The Consultative Committee for Space Data Systems

Recommendation for Space Data System Practices

SPACECRAFT ONBOARD INTERFACE SERVICES—SPECIFICATION FOR DICTIONARY OF TERMS FOR ELECTRONIC DATA SHEETS

RECOMMENDED PRACTICE

CCSDS 876.1-M-1

MAGENTA BOOK

March 2024

The Consultative Committee for Space Data Systems

Recommendation for Space Data System Practices

SPACECRAFT ONBOARD INTERFACE SERVICES—SPECIFICATION FOR DICTIONARY OF TERMS FOR ELECTRONIC DATA SHEETS

RECOMMENDED PRACTICE

CCSDS 876.1-M-1

MAGENTA BOOK

March 2024

AUTHORITY

Issue:	Recommended Practice, Issue 1
Date:	March 2024
Location:	Washington, DC, USA

This document has been approved for publication by the Management Council of the Consultative Committee for Space Data Systems (CCSDS) and represents the consensus technical agreement of the participating CCSDS Member Agencies. The procedure for review and authorization of CCSDS documents is detailed in *Organization and Processes for the Consultative Committee for Space Data Systems* (CCSDS A02.1-Y-4), and the record of Agency participation in the authorization of this document can be obtained from the CCSDS Secretariat at the email address below.

This document is published and maintained by:

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

STATEMENT OF INTENT

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

CCSDS Recommendations take two forms: **Recommended Standards** that are prescriptive and are the formal vehicles by which CCSDS Agencies create the standards that specify how elements of their space mission support infrastructure shall operate and interoperate with others; and **Recommended Practices** that are more descriptive in nature and are intended to provide general guidance about how to approach a particular problem associated with space mission support. This **Recommended Practice** is issued by, and represents the consensus of, the CCSDS members. Endorsement of this **Recommended Practice** is entirely voluntary and does not imply a commitment by any Agency or organization to implement its recommendations in a prescriptive sense.

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

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

FOREWORD

This document is a technical Recommended Practice for use in developing flight and ground systems for space missions and has been prepared by the Consultative Committee for Space Data Systems (CCSDS). The Dictionary of Terms described herein is intended for missions that are cross-supported between Agencies of the CCSDS, in the framework of the Spacecraft Onboard Interface Services (SOIS) CCSDS area.

This Recommended Practice specifies a dictionary of terms to be used as a vocabulary in electronic data sheets which describe components that communicate within a spacecraft network. The data sheets are for use by tool chains in the design, assembly, integration, testing, and operation of space missions. The SOIS Dictionary of Terms provides a common vocabulary regardless of the particular tool chain being used.

Through the process of normal evolution, it is expected that expansion, deletion, or modification of this document may occur. This Recommended Practice is therefore subject to CCSDS document management and change control procedures, which are defined in the *Organization and Processes for the Consultative Committee for Space Data Systems* (CCSDS A02.1-Y-4). Current versions of CCSDS documents are maintained at the CCSDS Web site:

http://www.ccsds.org/

Questions relating to the contents or status of this document should be sent to the CCSDS Secretariat at the email address indicated on page i.

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

Member Agencies

- Agenzia Spaziale Italiana (ASI)/Italy.
- Canadian Space Agency (CSA)/Canada.
- Centre National d'Etudes Spatiales (CNES)/France.
- China National Space Administration (CNSA)/People's Republic of China.
- Deutsches Zentrum für Luft- und Raumfahrt (DLR)/Germany.
- European Space Agency (ESA)/Europe.
- Federal Space Agency (FSA)/Russian Federation.
- Instituto Nacional de Pesquisas Espaciais (INPE)/Brazil.
- Japan Aerospace Exploration Agency (JAXA)/Japan.
- National Aeronautics and Space Administration (NASA)/USA.
- UK Space Agency/United Kingdom.

Observer Agencies

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

DOCUMENT CONTROL

Document	Title	Date	Status
CCSDS 876.1-M-1	Spacecraft Onboard Interface Services—Specification for Dictionary of Terms for Electronic Data Sheets, Recommended Practice, Issue 1	March 2024	Original issue

CONTENTS

Section Page

1 INTRODUCTION 1-1

1.1 PURPOSE AND SCOPE OF THIS DOCUMENT 1-1
1.2 APPLICABILITY 1-1
1.3 RATIONALE 1-1
1.4 DOCUMENT STRUCTURE 1-1
1.5 TERMS DEFINED IN THIS RECOMMENDED PRACTICE 1-2
1.6 NOMENCLATURE 1-22
1.7 REFERENCES 1-22

2 OVERVIEW 2-1

2.1 INTRODUCTION 2-1
2.2 THE SUBJECT MATTER OF ELECTRONIC DATA SHEETS 2-2
2.3 PURPOSE AND FUNCTION OF SOIS ELECTRONIC DATA SHEETS 2-3
2.4 USE CASES FOR DOT 2-4
2.5 HOW THE DICTIONARY OF TERMS RELATES TO ELECTRONIC DATA SHEETS 2-4
2.6 USE OF PRE-EXISTING STANDARDS 2-6
2.7 PRINCIPLES OF THE DICTIONARY OF TERMS 2-6
2.8 METADATA 2-12
2.9 MAINTAINING THE DICTIONARY OF TERMS 2-13

3 STRUCTURE OF THE DICTIONARY OF TERMS 3-1

3.1 OVERVIEW 3-1
3.2 ACCESS 3-1
3.3 BASIC CONCEPTS 3-2
3.4 SEMANTIC ATTRIBUTES 3-3
3.5 METADATA 3-8
3.6 USER-DEFINED ONTOLOGIES 3-10

ANNEX A DICTIONARY OF TERMS FOR ELECTRONIC DATA SHEETS IMPLEMENTATION CONFORMANCE STATEMENT PROFORMA (NORMATIVE) A-1
ANNEX B SECURITY CONSIDERATIONS (INFORMATIVE) B-1
ANNEX C ABBREVIATIONS (INFORMATIVE) C-1
ANNEX D INFORMATIVE REFERENCES (INFORMATIVE) D-1
ANNEX E EXAMPLE DOT/XML ONTOLOGY INSTANTIATIONS (INFORMATIVE) E-1

CONTENTS (continued)

Figure Page

2-1 Major Concepts of This Book, and Their Relationships....2-1
2-2 SEDS Describe Data Interfaces in a Spacecraft....2-2
2-3 Described by SEDS....2-3
2-4 How the DoT Provisions the SEDS Schema....2-5
2-5 References to a Model of Operation....2-9
2-6 Defining a Discrete Variable Type....2-11

1 INTRODUCTION

1.1 PURPOSE AND SCOPE OF THIS DOCUMENT

This document is one of a family of documents specifying the Spacecraft Onboard Interface Services (SOIS)-compliant services to be provided in support of applications.

This document defines the SOIS Specification for Dictionary of Terms (DoT) for Electronic Data Sheets (EDSes) for onboard components. The SOIS DoT provides the vocabulary for electronically defining the interfaces offered by flight components such as sensors, actuators, and software components. This document describes the basic format of the vocabulary, while a SANA registry contains the actual normative details of the vocabulary.

This edition describes the structure of vocabulary for description of data interfaces including functional interfaces and protocols used to access the data interfaces. This edition also describes how to continue to gather vocabulary from subject matter experts for publication through SANA.

1.2 APPLICABILITY

This document applies to any mission or equipment claiming to provide CCSDS SOIS EDS for onboard components. The terms in the SANA publication have been collected from subject matter experts, representing engineering knowledge that applies not only in SOIS but also in other architectures of space data systems.

1.3 RATIONALE

SOIS provides a DoT specification facilitating toolchain compatibility and optional portability of components amongst systems implementing interfaces defined by SOIS EDS.

1.4 DOCUMENT STRUCTURE

This document has the following major sections:

- Section 1, this section, contains administrative information, definitions, and references.
- Section 2 provides an overview of the DoT for EDS and maintenance procedures.
- Section 3 provides a normative description of the structure of the ontology.

In addition, the following annexes are provided:

- Annex A comprises an Implementation Conformance Statement (ICS) Proforma.
- Annex B discusses security considerations.
- Annex C contains a list of acronyms.

– Annex D contains a list of informative references.

– Annex E provides for illustrative purposes example fragments of SOIS EDS.

1.5 TERMINOLOGY

1.5.1 TERMS DEFINED IN THIS RECOMMENDED PRACTICE

For the purposes of this Recommended Practice, the following definitions apply:

actuator: A part of a device that transfers action from an application to the physical world (see *transducer*).

application: An algorithm that applies SOIS services to accomplish the goals of a mission.

component: A logical element of a system accessed through defined communication interfaces. A component can be purely conceptual or realized in software or hardware (e.g., as a field-programmable gate array).

coordinate system: A frame for measurement of physical quantities, which may have one or more dimensions.

device abstraction control procedure, DACP: The control procedure that provides the abstraction of a *device-specific access protocol* to a functional interface. This may involve, for example, the application of calibrations to raw values provided by the device or combination of multiple raw values to determine a derived value in SI units.

Device Access Service, DAS: The aggregation of the *device-specific access protocol*s for each onboard device.

Device Virtualization Service, DVS: The aggregation of *device abstraction control procedure*s for onboard devices.

device: A physical element of a system accessed through subnetwork-layer interfaces (see reference [D2]).

device-specific access protocol, DSAP: The protocol that makes use of a subnetwork service to command and/or acquire data from a device. This is specific to each device as no standardization of access protocols at the device level exists.

dictionary of terms, DoT: Ontology of terms used to describe data in interfaces in Electronic Data Sheets (see reference [D2]).

electronic data sheet, EDS: Electronic description of some details of a device, software component or standard. Unless qualified with the acronym ‘SOIS’, this term is general, referring to any machine-readable data sheet (see *SOIS Electronic Data Sheet*).

glossary: A collection of terms with brief informal explanations of their usage in a particular document.

model of operation: A representation of the parts of a device and their relations, and optionally objects outside the device, any of which can be referenced by semantic attributes in a *SOIS Electronic Data Sheet* (see 2.7.4).

ontology: A collection of descriptions of entities, named by terms, and relationships among those entities (see 2.7). The information in a glossary is a subset of the information in an ontology.

path expression: A string used in referential semantics to identify a part in a model of operation. The string is delimited by '.' and consists of alternating names of object relations and names of classes or individuals in the model of operations. The concept is similar to an XPath expression, but the graph to be traversed is not XML document syntax; instead, it is the graph of relations among classes in the ontology (see 2.7.4.3 for examples).

portability: The capability of a component to be integrated into an assembly without change either to the component or to the assembly interfaces. Portability requires that the definitions of interfaces be consistent across all systems to which they may be ported. Consistency requires that the terms used to define an interface are defined in the DoT (see *toolchain compatibility*).

referential class: A class in the DoT ontology that relates an item of data to a part of a model of operation, as part of the description of an item of data in an interface.

SANA DoT: The normative DoT published in the SANA Web site.

semantic attribute: A property of an item of data, such as reference frame or unit of measure.

sensor: A part of a device that transfers data from the physical world to an application (see *transducer*).

SOIS Electronic Data Sheet(s), SEDS: Electronic description of a device's metadata, device-specific functional and access interfaces, device-specific access protocol, and, optionally, device abstraction control procedure (reference [D2]), compliant with SOIS standards (see *electronic data sheet*).

syntactic attribute: A property of an item of data that describes encoding the data for communication. An example of syntactic attribute is the choice of interpretation of bits as an integer or as a floating-point number (reference [D2]).

term: A word or phrase that has a formally defined interpretation in a particular context of usage. The terms in the SOIS DoT are defined in the context of describing spacecraft components in SEDS.

toolchain compatibility: The capability of a component to be integrated into an assembly using an automated process that can generate any needed interface transformation (see *portability*).

transducer: A measurement probe in a sensor; the active part of an actuator. A transducer is the part of a device that has a coordinate system for measurement or for action.

type: A conceptual class that is defined in an EDS. The instances of a type share the attributes that define the type. The syntactic attributes are defined in the SEDS schema. The semantic attributes are defined in the DoT.

user-defined DoT: A non-portable extension to the SANA DoT used within a project.

value: A formatted instance of data that is acquired from or used as a command to a component.

vocabulary: A collection of terms used as common knowledge in some context. In this document, a vocabulary is the same as an ontology (see 2.7).

1.5.2 PROGRAMMATIC TERMS

1.5.2.1 Overview

The following terms are defined in the standard and core SEDS schemas. These terms were developed to facilitate matching of semantics of variables in interfaces. If these normative terms are present in the SEDS definition of a required interface, then they can be used to search a catalog for provided interfaces that are compatible. To assure successful matching, these terms must be normative, including upper and lower case.

1.5.2.2 Classes

DataTransferRateRange: A class that defines a range of data transfer rates.

SpaceWireLinkEnablement: A class that describes the mechanism to enable a SpaceWire link.

SpaceWireLogicalAddressRange: A class that specifies the included bounds on a range of SpaceWire logical addresses.

SpaceWireProtocol: A class that describes the parameters that constrain the content and sequencing of data sent and received through a SpaceWire link. SpaceWireProtocol has the following subclasses:

- SpaceWireCPTP;
- SpaceWireCustomProtocol;
- SpaceWireRMAP.

ModelOfOperation: A description of the parts of a component and the context in which the component operates. ModelOfOperation has the following subclasses:

- SpaceWireLinkEnablement;
- SpaceWireProtocol;

- SpaceWireRMAPWriteSupport;
- MetadataCategory;
- IntegerRange;
- SubnetInterfaceType;
- Enumeration;
- ParameterOpsType;
- Subnet;
- SpaceWireCPTP;
- SpaceWireCustomProtocol;
- SpaceWireRMAP;
- DataSheetLanguage;
- ModelOfProduction;
- Version;
- DataTransferRateRange;
- SpaceWireLogicalAddressRange;
- SpaceWireRMAPAddressRange;
- DesignParameters;
- SpaceWireSubnet;
- DoT;
- QUDV;
- SEDS;
- Manufacturer;
- ManufacturersModel;
- SerialNumber;
- SOIS_Version.

NOTE – A ModelOfOperation may be expressed in a data modelling language, such as MOF or SysML, and then extracted into a set of OWL classes and relations. The OWL ModelOfOperation contains an individual in the ModelOfOperation class, called the anchor. The anchor has object relations to individuals in other classes that form a graph. A semantic attribute that corresponds to a referential class has for its value a string that describes a path through the graph starting from the name of the anchor and traversing relations by name to a set of target individuals. The meaning of the attribute is a relationship between the EDS object in which the attribute appears and target individuals in the ModelOfOperation.

