            <sequence>
                <element name="EntryList" type="xtce:CommandContainerEntryListType"/>
                <element name="BaseContainer" minOccurs="0">
                    <complexType>
                        <sequence>
                            <element name="RestrictionCriteria" minOccurs="0">
                                <annotation>
                                    <documentation>Given that this Container is the Base container type, RestrictionCriteria lists
conditions that must be true for this Container to be 'this' subContainer type. May be a simple Comparison List, a Boolean Expression, and/or
in a Graph of containers established by the NextContainer</documentation>
                                </annotation>
                            </complexType>
                        <complexContent>
                            <extension base="xtce:MatchCriteriaType">
                                <choice>
                                    <element name="NextContainer" type="xtce:ContainerRefType" minOccurs="0"/>
                                </choice>
                            </extension>
                        </complexContent>
                    </complexType>
                </element>
            </sequence>
            <attribute name="containerRef" type="xtce:NameReferenceType" use="required"/>
        </complexType>
    </element>
</sequence>
</extension>
</complexContent>
</complexType>
<complexType name="CommandVerifierType">

```

```

<annotation>
  <documentation>A command verifier is used to check that the command has be successfully executed. Command Verifiers may
  be either a Custom Algorithm or a Boolean Check or the presence of a Container for a relative change in the value of a Parameter. The
  timeToWait is a time period where the verification must test true.</documentation>
</annotation>
<choice>
  <element name="Comparison" type="xtce:ComparisonType"/>
  <element name="ComparisonList">
    <annotation>
      <documentation>All comparisons must be true</documentation>
    </annotation>
    <complexType>
      <sequence>
        <element name="Comparison" type="xtce:ComparisonType" maxOccurs="unbounded"/>
      </sequence>
    </complexType>
  </element>
  <element name="BooleanExpression" type="xtce:BooleanExpressionType"/>
  <element name="ContainerRef" type="xtce:ContainerRefType">
    <annotation>
      <documentation>When verification is the existence of a Container</documentation>
    </annotation>
  </element>
  <element name="ParameterValueChange">
    <annotation>
      <documentation>Used to look for relative change in a Parameter value. Only usefull for numeric Parameters</
documentation>
    </annotation>
    <complexType>
      <sequence>
        <element name="ParameterRef" type="xtce:ParameterRefType"/>
        <element name="Change">
          <complexType>
            <attribute name="value" type="decimal" use="required"/>
          </complexType>
        </element>
      </sequence>
    </complexType>
  </element>
  <element name="CustomAlgorithm" type="xtce:InputAlgorithmType"/>
</choice>
<attribute name="timeToWait" type="xtce:RelativeTimeType" use="required">
  <annotation>
    <documentation>Specifies how much time to provide for the verification. </documentation>
  </annotation>
</attribute>
</complexType>
<complexType name="ParameterToSetType">
  <annotation>
    <documentation>Used by Meta Command to indicate ground Parameters that should be set after completion of a command. </
documentation>
  </annotation>
  <sequence>
    <element name="ParameterRef" type="xtce:ParameterRefType"/>
    <element name="Derivation" type="xtce:MathOperationType"/>
  </sequence>

```

```

</complexType>
<complexType name="CommandContainerSetType">
  <annotation>
    <documentation>Contains an unordered Set of Command Containers</documentation>
  </annotation>
  <sequence>
    <element name="CommandContainer" type="xtce:SequenceContainerType" maxOccurs="unbounded"/>
  </sequence>
</complexType>
<complexType name="SignificanceType" mixed="false">
  <annotation>
    <documentation>Significance provides some cautionary information about the potential consequence of each MetaCommand.</
documentation>
  </annotation>
  <attribute name="spaceSystemAtRisk" type="xtce:NameReferenceType">
    <annotation>
      <documentation>If none is supplied, then the current SpaceSystem is assumed to be the one at risk by the issuance of this
command</documentation>
    </annotation>
  </attribute>
  <attribute name="reasonForWarning" type="string"/>
  <attribute name="consequenceLevel">
    <annotation>
      <documentation>No specific meanings have been assigned to these different levels, but they mirror the Alarm levels of
Telemetry.</documentation>
    </annotation>
  </attribute>
  <simpleType>
    <restriction base="string">
      <enumeration value="none"/>
      <enumeration value="watch"/>
      <enumeration value="warning"/>
      <enumeration value="distress"/>
      <enumeration value="critical"/>
      <enumeration value="severe"/>
    </restriction>
  </simpleType>
</attribute>
</complexType>
<!-- ***** End of Command Definition Schema -->
<!-- ***** Algorithm Schema -->
<annotation>
  <documentation xml:lang="en">This schema defines the structure for an Algorithm. An Algorithm may be one of a growing set of pre-
defined algorithms or a named escape into a user defined algorithm where (depending on the system) the name of the algorithm may be a java
class, a function in a shared library, an external program or some other reference to an outside algorithm. At some later date, this schema may
also allow the logic of the user defined algorithm to be defined within the instance document itself (perhaps using MathML?).</documentation>
</annotation>
<complexType name="SimpleAlgorithmType">
  <annotation>
    <documentation xml:lang="en">The simplest form of algorithm, a SimpleAlgorithmType contains an area for a free-form pseudo
code description of the algorithm plus a Set of references to external algorithms. External algorithms are usually unique to a ground system
type. Multiple external algorithms are possible because XTCE documents may be used across multiple ground systems.</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:NameDescriptionType">
      <sequence>

```

```

    <element name="AlgorithmText" minOccurs="0">
      <annotation>
        <documentation xml:lang="en">This optional element may be used to enter Pseudo or actual code for the
algorithm. The language for the algorithm is specified with the language attribute</documentation>
      </annotation>
    </element>
    <complexType>
      <simpleContent>
        <extension base="string">
          <attribute name="language" type="string" default="pseudo"/>
        </extension>
      </simpleContent>
    </complexType>
  </element>
  <element name="ExternalAlgorithmSet" minOccurs="0">
    <complexType>
      <sequence>
        <element name="ExternalAlgorithm" maxOccurs="unbounded">
          <annotation>
            <documentation>This is the external algorithm. Multiple entries are provided so that the same
database may be used for multiple implementation s</documentation>
          </annotation>
        </element>
      </sequence>
    </complexType>
  </element>
  <extension base="xtce:SimpleAlgorithmType">
    <complexType name="InputAlgorithmType">
      <annotation>
        <documentation>A set of labeled inputs is added to the SimpleAlgorithmType</documentation>
      </annotation>
    </complexType>
  </extension>
  <complexType name="InputSet" minOccurs="0">
    <sequence>
      <element name="InputParm" maxOccurs="unbounded">
        <complexType>
          <choice maxOccurs="unbounded">
            <element name="ParameterInstanceRef">
              <annotation>
                <documentation>Names an input parameter to the algorithm. There are two attributes to InputParm,
inputName and parameterName. parameterName is a parameter reference name for a parameter that will be used in this algorithm. inputName
is an optional "friendly" name for the input parameter. </documentation>
              </annotation>
            </element>
          </choice>
        </complexType>
      </element>
    </sequence>
  </complexType>

```

```

        </element>
        <element name="Constant" minOccurs="0">
          <annotation>
            <documentation xml:lang="en">Names and provides a value for a constant input to the algorithm.

```

There are two attributes to Constant, constantName and value. constantName is a variable name in the algorithm to be executed. value is the value of the constant to be used.</documentation>

```

          </annotation>
          <complexType>
            <attribute name="constantName" type="string"/>
            <attribute name="value" type="string" use="required"/>
          </complexType>
        </element>
      </choice>
    </complexType>
  </element>
</sequence>
</extension>
</complexContent>
</complexType>
<complexType name="InputOutputAlgorithmType">
  <annotation>
    <documentation>A set of labeled outputs are added to the SimpleInputAlgorithmType</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:InputAlgorithmType">
      <sequence>
        <element name="OutputSet" minOccurs="0">
          <complexType>
            <sequence>
              <element name="OutputParameterRef" maxOccurs="unbounded">
                <annotation>
                  <documentation>Names an output parameter to the algorithm. There are two attributes to

```

OutputParm, outputName and parameterName. parameterName is a parameter reference name for a parameter that will be updated by this algorithm. outputName is an optional "friendly" name for the output parameter.</documentation>

```

                </annotation>
                <complexType>
                  <simpleContent>
                    <extension base="xtce:ParameterRefType">
                      <attribute name="outputName" type="string"/>
                    </extension>
                  </simpleContent>
                </complexType>
              </element>
            </sequence>
          </complexType>
        </element>
      </sequence>
      <attribute name="thread" type="boolean" use="optional"/>
    </extension>
  </complexContent>
</complexType>
<complexType name="InputOutputTriggerAlgorithmType">
  <annotation>
    <documentation>A set of labeled triggers is added to the SimpleInputOutputAlgorithmType</documentation>
  </annotation>
  <complexContent>

```

