**Date:** July 2025

# System Profile for Effective Cyber Threat-based Risk Assessments (SPECTRA), version 1.0

*Volume 4: SPECTRA Comma Separated Values, v1.0 – beta 1*

---

**OMG Document Number: ptc/25-07-01**

**Standard Document URL: https://www.omg.org/spec/SPECTRA/**

---

This OMG document replaces the submission document (sysa/25-03-01). It is an OMG Adopted Beta Specification and is currently in the finalization phase. Comments on the content of this document are welcome and should be directed to issues@omg.org by June 2025.

You may view the pending issues for this specification from the OMG revision issues web page https://issues.omg.org/issues/lists.

The FTF Recommendation and Report for this specification will be published in April 2026. If you are reading this after that date, please download the available specification from the OMG Specifications Catalog.

Copyright © 2025, KDM Analytics
Copyright © 2025, Ontogenesis Solutions
Copyright © 2025, Model Driven Solutions
Copyright © 2025, 88 Solutions
Copyright © 2025, OMG

## USE OF SPECIFICATION - TERMS, CONDITIONS & NOTICES

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

## LICENSES

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

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

## PATENTS

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

## GENERAL USE RESTRICTIONS

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

## DISCLAIMER OF WARRANTY

WHILE THIS PUBLICATION IS BELIEVED TO BE ACCURATE, IT IS PROVIDED "AS IS" AND MAY CONTAIN ERRORS OR MISPRINTS. THE OBJECT MANAGEMENT GROUP AND THE COMPANIES LISTED ABOVE MAKE NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS PUBLICATION, INCLUDING BUT NOT LIMITED TO ANY WARRANTY OF TITLE OR OWNERSHIP, IMPLIED WARRANTY OF MERCHANTABILITY OR WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE OR USE. IN NO EVENT SHALL THE OBJECT MANAGEMENT GROUP OR ANY OF THE COMPANIES LISTED ABOVE BE LIABLE FOR ERRORS CONTAINED HEREIN OR FOR DIRECT, INDIRECT, INCIDENTAL, SPECIAL, CONSEQUENTIAL, RELIANCE OR COVER DAMAGES, INCLUDING LOSS OF PROFITS, REVENUE, DATA OR USE, INCURRED BY ANY USER OR ANY THIRD PARTY IN CONNECTION WITH THE FURNISHING, PERFORMANCE, OR USE OF THIS MATERIAL, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

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

## RESTRICTED RIGHTS LEGEND

Use, duplication or disclosure by the U.S. Government is subject to the restrictions set forth in subparagraph (c) (1) (ii) of The Rights in Technical Data and Computer Software Clause at DFARS 252.227-7013 or in subparagraph (c)(1) and (2) of the Commercial Computer Software - Restricted Rights clauses at 48 C.F.R. 52.227-19 or as specified in 48 C.F.R. 227-7202-2 of the DoD F.A.R. Supplement and its successors, or as specified in 48 C.F.R. 12.212 of the Federal Acquisition Regulations and its successors, as applicable. The specification copyright owners are as indicated above and may be contacted through the Object Management Group, 9C Medway Road PMB 274, Milford, MA 01757, U.S.A.

## TRADEMARKS

CORBA®, CORBA logos®, FIBO®, Financial Industry Business Ontology®, FINANCIAL INSTRUMENT GLOBAL IDENTIFIER®, IIOP®, IMM®, Model Driven Architecture®, MDA®, Object Management Group®, OMG®, OMG Logo®, SoaML®, SOAML®, SysML®, UAF®, Unified Modeling Language®, UML®, UML Cube Logo®, VSIPL®, and XMI® are registered trademarks of the Object Management Group, Inc.

For a complete list of trademarks, see: https://www.omg.org/legal/tm_list.htm. All other products or company names mentioned are used for identification purposes only, and may be trademarks of their respective owners.

## COMPLIANCE

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

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

Management Group, Inc., software developed using this specification may claim compliance or conformance with the specification only if the software satisfactorily completes the testing suites.