DataSheetLanguage: A class that specifies the names of documents that control the syntax and semantics of the SEDS instance in which they appear. DataSheetLanguage has the following subclasses:

- DoT;
- QUDV;
- SEDS.

DesignParameters: A class that adjusts the behavior of a component to suit the context in which it operates. The context includes the assembly of components in which the component operates. It also includes the standards of the agency that builds and operates the assembly of components.

NOTE – Often, the assembly of components is a vehicle. DesignParameters must be delivered to a component from the designers of the assembly in which it operates. For this reason, the description of DesignParameters takes the form of an interface to the component.

DoTEncodingAndPrecisionType: A standard type of encoding and precision of floating point numbers.

ExtensibleEnumeration: A class of the DoT that specifies extensions to the enumerations in the seds.xsd file. ExtensibleEnumeration has the following subclasses:

- DoTEncodingAndPrecisionType;
- DoTEncodingRulesType;
- DoTErrorControlType;
- DoTIntegerEncodingType;
- DoTMathOperatorsType;
- DoTStringEncodingType.

DoTEncodingRulesType: A standard type of encoding rules for composite data types.

DoTErrorControlType: A standard type of error control algorithm.

DoTIntegerEncodingType: A standard type of integer encoding.

DoTMathOperatorsType: A standard math operator.

DoTStringEncodingType: A standard type of encoding for string data.

Enumeration: A class that enumerates the possible values of a discrete variable.

Manufacturer: A subclass of class 'ModelOfProduction' that specifies the manufacturer (i.e., the organization that constructed a device or software component), as defined in the DoT.

ModelOfProduction: A class that describes the provenance of a device or software component. ModelOfProduction includes the following subclasses:

- Manufacturer;
- ManufacturersModel;
- SerialNumber.

ManufacturersModel: A string assigned by a Manufacturer to identify a type of product that it makes.

ParameterOpsType: A class that provides an element with which to describe the operational constraints on a parameter.

Quantities, Units, Dimensions, and Values; QUDV: An ontology that defines units of measurement and quantity kinds.

NOTE – QUDV is published by the Object Management Group, so its version numbers follow the conventions of that group. The DoT uses QUDV for the subject matter that it covers.

RefersToModel: A subclass of the class 'SemanticProperty'. RefersToModel includes the following subclasses:

- foreignKeyTo;
- memberOf;
- subject.

NOTE – A class derived from RefersToModel may be called a referential class and shall have no individual members. A semantic attribute in the EDS schema that corresponds to a referential class shall have a range of values in 'xsd:string'. The prefix of the string shall be the name of an individual in the 'ModelOfOperation' class or a derivative of that class. The remainder of the string shall be a path expression delimited by '.', which consists of alternating names of object relations and names of classes or individuals. The definition of a referential class shall relate the part of the EDS that bears the attribute to the individuals or classes in the ontology that are at the end of the path specified by the value of the attribute.

SemanticProperty: An enumeration of properties that describes the intended usage of data in a SEDS. SemanticProperty includes the following subclasses:

- RefersToModel;
- chirality;
- coordinateType;
- differenceConvention;
- enumeration;
- interpretation;
- purpose;
- referenceFrame;
- relationToTimestamp;
- statusConvention;
- toCoordinateType;
- toFrame;
- transformationType;
- foreignKeyTo;
- memberOf;
- subject;
- healthStatus;
- operatingMode;
- starTrackerOperatingMode;
- tecOperatingMode.

SOIS_Version: A subclass of the class 'Version'.

NOTE – The version numbers for the schema and for the DoT have the form of three positive integers concatenated with period characters, '.'. The first integer shall be 0 for the first version and incremented by 1 for versions that are not backward compatible with the previous version. The second integer shall be 0 for each first version that breaks compatibility and incremented by 1 for versions that extend the previous version while maintaining compatibility. The third integer shall be 0 for each first version that breaks compatibility or that extends compatibly, and incremented by 1 for versions that correct errors in previous versions without breaking compatibility or extending compatibly.

Version: A string assigned by the manufacturer to identify a stage in the development of a type of product that it makes. Some versions may be offered in a market, while others may be used only internally for experimental development. Version includes the following subclass:

– SOIS_Version.

SerialNumber: A string assigned by a manufacturer to identify a particular instance of a kind of product that it makes.

SpaceWireSubnet: A model of operation of SpaceWire as a subnetwork in SOIS. The attributes of this class and of associated classes define the SpaceWire configuration parameters.

Subnet: A description of the parameters of a technology that implements the functions of the physical, datalink, and network layers of a standard OSI protocol stack in a SOIS protocol stack. The Subnet description provides a standard framework in which to specify the particular technology parameters for a device in a SEDS.

UnitsNotYetIntegratedIntoQUDV: A temporary category of discrete concept in which the concepts that represent units not covered by QUDV conventions can reside until the QUDV conventions have been implemented.

chirality: The handedness (i.e., left- or right-handed) of a coordinate system.

coordinateType: A class that indicates how to interpret a semantic type that is an array of coordinates.

NOTE – The arrays to which the coordinate type applies may be interpreted as locations in a phase space, such as position and velocity vectors. For arrays that may be interpreted as transformation from one frame to another, such as quaternions and direction cosine matrices, the transformation type applies. The term specifies a particular combination of the following terms: coordinate system, array size, unit of measure, reference frame. Some of the terms listed above may be incompletely determined by a coordinateType value, and so must be specified in an EDS with that value when defining an item of data. Those terms above that are completely determined by a coordinateType value need not be specified with that value in an

EDS. The definition of a value of coordinateType in the DoT must provide the information about the terms listed above. In an EDS, this term is the coordinateType attribute of a data item. Use cases include interpreting a semantic type that is an array of numbers.

differenceConvention: A class used to compute the difference between 'relative' variables.

NOTE – In an EDS, this term is the 'differenceConvention' attribute of a data item. Use cases include the specification that the change in speed reported by a single-axis accelerometer is the difference between the speed at the time of reporting and the speed at the time of the previous report.

Enumeration: A subclass of ModelOfOperation whose derived classes contain enumerations of the possible interoperable meanings of discrete data items described by a SEDS. Enumeration includes the following subclasses:

- healthStatus;
- operatingMode;
- starTrackerOperatingMode;
- tecOperatingMode.

NOTE – The classes derived from this class represent enumerations that may be the range of a data item in an interface described by an EDS. The individuals in classes derived from this class are discrete concepts that are values in the range of an enumeration data item in an interface described by an EDS.

foreignKeyTo: A class that identifies a keyed collection in a model. The data item that bears the foreignKeyTo attribute contains as its value a key to the identified keyed collection.

healthStatus: A class that enumerates the states of health of a device.

interpretation: A class that identifies a broad class of data types to which an item of data belongs.

NOTE – In an EDS, this term is the 'interpretation' attribute of a data item. Use cases include determining how to interpret the semantic type string for a data item, distinguishing an index from a count, and distinguishing a transformation from a position.

memberOf: A referential attribute for referencing individuals defined in an EDS.

NOTE – An element bearing this semantic attribute is a member of the class in a standard or user-provided model of operation, named by the value of the attribute. Standard models of operation cannot efficiently represent the variety of details that may occur in a specific instrument. To address this difficulty, the parts of a standard model of operation, which are classes, can be interpreted as a skeletal

model. Individuals in the classes of a standard model of operation can be defined in an EDS, and can be related to those classes by the referential attribute 'memberOf'. This convention allows EDS to define a variety of operating models without needing a user-provided ontology, which would diminish their portability.

operatingMode: A class that enumerates the modes of operation of a device or of a software component, which can simply be turned on or off.

purpose: A class that specifies how to apply an instance of a semantic type.

NOTE – One may think that the purpose of a semantic type derives from its context of usage, for example, thinking that a parameter of a command must be a set-point. However, measured values that are not set-points may also be parameters of a command. Both set-points and measured values may appear in published data. The purpose may be clear for simple commands and telemetry that set a data item in a device; in those cases, the purpose attribute can be left implicit. In some housekeeping telemetry the distinction is not so clear; both set points and measurements may appear in the same relation; in those cases, the purpose attribute must be explicit on each data item. In an electronic data sheet, this term is the 'purpose' attribute of a data item. Use cases include distinguishing a set-point from a measurement to be controlled, distinguishing a central value from a measured value, and distinguishing an adjustment from a measurement.

referenceFrame: A class that identifies the frame of reference in which a semantic type is relevant.

NOTE – The generic names of frames, such as 'device' and 'vehicle', are employed here and do not imply a specific coordinate system. For example, 'ECI' is a value of this enumeration, while the more specific 'J2000', 'TOD', and 'MOD' are reserved for the 'coordinate type' enumeration. In an EDS, this term is the 'referenceFrame' attribute of a data item. Use cases include determining the frame in which a semantic type is represented in order to determine what transformation is needed, if any, to represent the semantic type in a frame that is appropriate for an application and determining the frame of reference that is input to a transformation of coordinates.

relationToTimestamp: A class that specifies the age of the other data items in relation to the time stamp.

NOTE – When a time stamp appears in a relation with other data items, it may be important to a designer of a control system to know how the time stamp relates to the age of the other data items in the relation. In an EDS, this term is the timeRelation attribute of a time stamp data item. Use cases include enabling a design of a control system to estimate the delay between measurement of a data value and the time when the control system obtains the data item.

starTrackerOperatingMode: A class that enumerates the modes of operation of a star tracker.

statusConvention: A class that describes how a data item represents the quality of function of a device.

subject: A referential class that references a component of the system in which the data item using the subject property appears.

NOTE – The data item applies to the component. For example, if the purpose property of the data item is 'measurement', then the data item is a measurement of the component referenced in the subject property.

toCoordinateType: A class that identifies the coordinate type that results from a transformation from the coordinate system identified by the coordinateType.

toFrame: A class that identifies the frame of reference that results from a transformation from the frame identified by the referenceFrame.

transformationType: A class that indicates how to interpret a semantic type that is an array of numbers.

NOTE – The arrays to which the transformation type applies may be interpreted as a transformation from one frame to another, such as quaternions and direction cosine matrices. For arrays that may be interpreted as locations in a phase space, such as position and velocity vectors, the coordinate type applies. A coordinate type that is a vector may be treated as a transformation by adding it to other vectors; for example, such a transformation would be a called a translation. The 'transformation type' term specifies a particular combination of the following terms: coordinate system, array size, unit of measure, and computation. The definition of a value of the 'transformation type' term must provide the information about the terms listed above. The 'reference frame' and 'to frame' must be specified with the transformation type in an EDS when defining an item of data. In an EDS, this term is the 'transformationType' attribute of a data item. Use cases include interpreting the quaternion that is emitted by a star tracker.

1.5.2.3 Instances

SpaceWireAlwaysEnabled: An instance of SpaceWireLinkEnablement and ModelOfOperation.

SpaceWireAutoStart: An instance of SpaceWireLinkEnablement and ModelOfOperation.

SpaceWireOutOfBandEnable: An instance of SpaceWireLinkEnablement and ModelOfOperation.

SpaceWireRMAPWriteAcknowledged: An instance of SpaceWireRMAPWriteSupport and ModelOfOperation

SpaceWireRMAPWriteVerified: An instance of SpaceWireRMAPWriteSupport and ModelOfOperation.

SpWLogicalAddressRange: A class that defines a range of logical addresses that can be used to address an endpoint on a SpaceWire link. The syntax is low value followed by high value, separated by a colon. The values are unsigned integers that can fit in an octet. Instance of SpecificValue.

SpWProtocolId: A class that constrains the content and sequencing of data sent and received through a SpaceWire link. One protocol id should be specified per interface, and the interface describes the content of data sent and received on a SpaceWire link. The state machine of the component that interacts across the interface describes the sequencing of data sent and received. The format of a protocol id is an unsigned integer that fits in an octet, or one of the standard protocol names in {RMAP, CPTP}. Instance of SpecificValue and SpaceWireParameter.

SpWRMAPAddressRange: A class that identifies a range of memory addresses in an endpoint that are accessible through the RMAP protocol. The format of this specification is a list of memory ranges separated by commas. Each memory range consists of a low value, followed by a high value, followed by a memory identifier, followed by a SpaceWire RMAP write support code, with colons between. The values and the memory identifier are unsigned integers. The SpaceWire RMAP write support code is a string in {"ack", "ver"}. Instance of SpecificValue and SpaceWireParameter.

SpWReceiveRate: A class that defines a range of transfer rates at which an endpoint can receive data through a link. The syntax is low value followed by high value, separated by a colon. The values are integers, in units of Mbit/sec. Instance of SpecificValue and of SpaceWireParameter.

SpWTransmitRate: A class that defines a range of transfer rates at which an endpoint can send data through a link. The syntax is low value followed by high value, separated by a colon. The values are integers, in units of Mbit/sec. SpWTransmitRate is an instance of SpecificValue and SpaceWireParameter.

DummyEncodingAndPrecision: A dummy item to avoid an empty enumeration in a derived schema. Instance of DoTEncodingAndPrecisionType and ExtensibleEnumeration.

NOTE – Do not use.

DummyEncodingRules: A dummy item to avoid an empty enumeration in a derived schema. Instance of ExtensibleEnumeration and ExtensibleEnumeration.

NOTE – Do not use.

CRC16: (See ITU-T Recommendation X.25, p. 9, section 2.2.7.4, International Telecommunications Union, October 1996.) Instance of ExtensibleEnumeration and DoTErrorControlType.

CRC32:
x^{{32}}+x^{{26}}+x^{{23}}+x^{{22}}+x^{{16}}+x^{{12}}+x^{{11}}+x^{{10}}+x^{8}+x^{7}+x^{5}+x^{4}+x^{2}+x+1. Instance of ExtensibleEnumeration and DoTErrorControlType.

CRC32C: (See Castagnoli, G., Brauer, S., and M. Herrmann, "Optimization of Cyclic Redundancy-Check Codes with 24 and 32 Parity Bits", IEEE Transact. on Communications, Vol. 41, No. 6, June 1993.) An instance of ExtensibleEnumeration and of DoTErrorControlType.

SDNV: A self-delimiting numeric value encoding scheme adopted by Bundle Protocol in IETF RFC 5050. Instance of ExtensibleEnumeration and DoTIntegerEncodingType.