```

<extension base="xtce:InputOutputAlgorithmType">
  <sequence>
    <element name="TriggerSet" type="xtce:TriggerType" minOccurs="0"/>
  </sequence>
  <attribute name="triggerContainer" type="xtce:NameType" use="optional">
    <annotation>
      <documentation xml:lang="en">First telemetry container from which the output parameter should be calculated.</
documentation>
    </annotation>
  </attribute>
  <attribute name="priority" type="integer" use="optional">
    <annotation>
      <documentation xml:lang="en">Algorithm processing priority.</documentation>
    </annotation>
  </attribute>
</extension>
</complexContent>
</complexType>
<complexType name="CalibratorType">
  <annotation>
    <documentation xml:lang="en">Calibrators are normally used to convert to and from bit compacted numerical data</
documentation>
  </annotation>
  <choice>
    <element name="SplineCalibrator">
      <annotation>
        <documentation xml:lang="en">A calibration type where a segmented line in a raw vs calibrated plane is described using
a set of points. Raw values are converted to calibrated values by finding a position on the line coorresponding to the raw value. The algorithm
triggers on the input parameter.</documentation>
      </annotation>
      <complexType>
        <sequence>
          <element name="SplinePoint" type="xtce:SplinePointType" minOccurs="2" maxOccurs="unbounded"/>
        </sequence>
        <attribute name="order" type="positiveInteger" default="1"/>
        <attribute name="extrapolate" type="boolean" default="false"/>
      </complexType>
    </element>
    <element name="PolynomialCalibrator" type="xtce:PolynomialType">
      <annotation>
        <documentation>A calibration type where a curve in a raw vs calibrated plane is described using a set of polynomial
coefficients. Raw values are converted to calibrated values by finding a position on the curve corresponding to the raw value. The first
coefficient belongs with the X^0 term, the next coefficient belongs to the X^1 term and so on. </documentation>
      </annotation>
    </element>
  </choice>
  <attribute name="name" type="string" use="optional"/>
</complexType>
<complexType name="MathAlgorithmType">
  <annotation>
    <documentation>A simple mathematical operation</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:NameDescriptionType">
      <sequence>
        <element name="MathOperation">

```



```

    <complexType>
      <complexContent>
        <extension base="xtce:MathOperationType">
          <sequence>
            <element name="TriggerSet" type="xtce:TriggerType"/>
          </sequence>
          <attribute name="outputParameterRef" type="xtce:NameReferenceType" use="required"/>
        </extension>
      </complexContent>
    </complexType>
  </element>
</sequence>
</extension>
</complexContent>
</complexType>
<complexType name="TriggerType">
  <annotation>
    <documentation>A trigger is used to initiate the processing of some algorithm. A trigger may be based on an update of a
Parameter or on a time basis. Triggers may also have a rate that limits their firing to a 1/rate basis.</documentation>
  </annotation>
  <choice maxOccurs="unbounded">
    <element name="ParameterRef" minOccurs="0">
      <annotation>
        <documentation>Names a parameter that upon change will start the execution of the algorithm. Holds a parameter
reference name for a parameter that when it changes, will cause this algorithm to be executed.</documentation>
      </annotation>
      <complexType>
        <simpleContent>
          <extension base="xtce:ParameterRefType">
            <attribute name="triggerName" type="string"/>
          </extension>
        </simpleContent>
      </complexType>
    </element>
    <element name="TriggerFrequency" type="xtce:RelativeTimeType" minOccurs="0"/>
  </choice>
  <attribute name="name" type="string" use="optional"/>
  <attribute name="triggerRate" type="nonNegativeInteger" use="optional" default="1"/>
</complexType>
<!--***** End of Algorithm Schema -->
<!--***** Stream Definitions Schema -->
<annotation>
  <documentation xml:lang="en">This schema provides a language for defining binary stream data.</documentation>
</annotation>
<complexType name="FrameStreamType">
  <annotation>
    <documentation xml:lang="en">The top level type definition for all data streams that are frame based.</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:PCMStreamType">
      <sequence>
        <choice>
          <element name="ContainerRef" type="xtce:ContainerRefType">
            <annotation>
              <documentation>This Container (usually abstract) is the container that is in the fixed frame stream.
Normally, this is an generalcontainer type from which many specific containers are inherited.</documentation>
            </annotation>
          </element>
        </choice>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

```

        </annotation>
      </element>
      <element name="ServiceRef" type="xtce:ServiceRefType"/>
    </choice>
    <element name="StreamRef" type="xtce:StreamRefType" minOccurs="0">
      <annotation>
        <documentation>This is a reference to a connecting stream - say a custom stream.</documentation>
      </annotation>
    </element>
  </sequence>
</extension>
</complexContent>
</complexType>
<complexType name="FixedFrameStreamType">
  <annotation>
    <documentation xml:lang="en">For streams that contain a series of frames with a fixed frame length where the frames are found
    by looking for a marker in the data. This marker is sometimes called the frame sync pattern and sometimes the Asynchronous Sync Marker
    (ASM).</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:FrameStreamType">
      <sequence>
        <element name="SyncStrategy">
          <complexType>
            <complexContent>
              <extension base="xtce:SyncStrategyType">
                <sequence>
                  <element name="SyncPattern">
                    <annotation>
                      <documentation xml:lang="en">The pattern of bits used to look for frame synchronization.</
documentation>
                    </annotation>
                    <complexType>
                      <attribute name="pattern" type="hexBinary" use="required">
                        <annotation>
                          <documentation>CCSDS ASM for non-turbocoded frames = 1acffc1d</
documentation>
                        </annotation>
                      </attribute>
                      <attribute name="bitLocationFromStartOfContainer" type="integer" default="0"/>
                      <attribute name="mask" type="hexBinary"/>
                      <attribute name="maskLengthInBits" type="positiveInteger">
                        <annotation>
                          <documentation>truncate the mask from the left</documentation>
                        </annotation>
                      </attribute>
                      <attribute name="patternLengthInBits" type="positiveInteger" use="required">
                        <annotation>
                          <documentation>truncate the pattern from the left</documentation>
                        </annotation>
                      </attribute>
                    </complexType>
                  </element>
                </sequence>
              </extension>
            </complexContent>
          </complexType>
        </element>
      </sequence>
    </extension>
  </complexContent>

```

```

        </complexType>
      </element>
    </sequence>
    <attribute name="syncApertureInBits" type="nonNegativeInteger" default="0">
      <annotation>
        <documentation>Allowed slip (in bits) in either direction for the sync pattern</documentation>
      </annotation>
    </attribute>
    <attribute name="frameLengthInBits" type="long" use="required"/>
  </extension>
</complexContent>
</complexType>
<complexType name="VariableFrameStreamType">
  <annotation>
    <documentation xml:lang="en">For streams that contain a series of frames with a variable frame length where the frames are found by looking for a series of one's or zero's (usually one's). The series is called the flag. in the PCM stream that are usually made to be illegal in the PCM stream by zero or one bit insertion. </documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:FrameStreamType">
      <sequence>
        <element name="SyncStrategy">
          <complexType>
            <complexContent>
              <extension base="xtce:SyncStrategyType">
                <sequence>
                  <element name="Flag">
                    <annotation>
                      <documentation xml:lang="en">The pattern of bits used to look for frame synchronization.</documentation>
                    </annotation>
                  </element>
                </sequence>
              </extension>
            </complexContent>
          </complexType>
          <attribute name="flagSizeInBits" type="positiveInteger" default="6"/>
          <attribute name="flagBitType" default="ones">
            <simpleType>
              <restriction base="string">
                <enumeration value="zeros"/>
                <enumeration value="ones"/>
              </restriction>
            </simpleType>
          </attribute>
        </complexType>
      </element>
    </sequence>
  </extension>
</complexContent>
</complexType>
</element>
</sequence>
</extension>
</complexContent>
</complexType>
<complexType name="CustomStreamType">
  <annotation>
    <documentation xml:lang="en">A stream type where some level of custom processing (e.g. convolutional, encryption, compression) is performed. Has a reference to external algorithms for encoding and decoding algorithms.</documentation>
  </annotation>