## OMG's Issue Reporting Procedure

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

# Table of Contents

# Preface

## OMG

Founded in 1989, the Object Management Group, Inc. (OMG) is an open membership, not-for-profit computer industry standards consortium that produces and maintains computer industry specifications for interoperable, portable, and reusable enterprise applications in distributed, heterogeneous environments. Membership includes Information Technology vendors, end users, government agencies, and academia.

OMG member companies write, adopt, and maintain its specifications following a mature, open process. OMG's specifications implement the Model Driven Architecture® (MDA®), maximizing ROI through a full-lifecycle approach to enterprise integration that covers multiple operating systems, programming languages, middleware and networking infrastructures, and software development environments. OMG's specifications include: UML® (Unified Modeling Language™); CORBA® (Common Object Request Broker Architecture); CWM™ (Common Warehouse Metamodel); and industry-specific standards for dozens of vertical markets.

More information on the OMG is available at https://www.omg.org/.

# OMG Specifications

As noted, OMG specifications address middleware, modeling and vertical domain frameworks. All OMG Specifications are available from the OMG website at:
*https://www.omg.org/spec*

All of OMG's formal specifications may be downloaded without charge from our website. (Products implementing OMG specifications are available from individual suppliers.) Copies of specifications, available in PostScript and PDF format, may be obtained from the Specifications Catalog cited above or by contacting the Object Management Group, Inc. at:

OMG Headquarters
9C Medway Road, PMB 274
Milford, MA 01757
USA
Tel: +1-781-444-0404
Fax: +1-781-444-0320
Email: *pubs@omg.org*

Certain OMG specifications are also available as ISO standards. Please consult https://www.iso.org

# 1 Scope

SPECTRA is a language for describing cyber and cyber-physical systems for the purposes of risk assessments, cybersecurity assessments and vulnerability assessments. System descriptions - including models, consist of many artifacts which are of importance for one or more lifecycle phases. For the purposes of a cybersecurity assessment, certain artifacts are of essence - for example, what are the parts of the system, how these parts are connected to convey information, what information is being conveyed, and what is the nature of the parts. Cybersecurity implies a filter for the level of technical detail, compared to other disciplines involved in the system lifecycle. Effectively extracting only the relevant cybersecurity assertions for a system description is a challenging task. SPECTRA language - a set of conceptual entities and relations, collectively referred to as Core Assertions for cybersecurity - extends Systems Engineering languages with means to identify the core entities and their relationships to support the task of interpreting and postprocessing a system description by automated tools and enabling cybersecurity analytics. SPECTRA facilitates ingesting normalized machine-consumable system descriptions into compliant tools for big data analytics in cybersecurity.

SPECTRA's objective is to provide a standard compliance reference for acquisition contracts soliciting models for various assessments, as well as tools and services for performing such assessments automatically.

This specification defines the Comma-Separated Values (CSV) binding for the Core Assertions Metamodel.

# 2 Conformance

## 2.1. Introduction

The SPECTRA Comma-Separated Values specification defines the following three compliance points:

1. Model compliance (CSV)
2. Analytics Tool (Consumer) compliance (CSV)
3. Core Assertions Compliance Point

## 2.1. Model Compliance Point (CSV)

A CSV file conforming to the SPECTRA Model compliance point shall be a well-formed SPECTRA Core Assertions (CA) CSV document where the assertions about the system of interest are made according to the meaning, semantics and constraints described in this specification. It is the responsibility of the producer of the compliant model to choose which meanings to use, based on the need to communicate certain assertions about the system of interest. This involves the mandatory elements to communicate the key assertions about the system of interest, as well as any extended meanings and optional elements suggested by this SPECTRA CSV specification.

This compliance point facilitates interchange of SPECTRA models for cyber and cyber-physical systems that can be unambiguously interpreted by the software compliant to "Analytics Tool (consumer) Compliance Point" (including but not limited to cyber risk assessment tools).

## 2.2. Analytics Tool (Consumer) Compliance Point (CSV)

Software that conforms to the SPECTRA Consumer compliance point shall ingest SPECTRA CSV documents defined according to the Model Compliant Point (XMI). A compliant consumer tool shall be able to ingest all elements described in this specification. This compliance point does not restrict the capabilities of the compliant software. For example, the compliant consumer may choose to ignore some of the extended SPECTRA meanings and focus of the mandatory meanings. At this level of compliance SPECTRA allows compliant tools to have the same interpretation of the input SPECTRA CA model of the SOI, as intended by the modeler.