NOTE – From IETF RFC 5050: 'The SDNV encoding scheme is closely adapted from the Abstract Syntax Notation One Basic Encoding Rules for subidentifiers within an object identifier value. A SDNV is a numeric value encoded in N octets, the last of which has its Most Significant Bit (MSB) set to 0; the MSB of every other octet in the SDNV must be set to one. The value encoded in an SDNV is the unsigned binary number obtained by concatenating into a single bit string the 7 least significant bits of each octet of the SDNV'.

DummyMathOperator: A dummy item, to avoid an empty enumeration in a derived schema. DummyMathOperator is an instance of ExtensibleEnumeration and of DoTMathOperatorsType.

NOTE – Do not use.

CBORbytes: An encoding that uses IETF RFC 7049 with major type 2 to encode a string of bytes. Instance of ExtensibleEnumeration and DoTStringEncodingType.

VCHAR: Visible ASCII characters 0x21 through 0x7e. (See IETF RFC 5234.) Instance of ExtensibleEnumeration and DoTStringEncodingType.

fractionOfMaximum: A number in which the value 0 represents the minimum value of a scale and the value 1 represents the maximum value of the scale. The minimum and maximum values of the scale must be explained separately. Instance of UnitsNotYetIntegratedIntoQUDV.

leftHanded: A class that specifies a coordinate system that has three dimensions and that the order of the presentation of positive values of the dimensions in a vector corresponds to the directions of thumb, index, and medius fingers of a human left hand. Instance of chirality.

rightHanded: A class that specifies a coordinate system that has three dimensions and that the order of the presentation of positive values of the dimensions in a vector corresponds to the directions of thumb, index, and medius fingers of a human right hand. Instance of chirality.

J2000: A value of the 'coordinate type' term that specifies the following information for a semantic type: coordinate system = Cartesian, array size = 3, unit of measure = length, reference frame = ECI. The elements of the vector are interpreted as Cartesian coordinates. This approximately inertial coordinate frame has its origin at the center of the Earth. It is defined with the Earth's mean equator and mean equinox at 12:00 Terrestrial Time on 1 January 2000. The x axis points toward the mean equinox corresponding to northern spring. The z axis points north along the Earth's spin axis. The y axis is 90 degrees east of the x axis in the plane

of the equator, such that it forms a right-handed coordinate system. The ‘unit of measure’ must be specified with this coordinate type. Use cases include specifying that a semantic type is a vector with a coordinate system that corresponds to ECI J2000. Instance of coordinateType and toCoordinateType.

LLA: A value of the ‘coordinate type’ term that specifies the following information for a semantic type: coordinate system = latitude, longitude, altitude; array size = 3; unit of measure = degree, degree, meter; reference frame = ECEF. The elements of the vector are interpreted as spherical coordinates, and the two angle coordinates are treated as latitude and longitude; the radial coordinate is 0 at the radius of Earth’s mean sea level. This coordinate system has its origin at the intersection of the prime meridian and the equator. Latitude is often marked north or south, instead of using negative angles. Similarly, angular distance around the equator is often marked as east or west. Use cases include specifying the semantic type of a vector location of a building on Earth that is a target of surveillance. Instance of coordinateType and toCoordinateType.

MOD: A value of the ‘coordinate type’ term that specifies the following information for a semantic type: coordinate system = Cartesian, array size = 3, unit of measure = length, reference frame = ECI. The elements of the vector are interpreted as Cartesian coordinates. This approximately inertial coordinate frame has its origin at the center of the Earth. It is defined with the Earth’s mean equator and mean equinox at a particular date, assumed here to be the time of the measurement. The x axis points toward the mean equinox corresponding to northern spring. The z axis points north along the Earth’s spin axis. The y axis is 90 degrees east of the x axis in the plane of the equator, such that it forms a right-handled coordinate system. The ‘unit of measure’ must be specified with this coordinate type. Use cases include specifying that a semantic type is a vector with a coordinate system that corresponds to ECI MOD at the time of measurement. Instance of coordinateType and toCoordinateType.

TOD: A value of the ‘coordinate type’ term that specifies the following information for a semantic type: coordinate system = Cartesian, array size = 3, unit of measure = length, reference frame = ECI. The elements of the vector are interpreted as Cartesian coordinates. This approximately inertial coordinate frame has its origin at the center of the Earth. It is defined with the Earth’s true equator and equinox at a particular date, assumed here to be the time of the measurement. The x axis points toward the true equinox corresponding to northern spring. The z axis points north along the Earth’s spin axis. The y axis is 90 degrees east of the x axis in the plane of the equator, such that it forms a right-handed coordinate system. The ‘unit of measure’ must be specified with this coordinate type. Use cases include specifying that a semantic type is a vector with a coordinate system that corresponds to ECI TOD at the time of measurement. Instance of coordinateType and toCoordinateType.

UPS: A value of the ‘coordinate type’ term that specifies the following information for a semantic type: coordinate system = latitude, longitude; array size = 2; unit of measure = degree; reference frame = ECEF. The elements of the vector are interpreted as universal polar stereographic coordinates. Use cases include specifying that a semantic type is a vector location on Earth, with a coordinate system that corresponds to universal polar stereographic coordinates. Instance of coordinateType and toCoordinateType.

UTC: A value of the ‘coordinate type’ term that specifies that a number is coordinated universal time as defined by the International Telecommunication Union and the International Astronomical Union. The abbreviation is a compromise between ‘coordinated universal time’ and ‘temps universel coordonné’. Use cases include representing the time obtained from a global positioning system and representing the time obtained from an atomic clock. Instance of coordinateType and toCoordinateType.

UTM: A value of the ‘coordinate type’ term that specifies the following information for a semantic type: coordinate system = latitude, longitude; array size = 2; unit of measure = degree; reference frame = ECEF. The elements of the vector are interpreted as universal transverse Mercator coordinates. Use cases include specifying that a semantic type is a vector location on Earth with a coordinate system that corresponds to universal transverse Mercator coordinates. Instance of coordinateType and toCoordinateType.

latLon: A value of the ‘coordinate type’ term that specifies the following information for a semantic type: coordinate system = latitude, longitude; array size = 2; unit of measure = degree; reference frame = ECEF. The elements of the vector are interpreted as spherical coordinates, but only the two angle coordinates are used; the radial coordinate is fixed at the mean radius of Earth. This coordinate system has its origin at the intersection of the prime meridian and the equator. Latitude is often marked north or south instead of using negative angles. Similarly, angular distance around the equator is often marked as east or west. Both of these directional conventions must be specified if a device uses them; the term is ‘latlon convention’. Use cases include specifying the semantic type of a vector location of a ship at sea on Earth. Instance of coordinateType and toCoordinateType.

comparison: A class that identifies the difference convention for variables without a specific absolute reference frame. Use cases include specifying that the voltage reported by a voltage meter is a difference in potential without a reference ground. Instance of differenceConvention.

delta: A class that identifies a difference between the value of a variable at one time compared to the value at a prior time. For measurements, the prior time is the time that the variable was last reported, or since initialization. For commanded values, the prior time is the time of the current command. Use cases include identifying the change in speed reported by an accelerometer, identifying the elapsed time since the last report of a measurement from a sensor, and specifying the time during which a command parameter is valid. Instance of differenceConvention.

gradient: A class that identifies a difference computed as the ratio of a delta variable to a path variable. Such a variable is an average obtained by dividing the differences over the same reporting interval. Use cases include identifying the rate of change of atmospheric density with altitude at a particular time and place. Instance of differenceConvention.

path: A class that identifies a difference between the value of a variable at one point on a path and the value at a prior point on the path. The path is the continuously changing location and orientation of the device frame for the variable relative to the reference frame for the variable. The prior point is the location at the time of the prior report or command. Use cases include

identifying the change in the Earth's magnetic field reported by a magnetometer during the flight of the vehicle on which it is mounted. Instance of differenceConvention.

ratio: A class that indicates that a variable is a quotient of two quantities. Instance of differenceConvention.

sinceEpoch: A class that identifies the amount of time that has elapsed since a calendar epoch. The data item it describes is a date and time. Use cases include providing a time stamp on housekeeping data and specifying the time when an action should be performed. Instance of differenceConvention.

sinceSync: A class that identifies the amount of time that has elapsed since the last synchronization signal. Use cases include identifying the time at which a measurement was reported by a star tracker relative to the last synchronization sent to the star tracker. Instance of differenceConvention.

healthGood: A class that indicates that a device is in good health is operating correctly, with no errors detected. Instance of enumeration and healthStatus.

healthSevereWarning: A class that indicates that errors have been detected and reset is recommended. Instance of enumeration and healthStatus.

healthWarning: A class that indicates that some errors have been detected but reset is not required. Instance of enumeration and healthStatus.

notOperational: A class that indicates that the device is not operating. Instance of enumeration. Instance of operatingMode, starTrackerOperatingMode, and tecOperatingMode.

operational: A class that indicates that the device is operating. Instance of enumeration and operatingMode.

starTrackerAcquiring: A class that indicates that the star tracker is comparing the image in its focal plane to known configurations of stars but has not yet recognized the image. Instance of enumeration and starTrackerOperatingMode.

starTrackerReportingRates: A class that indicates that the star tracker is reporting the angular rate vector. This condition implies that the star tracker is tracking or acquiring. Instance of enumeration. Instance of starTrackerOperatingMode.

starTrackerTracking: A class that indicates that the star tracker has recognized the configuration of stars in its image, and is reporting its attitude. Instance of enumeration and starTrackerOperatingMode.

controlling: A class that indicates that the controller is actively attempting to achieve its set point. Instance of enumeration. Instance of tecOperatingMode.

inBand: A class that indicates that the controller is within the specified tolerance of its set point. Instance of enumeration and tecOperatingMode.

command: A class that specifies the verb form of a command. Instance of interpretation.

difference: A class that represents a difference between one physical quantity and another on the same scale. Instance of interpretation.

mode: A class that represents a mode of operation of a device. Instance of interpretation.

name: A class that specifies names of objects or variables in the management model of a system. Instance of interpretation.

orbit: A class that defines an orbit. Instance of interpretation.

position: A class that represents a position in phase space. Instance of interpretation.

scope: A class that represents a region or subset of a device. Instance of interpretation.

status: A class that represents a status, such as an error condition, of a device. Instance of interpretation.

transformation: A class that represents a change of coordinate systems. Instance of interpretation.

checksum: A compressed form of a larger string of bits. A checksum cannot be decompressed to recover the larger string. A checksum can be transmitted with the larger string in order to detect errors in transmission. If both the receiver and the sender apply the same compression algorithm to the larger string, they should expect to obtain the same checksum value. If the receiver finds a different checksum value than was transmitted with the larger string, then a transmission error is likely. Instance of interpretation.

validityPeriod: An interpretation that can be assigned to a time duration argument in a command in order to define the period of time allowed for execution of the command. Instance of interpretation.

action: A semantic type representing a specific amount of action to be applied to the real world through an actuator. Use cases include distinguishing a torque to be applied to a reaction wheel from a stress measured by a strain gauge nearby. Instance of purpose.

calibration: A semantic type representing a number that is a measurement obtained through a calibration procedure. It is a value obtained specifically for each instance of a device. Use cases include distinguishing an adjustment from a measurement. Instance of purpose.

measurement: A semantic type representing a measurement of the real world obtained by a sensor. Use cases include distinguishing a set-point from a measurement to be controlled, distinguishing a central value from a measured value, and distinguishing an adjustment from a measurement. Instance of purpose.

nominal: A semantic type representing a number that is a non-specific central value over all instances of a device that result from a reproducible manufacturing process. Use cases include distinguishing a central value from a measured value. Instance of purpose.

setpoint: A semantic type representing a value to be achieved by a control system or a setting to be applied to a component. Use cases include distinguishing a set-point from a measurement to be controlled. Instance of purpose.

Earth-centered, Earth fixed; ECEF: A class that identifies the frame that rotates with the Earth to describe semantic types that represent data about the surface of the Earth. This reference frame must be accompanied by specification of coordinate type when used to describe a data item in an EDS. Use cases include specifying the frame of latitude and longitude measurements and specifying an approximate frame of geosynchronous spacecraft. Instance of referenceFrame and toFrame.

Earth-centered inertial, ECI: The inertial frame of the Earth, generally based on the positions of remote stellar objects. This reference frame is only approximately inertial because of the acceleration of Earth to maintain its orbit. This reference frame must be accompanied by specification of coordinate type when used to describe a data item in an EDS. Use cases include specifying the frame of the remote stars that form the basis for attitude measurement by a star tracker and specifying the location of a vehicle in interplanetary flight. Instance of referenceFrame and toFrame.

local vertical, local horizontal; LVLH: The vehicle frame of an orbiter. The LVLH frame typically uses a Cartesian coordinate type oriented so that the x axis points in the direction of flight, the z axis points toward nadir, and the y axis points to the right in the viewpoint of a person looking in the direction of flight, to make a right-handed rectangular coordinate system. Use cases include specifying the frame from which a surveillance satellite observes the Earth. Instance of referenceFrame and toFrame.

device: A class that identifies the frame of an instrument to describe semantic types that represent measurements or actuations in the frame of a device. Use cases include specifying the frame of torque of a reaction wheel, specifying the frame of rotation in the transformation reported by a star tracker, and specifying the frame of measurement of a coarse sun sensor. Instance of referenceFrame and toFrame.

externalObject: A class that identifies an external object. Use cases include exchanging position and attitude information with an approaching vehicle during docking, and distinguishing the location of a ship at sea from the nadir point of an orbiter. Instance of referenceFrame and toFrame.

NOTE – The data managed by the command and data handling system of a vehicle often contains information about other objects, such as targets of surveillance or approaching vehicles. An identifier of the external object should appear in the same aggregation relation with this reference frame, or in the electronic data sheet description of a data item.

mount: The reference frame associated with the face of a bracket or mounting point (when no bracket is present) that positions and orients an instrument relative to a spacecraft to which it is attached. In order to transform instrumentation measurements and actuations between device frame and vehicle frame, it is necessary to know the mounting frame as an intermediate step. Use cases include specifying the frame in which a reaction wheel is mounted to the structure of a vehicle and specifying the frame in which a star tracker is mounted to the structure of a vehicle. Instance of referenceFrame and toFrame.

transducer: A special variation on the device reference frame. Use cases include specifying the frame of a coarse sun sensor in a device that contains multiple coarse sun sensors arranged to face in non-orthogonal directions and specifying the frame of each of the optical axes of a star tracker with multiple sights. Instance of referenceFrame and toFrame.