```

```

    <appinfo>Must check to ensure that the attributes encodedStreamRef and decodedStreamRef point to valid Streams</appinfo>
</annotation>
<complexContent>
  <extension base="xtce:PCMStreamType">
    <sequence>
      <element name="EncodingAlgorithm" type="xtce:InputAlgorithmType"/>
      <element name="DecodingAlgorithm" type="xtce:InputOutputAlgorithmType">
        <annotation>
          <documentation>Algorithm outputs may be used to set decoding quality parameters.</documentation>
        </annotation>
      </element>
    </sequence>
    <attribute name="encodedStreamRef" type="xtce:NameReferenceType" use="required"/>
    <attribute name="decodedStreamRef" type="xtce:NameReferenceType" use="required"/>
  </extension>
</complexContent>
</complexType>
<complexType name="PCMStreamType" abstract="true">
  <annotation>
    <documentation xml:lang="en">A PCM Stream Type is the high level definition for all Pulse Code Modulated (PCM) (i.e., binary)
streams.</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:NameDescriptionType">
      <attribute name="bitRateInBPS" type="double"/>
      <attribute name="pcmType" default="NRZL">
        <simpleType>
          <restriction base="string">
            <enumeration value="NRZL"/>
            <enumeration value="NRZM"/>
            <enumeration value="NRZS"/>
            <enumeration value="BiPhaseL"/>
            <enumeration value="BiPhaseM"/>
            <enumeration value="BiPhaseS"/>
          </restriction>
        </simpleType>
      </attribute>
      <attribute name="inverted" type="boolean" default="false"/>
    </extension>
  </complexContent>
</complexType>
<complexType name="StreamRefType">
  <annotation>
    <documentation xml:lang="en">Holds a reference to a stream</documentation>
  </annotation>
  <attribute name="streamRef" type="xtce:NameReferenceType" use="required">
    <annotation>
      <documentation xml:lang="en">name of reference stream</documentation>
    </annotation>
  </attribute>
</complexType>
<complexType name="StreamSetType">
  <annotation>
    <documentation>Contains an unordered set of Streams.</documentation>
  </annotation>
  <choice maxOccurs="unbounded">

```

```

        <element name="FixedFrameStream" type="xtce:FixedFrameStreamType"/>
        <element name="VariableFrameStream" type="xtce:VariableFrameStreamType"/>
        <element name="CustomStream" type="xtce:CustomStreamType"/>
    </choice>
</complexType>
<complexType name="FixedFrameSyncStrategyType">
    <annotation>
        <documentation>A Sync Strategy specifies the requirements to deem a PCM Fixed Frame Stream "in-sync" or out of sync.</
documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:SyncStrategyType">
            <sequence>
                <element name="SyncPattern">
                    <annotation>
                        <documentation xml:lang="en">The pattern of bits used to look for frame synchronization.</documentation>
                    </annotation>
                    <complexType>
                        <attribute name="pattern" type="hexBinary" use="required"/>
                        <attribute name="bitLocation" type="integer" default="0"/>
                        <attribute name="mask" type="hexBinary"/>
                        <attribute name="maskLengthInBits" type="positiveInteger"/>
                        <attribute name="patternLengthInBits" type="positiveInteger" use="required"/>
                    </complexType>
                </element>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<complexType name="SyncStrategyType">
    <annotation>
        <documentation>A Sync Strategy specifies the strategy on how to find frames within a stream of PCM data. The sync strategy is
based upon a state machine that begins in the 'Search' state until the first sync marker is found. Then it goes into the 'Verify' state until a
specified number of successive good sync markers are found. Then, the state machine goes into the 'Lock' state, in the 'Lock' state frames are
considered good. Should a sync marker be missed in the 'Lock' state, the state machine will transition into the 'Check' state, if the next sync
marker is where it's expected within a specified number of frames, then the state machine will transition back to the 'Lock' state, if not it will
transition back to 'Search'. </documentation>
    </annotation>
    <sequence>
        <element name="AutoInvert" minOccurs="0">
            <annotation>
                <documentation xml:lang="en">After serching for the frame sync marker for some number of bitss, it may be desirable to
invert the incoming data, and then look for frame sync. In some cases this will require an external algorithm</documentation>
            </annotation>
            <complexType>
                <sequence>
                    <element name="InvertAlgorithm" type="xtce:InputAlgorithmType" minOccurs="0"/>
                </sequence>
                <attribute name="badFramesToAutoInvert" type="positiveInteger" default="1024"/>
            </complexType>
        </element>
    </sequence>
    <attribute name="verifyToLockGoodFrames" type="nonNegativeInteger" default="4"/>
    <attribute name="checkToLockGoodFrames" type="nonNegativeInteger" default="1"/>
    <attribute name="maxBitErrorsInSyncPattern" type="nonNegativeInteger" default="0">
    </annotation>

```

```

        <documentation>Maximum number of bit errors in the sync pattern (marker).</documentation>
    </annotation>
</attribute>
</complexType>
<!-- ***** End of Stream Definition Schema -->
<!-- ***** -->
<!-- ***** DataTypes-->
<complexType name="AbsoluteTimeDataType">
    <annotation>
        <documentation>Used to contain an absolute time. Contains an absolute (to a known epoch) time. Use the [ISO 8601] extended
format CCYY-MM-DDThh:mm:ss where "CC" represents the century, "YY" the year, "MM" the month and "DD" the day, preceded by an optional
leading "-" sign to indicate a negative number. If the sign is omitted, "+" is assumed. The letter "T" is the date/time separator and "hh", "mm",
"ss" represent hour, minute and second respectively. Additional digits can be used to increase the precision of fractional seconds if desired i.e
the format ss.ss... with any number of digits after the decimal point is supported.
</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:BaseTimeDataType"/>
    </complexContent>
</complexType>
<complexType name="BaseDataType" abstract="true">
    <annotation>
        <documentation>An abstract type used by within the schema to derive other data types by the ground system. </documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:NameDescriptionType">
            <sequence>
                <element name="UnitSet">
                    <complexType>
                        <sequence>
                            <element name="Unit" type="xtce:UnitType" minOccurs="0" maxOccurs="unbounded"/>
                        </sequence>
                    </complexType>
                </element>
                <choice minOccurs="0">
                    <element name="BinaryDataEncoding" type="xtce:BinaryDataEncodingType"/>
                    <element name="FloatDataEncoding" type="xtce:FloatDataEncodingType"/>
                    <element name="IntegerDataEncoding" type="xtce:IntegerDataEncodingType"/>
                    <element name="StringDataEncoding" type="xtce:StringDataEncodingType"/>
                </choice>
            </sequence>
        </extension>
    </complexContent>
</complexType>
<complexType name="BaseTimeDataType" abstract="true">
    <annotation>
        <documentation>An abstract type used by within the schema to describe derive other data types by the ground system. </
documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:NameDescriptionType">
            <sequence>
                <sequence minOccurs="0">
                    <element name="Encoding">
                        <annotation>

```

`<documentation>`Scale and offset are used in a $y = mx + b$ type relationship (m is the scale and b is the offset) to make adjustments to the encoded value so that it matches the time units. Binary Encoded time is typically used with a user supplied transform algorithm to convert time data formats that are too difficult to describe in XTCE.`</documentation>`

```

</annotation>
<complexType>
  <choice>
    <element name="BinaryDataEncoding" type="xtce:BinaryDataEncodingType"/>
    <element name="FloatDataEncoding" type="xtce:FloatDataEncodingType"/>
    <element name="IntegerDataEncoding" type="xtce:IntegerDataEncodingType"/>
    <element name="StringDataEncoding" type="xtce:StringDataEncodingType"/>
  </choice>
  <attribute name="units" default="seconds">
    <simpleType>
      <restriction base="string">
        <enumeration value="seconds"/>
        <enumeration value="years"/>
        <enumeration value="picoseconds"/>
        <enumeration value="days"/>
        <enumeration value="month"/>
      </restriction>
    </simpleType>
  </attribute>
  <attribute name="scale" type="double" default="1"/>
  <attribute name="offset" type="double" default="0"/>
</complexType>
</element>
</sequence>
<sequence minOccurs="0">
  <element name="ReferenceTime" type="xtce:ReferenceTimeType"/>
</sequence>
</extension>
</complexContent>
</complexType>
<complexType name="BinaryDataType">
  <annotation>
    <documentation>Contains an arbitrarily large binary value </documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:BaseDataType">
      <attribute name="initialValue" type="hexBinary"/>
    </extension>
  </complexContent>
</complexType>
<complexType name="BooleanDataType">
  <annotation>
    <documentation>Contains a boolean value</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:BaseDataType">
      <attribute name="initialValue" type="boolean"/>
      <attribute name="oneStringValue" type="string" default="True"/>
      <attribute name="zeroStringValue" type="string" default="False"/>
    </extension>
  </complexContent>
</complexType>