This compliance point allows various analytics to be performed on the ingested models, included but not limited to the Threat-based Cyber Risk Assessment.

## 2.3. Core Assertions Compliance Point

Software that conforms to the SPECTRA Consumer compliance point shall ingest SPECTRA CSV documents that conform to Model Compliance Point (CSV) defined in this SPECTRA specification. A compliant consumer tool shall be able to ingest all tables described in this specification. The software compliant at the level of the Core Assertions shall provide capability to export the Core Assertions from the ingested compliant SysML model in one of the supported formats.
This compliance point does not restrict other capabilities of the compliant software.
Software compliant at the Core Assertions compliance point shall use the semantic rules describing the process of deriving the core assertions from the CSV tables

# 3 References

## 3.1 Normative References

- SPECTRA Core Assertions
- IETF RFC 4180 Common Format and MIME Type for Comma-Separated Values (CSV) Files
- ISO/IEC IEEE 15288:2015, Systems and software engineering - System life cycle process

## 3.2 Non-normative References

- NIST SP-800-30
- ISO/IEC 27005
- NIST SP-800-37
- SPDX
- CycloneDX v1.6 Ecma, OWASP, 2024
- NIST SP-800-53
- NIST SP-800-53a
- KDM
- Prolog

# 4 Terms and Definitions

No additional terms or definitions.

# 5 Symbols

No additional symbols/abbreviations.

# 6 Additional Information

## 6.1 How to read this specification

SPECTRA Comma-Separated Value is organized as a collection of ***tables***. Each table shall be represented by one or more comma-separated value files. Each table includes multiple **columns**. Each tables includes 2 **header rows** and zero or more **data rows**.

- The first header row shall contain the table identification
- When multiple tables have the same table identification, the data rows shall be merged such that the multiple tables are treated same as a single table with each column having all data rows from each table
- Constraints defined for each table shall be applied to the combined data rows
- The second header row shall contain column titles
- Columns shall be identified by its unique column name
- SPECTRA column names shall be case insensitive
- Columns can be in any order
- Other user-defined columns can be inserted
- The data rows for each column shall follow the data type specification
- Boolean value true can be represented as true, True, yes, y, Y
- Boolean value false can be represented as false, False, no, n, N or blank

Semantics of the SPECTRA Comma-Separated Value is defined in terms of the implied Core Assertions about the SOI and how they can be derived from the tables represented by the files in a comma-separated value format. Each row of each tables makes one or more core assertions about the SOI.

SPECTRA semantic description uses Prolog to describe the claims, the core assertion rules, inference rules and the core assertion infrastructure involved in managing core assertions. Prolog is an adequate approach since it combines a database of assertions and logical inference rules.

The rules for deriving parameters to claims from the combination of metadata attributes and information in the SysML model are described informally.

SPECTRA Core Assertions are outlined in Annex A of this specification.

# 7 SPECTRA CSV Tabular Representation

## 7.1 Sections Table

Section - next lower level of structural decomposition of the SOI, into independent or autonomous areas (possibly each under a distinct authority), e.g. spacecraft, ground station, launch vehicle. Sections often participate in distinct mission phases, and therefore in distinct mission (and/or capability) configurations. Therefore, sections are significant for the purposes of risk assessment.

**Table Identification**
SPECTRA Sections for system <SOI name>

**Table Columns**

| Column title | Column data | Description |
|---|---|---|
| Name | Identifier | Specifies the identifying name of the element |
| Purpose | Text | Provides the description of the element |
| isInternal | Boolean | Specifies whether the element in internal to SOI or external (part of the environment in the context of the SOI |
| Notes | Text | Informal notes to the element |

**Constraints**

- Names in the table shall be unique
- This table defines sections. A reference to a section in the units table shall match exactly one name in this table.

**Semantics**

## 7.2 Subsystems Table

Subsystem - arrangement of replaceable units that collaborate to provide related functions and/or capabilities, e.g. communications subsystems or thermal subsystems. Subsystems are significant for the purposes of cyber risk assessment, because of their alignment to system functions and/or capabilities, and thus the related operational impacts.