NOTE – Some instruments contain multiple transducers. For example, a three-axis magnetometer may consist of three one-axis magnetometers. Usually, the one-axis transducers are arranged parallel to the coordinate axes of the device frame, so no transformation is needed to transform from measurement frame to mount frame. However, some devices may have multiple transducers that are not so fortuitously positioned. In that case, it is necessary to be able to refer to the reference frame of the transducers of a device.

vehicle: The base from which a spacecraft conducts its mission. Use cases include controlling the direction in which an articulated antenna is pointing relative to the attitude of the vehicle. Instance of referenceFrame and toFrame.

NOTE – Relations between sensors and actuators are computed through the vehicle frame.

acquiring: A class that indicates that the timestamp is simultaneous with the acquisition of a datum. Instance of relationToTimestamp.

generation: A class that indicates that the timestamp is simultaneous with the generation of a datum. Instance of relationToTimestamp.

sending: A class that indicates that the timestamp is the time when the message that contains it was sent. Instance of relationToTimestamp.

binaryStatus: A class that indicates whether a device is functioning correctly or erroneously. The data item has the value 0 if no error is being detected, and it has the value 1 if an error is present. Typically, the binary representation of such a variable would be bit or unsigned integer. Use cases include the SOIS accelerometer interface uses this convention to report device status. Instance of statusConvention.

qualityFraction: A status convention that indicates a fractional quality of service, with 0 representing complete failure and 1 representing completely successful operation. The method of computing this value is an algorithm that varies from device to device and is explained in natural language in an unstructured text field in an EDS. Use cases include specifying the quality of data produced by a star tracker. Instance of statusConvention.

ternaryStatus: A status convention that adds a warning state to the binary status. The three states follow: OK – no error detected, Warning – action (such as restart) is not needed, and Severe – action (such as restart) is needed. Use cases include distinguishing minor errors from major errors. Instance of statusConvention.

DCM: A value of the 'transformation type' term that specifies the following information for a semantic type: coordinate system = rectangular, array size = (n, n), row = major order, unit of measure = not applicable. The transformation is computed by matrix multiplication. The elements of the matrix are interpreted as direction cosines. Instance of transformationType.

quaternion: A value of the 'transformation type' term that specifies the following information for a semantic type: coordinate system = Cartesian, tarray size = 4, unit of measure = not applicable. The transformation is computed by conjugation. The four coordinates are a scalar first, followed by three vector coordinates, corresponding to the orthogonal Cartesian coordinate axes x, y, and z, with the same chirality as the coordinate system. Chirality of the coordinate system must be specified in an EDS when using this term. Instance of transformationType.

rollPitchYaw: A value of the 'transformation type' term that specifies the following information for a semantic type: coordinate system = Cartesian, right-handed; tarray size = 3; unit of measure = degree. The transformation is computed by constructing a rotation based on the description below. The three coordinates are a variant of Euler angles, based on the convention that the Cartesian coordinates of the vehicle frame are aligned with LVLH. The three coordinates are not distances in this coordinate system; rather, they are rotation angles around the axes of the coordinate system. Roll is rotation around the direction of flight (x). Pitch is rotation around the y axis. Yaw is rotation around the z axis. All these rotations use the right-handed convention for sign. Instance of transformationType.

rotationRate: A value of the 'transformation type' term that specifies the following information for a semantic type: coordinate system = Cartesian, tarray size = n, unit of measure = angle/time. The transformation is generally not computed because this vector is often used directly as a measurement and as a set point for an attitude control system. The vector represents the rate of rotation of the 'to frame' relative to the 'reference frame'. The elements of the vector are the angular rates around each axis of the reference frame. Chirality of the coordinate system must be specified in an EDS when using this term. Unit of measure for the rate must also be specified in an EDS when using this term. Instance of transformationType.

translation: A value of the 'transformation type' term that specifies the following information for a semantic type: coordinate system = Cartesian, array size = n, unit of measure = length. The transformation is computed by adding this vector to the vector in the reference frame. The elements of the vector are interpreted as changes in Cartesian coordinates. The vector is not a position but is a translation transformation that can be applied to position vectors by adding the translation coordinates to the position coordinates. Chirality of the coordinate system must be specified in an EDS when using this term. The unit of measure must also be specified in an EDS when using this term. Instance of transformationType.

velocity: A value of the 'transformation type' term that specifies the following information for a semantic type: coordinate system = Cartesian, array size = n, unit of measure = length/ time. The transformation is generally not computed because this vector is often used directly as a measurement and as a set point for a navigation system. The vector represents the rate of translation of the 'to frame' relative to the 'reference frame'. The elements of the vector are the linear speeds along each axis of the reference frame. Chirality of the coordinate system must be specified in an EDS when using this term. Unit of measure for the speed must also be specified in an EDS when using this term. Instance of transformationType.

1.6 NOMENCLATURE

1.6.1 NORMATIVE TEXT

The following conventions apply for the normative specifications in this Recommended Practice:

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

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

c) the word 'may' implies an optional specification;

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

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

1.6.2 INFORMATIVE TEXT

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

– Overview;

– Background;

– Rationale;

– Discussion.

1.7 REFERENCES

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

[1] *Spacecraft Onboard Interface Services—XML Specification for Electronic Data Sheets*. Issue 1. Recommendation for Space Data System Standards (Blue Book), CCSDS 876.0-B-1. Washington, D.C.: CCSDS, April 2019.

[2] Tim Bray, et al., eds. *Extensible Markup Language (XML) 1.0*. 5th ed. W3C Recommendation. N.p.: W3C, 26 November 2008.

[3] Shudi (Sandy) Gao, C. M. Sperberg-McQueen, and Henry S. Thompson, eds. *W3C XML Schema Definition Language (XSD) 1.1 Part 1: Structures*. Version 1.1. W3C Recommendation. N.p.: W3C, 5 April 2012.

[4] David Peterson, et al., eds. *W3C XML Schema Definition Language (XSD) 1.1 Part 2: Datatypes*. Version 1.1. W3C Recommendation. N.p.: W3C, 5 April 2012.

[5] W3C OWL Working Group, ed. *OWL 2 Web Ontology Language Document Overview*. 2nd ed. W3C Recommendation. N.p.: W3C, 11 December 2012.

[6] "Dublin Core Annotation Properties." http://protege.stanford.edu/plugins/owl/dc/protege-dc.owl.

[7] *Spacecraft Onboard Interface Services—Subnetwork Packet Service*. Issue 1. Recommendation for Space Data System Practices (Magenta Book), CCSDS 851.0-M-1. Washington, D.C.: CCSDS, December 2009.

[8] *Spacecraft Onboard Interface Services—Subnetwork Memory Access Service*. Issue 1. Recommendation for Space Data System Practices (Magenta Book), CCSDS 852.0-M-1. Washington, D.C.: CCSDS, December 2009.

[9] *Spacecraft Onboard Interface Services—Subnetwork Synchronisation Service*. Issue 1. Recommendation for Space Data System Practices (Magenta Book), CCSDS 853.0-M-1. Washington, D.C.: CCSDS, December 2009.

[10] *Spacecraft Onboard Interface Services—Subnetwork Device Discovery Service*. Issue 1. Recommendation for Space Data System Practices (Magenta Book), CCSDS 854.0-M-1. Washington, D.C.: CCSDS, December 2009.

[11] *Spacecraft Onboard Interface Services—Subnetwork Test Service*. Issue 1. Recommendation for Space Data System Practices (Magenta Book), CCSDS 855.0-M-1. Washington, D.C.: CCSDS, December 2009.

[12] "Spacecraft Onboard Interface Services Electronic Data Sheets and Dictionary of Terms." Space Assigned Numbers Authority. https://sanaregistry.org/r/sois/.

[13] "Quantities, Units, Dimensions, Values (QUDV)." Object Management Group. http://www.omgwiki.org/OMGSysML/doku.php?id=sysml-qudv:quantities_units_dimensions_values_qudv.

NOTE – Informative references are contained in annex D.

2 OVERVIEW

2.1 INTRODUCTION

The SOIS DoT provides definitions of terms that are used in SEDS. This section is a brief overview of the relationship between DoT and SEDS, arranged in the following topics:

- the subject matter of SEDS;
- using and storing SEDS;
- the mechanism that associates DoT content with SEDS content;
- pre-existing standards applied in this document;
- the content of the DoT;
- how the DoT relates to SEDS metadata;
- how to maintain the DoT.

Figure 2-1 is a map of the major artefacts for DoT and SEDS and the relationships among them. The topics this document focuses on are highlighted in blue.

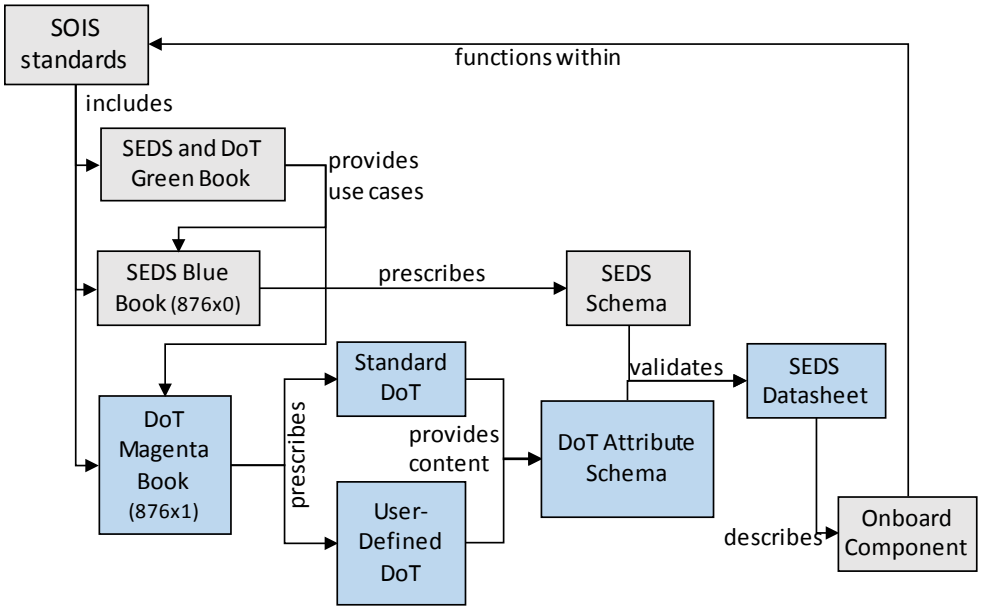


Figure 2-1: Major Concepts of This Book, and Their Relationships

A SEDS specifies access to an onboard device or software component using SOIS standards. The XML schema used to validate such a datasheet is defined in two parts:

- the SEDS schema is fixed and described in reference [1];
- the DoT attribute schema is extensible, and comes from combining the standard DoT, and optionally a user-defined DoT, as described in this document.

2.2 THE SUBJECT MATTER OF ELECTRONIC DATA SHEETS

The SEDS are defined within the context of the overall SOIS architecture (see reference [D2]). Figure 2-2 shows how SEDS describe data interfaces at various points in a spacecraft data system. Devices appear on the left side, with increasing degrees of aggregation of data interfaces in moving to the right side of the diagram.

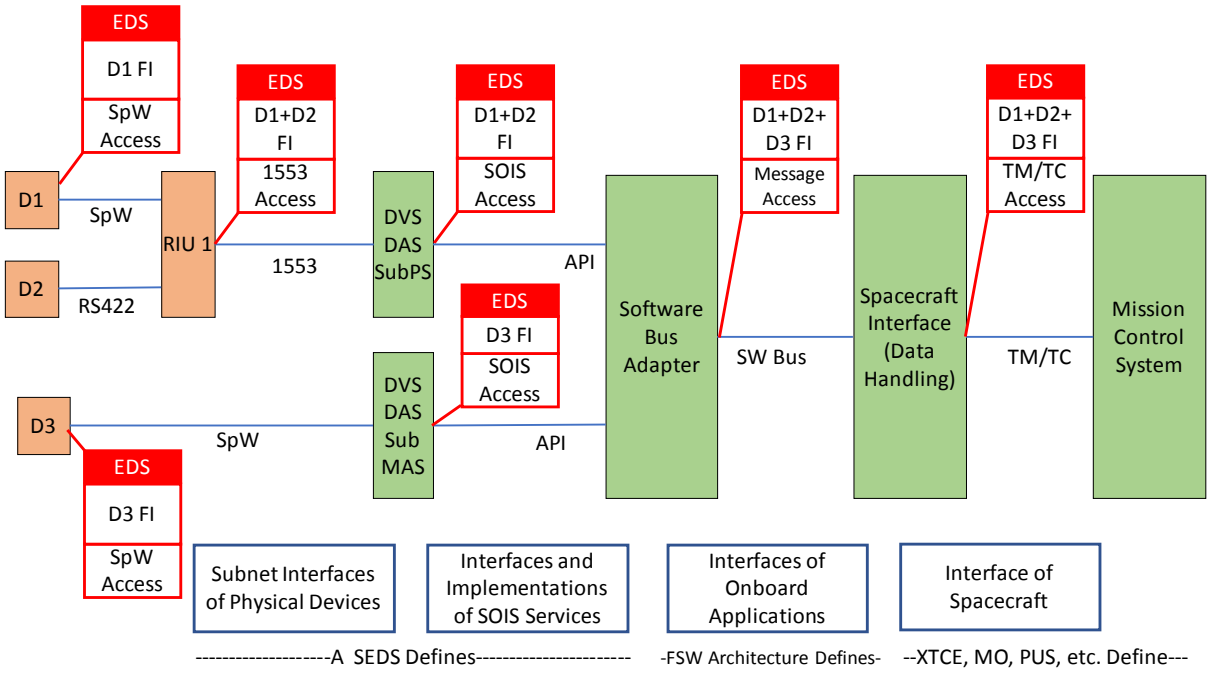


Figure 2-2: SEDS Describe Data Interfaces in a Spacecraft

A SEDS describes the format of information in a data interface for an onboard device accessed using the Packet, Memory Access, and Synchronization Services of the Subnetwork Layer, as illustrated in figure 2-3.