```

```

<complexType name="EnumeratedDataType">
  <annotation>
    <documentation>Contains an enumerated value - a value that has both an integral and a string representation.</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:BaseDataType">
      <sequence>
        <element name="EnumerationList">
          <complexType>
            <sequence>
              <element name="Enumeration" type="xtce:ValueEnumerationType" maxOccurs="unbounded"/>
            </sequence>
          </complexType>
        </element>
      </sequence>
      <attribute name="initialValue" type="string"/>
    </extension>
  </complexContent>
</complexType>
<complexType name="FloatDataType">
  <annotation>
    <documentation>Contains a floating point value</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:NumericDataType">
      <attribute name="initialValue" type="decimal"/>
      <attribute name="sizeInBits" default="32">
        <simpleType>
          <restriction base="positiveInteger">
            <enumeration value="32"/>
            <enumeration value="64"/>
            <enumeration value="128"/>
          </restriction>
        </simpleType>
      </attribute>
    </extension>
  </complexContent>
</complexType>
<complexType name="IntegerDataType">
  <annotation>
    <documentation>Contains an integral value</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:NumericDataType">
      <attribute name="initialValue" type="integer">
        <annotation>
          <documentation>base 10 integer value</documentation>
        </annotation>
      </attribute>
      <attribute name="sizeInBits" type="positiveInteger" default="32"/>
      <attribute name="signed" type="boolean" default="true"/>
    </extension>
  </complexContent>
</complexType>
<complexType name="NumericDataType">
  <annotation>

```



```

    <documentation>An abstract type that is a super type of either an Integer or Float Data type.</documentation>
</annotation>
<complexContent>
  <extension base="xtce:BaseDataType">
    <sequence>
      <element name="ToString" type="xtce:NumberToStringType" minOccurs="0"/>
      <element name="ValidRange" minOccurs="0">
        <complexType>
          <complexContent>
            <extension base="xtce:DecimalRangeType"/>
          </complexContent>
        </complexType>
      </element>
      <element name="DefaultCalibrator" type="xtce:CalibratorType" minOccurs="0"/>
      <element name="ContextCalibratorList" minOccurs="0">
        <complexType>
          <sequence>
            <element name="ContextCalibrator" type="xtce:ContextCalibratorType" maxOccurs="unbounded"/>
          </sequence>
        </complexType>
      </element>
    </sequence>
    <attribute name="validRangeAppliesToCalibrated" type="boolean" default="true"/>
  </extension>
</complexContent>
</complexType>
<complexType name="RelativeTimeDataType">
  <annotation>
    <documentation>Used to contain a relative time value. Used to describe a relative time. Normally used for time offsets. A
Relative time is expressed as PnYn MnDTnH nMnS, where nY represents the number of years, nM the number of months, nD the number of
days, 'T' is the date/time separator, nH the number of hours, nM the number of minutes and nS the number of seconds. The number of seconds
can include decimal digits to arbitrary precision. For example, to indicate a duration of 1 year, 2 months, 3 days, 10 hours, and 30 minutes, one
would write: P1Y2M3DT10H30M. One could also indicate a duration of minus 120 days as: -P120D. An extension of Schema duration type. </
documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:BaseTimeDataType"/>
  </complexContent>
</complexType>
<complexType name="StringDataType">
  <annotation>
    <documentation>Contains a String Value</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:BaseDataType">
      <sequence>
        <element name="SizeRangeInCharacters" type="xtce:IntegerRangeType" minOccurs="0"/>
        <element name="DefaultCalibrator" type="xtce:CalibratorType" minOccurs="0"/>
        <element name="ContextCalibratorList" minOccurs="0">
          <complexType>
            <sequence>
              <element name="ContextCalibrator" type="xtce:ContextCalibratorType"/>
            </sequence>
          </complexType>
        </element>
      </sequence>
    </extension>
  </complexContent>
</complexType>

```

```

<attribute name="initialValue" type="string"/>
<attribute name="restrictionPattern" type="string"/>
<attribute name="characterWidth">
  <simpleType>
    <restriction base="integer">
      <enumeration value="8"/>
      <enumeration value="16"/>
    </restriction>
  </simpleType>
</attribute>
</extension>
</complexContent>
</complexType>
<complexType name="DataEncodingType">
  <annotation>
    <documentation xml:lang="en">Describes how a particular piece of data is sent or received from some non-native, off-platform
device. (e.g. a spacecraft)</documentation>
  </annotation>
  <sequence>
    <element name="ErrorDetectCorrect" type="xtce:ErrorDetectCorrectType" minOccurs="0"/>
    <element name="ByteOrderList" type="xtce:ByteOrderType" minOccurs="0">
      <annotation>
        <documentation>Used to describe an arbitrary byte order in multibyte parameters. First byte in list is the first in the
stream. Byte significance is the is highest for most significant bytes. If not included, it is assumed that the most significant byte is first, least
significant byte last.</documentation>
      </annotation>
    </element>
  </sequence>
  <attribute name="bitOrder" default="mostSignificantBitFirst">
    <simpleType>
      <restriction base="string">
        <enumeration value="leastSignificantBitFirst"/>
        <enumeration value="mostSignificantBitFirst"/>
      </restriction>
    </simpleType>
  </attribute>
</complexType>
<complexType name="IntegerDataEncodingType">
  <annotation>
    <documentation xml:lang="en">For all major encodings of integer data</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:DataEncodingType">
      <sequence>
        <element name="DefaultCalibrator" type="xtce:CalibratorType" minOccurs="0"/>
        <element name="ContextCalibratorList" minOccurs="0">
          <annotation>
            <documentation>Use when different calibrations must be used on the Parameter in different contexts. Use the
first one that tests true</documentation>
          </annotation>
          <complexType>
            <sequence>
              <element name="ContextCalibrator" maxOccurs="unbounded">
                <complexType>
                  <sequence>
                    <element name="UseWhenCondition" type="xtce:BooleanExpressionType"/>

```

```

        <element name="Calibrator" type="xtce:CalibratorType"/>
    </sequence>
</complexType>
</element>
</sequence>
</complexType>
</element>
</sequence>
<attribute name="encoding" default="unsigned">
    <simpleType>
        <restriction base="string">
            <enumeration value="unsigned"/>
            <enumeration value="signMagnitude"/>
            <enumeration value="twosCompliment"/>
            <enumeration value="onesCompliment"/>
            <enumeration value="BCD"/>
            <enumeration value="packedBCD"/>
        </restriction>
    </simpleType>
</attribute>
<attribute name="sizeInBits" type="positiveInteger" default="8"/>
</extension>
</complexContent>
</complexType>
<complexType name="FloatDataEncodingType">
    <annotation>
        <documentation xml:lang="en">For common encodings of floating point data</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:DataEncodingType">
            <sequence>
                <element name="DefaultCalibrator" type="xtce:CalibratorType" minOccurs="0"/>
                <element name="ContextCalibratorList" minOccurs="0">
                    <annotation>
                        <documentation>Use when different calibrations must be used on the Parameter in different contexts. Use the
first one that tests true</documentation>
                    </annotation>
                    <complexType>
                        <sequence>
                            <element name="ContextCalibrator" maxOccurs="unbounded">
                                <complexType>
                                    <sequence>
                                        <element name="UseWhenCondition" type="xtce:BooleanExpressionType"/>
                                        <element name="Calibrator" type="xtce:CalibratorType"/>
                                    </sequence>
                                </complexType>
                            </element>
                        </sequence>
                    </complexType>
                </element>
            </sequence>
            <attribute name="encoding" default="IEEE754_1985">
                <simpleType>
                    <restriction base="string">
                        <enumeration value="IEEE754_1985"/>
                        <enumeration value="MILSTD_1750A"/>
                    </restriction>
                </simpleType>
            </attribute>
        </extension>
    </complexContent>
</complexType>

```

```

        </restriction>
      </simpleType>
    </attribute>
    <attribute name="sizeInBits" default="32">
      <simpleType>
        <restriction base="positiveInteger">
          <enumeration value="32"/>
          <enumeration value="64"/>
          <enumeration value="128"/>
        </restriction>
      </simpleType>
    </attribute>
  </extension>
</complexContent>
</complexType>
<complexType name="StringDataEncodingType">
  <annotation>
    <documentation xml:lang="en">For common encodings of string data</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:DataEncodingType">
      <sequence>
        <element name="DefaultCalibrator" type="xtce:CalibratorType" minOccurs="0"/>
        <element name="ContextCalibratorList" minOccurs="0">
          <annotation>
            <documentation>Use when different calibrations must be used on the Parameter in different contexts. Use the
first one that tests true</documentation>
          </annotation>
          <complexType>
            <sequence>
              <element name="ContextCalibrator" maxOccurs="unbounded">
                <complexType>
                  <sequence>
                    <element name="UseWhenCondition" type="xtce:BooleanExpressionType"/>
                    <element name="Calibrator" type="xtce:CalibratorType"/>
                  </sequence>
                </complexType>
              </element>
            </sequence>
          </complexType>
        </element>
      </sequence>
    </extension>
  </complexContent>
  <element name="SizeInBits">
    <complexType>
      <choice>
        <element name="Fixed" type="xtce:IntegerValueType"/>
        <element name="TerminationChar" type="hexBinary">
          <annotation>
            <documentation>Like C strings, they are terminated with a special string, usually a null character.</
documentation>
          </annotation>
          <!-- default="00" (does not work with CASTOR 0.9.5.3) -->
        </element>
        <element name="LeadingSize">
          <annotation>
            <documentation>Like PASCAL strings, the size of the string is given as an integer at the start of the
string. SizeTag must be an unsigned Integer</documentation>
          </annotation>