**Table Identification**
SPECTRA Sections for system <SOI name>

**Table Columns**

| Column title | Column data | Description |
|---|---|---|
| Name | Identifier | Specifies the identifying name of the element |
| Purpose | Text | Provides the description of the element |
| isInternal | Boolean | Specifies whether the element in internal to SOI or external (part of the environment in the context of the SOI |
| Notes | Text | Informal notes to the element |

**Constraints**

- Names in the table shall be unique
- This table defines subsystems. A reference to a subsystem in units table shall match exactly one names in this table.

**Semantics**

## 7.3 Units Table

Units Table represents Replaceable Units and Subcomponents.

A Replaceable Unit represents a tangible node in SOI that plugs into a unique place in the web of buses and links, and thus produces and consumes a unique set of flows and performs a unique set of functions. A replaceable unit fits into a unique place defined by its connections to buses and links in SOI.

A replaceable unit may represent a set of interdependent attack opportunities (in terms of suppliable artifacts) (e.g. a custom software image running on a specific type of hardware and a specific type of operating system). A replaceable unit is usually pre-integrated and pre-configured, and then supplied to be integrated into the SOI.

From the risk assessment perspective, it is often preferable to consider the functions of a replaceable unit as irreducible, rather than as collaborations between subcomponents, providing a level of separation between replaceable units and any of its subcomponents in the end-to-end information pathways involving other replaceable units and channels.

A replaceable unit is involved in defining functional threads (various replaceable units are the “places” through which the thread “traverses”).

A set of replaceable units is the scope of collaboration.

Subcomponent - any block that is part of a replaceable unit (or a conveying element); a model of a SOI may choose to describes more levels of structural decomposition for certain replaceable units, with internal channels and flows and more elementary functions, collaborating to produce the functions of the entire replaceable units as emergent behaviors; and specific suppliable artifacts representing unique attack opportunities; for cyber and cyber-physical systems such collaborations may be quite complex, and the model may or may not have a high fidelity description of such behaviors; for the purposes of cyber risk assessment, subcomponents are ignored and their "boundary" flows and abstracted to the corresponding replaceable unit, their emergent functions abstracted to the replaceable unit, and all their artifacts aggregated and assigned to the replaceable unit. Since the corresponding replaceable unit is pre-integrated prior to its deployment into the SOI, this allows normalization of the information pathways and reduction in their complexities. The functions and flows of each replaceable unit are treated as axioms regardless of subcomponents. Marking an element as a subcomponent (of some replaceable unit) hides this element from information pathways.

**Table Identification**
SPECTRA Units for system <SOI name>

**Table Columns**

| Column title | Column data | Description |
|---|---|---|
| Name | Identifier | Specifies the identifying name of the element |
| Purpose | Text | Provides the description of the element |
| Replication | Integer | Specifies whether the element describes a set of similar elements |
| Section | Reference | Specifies the section of the element |
| Subsystem | Reference | Specifies the subsystem of the element |
| isInternal | Boolean | Specifies whether the element in internal to SOI or external (part of the environment in the context of the SOI |
| isBoundary | Boolean | Specifies whether the element is boundary (validation purposes) |
| isRedundant | Boolean | Specifies whether the element has secondary redundant units |
| Parent | Reference | Specifies the parent of the element |
| Domain | DomainEnum | Specifies the domain of the element |

| | | |
|---|---|---|
| Characteristics | List of CharacteristicEnum | Specifies the et of unique characteristics of the element |
| Traits | List of UnitTraitEnum | Specifies the set of unique trait of the element |
| Notes | Text | Informal notes to the element |

**Constraints**

- Names in the table shall be unique
- This table defines units. A reference to a unit in other table shall match exactly one name in this table
- Reference to section shall match exactly one name in the sections table
- Reference to subsystem shall match exactly one name in the subsystems table
- Reference to parent shall match exactly one name in the units table

**Validation**

- The value of isBoundary is used for validation. A ReplaceableElement is considered boundary (isBoundary=true) if there exist at least one Exchange for which that ReplaceableElement is either producer or consumer, such that the isInternal properties of the produce and the consumer of that Exchange have opposite values (i.e. the Exchange crosses the boundary between SOI and its environment).