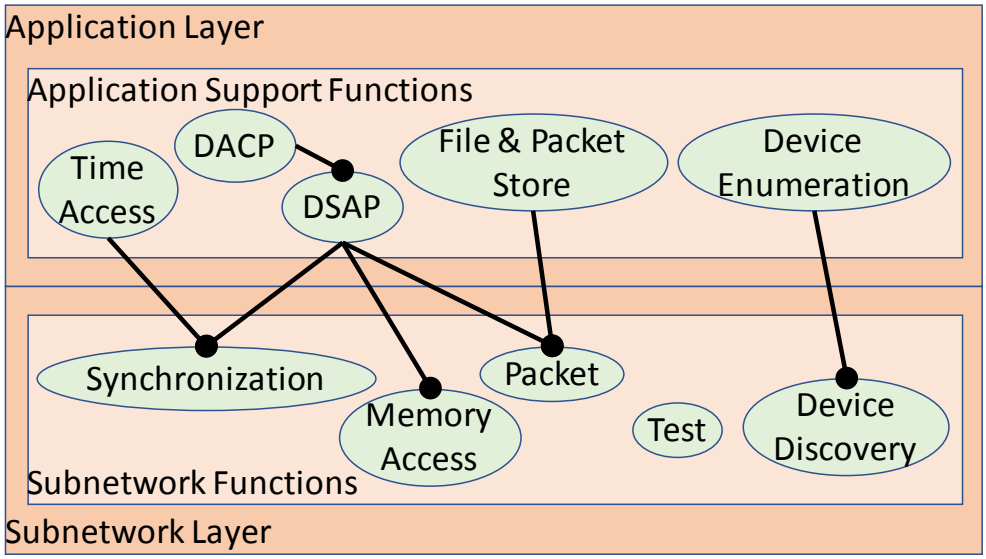


Figure 2-3: Described by SEDS

The Subnetwork Layer provides standard services mapped onto subnetwork-specific protocols to send and receive discrete packets (reference [7]), access remote memory (reference [8]), synchronize with the subnetwork (reference [9]), and discover and test devices on the subnetwork (references [10] and [11], respectively).

Some aspects of a device data interface may correspond to standardized protocols directly usable at the Application Layer. The remainder is specified by the SEDS for that device. This captures all device-specific aspects, including those specified at some other level of commonality (e.g., agency, company, product line).

The interfaces defined in a datasheet may be used in the specification or implementation of applications and Application Support Layer services.

2.3 PURPOSE AND FUNCTION OF SOIS ELECTRONIC DATA SHEETS

When a manufacturer produces a component, it can provide information about the usage of the component in the form of a SEDS. By using a structured document, which is accessible to algorithms, a manufacturer facilitates the integration of its product into a space system. A space agency can use its own toolchain to assure successful adaptation of the component, given the SEDS for the component.

A SEDS is intended to be a machine-interpretable mechanism for describing devices which may be accessed using the SOIS DAS and DVS. The SEDS is intended, in its fullest form, to replace the traditional user manuals, specifications, and data sheets that accompany a device

and are necessary to determine the operation of the device and how to communicate with it. The function and syntax of SEDS is described in reference [1]. The DoT provides the formal vocabulary for SEDS, enabling the functions listed below. The indexes and other functions mentioned in the list are not specified by this document; rather, they are phenomena that are expected to develop as engineers and entrepreneurs write software to exploit the machine-readable information that will be available in SEDS. Prior to SEDS technology, information about quantities, units, dimensions, values, provenance, and usage of data was informal and therefore inaccessible for these functions. A more complete discussion of these use cases appears in the SEDS and DoT Green Book (reference [D4]).

2.4 USE CASES FOR DoT

Use cases for DoT include:

- assuring that the terms used in SEDS have clear meanings;
- enabling a market index to components based on their interfaces;
- enabling an index to components in a vehicle, based on their interfaces;
- enabling an index to data in the spacecraft database;
- enabling lookup of topics published and subscribed on a software bus;
- enabling matching of interfaces of two components during design or during adaptive reconfiguration so the two components can interact through the mutual interface, one component acting as a provider of data and services through the interface, and the other acting as a consumer of data and services through the interface;
- enabling matching of interfaces of a device with interfaces of simulation models, to configure and to validate testing and simulation software;
- providing a glossary for human-readable documentation generated from SEDS.

2.5 HOW THE DICTIONARY OF TERMS RELATES TO ELECTRONIC DATA SHEETS

A schema determines the syntax of SEDS, which are XML files. The schema consists of a base syntax and semantic inclusions. The base schema is described in the companion volume, *Spacecraft Onboard Interface Services—XML Specification for Electronic Data Sheets* (reference [1]). The base schema controls how a SEDS describes the syntax of data that passes across the data interfaces of a component. The DoT provides additional terms that go beyond syntax to describe the suitability of data items for usage in applications. Because these terms are ‘beyond syntax’, they are referred to as ‘semantic’ attributes of data. This section describes how the DoT provisions the SEDS schema with semantic attributes.

The SANA DoT is a normative specification of terms to be used in SEDS instances. The SANA DoT is the ontology published at reference [12]. This document is not the SANA DoT. This Recommended Practice specifies the basic structure, usage, and maintenance of the SANA DoT.

The SANA DoT stores information in an ontological syntax that facilitates management of terminology. That information is extracted in the form of an XML schema fragment, which is then included in the SEDS schema. The schema fragment defines semantic attributes that can be applied to items of data, as described in this document (see figure 2-4).

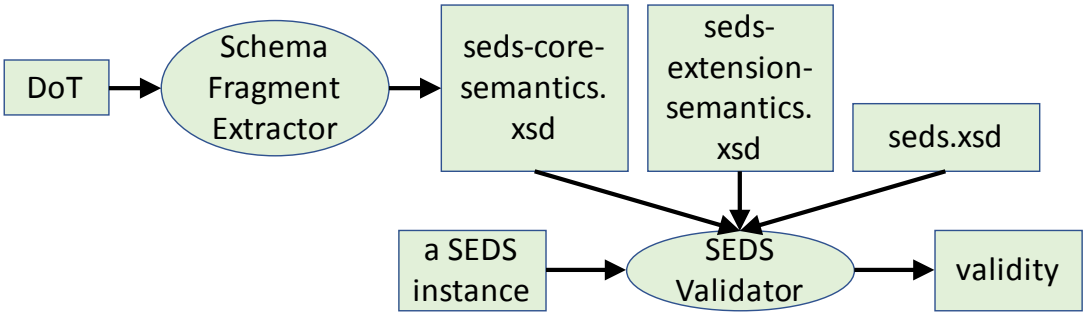


Figure 2-4: How the DoT Provisions the SEDS Schema

In figure 2-4, a schema fragment extractor reads the DoT and constructs a schema fragment named seds-core-semantics.xsd. A project may provide its own additional schema fragment named seds-extension-semantics.xsd, either by writing it directly or by generating it from a project ontology. The standard SEDS schema, named seds.xsd, includes the two schema fragments. A standard validating XML reader uses the combined schema to validate a SEDS instance.

In order to be able to relate the elements of a data sheet to concepts, and to promote standardization and interoperability, the SANA DoT provides a core ontology for data sheet authors and users. These semantic terms effectively form part of the language that is used to write SEDS. The file seds-core-semantics.xsd in figure 2-4 carries these standard, interoperable terms.

The SEDS schema supports the use of semantic attributes to annotate the SEDS elements that have a ‘name’ attribute. These include the following elements:

- interfaces;
- commands;
- parameters;
- components;
- metadata;
- enumeration members.

When the semantics provided by the SANA DoT are insufficient, a data sheet author may utilize an additional user-defined schema fragment in a file named seds-extension-sematics.xsd, which can be used within a project. SANA provides an example extension schema file. This arrangement provides a standard, flexible, and extensible mechanism for capturing the semantics of component operation in a machine-interpretable form. The terms

in the extension schema can later be offered for integration into the SANA DoT. The use of an extension schema reduces the portability of a SEDS in order to be toolchain compatible in the project in which the extension ontology was developed. To become truly portable outside the original project, it is necessary to assimilate the extension terms into the SANA ontology and adapt the project SEDS to use the terms of the new SANA ontology.

The DoT can provide a mechanism for future extensions of the SEDS in cases in which the extension can be obtained by adding terms to the structure that is already present in the SEDS schema.

2.6 USE OF PRE-EXISTING STANDARDS

The specification and use of SEDS make use of the following World Wide Web Consortium (W3C) standards:

- XML—The Extensible Markup Language (reference [2]) is used to mark up data sheet documents in a machine-readable manner.
- XSD—The XML Schema Definition language (references [3] and [4]) is used to specify valid construction rules for data sheet documents. Version 1.1 of the XSD recommendation is used.
- OWL/RDF—In some cases, a data sheet author may wish to specify a user-defined DoT. This may be accomplished by accompanying the data sheet document with a DoT document specified according to the Web Ontology Language and using the syntax of the Resource Description Framework (reference [5]).

2.7 PRINCIPLES OF THE DICTIONARY OF TERMS

2.7.1 GENERAL

The DoT relies upon conceptual models to define terms in an ontology and enable successful interpretation of SEDS by people and by algorithms. A collection of descriptions of entities named by terms and their relationships constitutes an ontology. This section describes major models used in the DoT.

2.7.2 THE REPRESENTATION OF THE DOT

The DoT is a model-based vocabulary. This is a variation on the idea of model-based engineering, in which a central data model stores knowledge, and peripheral software tools render and exploit that knowledge in engineering documents.

The model selected for the DoT is an ontological description language known as Web Ontology Language (OWL). Tools for working with this language are available without monetary cost to the user, and commercial tools are also available. These tools provide graphical user interfaces for writing, interpreting, and validating associations between terms. As a bonus, it is also possible to invoke a reasoning program to infer implicit relations among terms, given the

relations that are present. The description logic sublanguage of OWL has adequate expressive capabilities for the use cases considered. OWL is a W3C standard, and it has a community of users. The syntax for the ontology files provided by SANA is OWL, represented in RDF/XML.

A SEDS instance accompanies the component that it describes; however, the DoT exists in a centrally accessible location (reference [12]) where it can be used by toolchain software to interpret SEDS.

In a model-based vocabulary, the model is the single source of information that is distributed through the medium of a variety of artefacts. Among those artefacts are a glossary of terms for humans to read and a schema of terms to be included in the SEDS schema. The latter was described in 2.5. The glossary is discussed below. Some toolchain elements may use the DoT directly, such as a hypothetical design-checking tool that checks the match between interfaces provided and required by components to assure that each data item is being used appropriately. Another useful feature of a model-based DoT is the capability to generate more than one form of schema fragment. There is a schema fragment that is appropriate for use by the SOIS EDS schema. There may be a need for similar fragments to be used in other descriptive tools, such as *XML Telemetric and Command Exchange* (reference [D3]). By using the same terminology model for both schemas, the usage of terms will be consistent across both spacecraft and ground systems.

2.7.3 MODEL OF MEASUREMENT

The DoT ontology is built on top of a proof-of-concept ontology called Quantities, Units, Dimensions, and Values (QUDV, reference [13]). The QUDV ontology provides a scheme for representing units of measure and relating them to quantity kinds. The DoT extends this scheme to cover more units of measure along with their quantity kinds. The extension will grow where new units of measure are needed.

The ability to specify quantity kinds can be useful in situations when specifying the unit of measure is insufficient to identify the physical property in a measurement, such as in the case of torque and energy, for which the units of measure are the same when reduced to base units. In such situations, there is an informal reliance upon convention in the use of derived units for disambiguation; for example, torque is often expressed as newton meters, while energy is often expressed as joules. The fact that torque is a vector is not always explicit when the axis of rotation is obvious to human engineers. These conventions may be widespread, but they are conventions that may not be accessible to algorithmic interpretation of SEDS. In contrast, the quantity kind property provides an explicit disambiguation, which is defined in the ontology.

A specification of a required interface can be viewed during design of a spacecraft as a search argument for a component that provides a compatible interface. Software components may have required interfaces that specify a unit of measure, but the available providers of such an interface may offer different units of measure. The information in the DoT ontology can be used by a toolchain to generate a shim that converts the units of measure provided by a service to that required by a software component that uses the service.

The following example explains a way in which an ontological model can check the usage of terms. Accidental misuse of the semantic attributes of items of data in an interface can be

detected. For example, a SEDS author might specify that a given item of data has quantity kind ‘length’ and unit of measure ‘arc-second’. The mistake here is that only certain units of measure are possible for a given quantity kind. The author is going to have to either change the quantity kind to ‘angle’ or choose a unit of measure that measures length, such as ‘meter’. The model for the terms contains this kind of information, so it can be enforced at some point in the validation of SEDS. In this example, the model contains an association between quantity kinds and their meaningful units of measure. At present, this check is not implemented in the SEDS schema, but it can be implemented in a toolchain that reads the DoT, if needed.

2.7.4 MODELS OF OPERATION

2.7.4.1 General

The purpose of a model of operation is to identify the objects whose properties are represented by data items in interfaces described in a SEDS. References to a model of operation link the information view in a SEDS to other viewpoints in a larger architectural design of the component described by the SEDS.

2.7.4.2 Standard Models of Operation

A model of operation is a description of the parts of a component and the context in which the component operates. A standard model of operation is one that appears in the SANA DoT ontology. The DoT model of operation contains an individual in the ModelOfOperation class called the anchor. The anchor has object relations to individuals in other classes that form a graph. A semantic attribute that corresponds to a referential class has for its value a string that describes a path through the graph starting from the name of the anchor and traversing relations by name to a set of target individuals. The meaning of the attribute is a relationship between the SEDS object in which the attribute appears and target individuals in the model of operation.

For example, the DoT defines a referential class named ‘subject’. The description of the class says that a data item in a SEDS that has the ‘subject’ attribute is a property of the referenced individuals in the model of operation.

More specifically, and continuing the ‘subject’ example above, an imaging device may carry some thermistors to measure temperature at different points in the instrument. The focal plane is often a point of interest. There could be an electronics package attached to the imager for processing the images, and the temperature of that package could be of interest. In order to recognize which thermistor measures the temperature of which part of the imager, it is necessary to define a model of the parts of the imager, and then it is necessary to attach a semantic attribute to the measurement from each thermistor that refers to the model part where the thermistor is located (see figure 2-5).

For an example of context in a model of operation, a device that tracks signals from ships at sea would have a model of operation that contains the device, the vehicle, the Earth, and the ships at sea. This model of operation would likely be defined in a user-defined context (3.6) during its development.

2.7.4.3 User Provided Models of Operation

Authors of SEDS may write a user-defined ontology that adds individuals to the ModelOfOperation class, plus related classes for reference by attributes in the SEDS. This can be done in the metadata section of a SEDS. The convention that enables this is the treatment of a SEDS document as a model of operations.