```

```

        </annotation>
        <complexType>
          <attribute name="sizeInBitsOfSizeTag" type="positiveInteger" default="16"/>
        </complexType>
      </element>
    </choice>
  </complexType>
</element>
</sequence>
<attribute name="encoding" default="UTF-8">
  <simpleType>
    <restriction base="string">
      <enumeration value="UTF-8"/>
      <enumeration value="UTF-16"/>
    </restriction>
  </simpleType>
</attribute>
</extension>
</complexContent>
</complexType>
<complexType name="BinaryDataEncodingType">
  <annotation>
    <documentation xml:lang="en">For binary data or for integer, float, string, or time data that is not in any of the known encoding
formats. For any data that is not encoded in any of the known integer, float, string, or time data formats use a To/From transform algorithm.</
documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:DataEncodingType">
      <sequence>
        <element name="SizeInBits" type="xtce:IntegerValueType"/>
        <element name="FromBinaryTransformAlgorithm" type="xtce:InputAlgorithmType" minOccurs="0">
          <annotation>
            <documentation>Used to convert binary data to an application data type</documentation>
          </annotation>
        </element>
        <element name="ToBinaryTransformAlgorithm" type="xtce:InputAlgorithmType" minOccurs="0">
          <annotation>
            <documentation>Used to convert binary data from an application data type to binary data</documentation>
          </annotation>
        </element>
      </sequence>
    </extension>
  </complexContent>
</complexType>
<simpleType name="EpochType">
  <annotation>
    <documentation>Epochs may be specified as a date or TAI (which correlates to 1 January 1958)</documentation>
  </annotation>
  <union memberTypes="date">
    <simpleType>
      <restriction base="string">
        <enumeration value="TAI"/>
      </restriction>
    </simpleType>
  </union>
</simpleType>

```

```

<!--***** DataTypes-->
<!--***** -->
<!--***** Common Types Schema -->
<!-- Basic elements used for in all dictionaries -->
<complexType name="AliasSetType">
  <annotation>
    <documentation>Contains an unordered collection of Alias's</documentation>
  </annotation>
  <sequence>
    <element name="Alias" maxOccurs="unbounded">
      <annotation>
        <documentation xml:lang="en">Used to contain an alias (alternate) name or ID for the the object. For example, a
parameter may have a mnemonic, an on-board id, and special IDs used by various ground software applications; all of these are alias's. Some
ground system processing equipent has some severe naming restrictions on parameters (e.g., names must less then 12 characters, single case
or integral id's only); their alias's provide a means of capturing each name in a "nameSpace".</documentation>
      </annotation>
      <complexType>
        <attribute name="nameSpace" type="string" use="required"/>
        <attribute name="alias" type="string" use="required"/>
      </complexType>
    </element>
  </sequence>
</complexType>
<complexType name="ANDedConditionsType">
  <annotation>
    <documentation>A list of boolean comparisons, or boolean groups that are logically ANDed together. Any ORed conditions in the
list are evaluated first.</documentation>
  </annotation>
  <choice minOccurs="2" maxOccurs="unbounded">
    <element name="Condition" type="xtce:ComparisonCheckType"/>
    <element name="ORedConditions" type="xtce:ORedConditionsType"/>
  </choice>
</complexType>
<simpleType name="BinaryType">
  <annotation>
    <documentation>A simple restriction on string for hexadecimal numbers. Must be in 0b or 0B form.</documentation>
  </annotation>
  <restriction base="string">
    <pattern value="0[bB][0-1]+"/>
  </restriction>
</simpleType>
<complexType name="BooleanExpressionType">
  <annotation>
    <documentation>Holds an arbitrarily complex boolean expression</documentation>
  </annotation>
  <choice>
    <element name="Condition" type="xtce:ComparisonCheckType"/>
    <element name="ANDedConditions" type="xtce:ANDedConditionsType"/>
    <element name="ORedConditions" type="xtce:ORedConditionsType"/>
  </choice>
</complexType>
<complexType name="ByteOrderType">
  <annotation>
    <documentation>An ordered list of bytes where the order of the bytes is in stream order. Each byte has an attribute giving its
significance.</documentation>

```

`<appinfo>`The software must check to ensure that the significance of each byte is unique, and does not contain bytes of greater significance greater than the size of the object`</appinfo>`

```

</annotation>
<sequence minOccurs="2" maxOccurs="unbounded">
  <element name="Byte">
    <complexType>
      <attribute name="byteSignificance" type="nonNegativeInteger" use="required"/>
    </complexType>
  </element>
</sequence>
</complexType>
<complexType name="ComparisonCheckType">
  <annotation>
    <documentation xml:lang="en">A ParameterInstanceRef to a value or another parameter instance</documentation>
  </annotation>
  <sequence>
    <element name="ParameterInstanceRef" type="xtce:ParameterInstanceRefType"/>
    <element name="ComparisonOperator" type="xtce:ComparisonOperatorsType"/>
    <choice>
      <element name="ParameterInstanceRef" type="xtce:ParameterInstanceRefType">
        <annotation>
          <documentation>Parameter is assumed to be of the same type as the comparison Parameter</documentation>
        </annotation>
      </element>
      <element name="Value" type="string">
        <annotation>
          <documentation>Value is assumed to be of the same type as the comparison Parameter</documentation>
        </annotation>
      </element>
    </choice>
  </sequence>
</complexType>
<complexType name="ComparisonType">
  <annotation>
    <documentation>A simple ParameterInstanceRef to value comparison. The string supplied in the value attribute needs to be converted to a type matching the Parameter being compared to. Numerical values are assumed to be base 10 unless proceeded by 0x (hexadecimal), 0o (octal), or 0b (binary). The value is truncated to use the least significant bits that match the bit size of the Parameter being compared to.</documentation>
  </annotation>
  <simpleContent>
    <extension base="xtce:ParameterInstanceRefType">
      <attribute name="comparisonOperator" type="xtce:ComparisonOperatorsType" default="="/>
      <attribute name="value" type="string" use="required"/>
    </extension>
  </simpleContent>
</complexType>
<simpleType name="ComparisonOperatorsType">
  <annotation>
    <documentation xml:lang="en">Operators to use when testing a boolean condition for a validity check</documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="="/>
    <enumeration value="!="/>
    <enumeration value="&lt;"/>
    <enumeration value="&lt;="/>
    <enumeration value=">"/>
  </restriction>

```

```

        <enumeration value=">="/>
    </restriction>
</simpleType>
<complexType name="ContextCalibratorType">
    <annotation>
        <documentation>Context calibrations are applied when the ContextMatch is true. Context calibrators override Default calibrators</
documentation>
    </annotation>
    <sequence>
        <element name="ContextMatch" type="xtce:MatchCriteriaType"/>
        <element name="Calibrator" type="xtce:CalibratorType"/>
        <!-- <element name="Context" type="xtce:MatchCriteriaType"/> -->
    </sequence>
</complexType>
<complexType name="DecimalValueType">
    <annotation>
        <documentation>Contains an Numeric value; value may be provided directly or via the value in a parameter.</documentation>
    </annotation>
    <choice>
        <element name="FixedValue" type="decimal"/>
        <element name="DynamicValue">
            <annotation>
                <documentation>Uses a parameter to for the value. The parameter value may be optionally adjusted by a Linear function
or use a series of boolean expressions to lookup the value. Anything more complex and a DynamicValue with a CustomAlgorithm may be used
</documentation>
            </annotation>
            <complexType>
                <sequence>
                    <element name="ParameterInstanceRef" type="xtce:ParameterInstanceRefType"/>
                    <element name="LinearAdjustment" minOccurs="0">
                        <annotation>
                            <documentation>A slope and intercept may be applied to scale or shift the value of the parameter in the
dynamic value</documentation>
                        </annotation>
                        <complexType>
                            <attribute name="slope" type="decimal" default="0"/>
                            <attribute name="intercept" type="decimal" default="0"/>
                        </complexType>
                    </element>
                </sequence>
            </complexType>
        </element>
    </choice>
</complexType>
<complexType name="ErrorDetectCorrectType">
    <annotation>
        <documentation xml:lang="en">A simple element that provides for simple, but common error checking and detection.</
documentation>
    </annotation>
    <choice>
        <element name="Parity">
            <annotation>
                <documentation xml:lang="en">Bit position starts with 'zero'.</documentation>
            </annotation>
            <complexType>
                <attribute name="type" use="required">