**Semantics**

External - any block or part outside of the assessment boundary; usually owned by the system context directly or indirectly.

Internal - any block or part inside the assessment boundary; usually owned by the context directly or indirectly.

## 7.3.1 Unit Trait Enumeration

Traits (together with Artifacts and Characteristics) provide a shared vocabulary to make assertions about the nature of the elements of SOI. Traits describe the common assertions related to the nature of the replaceable elements of the SOI and their role in the missions and capabilities supported by the SOI. The vocabulary of Traits is coordinated with the vocabulary of Operational Data (hence this package is named “Traits and Data”).

Trait describes the nature of a replaceable unit. Specific subclasses of Trait provide a vocabulary of shared meanings to describe the nature of the replaceable elements of cyber and cyber-physical systems. The vocabulary of Application Traits provides only a useful top-level vocabulary across multiple application domains (e.g. avionics, electrical substations, electrical vehicles, medical devices, etc. can have their own extensive domain ontologies, not addressed by SPECTRA). On the other hand, the vocabulary for Security Traits and Maintenance Traits is more detailed, as these domains are common across various cyber and cyber-physical systems.

### Application Traits

**Table 1 Enumeration Literals for Application Traits**

| Enum Value | Definition |
|---|---|
| UI | User Interface (UI) - user interface component that is used by some human operator. Usually implemented as part of some custom software. |
| CommandControl | Command control - decision making capability, usually implemented as part of some custom application software. |
| Controller | Controller (a control device) - is a component of a cyber-physical system that provides control signals to actuators or other physical |

| | |
|---|---|
| | devices. Usually, a controller is receiving digital information from other parts of the system. |
| Sensor | A sensor train represents the role of an element of a cyber-physical system that detects events or changes in its physical environment and sends the digital information to other elements, usually a processing element of some kind. |
| Actuator | An actuator is a component of a cyber-physical system that produces force, torque, or displacement, when a digital, electrical, pneumatic or hydraulic input is supplied to it. The effect is usually produced in a controlled way. An actuator translates such an input signal into the required form of mechanical energy. It is a special type of transducer.<br>Some systems make distinction between a control device and an actuator, where the control signal is physical rather than digital. Modern systems often involve digitally controlled actuators.<br>An actuator requires a control device (which provides control signal) and a source of energy. The control signal is usually relatively low in energy and may be voltage, electric current, pneumatic, or hydraulic fluid pressure. In the electric, hydraulic, and pneumatic sense, it is a form of automation or automatic control.<br>SPECTRA traits can cover various situations. Several traits can be added to the same replaceable unit or subcomponents, if needed.  See other Application Traits, Command Control, Controller, Sensor, Transducer. |
| Transducer | A transducer is a device that converts energy from one form to another. Usually, a transducer converts a signal in one form of energy to a signal in another. Transducers are often employed at the boundaries of automation, measurement, and control systems, where electrical signals are converted to and from other physical quantities (energy, force, torque, light, motion, position, etc.). An example of a transducer is an antenna, which can convert radio waves (electromagnetic waves) into an electrical signal to be processed by a radio receiver or translate an electrical signal from a transmitter into radio waves. Another example is a voice coil, which is used in loudspeakers to translate an electrical audio signal into sound, and in dynamic microphones to translate sound waves into an audio signal. Sensors and Actuators can be considered as specialized transducers. |
| ProcessingElement | An element that processes (and possibly stored) digital information. A cyber-physical system can be described as a combination of sensors, actuators and processing elements. Some processing elements can be identified as Command Control elements or Controllers. |

### Security Traits

Security Traits entail Security Domain. The elements in this package represent the common security mechanisms for cyber and cyber-physical systems and are aligned with NIST-800-53a “Assessing Security and Privacy Controls in Information Systems and Organizations”.