Considering each SEDS to be a model of operation enables the semantic properties of one parameter to refer to another parameter. For example, a measure of variance could refer to a separate parameter that contains the most recent measurement in the distribution.

Standard models of operation cannot efficiently represent the variety of details that may occur in a specific instrument. To address this difficulty, the parts of a standard model of operation, which are classes, can be interpreted as a skeletal model. Individuals in the classes of a standard model of operation can be defined in a SEDS and can be related to those classes by a referential attribute, 'memberOf'. This convention allows SEDS to define a variety of operating models without needing a user-defined ontology, which would diminish their portability (see figure 2-5).

Continuing the example of an imaging device above, the SEDS could describe part of the model of operation of a particular instrument by means of an element that represents the focal plane, containing the attributes <… name="focalPlane" memberOf="imager.hasA.focalPlane" …>, where the imager.hasA.focalPlane is a path through a standard model of operation of an imager in the SANA ontology.

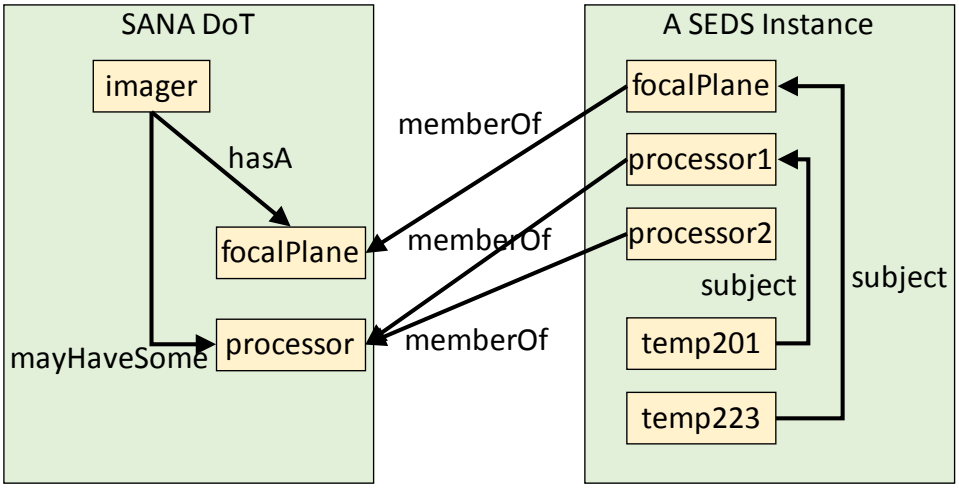


Figure 2-5: References to a Model of Operation

The SEDS example of the imaging device in figure 2-5 contains a focal plane and two processors. Suppose that the housekeeping data reported by the device contains measurements of temperature of those parts. It is possible to describe explicitly which measurement applies to which part. The SEDS could also contain two elements: one with attributes <… name="processor1" memberOf="imager.mayHaveSome.processor" …>, and the other with attributes <… name="processor2" memberOf="imager.mayHaveSome.processor" …>. In a

description of a housekeeping interface in the SEDS, one parameter that is a measure of temperature could have the attributes <… subject="focalPlane" quantityKind="Temperature" …>, while another parameter in the same interface that is also a measure of temperature could have the attributes <… subject="processor1" quantityKind="Temperature" …>.

The example of the imaging device in figure 2-5 contains the names 'temp201' and 'temp223', which demonstrate the ambiguity that is natural in names. The 'temp' in the names could be mistaken to mean 'temporary', instead of 'temperature'. The '201' and the '223' could be references to a table in a paper document that tells the locations of thermistors, but a person or algorithm trying to make sense of those names would need access to that table, and the convention used to construct the names is not a standard that can be encoded in an algorithm that works across multiple projects. Compare this ambiguity to the greater clarity of expression in using the attributes. The attribute <… quantityKind="Temperature" …> states explicitly that the data items represent temperatures. Other attributes not shown in the example provide additional information. The attribute <… unit …> can identify the units of measurement. The attribute <… purpose…> can state whether the data item is a measurement, a set point, or another intended use.

It is important to note that, while the word 'parameter' was used in the example of the imaging device, the use of semantic attributes is not limited to parameters. Any item of data described by a SEDS can have semantic attributes.

2.7.5 MODELS OF DISCRETE DATA

An enumeration of labels, with descriptions, but without assigning numbers to the labels, is an abstract model of a discrete variable. The DoT contains this kind of description of a discrete variable. A SEDS instance refers to the description in the DoT and assigns numbers to the labels to realize a discrete variable. A 'discrete variable', as used in this section, could appear in a SEDS instance as a parameter, a variable, or an argument of a command.

There are at least three stakeholders in describing the meanings of discrete variables.

- A manufacturer of an instrument would like to describe some of the standard meanings of the values of a discrete variable, while possibly including some extra special meanings that differentiate their product from others on the market.
- A spacecraft designer would like to design systems that respond to the values of discrete variables appropriately, so the meanings of each value must be described explicitly.
- A mission operations team would like to see discrete variables with similar semantics presented in a way that is consistent across missions in order to minimize the chance of errors of interpretation.

In order to satisfy the different stakeholders, the semantics of discrete variables in the DoT map to the values of discrete variables in SEDS instances, as described in this section.

There is a possibility of confusion in this discussion of enumerations, because the DoT defines two kinds of enumerations. There are not only enumerations for semantic attributes in the schema, but also enumerations for discrete variables that appear in interfaces as data during the operation of a component. The enumerations discussed in this section do not enumerate the values possible for a semantic attribute in a SEDS schema. Instead, these enumerations apply to the data that appears in an interface during operation of a component described by a SEDS instance.

SEDS assign semantics to enumeration members for discrete data variables in interfaces as shown in figure 2-6. The example on the left side of the figure is the operating modes of a thermo-electric controller. The left side of the figure is a model of operations defined in metadata in a package file package named 'CCSDS/SOIS/modops'. On the right side of the figure is a SEDS instance that refers to the model of operations. The <… subject …> attributes assign enumeration labels in the SEDS instance to members of the operatingMode enumeration model in a model of operation. The manufacturer is free to define their own labels and values for the discrete variable when a standard model does not match. The mapping specified by the attributes provides the explanation of meanings of operating modes for vehicle designers. The labels in the model of operation may be used in the user interface of mission control systems to present discrete variables to human operators consistently across missions. The annotation properties in the ontology may be copied to an operator's manual to provide additional help to operators.

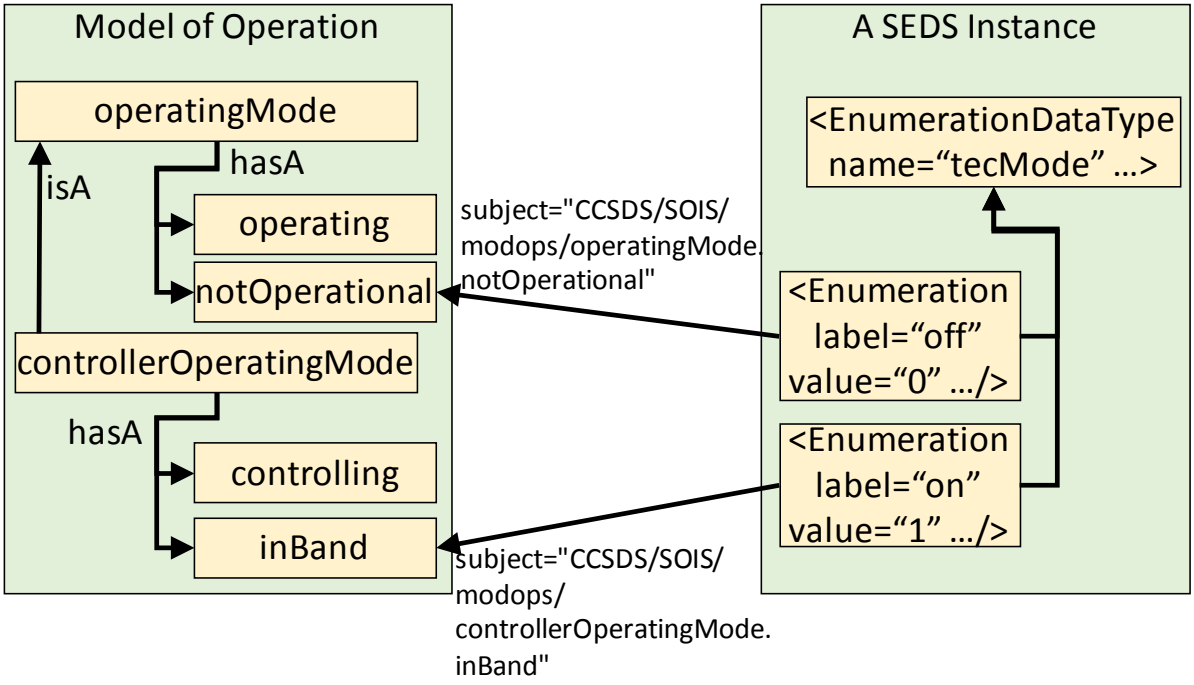


Figure 2-6: Defining a Discrete Variable Type

2.8 METADATA

The SEDS schema supports a metadata section in a SEDS instance. The DoT defines tags that can be applied to metadata elements to indicate how to interpret the metadata. Metadata consists of categories and values, both of which can be tagged.

The purposes to which metadata has been applied include:

- identification of static properties of the component described by a SEDS instance, such as mass of a device or memory space used by a software object (dynamic properties, such as the mass of a gas tank, may be reported in telemetry messages, and so would be defined in the `DataTypeSet` of a SEDS instance);
- definition or extension of a model of operation of the component described by a SEDS instance.

Among the static properties of a component are a small set of terms that describe its provenance. The DoT contains a class named 'ModelOfProduction', which has three subclasses named 'Manufacturer', 'ManufacturersModel', and 'SerialNumber'. When one of these class names is used as the name of a category or of a value, it indicates the meaning of the element of metadata as defined by the DoT, as is illustrated in the following example.

```
<Category name="ModelOfProduction" >
      <MetadataValueSet>
            <StringValue name="Manufacturer" value="Acme"/>
            <StringValue name="ManufacturersModel" value="X"/>
            <StringValue name="SerialNumber" value="123"/>
      </MetadataValueSet>
</Category>
```

Sometimes it is necessary to use the name attribute to distinguish more than one instance of a part of a model of operation. In the example in figure 2-5, the DoT contains a class named 'imager', which may have some instances of a class named 'processor'. In a SEDS instance that describes an imager with two processors, the metadata would extend the DoT model of operation by defining two processors named 'processor1' and 'processor2', as illustrated in the example below. The addition of the attribute memberOf="imager.mayHaveSome.processor" to the value elements indicates that the values represent instances of the processor class of an imager.

```
<Category name="imager" memberOf="ModelOfOperation" >
      <MetadataValueSet>
            <StringValue              name="processor1"              value="hasA"
memberOf="imager.mayHaveSome.processor"/>
            <StringValue              name="processor2"              value="hasA"
memberOf="imager.mayHaveSome.processor"/>
      </MetadataValueSet>
</Category>
```

Expressions elsewhere in the SEDS instance can refer to metadata values for the device using the names of the metadata values. References to those metadata values are made by a period-delimited path through the tree of category names; in the model of production example above, a reference to the serial number would be, 'ModelOfProduction.serialNumber'.

The following conventions simplify path expressions for references to items in a model of operations. The most common relation in models of operation is 'hasA', so the phrase 'hasA.' in a path expression can be abbreviated '.'. Containment of one category by another in the metadata category tree indicates that the latter category 'hasA' former category.

The following example shows that it may be necessary to indicate explicitly the subject of a metadata value when the position in the metadata category tree does not represent the subject implicitly.

```
<Category name="cm" memberOf="Version">
       <MetadataValueSet>
              <StringValue name="sedsVersion" value="0.1.0" memberOf="SOIS_Version"
subject="SEDS"/>
              <StringValue name="dotVersion" value="0.1.0" memberOf="SOIS_Version"
subject="DoT"/>
       </MetadataValueSet>
</Category>
```

In the example above, the version numbers of both the schema and the DoT are specified to indicate the versions of those documents to be used to parse a SEDS instance.

2.9 MAINTAINING THE DICTIONARY OF TERMS

While standard terminology is important for reusable application software, innovative applications and new technologies will not always fit within those constraints. The freedom to define new terms and new relationships among terms is built into the SOIS architecture for SEDS.

The following stakeholders participate in the maintenance of the SANA DoT:

- Manufacturers find that portability can increase the size of the market for their products.
- Integrators can reuse portable components in multiple missions.
- Subject matter experts understand the descriptions of the real world that must appear in an ontology in order to describe data efficiently. Agencies enlist members of their organizations to provide the role of subject matter experts by matching the subject matter of the topic under discussion to the appropriate experts.
- A managing authority builds consensus and provides arbitration to determine the content of the SANA DoT that optimizes the usefulness of the SANA DoT. The CCSDS can organize a Special Interest Group (reference [D1]) to provide this role.

The procedure for extending the DoT is the same as the procedure for constructing the DoT initially:

a) A manufacturer builds a new component that has innovative features that must be described in a SEDS.

b) The manufacturer writes the SEDS using new terms as necessary. The new terms are defined in a user-defined ontology. If the manufacturer prefers, the new terms can simply be defined directly in the seds-extension-semantics.xsd file, with comments.

c) The managing authority for the DoT reviews the user-defined terms and decides whether they should be integrated with the existing ontology.

d) If the decision in step c is to proceed, then the managing authority integrates the user-defined terms:

    1) As a result of step c, it may be discovered that some of the novel terms represent concepts that can be expressed by other existing terms in the DoT.

    2) Another result of step c is the recognition of new terms for the DoT.

    3) Yet another result is the addition of new structure to the DoT ontology.

    4) If the integration results in changes to terms that the user defined, the managing authority notifies the manufacturer. The manufacturer can decide to keep the user-defined ontology (not portable) or to issue new SEDS that are compatible with the revised SANA ontology (portable).

e) The managing authority for the DoT publishes the latest version periodically.

3 STRUCTURE OF THE DICTIONARY OF TERMS

3.1 OVERVIEW

This section describes the structure of the DoT ontology. The structure defined here is normative, in order to assure the capability of the DoT to provide terms that are compatible with usage in SEDS and in the SEDS schema. Each part of this section addresses a separate issue of expression or usage in SEDS.

Beyond the structure of the DoT ontology, this document defines a minimal set of concepts to support that structure in the ontology. This document does not specify all details of the actual content of the DoT ontology. Instead, this document designates the SANA DoT ontology as its formal, normative extension, to be curated by an expert group within CCSDS (see reference [12]).