```



```

        <simpleType>
          <restriction base="string">
            <enumeration value="Even"/>
            <enumeration value="Odd"/>
          </restriction>
        </simpleType>
      </attribute>
      <attribute name="bitsFromReference" type="nonNegativeInteger" use="required"/>
      <attribute name="reference" default="start">
        <simpleType>
          <restriction base="string">
            <enumeration value="start"/>
            <enumeration value="end"/>
          </restriction>
        </simpleType>
      </attribute>
    </complexType>
  </element>
  <element name="CRC">
    <annotation>
      <documentation xml:lang="en">Cyclic Redundancy Check definition. Legal values for coefficient's are 0 or 1. Exponents
must be integer values.</documentation>
    </annotation>
    <complexType>
      <sequence>
        <element name="Polynomial" type="xtce:PolynomialType"/>
      </sequence>
      <attribute name="bitsFromReference" type="nonNegativeInteger"/>
      <attribute name="reference" default="start">
        <simpleType>
          <restriction base="string">
            <enumeration value="start"/>
            <enumeration value="end"/>
          </restriction>
        </simpleType>
      </attribute>
    </complexType>
  </element>
</choice>
</complexType>
<simpleType name="FixedIntegerValueType">
  <annotation>
    <documentation>A simple union type combining integer, octal, binary, and hexadecimal types</documentation>
  </annotation>
  <union memberTypes="integer xtce:HexadecimalType xtce:OctalType xtce:BinaryType"/>
</simpleType>
<complexType name="HeaderType">
  <annotation>
    <documentation xml:lang="en">Schema for a Header record. A header contains general information about the system or
subsystem.</documentation>
  </annotation>
  <sequence>
    <element name="AuthorSet" minOccurs="0">
      <complexType>
        <sequence>
          <element name="Author" type="string" minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
      </complexType>
    </element>
  </sequence>

```

```

        </sequence>
    </complexType>
</element>
<element name="NoteSet" minOccurs="0">
    <complexType>
        <sequence>
            <element name="Note" type="string" minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
    </complexType>
</element>
<element name="HistorySet" minOccurs="0">
    <complexType>
        <sequence>
            <element name="History" type="string" minOccurs="0" maxOccurs="unbounded"/>
        </sequence>
    </complexType>
</element>
</sequence>
<attribute name="version" type="string"/>
<attribute name="date" type="string"/>
<attribute name="classification" type="string" default="NotClassified"/>
<attribute name="classificationInstructions" type="string"/>
<attribute name="validationStatus" use="required">
    <simpleType>
        <restriction base="string">
            <enumeration value="Unknown"/>
            <enumeration value="Working"/>
            <enumeration value="Draft"/>
            <enumeration value="Test"/>
            <enumeration value="Validated"/>
            <enumeration value="Released"/>
            <enumeration value="Withdrawn"/>
        </restriction>
    </simpleType>
</attribute>
</complexType>
<simpleType name="HexadecimalType">
    <annotation>
        <documentation>A simple restriction on string for hexadecimal numbers. Must be in 0x or 0X form.</documentation>
    </annotation>
    <restriction base="string">
        <pattern value="0[xX][0-9a-fA-F]+"/>
    </restriction>
</simpleType>
<complexType name="IntegerValueType">
    <annotation>
        <documentation>Contains an Integer value; value may be provided directly or via the value in a parameter.</documentation>
    </annotation>
    <choice>
        <element name="FixedValue" type="xtce:FixedIntegerValueType"/>
        <element name="DynamicValue">
            <annotation>
                <documentation>Uses a parameter to for the value. The parameter value may be optionally adjusted by a Linear function or use a series of boolean expressions to lookup the value. Anything more complex and a DynamicValue with a CustomAlgorithm may be used</documentation>
            </annotation>

```

```

    <complexType>
      <sequence>
        <element name="ParameterInstanceRef" type="xtce:ParameterInstanceRefType"/>
        <element name="LinearAdjustment" minOccurs="0">
          <annotation>
            <documentation>A slope and intercept may be applied to scale or shift the value of the parameter in the
dynamic value</documentation>
          </annotation>
          <complexType>
            <attribute name="slope" type="integer" default="0"/>
            <attribute name="intercept" type="integer" default="0"/>
          </complexType>
        </element>
      </sequence>
    </complexType>
  </element>
  <element name="DiscreteLookupList">
    <annotation>
      <documentation>Lookup a value using the lookup list supplied. Use the first match found.</documentation>
    </annotation>
    <complexType>
      <sequence>
        <element name="DiscreteLookup" maxOccurs="unbounded">
          <complexType>
            <complexContent>
              <extension base="xtce:MatchCriteriaType">
                <attribute name="value" type="integer" use="required"/>
              </extension>
            </complexContent>
          </complexType>
        </element>
      </sequence>
    </complexType>
  </element>
</choice>
</complexType>
<simpleType name="MathOperatorsType">
  <annotation>
    <documentation xml:lang="en">Mathematical operators</documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="+"/>
    <enumeration value="-"/>
    <enumeration value="mult"/>
    <enumeration value="div"/>
    <enumeration value="mod"/>
    <enumeration value="exp"/>
    <enumeration value="bitor"/>
    <enumeration value="bitand"/>
    <enumeration value="bitxor"/>
  </restriction>
</simpleType>
<complexType name="MatchCriteriaType">
  <annotation>
    <documentation>Contains either a simple Comparison, a ComparisonList, an arbitrarily complex BooleanExpression or an escape
to an externally defined algorithm</documentation>
  </annotation>

```

```

</annotation>
<choice>
  <element name="Comparison" type="xtce:ComparisonType">
    <annotation>
      <documentation>A simple comparison check</documentation>
    </annotation>
  </element>
  <element name="ComparisonList">
    <annotation>
      <documentation>All comparisons must be true</documentation>
    </annotation>
    <complexType>
      <sequence>
        <element name="Comparison" type="xtce:ComparisonType" maxOccurs="unbounded"/>
      </sequence>
    </complexType>
  </element>
  <element name="BooleanExpression" type="xtce:BooleanExpressionType">
    <annotation>
      <documentation>An arbitrarily complex boolean expression</documentation>
    </annotation>
  </element>
  <element name="CustomAlgorithm" type="xtce:InputAlgorithmType">
    <annotation>
      <documentation>An escape to an externally defined algorithm</documentation>
    </annotation>
  </element>
</choice>
</complexType>
<complexType name="MathOperationType">
  <annotation>
    <documentation xml:lang="en">A simple math operation</documentation>
  </annotation>
  <sequence>
    <choice>
      <element name="ParameterInstanceRef" type="xtce:ParameterInstanceRefType"/>
      <element name="Value" type="string">
        <annotation>
          <documentation>Value is assumed to be of the same type as the Parameter</documentation>
        </annotation>
      </element>
    </choice>
    <sequence minOccurs="0">
      <element name="Operator" type="xtce:MathOperatorsType"/>
      <choice>
        <element name="ParameterInstanceRef" type="xtce:ParameterInstanceRefType"/>
        <element name="Value" type="string">
          <annotation>
            <documentation>Value is assumed to be of the same type as the Parameter</documentation>
          </annotation>
        </element>
      </choice>
    </sequence>
  </sequence>
</complexType>
<simpleType name="NameType">