**Table 2 Enumeration Literals for Security Traits**

| Enum Value | Definition |
|---|---|
| Account Management | A user is an Individual, or (system) process acting on behalf of an individual, authorized to access a system. A user account is the information about the user, often involving the identifier (such as a user name), the authenticator (e.g. a password), the access permissions, etc. Account management mechanism supports managing user accounts. This can be a service, a library, etc. Usually such a mechanism has a specific location within the information pathways of the SOI, in one of the replaceable units. Understanding the location of account management is important for the purposes of cybersecurity risk assessments. |
| AccessControl | In physical security and information security, access control (AC) is the mechanism that provides selective restriction of access to a place, function, capability or resource. Usually such a mechanism has a specific location within the information pathways of the SOI, in one or more of the replaceable units. Understanding the location of access control mechanisms is important for the purposes of cybersecurity risk assessments. |
| AuditMechanism | In the context of cyber and cyber physical systems, the audit (AU) mechanism is responsible for generating and managing audit records. Auditing is a formal, systematic and disciplined approach designed to evaluate and improve the effectiveness of processes and related controls. Audit often supports the non-repudiation objective of cybersecurity. Usually, auditing is governed by professional standards, completed by individuals independent of the process being audited, and normally performed by individuals with one of several acknowledged certifications.<br>Usually, a mechanism that generates records for audit has a specific location within the information pathways of the SOI, in one or more of the replaceable units. Understanding the location of the audit mechanisms is important for the purposes of cybersecurity risk assessments. |
| MonitoringMechanism | In the context of cyber and cyber physical systems, the monitoring (MON) mechanism is a capability to monitor access and other security-relevant events, network traffic, etc. and to generate and manage monitoring records. Monitoring mechanism is different from audit. Monitoring is an on-going process usually directed by management to ensure processes are working as intended. Monitoring is an effective control within a process. Management uses monitoring tools and processes to verify that controls it has implemented are working on a routine basis and that business risks are being identified and addressed. However, because management is checking on their own operations, an inherent conflict is evident in that reporting may reflect what management prefers to report instead of what the actual results portray. Also, in many respects, operational personnel have a better understanding of the data and therefore may create the most appropriate and effective monitoring tools. However, those tools are not necessarily tempered by objectivity or the perspectives/ knowledge of an experienced auditor.<br>Usually, a monitoring mechanism has a specific location within the information pathways of the SOI, in one or more of the replaceable units. Understanding the location of the monitoring mechanisms is important for the purposes of cybersecurity risk assessments. |

| | |
|---|---|
| Crypto | Crypto – A cryptographic asset including algorithms, protocols, certificates, keys, tokens, and secrets. Aligned with Cyclone DX component type "cryptographic asset" |
| BackupMechanism | Backup, or data backup is a copy of computer data taken and stored elsewhere so that it may be used to restore the original after a data loss event. The backup mechanism is a capability to conduct system backups (whether automatically or with the assistance of humans). Backups can be used to recover data after its loss from data deletion or corruption, or to recover data from an earlier time. Backups provide a simple form of IT disaster recovery; Backups are involved in the process of reconstituting a computer system or other complex configuration such as a computer cluster, active directory server, or database server. See also the Restore mechanism. |
| RestoreMechanism | Restore is the term used for creating original data from a backup. Backups are involved in the process of reconstituting a computer system or other complex configuration such as a computer cluster, active directory server, or database server. See also Backup mechanism. |
| RedundantSecondaryUnit | Redundancy refers to the duplication of critical components or functions of a system with the intention of increasing reliability. Identifying a certain unit as a Redundant Secondary Unit is important for correct interpretations of the model of SOI. |
| IdentificationAuthentication | Identification and authentication (IA) of users is a mechanism involved in verifying identity of a user of SOI. Authentication (FIPS 200, NIST-800-53) is a process of verifying the identity of a user, process, or device, often as a prerequisite to allowing access to resources in a system. The process of authentication may be used to validate personal identifier, verifying the authenticity of a website or a protocol endpoint using a digital certificate, ensuring that a product or a document is not counterfeit, etc. IA might include an explicit examination of any key-based, “password-less” login capability and potential risks inherent from any deficiency in key management, account management, system and boundary protection, physical and environmental protection, and other safeguards to prevent identification and authentication bypass by unauthorized users or processes acting on behalf of users.<br>A multi-factor authentication mechanism (NIST-SP-800-63-3, NIST-800-53) is an authentication system or an authenticator that requires more than one authentication factor for successful authentication. Multi-factor authentication can be performed using a single authenticator that provides more than one factor or by a combination of authenticators that provide different factors.<br>The three authentication factors are something you know, something you have, and something you are. |