3.2 ACCESS

3.2.1 GENERAL

The DoT shall be accessible for public use at least to the extent defined in this subsection.

3.2.2 ACCESS TO ONTOLOGY

The DoT ontology shall be accessible for public use through SANA (reference [12]) with web and programmatic access. Any included ontologies that are not already publicly accessible shall be accessible or referenced on the same CCSDS resource.

NOTE – The DoT ontology is currently available in SANA (see reference [12]).

3.2.3 ACCESS TO DERIVATIVES

The files listed in this section, which are generated from the content of the ontology, as well as the ontology shall be accessible for public use through SANA:

- the human-readable DoT;
- the schema representing the DoT, which is included in the SEDS schema;
- a minimal schema for optional user-defined terms.

NOTE – The files above are currently available in SANA (see reference [12]).

3.3 BASIC CONCEPTS

3.3.1 GENERAL

The DoT shall contain at least the basic concepts and structure described in this section, for use in defining the terms that can be used in authoring SEDS.

NOTE – Extensions to the DoT are described in 3.6.

3.3.2 HUMAN-READABLE COMMENTS

The 'dc:description' annotation property shall contain a human-readable description of the meaning of each class, object property, data property, and individual defined in the DoT ontology.

NOTE – The annotation can be extracted from the ontology along with terms to build a human-readable artefact called the 'glossary'. The namespace 'dc' indicates that the annotation property is defined in the 'Dublin core' (reference [6]).

3.3.3 SEDS ATTRIBUTES

The DoT shall provide content for the SEDS schema through an includable schema, using the following mechanism:

a) The includable schema shall contain one or more attribute groups.

b) The includable schema shall have no target namespace.

c) The SEDS schema shall include the includable schema.

d) The SEDS schema shall refer to the attribute groups in the includable schema in the definitions of types or of elements where those attributes will be used in a SEDS.

NOTE – This mechanism allows for the definition of attributes to be used in various contexts in a SEDS schema. Potential examples of those contexts are:

- **semantic attributes**: (see 3.4);
- **subnetwork attributes**: attributes that define the properties of a subnetwork interface using a model of operation of a subnetwork;
- **metadata**: (see 3.5.1).

3.3.4 ENUMERATED SEDS ATTRIBUTES

A class that represents a schema attribute whose range is an enumeration, and which is used in an element of a SEDS, shall map to the SEDS schema such that the name of the class is the name of the attribute, and the names of the individuals in the class are the names of the labels of the enumeration.

NOTE – The enumerations described here are the ranges of attributes in a SEDS instance. Enumerations of possible values for discrete data variables in component interfaces are the subject of 3.4.3.5.

3.3.5 EXTENSIBLE ENUMERATED SEDS ATTRIBUTES

The DoT shall contain a class named ‘ExtensibleEnumeration’. Sub-classes derived immediately from the ExtensibleEnumeration class shall have names with prefix ‘DoT’ and with the remainder of the name matching a simple type in the main SEDS schema for a union of enumerations. Individuals in the sub-classes shall have names that are to be members of the union.

NOTES

1 Rationale: The SEDS schema contains a number of enumerations listing standards options for encoding and checksum algorithms. Component or project-specific algorithms can be supported by extending those enumerations to take additional values.

2 Example: The main SEDS schema provides a simple type named ‘ErrorControlType’, which is a union of ‘CoreErrorControlType’, ‘DotErrorControlType’, and ‘ExtErrorControlType’. The CoreErrorControlType is an enumeration in the main SEDS schema. The DotErrorControlType is provided by the inclusion schema generated from the DoT. The ExtErrorControlType is provided by the sample inclusion schema for use within a project. The DoT contains a class named ‘DotErrorControlType’ derived from ‘ExtensibleEnumeration’. There is an individual named ‘CRC32’ in the DotErrorControlType class in the DoT. This example allows both the DoT (interoperable) and the project extension schema (not interoperable) to extend the enumeration of algorithms that a toolchain can apply to detect errors in packets.

3.4 SEMANTIC ATTRIBUTES

3.4.1 OVERVIEW

Subsection 3.4 contains the following topics:

- relationship between SEDS schema and DoT for semantic terms;
- representation of semantic terms in the DoT;
- a mechanism for constraining the combinations of terms that can be applied together in an EDS element.

3.4.2 RELATIONSHIP BETWEEN SEDS SCHEMA AND DOT FOR SEMANTIC ATTRIBUTES

3.4.2.1 Overview

Semantic attributes describe the interpretation of data. Examples of such terms are reference frame and unit of measure. These terms are called 'semantic attributes'.

This subsection describes the relationship between the SEDS schema and the DoT for semantic attributes. Semantic attributes are defined by the DoT and transferred to the SEDS schema as specified in 3.3.

By defining semantic attributes in the DoT, the generation of human-readable artefacts from the DoT can include semantic attributes. The ontology can also be used to describe restrictions on the usage of the attributes.

3.4.2.2 The DoT shall define semantic attributes that can be used to describe the interpretation of data in SEDS.

NOTE – Examples of such terms are 'referenceFrame' and 'unit'.

3.4.3 ONTOLOGY FOR SEMANTIC ATTRIBUTES

3.4.3.1 Representation of Semantic Attributes in the DoT

The DoT shall represent each semantic attribute as a class derived from the class 'SemanticProperty', and the name of the class shall be the name of the attribute in the schema.

3.4.3.2 Enumerated Semantic Attributes

For semantic attributes whose range of values is an enumeration, the names of individuals in the class shall be the values of the attribute in the schema (this is an instance of the specification in 3.3.3).

NOTE – An example of an enumerated semantic attribute is 'referenceFrame', which may have individuals with names like 'device' or 'ECI'.

3.4.3.3 QUDV Semantics

3.4.3.3.1 The DoT shall include the QUDV ontology (reference [13]) to obtain definitions of quantity kinds and units of measure. Until a stable publication of QUDV allows reference to specific version numbers, the QUDV ontology shall be stored on SANA as a copy of the Object Management Group (OMG) proof-of-concept publication.

3.4.3.3.2 The DoT shall extend the QUDV ontology as necessary.

3.4.3.3.3 Users of the DoT, both people and software, shall treat the QUDV classes for quantity kinds and units of measure as subclasses of 'SemanticProperty'.

NOTE – This formula generalizes the description of enumerated semantic attributes to include subclasses of the class that names the attribute. The names of the subclasses are not used, but the names of the individuals in the subclasses are used as the enumeration values. The quantity-kind class and the unit class are not defined as subclasses of 'SemanticProperty' in order to keep the DoT ontology separate from the QUDV ontology.

3.4.3.4 Referential Semantics

NOTE – An informative discussion of models of operation appears in 2.7.4.

3.4.3.4.1 Models of Operation

NOTE – Models of operation provide a target for references.

3.4.3.4.1.1 The DoT shall contain a class named 'ModelOfOperation'.

3.4.3.4.1.2 The DoT shall define a subclass of the class 'SemanticProperty' named 'RefersToModel'.

3.4.3.4.1.3 A class derived from RefersToModel may be called a referential class and shall have no individual members.

3.4.3.4.1.4 A semantic attribute in the SEDS schema that corresponds to a referential class shall have a range of values in 'xsd:string'.

3.4.3.4.1.5 The prefix of the string value of the attribute shall be the name of an individual in the ModelOfOperation class or the name of a class derived from ModelOfOperation.

3.4.3.4.1.6 The remainder of the string value of the attribute, if any, shall be a path expression delimited by '.', which consists of alternating names of object relations and names of classes or individuals in the SANA or user-defined ontology. The object relation 'hasA' may appear explicitly in a path expression, or it may be omitted between the name of the aggregating class and the name of the aggregated class.

3.4.3.4.1.7 The definition of the semantic attribute shall relate the part of the SEDS that bears the attribute to the individuals or classes in the model of operation that are at the end of the path specified by the value of the attribute.

NOTE – As an example, a navigation application that uses the nadir point of its satellite could be designed to ignore latitude-longitude parameters whose 'subject' attribute is not 'GNS.onBoard.artificialSatellite.over.nadirPoint'. This would allow other applications on board the vehicle to produce latitude-longitude parameters that are relevant to other objects of interest on the planet orbited by the satellite without harming the navigation application.

3.4.3.4.2 Standard Models of Operation

3.4.3.4.2.1 The DoT shall define standard models of operation, with standard names for the parts of the model that can be referenced.

3.4.3.4.2.2 The standard models of operation shall be individuals of the class 'ModelOfOperation' or classes derived from ModelOfOperation.

3.4.3.4.2.3 The parts of the standard models of operation shall be related classes and individuals in related classes.

3.4.3.4.3 User-Defined Models of Operation

3.4.3.4.3.1 The document object model of a SEDS shall be treated as a model of operation with the anchor being the trunk element of the SEDS.

3.4.3.4.3.2 The DoT shall define a referential class named 'memberOf', which shall have the interpretation that an element bearing the attribute in the document object model of a SEDS is a member of the class in a standard or user-defined model of operation, named by the value of the attribute.

3.4.3.5 Enumeration of Discrete Data

NOTE – A SEDS can associate enumeration tags with numbers to describe the representation of discrete variables in a data interface. The SEDS provides the association between integer values and names of classes in a model of operations that represent the conceptual elements of an enumeration.

3.4.3.5.1 The DoT shall define the interoperable meanings of enumeration tags.

3.4.3.5.2 The association of tags with numbers shall be local to SEDS.

3.4.3.5.3 The DoT shall define a subclass of 'ModelOfOperation' named 'Enumeration', whose derived classes contain enumerations of the possible interoperable meanings of discrete data items described by a SEDS.

3.4.3.5.4 Standard Enumerations

3.4.3.5.4.1 The DoT shall define standard enumerations, with standard names for the enumerated labels.

3.4.3.5.4.2 The standard enumerations shall be classes derived from the class ‘Enumeration’.

3.4.3.5.5 User-Defined Enumerations

Authors of SEDS may write a user-defined ontology that adds classes derived from the ‘Enumeration’ class. Members of these classes shall be named in the ‘subject’ attribute of Enumeration elements in an EnumeratedDataType element to indicate their meanings.

3.4.3.6 Schema for Semantic Attributes

3.4.3.6.1 The DoT shall be accompanied by open-source software for extraction of a schema fragment that can be included in the SEDS schema to define semantic attributes.

NOTE – Schemas supporting SEDS and SOIS DoT can be found at https://sanaregistry.org/r/sois/.

3.4.3.6.2 The schema fragment shall contain an attribute group named ‘CoreSemanticsAttributeGroup’, which contains semantic attributes and restricts their values.

3.4.4 COMBINATORIAL CONSTRAINTS ON SEMANTICS

3.4.4.1 General

The constraints on combinations of semantic attributes form an open-world model: combinations that are not explicitly excluded are allowed. This policy limits unintentional restrictions on the community of users.

3.4.4.2 Excluded Semantics

3.4.4.2.1 The DoT may define any necessary constraints on legal combinations of semantic attributes.

3.4.4.2.2 Each such constraint shall be an individual in the class ‘ExcludedSemantics’.

3.4.4.2.3 The class ‘ExcludedSemantics’ shall have an object property for each enumerated semantic attribute, with the name [‘value’ prefixed to the name of the enumerated semantic attribute class].

3.4.4.2.4 An individual in ‘ExcludedSemantics’ shall be interpreted as an illegal combination of attributes.

NOTES

1 Rationale: this represents semantic constraints which can be used to check consistency of a data sheet.

2 Example: to indicate that chirality cannot be used with coordinateType latLon, there would be two individuals in ExcludedSemantics, one with valueChirality=leftHanded and one with valueChirality=rightHanded, both with valueCoordinateType=latLon.

3.4.4.3 External Constraints

The DoT shall include the 'sysml-qudv:quantityKind' object property for individuals in classes derived from 'unit'.

NOTE – External software can use this information to validate the pairing of quantityKind and unit attributes.

3.5 METADATA

NOTE – This subsection enables the DoT to define semantic tags for metadata in SEDS instances.

3.5.1 SEMANTIC ATTRIBUTES FOR METADATA

3.5.1.1 The semantic attributes in 3.4 shall be applicable to metadata categories and to metadata value items.

3.5.1.2 The DoT shall contain a class named 'MetadataCategory', which is a sub-class of the class named 'ModelOfOperation'.

NOTE – This requirement enables referential semantics for the parts of the DoT model of operations that are specific to metadata (see 3.5.2, 3.5.3, and 3.5.4).

3.5.2 MODEL OF PRODUCTION

NOTE – This subsection enables the DoT to define the following terms that identify a component by telling how it was made (see 2.8).

3.5.2.1 The DoT shall contain a class named 'ModelOfProduction', which is a sub-class of the class named 'MetadataCategory'.

3.5.2.2 The DoT shall contain a class named 'Manufacturer', which is a sub-class of the class named 'ModelOfProduction'. Manufacturer identifiers shall conform to ISO 9362.

3.5.2.3 The DoT shall contain a class named 'ManufacturersModel', which is a sub-class of the class named 'ModelOfProduction'.

3.5.2.4 The DoT shall contain a class named ‘SerialNumber’, which is a sub-class of the class named ‘ModelOfProduction’.

3.5.3 LANGUAGE OF DATA SHEETS

NOTE – This subsection enables the DoT to define terms that indicate the following documents that define the syntax and semantics of SEDS so a toolchain may use those documents to parse a SEDS instance (see 2.8).

3.5.3.1 The DoT shall contain a class named ‘DataSheetLanguage’, which is a sub-class of the class named ‘MetadataCategories’.

3.5.3.2 The DoT shall contain a class or individual named ‘QUDV’, which is in DataSheetLanguage.

3.5.3.3 The DoT shall contain a class or individual named ‘SEDS’, which is in DataSheetLanguage.

3.5.3.4 The DoT shall contain a class or individual named ‘DoT’, which is in DataSheetLanguage.

3.5.4 VERSION NUMBERS

NOTE – This subsection enables the DoT to define terms that indicate the versions of the documents that define the syntax and semantics of SEDS so a toolchain may use the interpretations appropriate to those versions (see 2.8).

3.5.4.1 The DoT shall contain a class named ‘Version’, which is a sub-class of the class named ‘MetadataCategories’.

3.5.4.2 The DoT shall contain a class named ‘SOIS_Version’, which is a sub-class of the class named ‘Version’.

3.5.4.3 The version numbers for the SEDS schema and for the DoT shall have the form of three positive integers concatenated with period characters, ‘.’.