```

```

    <annotation>
      <documentation xml:lang="en">Used for "directory" style unique names. We need to preclude '.', '/', ':', '[' and ']'. Only letters,
digits, '_', '' and "-" are allowed </documentation>
    </annotation>
    <restriction base="string">
      <pattern value="[a-zA-Z0-9_\- ]*" />
    </restriction>
  </simpleType>
  <complexType name="NameDescriptionType">
    <annotation>
      <documentation xml:lang="en">The type definition used by most elements that require a name with optional descriptions. The
short description is intended to be used for quick "memory jogger" descriptions of the object. </documentation>
    </annotation>
    <sequence>
      <element name="LongDescription" type="string" minOccurs="0">
        <annotation>
          <documentation>The Long Description is intended to be used for explanatory descriptions of the object and may include
HTML markup. Long Descriptions are of unbounded length</documentation>
        </annotation>
      </element>
      <element name="AliasSet" type="xtce:AliasSetType" minOccurs="0" />
    </sequence>
    <attribute name="name" type="xtce:NameType" use="required" />
    <attribute name="shortDescription" type="string" use="optional">
      <annotation>
        <documentation>It is strongly recommended that the short description be kept under 80 characters in length</documentation>
      </annotation>
    </attribute>
  </complexType>
  <simpleType name="NameReferenceType">
    <annotation>
      <documentation xml:lang="en">Used when referencing a directory style "NameType". No characters are prohibited.</
documentation>
    </annotation>
    <restriction base="string" />
  </simpleType>
  <complexType name="NumberToStringType">
    <annotation>
      <documentation xml:lang="en">There are two ways numeric data can be changed to string data: using a Java style
NumberFormat, or using an enumerated list. Enumerated lists can be assigned to a single value or a value range.</documentation>
    </annotation>
    <choice>
      <choice minOccurs="unbounded">
        <element name="ValueEnumeration" type="xtce:ValueEnumerationType">
          <annotation>
            <documentation xml:lang="en">A number or range assigned to a string.</documentation>
          </annotation>
        </element>
        <element name="RangeEnumeration">
          <annotation>
            <documentation xml:lang="en">A string value associated with a numerical range.</documentation>
          </annotation>
          <complexType>
            <complexContent>
              <extension base="xtce:DecimalRangeType">
                <attribute name="label" type="string" use="required" />
              </extension>
            </complexContent>
          </complexType>
        </element>
      </choice>
    </choice>
  </complexType>

```

```

        </extension>
      </complexContent>
    </complexType>
  </element>
</choice>
<element name="NumberFormat">
  <complexType>
    <attribute name="numberBase" type="xtce:RadixType" use="optional"/>
    <attribute name="minimumFractionDigits" type="nonNegativeInteger" use="optional"/>
    <attribute name="maximumFractionDigits" type="nonNegativeInteger" use="optional"/>
    <attribute name="minimumIntegerDigits" type="nonNegativeInteger" use="optional"/>
    <attribute name="maximumIntegerDigits" type="nonNegativeInteger" use="optional"/>
    <attribute name="negativeSuffix" type="string" use="optional"/>
    <attribute name="positiveSuffix" type="string" use="optional"/>
    <attribute name="negativePrefix" type="string" use="optional" default="-"/>
    <attribute name="positivePrefix" type="string" use="optional"/>
    <attribute name="showThousandsGrouping" type="boolean" use="optional" default="true"/>
    <attribute name="notation" use="optional" default="normal">
      <simpleType>
        <restriction base="string">
          <enumeration value="normal"/>
          <enumeration value="scientific"/>
          <enumeration value="engineering"/>
        </restriction>
      </simpleType>
    </attribute>
  </complexType>
</element>
</choice>
</complexType>
<simpleType name="OctalType">
  <annotation>
    <documentation>A simple restriction on string for hexadecimal numbers. Must be in 0o or 0O form.</documentation>
  </annotation>
  <restriction base="string">
    <pattern value="0[oO][0-7]+"/>
  </restriction>
</simpleType>
<complexType name="ORedConditionsType">
  <annotation>
    <documentation>A list of boolean comparisons, or boolean groups that are logically ORed together. Any ANDed conditions in the list are evaluated first.</documentation>
  </annotation>
  <choice minOccurs="2" maxOccurs="unbounded">
    <element name="Condition" type="xtce:ComparisonCheckType"/>
    <element name="ANDedConditions" type="xtce:ANDedConditionsType"/>
  </choice>
</complexType>
<complexType name="ParameterSetType">
  <annotation>
    <documentation>Used by both the TelemetryMetaData and the CommandMetaData components each may be built independently.</documentation>
  </annotation>
  <choice maxOccurs="unbounded">
    <element name="Parameter">
      <annotation>

```

```

    <appinfo>Need to ensure that the named types actually exist</appinfo>
  </annotation>
  <complexType>
    <complexContent>
      <extension base="xtce:NameDescriptionType">
        <sequence>
          <element name="ParameterProperties" type="xtce:ParameterPropertiesType" minOccurs="0"/>
        </sequence>
        <attribute name="parameterTypeRef" type="xtce:NameReferenceType" use="required"/>
      </extension>
    </complexContent>
  </complexType>
</element>
<element name="ParameterRef" type="xtce:ParameterRefType">
  <annotation>
    <documentation>Used to include a Parameter defined in another sub-system in this sub-system.</documentation>
  </annotation>
</element>
</choice>
</complexType>
<complexType name="PolynomialType">
  <annotation>
    <documentation xml:lang="en">A polynomial expression. For example: 3 + 2x</documentation>
  </annotation>
  <sequence>
    <element name="Term" maxOccurs="unbounded">
      <annotation>
        <documentation xml:lang="en">A term in a polynomial expression. </documentation>
      </annotation>
      <complexType>
        <attribute name="coefficient" type="double" use="required"/>
        <attribute name="exponent" type="double" use="required"/>
      </complexType>
    </element>
  </sequence>
</complexType>
<complexType name="PropertyType">
  <annotation>
    <documentation>Used for custom user properties</documentation>
  </annotation>
  <complexContent>
    <extension base="xtce:NameDescriptionType">
      <sequence minOccurs="0" maxOccurs="unbounded">
        <element name="Property" type="xtce:PropertyType"/>
      </sequence>
      <attribute name="value" type="string" use="required"/>
    </extension>
  </complexContent>
</complexType>
<simpleType name="RadixType">
  <annotation>
    <documentation xml:lang="en">Specifies the number base</documentation>
  </annotation>
  <restriction base="string">
    <enumeration value="Decimal"/>
    <enumeration value="Hexadecimal"/>
  </restriction>
</simpleType>

```

```

        <enumeration value="Octal"/>
        <enumeration value="Binary"/>
    </restriction>
</simpleType>
<complexType name="ReferenceTimeType">
    <annotation>
        <documentation>Most time values are relative to another time e.g. seconds are relative to minutes, minutes are relative to hours.
This type is used to describe this relationship starting with the least significant time Parameter to and progressing to the most significant time
parameter. </documentation>
    </annotation>
    <choice>
        <element name="OffsetFrom" type="xtce:ParameterInstanceRefType"/>
        <element name="Epoch" type="xtce:EpochType"/>
    </choice>
</complexType>
<simpleType name="RelativeTimeType">
    <annotation>
        <documentation>Used to describe a relative time. Normally used for time offsets. A Relative time is expressed as PnYn MnDTnH
nMnS, where nY represents the number of years, nM the number of months, nD the number of days, 'T' is the date/time separator, nH the
number of hours, nM the number of minutes and nS the number of seconds. The number of seconds can include decimal digits to arbitrary
precision. For example, to indicate a duration of 1 year, 2 months, 3 days, 10 hours, and 30 minutes, one would write: P1Y2M3DT10H30M.
One could also indicate a duration of minus 120 days as: -P120D. An extension of Schema duration type. </documentation>
    </annotation>
    <restriction base="duration"/>
</simpleType>
<complexType name="RepeatType">
    <annotation>
        <documentation xml:lang="en">Hold a structure that can be repeated X times, where X is the Count</documentation>
    </annotation>
    <sequence>
        <element name="Count" type="xtce:IntegerValueType">
            <annotation>
                <documentation xml:lang="en">Value (either fixed or dynamic) that contains the count of repeated structures.</
documentation>
            </annotation>
        </element>
        <element name="Offset" minOccurs="0">
            <annotation>
                <documentation xml:lang="en">Indicates the distance between repeating entries (the last bit of one entry to the start bit of
the next entry)</documentation>
            </annotation>
            <complexType>
                <complexContent>
                    <extension base="xtce:IntegerValueType">
                        <attribute name="offsetSizeInBits" type="positiveInteger" default="1"/>
                    </extension>
                </complexContent>
            </complexType>
        </element>
    </sequence>
</complexType>
<complexType name="SplinePointType">
    <annotation>
        <documentation xml:lang="en">a spline is a set on points from which a curve may be drawn to interpolate raw to calibrated
values</documentation>
    </annotation>

```



```

    <attribute name="order" type="positiveInteger" default="1"/>
    <attribute name="raw" type="double" use="required"/>
    <attribute name="calibrated" type="double" use="required"/>
</complexType>
<complexType name="UnitType" mixed="true">
    <annotation>
        <documentation>Used to hold the unit(s) plus possibly the exponents for the units</documentation>
    </annotation>
    <attribute name="power" type="decimal" use="optional" default="1"/>
    <attribute name="factor" type="string" default="1"/>
    <attribute name="description" type="string"/>
</complexType>
<complexType name="ValueEnumerationType">
    <annotation>
        <documentation>Contains a value and an associated string label</documentation>
    </annotation>
    <attribute name="value" type="integer" use="required"/>
    <attribute name="label" type="string" use="required"/>
</complexType>
<!--Types used with alarms-->
<complexType name="AlarmConditionsType">
    <annotation>
        <documentation>When the alarm is determined by boolean logic</documentation>
    </annotation>
    <sequence>
        <element name="WatchAlarm" type="xtce:MatchCriteriaType" minOccurs="0"/>
        <element name="WarningAlarm" type="xtce:MatchCriteriaType" minOccurs="0"/>
        <element name="DistressAlarm" type="xtce:MatchCriteriaType" minOccurs="0"/>
        <element name="CriticalAlarm" type="xtce:MatchCriteriaType" minOccurs="0"/>
        <element name="SevereAlarm" type="xtce:MatchCriteriaType" minOccurs="0"/>
    </sequence>
</complexType>
<complexType name="AlarmRangesType">
    <annotation>
        <documentation>Contains five ranges: Watch, Warning, Distress, Critical, and Severe each in increasing severity. Normally, only the Warning and Critical ranges are used and the color yellow is associated with Warning and the color red is associated with Critical. The ranges given are valid for numbers lower than the min and higher than the max values. These ranges should not overlap, but if they do, assume the most severe range is to be applied. All ranges are optional and it is quite allowed for there to be only one end of the range. </documentation>
    </annotation>
    <sequence>
        <element name="WatchRange" type="xtce:DecimalRangeType" minOccurs="0"/>
        <element name="WarningRange" type="xtce:DecimalRangeType" minOccurs="0"/>
        <element name="DistressRange" type="xtce:DecimalRangeType" minOccurs="0"/>
        <element name="CriticalRange" type="xtce:DecimalRangeType" minOccurs="0"/>
        <element name="SevereRange" type="xtce:DecimalRangeType" minOccurs="0"/>
    </sequence>
    <attribute name="appliesToCalibratedValues" type="boolean" default="true"/>
</complexType>
<complexType name="ContextAlarmType">
    <annotation>
        <documentation>Context alarms are applied when the ContextMatch is true. Context alarms override Default alarms</documentation>
    </annotation>
    <complexContent>
        <extension base="xtce:NumericAlarmConditionType">

```

```

        <sequence>
            <element name="ContextMatch" type="xtce:MatchCriteriaType"/>
        </sequence>
    </extension>
</complexContent>
</complexType>
<complexType name="DecimalRangeType">
    <annotation>
        <documentation xml:lang="en">A range of numbers. "minInclusive", "minExclusive", "maxInclusive" and "maxExclusive" attributes
are borrowed from the W3C schema language.</documentation>
    </annotation>
    <attribute name="minInclusive" type="decimal"/>
    <attribute name="minExclusive" type="decimal"/>
    <attribute name="maxInclusive" type="decimal"/>
    <attribute name="maxExclusive" type="decimal"/>
</complexType>
<complexType name="IntegerRangeType">
    <annotation>
        <documentation xml:lang="en">An integral range of numbers. "min", and "max".</documentation>
    </annotation>
    <attribute name="min" type="integer"/>
    <attribute name="max" type="integer"/>
</complexType>
<complexType name="NumericAlarmConditionType">
    <annotation>
        <documentation>Alarms associated with numeric data types</documentation>
    </annotation>
    <choice>
        <sequence>
            <element name="StaticAlarmRanges" type="xtce:AlarmRangesType" minOccurs="0">
                <annotation>
                    <documentation>StaticAlarmRanges are used to trigger alarms when the parameter value passes some threshold
value</documentation>
                </annotation>
            </element>
            <element name="ChangePerSecondAlarmRanges" type="xtce:AlarmRangesType" minOccurs="0">
                <annotation>
                    <documentation>ChangePerSecondAlarmRanges are used to trigger alarms when the parameter value's rate-of-
change passes some threshold value. An alarm condition that triggers when the value changes too fast (or too slow)</documentation>
                </annotation>
            </element>
        </sequence>
        <element name="ConditionalAlarm">
            <annotation>
                <documentation>A MatchCriteria may be specifiect for each of the 5 alarm levels. Each level is optional and the alarm
should be the highest level to test true.</documentation>
            </annotation>
            <complexType>
                <sequence>
                    <element name="StaticAlarmConditions" type="xtce:AlarmConditionsType" minOccurs="0"/>
                    <element name="ChangePerSecondAlarmConditions" type="xtce:AlarmConditionsType" minOccurs="0"/>
                </sequence>
            </complexType>
        </element>
        <element name="CustomAlarm" type="xtce:InputAlgorithmType">
            <annotation>

```

```

        <documentation>An escape for ridiculously complex alarm conditions. Will trigger on changes to the containing
Parameter. </documentation>
    </annotation>
</element>
</choice>
<attribute name="minViolations" type="positiveInteger" default="1">
    <annotation>
        <documentation>Number of successive values of the Parameter for the Alarm to trigger.</documentation>
    </annotation>
</attribute>
</complexType>
<!--***** End of Common Types Schema -->
</schema>

```


Annex B

(normative)

Schema Style Notes

A number of conventions were developed and adopted during the authorship of the schema to make understanding the schema easier and the wording of a presentation more consistent.

- Element and Type names begin with a capital letter.
- Type names end with the word "Type."
- Attribute names begin with a lowercase letter.
- Usually, when the UML class diagram references classes, W3C Elements are used, and whenever the UML references simple types (strings, ints), W3C Attributes are used. In general, attributes are preferred over elements because they're easier to deal with in SAX and DOM, but whenever the Element/Attribute may one day carry metadata, elements should be used. One exception is enumerated classes because enumerations may be defined for attributes but not for elements.
- Bias toward self-describing names over short, bandwidth conserving ones.
- Use mixed case in names rather than underscores to combine multiple words (camelCase).
- A documentation annotation is included in every element and type definition. Annotations for a type are included with the type definition, use of the type is annotated in the element definition.
- Hints on units (for values with units) are provided in the names of attributes and elements (e.g., "dataRateInBPS" is preferred over "dataRate" OR "frameLengthInBits" is preferred over "frameLength").
- Major elements or any elements used multiple times are first defined with a complexType definition.
- All collections are put inside either a "List" element or a "Set" Element depending on whether the collection is ordered or unordered.
- Simplicity in the XML files is favored over simplicity in the Schema.
- Whenever an additional validity check must be performed that is not describable in the schema language, an appinfo annotation describes that validity check.

Annex C

(normative)

Bibliography

- [1] *CCSDS Packet Telemetry*, CCSDS 102.0-B-4
- [2] *CCSDS Telecommand*, CCSDS 203.0-B-1
- [3] *Telemetry and Telecommand Packet Utilization*, Draft 5.3, 5 Apr 2001, ECSS-E-70-41
- [4] *SCOS-2000 Database Import ICD*, Issue 5.0, 19 Jun 2001, S2K-MCS-ICD-0001-TOS-GCI
- [5] *Telemetric and Command Data Specification*, Space RFP-1, 20 Aug 2001, Space/01-04-01
- [6] *Packet Utilization Standard*, Issue 1, May 1994, ESA PSS-07-101
- [7] *CCSDS Time Code Formats*, Issue 2, April 1990, CCSDS 301.0-B-2
- [8] *SCOS-2000 Synthetic Parameters Software User Manual*, Issue 3.1, September 2001, S2K-MCS-SUM-0019-TOS-GCI
- [9] *Telemetry Attributes Transfer Standard (TMATS)*, IRIG-106

INDEX

A

Abbreviations 3
AlgorithmSet 10
ArgumentList 13
ArgumentTypeSet 12

B

BaseMetaCommand 13

C

CCSDS packet formats v
CCSDS packets 1
CCSDS Telemetry and Commanding Packaging format 2
CommandContainer 14
CommandMetaData 5, 6, 11
Commands 2
ContainerSet 9

D

Defaults 15
DefaultSignificance and ContextSignificanceList 14
DOM 75

H

header record 6

I

Interlock 14
IRIG-106 minor frame standards v

L

lifecycle phases v

M

MessageSet 10
MetaCommandSet 13

N

Names 5

P

Packet v
Packet telemetry v
Packet telemetry and commanding v
ParameterSet 8
ParameterToSetList 14
ParameterTypeSet 7
Prefixes and Postfixes 3

S

Satellite iii
SAX 75
schema 15
Schema Style Notes 75
Schema Text 17
ServiceSet 14
Space mission implementations iii
spacecraft manufacturer v
SpaceSystem 5
SpaceSystem Schema 17
StreamSet 10

sub-SpaceSystems 5
symbols/abbreviations 2

T

TDM frames 1
Telemetry 2
Telemetric and Telecommand (TM/TC) v
TelemetryMetaData 5, 6
Time Division Multiplexing (TDM) v
Time Division Multiplexing (TDM) telemetry and commanding v
TransmissionConstraintList 14

V

Verifiers 14

W

W3C v
W3C XML schema 15

X

XML Telemetric and Command Exchange (XTCE) iii