NOTE – Example: ‘1.2.0’ indicates a version that resulted from one change that was not backwards compatible and two subsequent backward-compatible extensions of function.

3.5.4.3.1 The first integer shall be zero for the first version and incremented by one for versions that are not backward compatible with the previous version.

3.5.4.3.2 The second integer shall be zero for each new value of the first integer and incremented by one for each change that extends the previous version while maintaining compatibility.

3.5.4.3.3 The third integer shall be zero for each new value of the second integer and incremented by one for each change that corrects errors in previous versions without breaking compatibility and without extending compatibility.

3.6 USER-DEFINED ONTOLOGIES

3.6.1 OVERVIEW

This subsection describes how user-defined ontologies may be used to extend the SANA ontology when the latter lacks information necessary for a description.

NOTE – EDSes that apply user-defined ontologies may be useful within a project, but the components that they describe are not portable outside the project. Agencies with a policy of promoting portable components will have to require their suppliers to use only terms that are in the SANA DoT, or to participate actively in the process of integrating new terms into the SANA DoT.

3.6.2 STRUCTURE OF USER-DEFINED ONTOLOGIES

3.6.2.1 Structural Consistency

The structure of a user-defined ontology shall be consistent with the structure of the SANA DoT ontology.

3.6.2.2 Discussion

If the purpose is simply to provide additional terms within the existing structure, then a user-defined ontology can define the additional terms as individuals in the appropriate class. In this case, the programs that generate derivatives from the ontology will be able to include the term(s) without change.

When the user-defined ontology must add new structure that is absent in the SANA DoT, it may add some new classes to the DoT that do not inherit any information from classes already present. In this case, the programs that generate derivatives from the ontology will require modification in order to generate the new structure. This action is not a violation of this standard.

3.6.2.3 A Simpler Alternative

Instead of constructing a user-defined ontology, a project may choose simply to add to the file named seds-extension-semantics.xsd, an example of which accompanies the SEDS schema in SANA. This saves the cost of building an ontology. Project personnel need only understand the XML schema language for XSD files (reference [4]).

3.6.3 OMISSIONS

NOTE – Section 3 in this document may omit some issues that are needed for particular components or for particular kinds of interfaces. For example, the present description only covers data interfaces; it does not cover physical interfaces, such as thermal, electrical, mass, geometry, and others. Unforeseen issues of data interfaces may have been omitted.

3.6.3.1 In case an issue is omitted that is needed for a particular interface, the SEDS author may provide a user-defined ontology to cover the issue, or the author may provide an updated seds-extension-semantics.xsd file.

3.6.3.2 After appropriate consideration, the SANA DoT managing authority shall decide whether to integrate the user-defined ontology into the SANA DoT.

3.6.3.3 If the decision is positive, then the managing authority shall integrate the user-defined ontology and amend this document to cover the issue.

NOTE – The process of integration can alter the user-defined ontology.

ANNEX A

DICTIONARY OF TERMS FOR ELECTRONIC DATA SHEETS IMPLEMENTATION CONFORMANCE STATEMENT PROFORMA

(NORMATIVE)

A1 INTRODUCTION

This annex provides the ICS Requirements List (RL) for implementation of the DoT, CCSDS 876.1-M-1. The ICS for an implementation is generated by completing the RL in accordance with the instructions below. An implementation shall satisfy the mandatory conformance requirements of the base standards referenced in the RL.

The RL in this annex is blank. An implementation's complete RL is called a ICS. The ICS states which capabilities and options of the services have been implemented. The following can use the ICS:

- an author of a SEDS that contains a user-defined DoT, as a checklist to reduce the risk of failure to conform to the standard through oversight;
- an author of toolchain software, as a basis for extracting information from the SANA DoT and from user-defined DoT for use by the toolchain.

A2 NOTATION

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

Status Symbols

M	mandatory
O	optional

Support Column Symbols

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

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

A3 REFERENCED BASE STANDARDS

The base standards references in the RL are:

– Dictionary of Terms for Electronic Data Sheets (this document).

A4 GENERATION INFORMATION

A4.1 IDENTIFICATION OF ICS

Ref	Question	Response
1	Date of statement (DD/MM/YYYY)	
2	ICS serial number	
3	System conformance statement cross-reference	

A4.2 IDENTIFICATION OF IMPLEMENTATION UNDER TEST (IUT)

Ref	Question	Response
1	Implementation name	
2	Implementation version	
3	Special configuration	
4	Other information	

A4.3 IDENTIFICATION

Ref	Question	Response
1	Supplier	
2	Contact point for queries	
3	Implementation name(s) and versions	
4	Other information necessary for full identification, for example, name(s) and version(s) for machines and/or operating systems: System name(s)	

A4.4 ONTOLOGY SUMMARY

Ref	Question	Response
1	Service version	
2	Addenda implemented	
3	Amendments implemented	
4	Have any exceptions been required? NOTE – A YES answer means that the implementation does not conform to the service. Non-supported mandatory capabilities are to be identified in the ICS, with an explanation of why the implementation is non-conforming.	Yes _____ No _____

A4.5 INSTRUCTIONS FOR COMPLETING THE RL

An implementer of toolchain software shows the extent of compliance to the specification by completing the RL; that is, compliance to all mandatory requirements and the options that are not supported are shown. The resulting completed RL is called an ICS. In the Support column, each response shall be selected either from the indicated set of responses, or it shall comprise one or more parameter values as requested. If a conditional requirement is inappropriate, N/A shall be used. If a mandatory requirement is not satisfied, exception information must be supplied by entering a reference X*i*, where *i* is a unique identifier, to an accompanying rationale for the non-compliance.

A5 GENERAL/MAJOR CAPABILITIES OF DOT ONTOLOGY

Service Feature	Reference	Status	Support
Access to Ontology	3.2.2	M	
Access to Derivatives	3.2.3	M	
Human-Readable Comments	3.3.2	M	

Enumerated SEDS Attribute	3.3.4	M	
Model of Production	3.5.2	M	
Semantic Attributes	3.4.2.2	M	
Representation of Semantic Attributes	3.4.3.1	M	
Enumerated Semantic Attributes	3.4.3.2	M	
QUDV Semantics	3.4.3.3	M	
Referential Semantics	3.4.3.4	M	
Enumeration Discrete Variables	3.4.3.5	M	
Schema for Semantic Attributes	3.4.3.6	M	
Combinatorial Constraints on Semantics	3.4.4	O	
External Constraints	3.4.4.3	M	
User-Defined Ontologies	3.6	O	

A6 SOFTWARE EXTRACTING INFORMATION FROM ONTOLOGY

This subsection provides identification of the software that extracts information from the ontology.

Service Feature	Reference	Status	Support
Human-Readable Comments	3.3.2	M	
Enumerated SEDS Attributes	3.3.3	M	
Enumerations of Discrete Data	3.4.3.5	M	
Ontology for Semantic Attributes	3.4.3	M	
Combinatorial Constraints on Semantics	3.4.4	O	

ANNEX B

SECURITY CONSIDERATIONS

(INFORMATIVE)

B1 SECURITY BACKGROUND

The SOIS DoT for EDSes for onboard devices is publicly available for use in design toolchains and is designed to accommodate extension by its users. This openness may be exploited to affect the operation of a toolchain adversely. Users must rely upon trusted manufacturers to provide safe EDSes. The specification of such security services is outside the scope of this document.

B2 SECURITY CONCERNS

At the time of writing, there are no identified security concerns. If confidentiality of data is required within a project, some degree of proprietary control may be obtained by using user-defined ontologies that are never submitted to the DoT managing authority for integration into the SANA DoT.

B3 POTENTIAL THREATS AND ATTACK SCENARIOS

Potential threats and attack scenarios typically derive from outside the mission-manufacturer relationship and are therefore not the direct concern of the SOIS DoT. It is assumed that all EDSes within the spacecraft have been thoroughly tested and cleared for use by the mission implementer.

B4 CONSEQUENCES OF NOT APPLYING SECURITY

The security services are out of scope of this document and are expected to be applied at organizational layers above or below those specified in this document. If confidentiality is not implemented, science data or other parameters transmitted within the spacecraft might be misused.

B5 RELIABILITY

While it is assumed that the underlying mechanisms used to implement a toolchain operate correctly, the initial implementation of the DoT can make no promises of reliability. After a sufficient body of experience with real EDSes has developed, useful estimates of reliability will be possible.

ANNEX C

ABBREVIATIONS

(INFORMATIVE)

Term	Meaning
CCSDS	Consultative Committee for Space Data Standards
DACP	device abstraction control procedure
DAS	Device Access Service
DoT	dictionary of terms
DSAP	device-specific access protocol
DVS	Device Virtualization Service
EDS	electronic data sheet
ICS	implementation conformance statement
IUT	implementation under test
OMG	Object Management Group
OWL	Web Ontology Language
QUDV	Quantities, Units, Dimensions, Values
RDF	Resource Description Framework
RL	requirements list
SANA	Space Assigned Numbers Authority
SEDS	SOIS Electronic Data Sheet(s)
SOIS	Spacecraft Onboard Interface Services
W3C	World Wide Web Consortium
XML	Extensible Markup Language
XSD	XML Schema Definition language

ANNEX D

INFORMATIVE REFERENCES

(INFORMATIVE)

[D1] *Organization and Processes for the Consultative Committee for Space Data Systems*. Issue 4. CCSDS Record (Yellow Book), CCSDS A02.1-Y-4. Washington, D.C.: CCSDS, April 2014.

[D2] *Spacecraft Onboard Interface Services*. Issue 2. Report Concerning Space Data System Standards (Green Book), CCSDS 850.0-G-2. Washington, D.C.: CCSDS, December 2013.

[D3] *XML Telemetric and Command Exchange—Version 1.2*. Issue 2. Recommendation for Space Data System Standards (Blue Book), CCSDS 660.0-B-2. Washington, D.C.: CCSDS, February 2020.

[D4] *Electronic Data Sheets and Common Dictionary of Terms for Onboard Devices and Components.* Proposed.

ANNEX E

EXAMPLE DOT/XML ONTOLOGY INSTANTIATIONS

(INFORMATIVE)

The following excerpts from a SEDS for a star tracker demonstrate the information carried by semantic tags. More complete examples appear in the associated green book (reference [D4]).

The FloatDataType element below defines semantics that are common to many temperature measurements that appear in the housekeeping data for the star tracker.

```
<FloatDataType name="rawtemperature" shortDescription="a temperature reading from a thermistor">
  <Semantics purpose="measurement" quantityKind="celsiusTemperatureQK" unit="count"/>
  <FloatDataEncoding encodingAndPrecision="IEEE754_2008_single"/>
  <Range>
    <PrecisionRange>single</PrecisionRange>
  </Range>
</FloatDataType>
```

Any items of data that are of this type will be measurements, not set points. They will represent temperature in units of analogue-to-digital counts.

The Entry elements below use the FloatDataType above to define some of the items of data in a housekeeping packet produced by the star tracker.

```
<Entry name="eTemp" type="rawTemperature">
  <Semantics subject="device.hasA.electronicsBox" unit="degreeCelsius"/>
  <PolynomialCalibrator>
    <Term exponent="0" coefficient="2.71828"/>
    <Term exponent="1" coefficient="3.14159"/>
  </PolynomialCalibrator>
</Entry>
<Entry name="fTemp" type="rawTemperature">
  <Semantics subject="device.hasA.focalPlane" unit="degreeCelsius"/>
  <PolynomialCalibrator>
    <Term exponent="0" coefficient="2.71832"/>
    <Term exponent="1" coefficient="3.14164"/>
  </PolynomialCalibrator>
</Entry>
```

The first item is the temperature measured at the electronics box of the star tracker. A calibrator element describes how to convert the raw temperature counts into Celsius degrees. The second item is the temperature measured at the focal plane, with a slightly different calibrator. The ‘subject’ attributes in this case refer to parts in the model of operation of the star tracker, by tracing a path in model of operations.

The ContainerDataType below defines a quaternion, which represents a rotation. Such an object can be used to compute a rotational transformation from one coordinate system to another. This definition does not specify the coordinate systems related by the

transformation; that information is left to be specified in the particular items of data that use a quaternion. The semantic information in this definition is just the set of assumptions that are implicit in many implementations of quaternions; those assumptions appear once in this type definition and need not be repeated for each item of data that represents a quaternion.

```
<ContainerDataType name="quaternionRxyz">
  <Semantics interpretation="transformation" transformationType="quaternion" chirality="rightHanded"/>
  <EntryList>
    <Entry name="real" type="NumberType" shortDescription="the real part">
      <Semantics subject="quaternion.hasA.realPart"/>
    </Entry>
    <Entry name="ix" type="NumberType" shortDescription="the x-imaginary part">
      <Semantics subject="quaternion.hasA.xPart"/>
    </Entry>
    <Entry name="jy" type="NumberType" shortDescription="the y-imaginary part">
      <Semantics subject="quaternion.hasA.yPart"/>
    </Entry>
    <Entry name="kz" type="NumberType" shortDescription="the z-imaginary part">
      <Semantics subject="quaternion.hasA.zPart"/>
    </Entry>
  </EntryList>
</ContainerDataType>
```

This definition clarifies the arrangement of parts, which can differ between groups of users, such as computer graphics displays and attitude control logic. In this case, the real part of the quaternion appears first; in other contexts, the real part may appear last. The algorithmically accessible identification of the real part is the 'subject' attribute of the semantics element, which traces a path through the model of operation for quaternions in general. The 'shortDescription' is unstructured text, and so is useful only to human readers. By using standard semantic tags, quaternions from different sources, such as star trackers and graphics rendering packages, can be automatically adapted to the interface where they are used.

The Entry element below defines an item of data measured by the star tracker, which is the rotation from Earth-centred inertial coordinates 'J2000' to the coordinates of the device.

```
<Entry name="attitude" type="foundation/QuaternionRxyz" shortDescription="Final Attitude Quaternion">
  <Semantics referenceFrame="ECI" coordinateType="J2000" toFrame="device" purpose="measurement"/>
  <FloatDataEncoding encodingAndPrecision="IEEE754_2008_double"/>
</Entry>
```

Another item of data in the star tracker SEDS defines the orientation of the mounting face of the device. An item of data in the vehicle manifest defines the orientation of the star tracker mount relative to the vehicle coordinate system. By composing these rotational transformations, it is possible to compute the attitude of the vehicle.