## Maintenance Traits

**Table 3 Enumeration Literals for Maintenance Traits**

| Enum Value | Definition |
|---|---|
| DataLoader | Data loader is a utility program that receives data that represents an artifact image (or a patch) and updates the corresponding artifact in the SOI. For example, patches for proprietary software are typically distributed as executable files instead of source code. When executed these files load a program into memory which manages the installation of the patch code into the target program(s) on disk. |

|  |  |
|---|---|
|  | Patches for other software are typically distributed as data files containing the patch code. These are read by a patch utility program which performs the installation. This utility modifies the target program's executable file—the program's machine code—typically by overwriting its bytes with bytes representing the new patch code. |
| TestEquipment | Test and maintenance equipment is connected to the SOI (at least at some Mission Phases), therefore it is important to identify such elements in the model of the SOI. |
| DataRecorder | The Data Recorder element (sometimes referred to as a system monitor) is very similar to Monitoring Mechanism in the Security Domain. A system monitor is a hardware or software component used to monitor system resources and performance in a computer system and generate Maintenance records for the benefits of the maintenance team and their processes. |

## 7.4 Assemblies Table

An Independent Configuration is an arrangement of Replaceable Units in some different way in addition to the master Assembly configurations.  An Independent Configuration also refers to an identifiable "place" in the digital pathways of the SOI. An Independent Configuration usually provides the scope of common organization and technical controls.

In a SysML model this is usually represented as a block with an IBD, that may own ports and connectors that constitute segments of SPECTRA conveying elements (channels). Independent configurations may overlap with functional configurations by extent. Functional configurations are considered primary/master decomposition of the system, while independent configurations provide a multitude of secondary overlapping decompositions (arrangements of the same replaceable units and channels).

Independent configuration is an abstract element, its concrete subclasses are organization, facility/building, enclave, subnet, group. "Independent configurations" help define the hierarchy of security assets and asset owners that are available to implement security, security constraints (policy, guidance, laws and regulations) and details where they are located (security enclaves).

**Table Identification**
SPECTRA Assemblies for system <SOI name>

**Table Columns**

| Column title | Column data | Description |
|---|---|---|
| Name | Identifier | Specifies the identifying name of the element |
| Purpose | Text | Provides the description of the element |
| Category | AssemblyCategoryEnum | Specifies the meaning of the element by referencing a specific enumeration literal |
| isInternal | Boolean | Specifies whether the element is internal |
| Domain | DomainEnum | Specifies the domain of the element |
| Characteristics | List of CharacteristicEnum | Specifies the set of unique characteristics of the element |
| Nodes unique | List of Reference | Specifies the nodes that are referenced by this element and are not used to establish relationships to other assemblies |

| | | |
|---|---|---|
| Nodes shared | List of Reference | Specifies the nodes that are referenced by this element and are shared with other assemblies |
| Notes | Text | Informal notes to the element |

**Constraints**

- Names in the table shall be unique
- This table defines assemblies. A reference to an assembly in other tables shall match exactly one name in this table
- Reference to a unit shall match exactly one name in the units table or in the assemblies table

**Semantics**

### 7.4.1 Assembly Category Enumeration

**Table 4 Enumeration Literals for Assembly Category**

| Enum Value | Definition |
|---|---|
| Group | Group - generic arrangement of replaceable nodes |
| Organization | Organization - arrangement of replaceable nodes that has common organizational controls |
| Enclave | Enclave - arrangement of replaceable units that has common organizational and technical controls. An enclave is defined as a collection of information systems connected by one or more internal networks under the control of a single authority and security policy. |
| Subnet | Subnet - arrangement of replaceable units around a common subnet (possibly in an enclave) that has common security policy and other constraints, common configuration, as well as common organization and technical controls, such as a firewall, etc. |
| Facility | Facility - arrangement of replaceable nodes that has common physical controls |
| Vehicle | An arrangement of replaceable units that is mobile platform, e.g. an aircraft or a truck. |
| Zone | An (often spacial) arrangement of replaceable units, important for physical security |
| Container | An (often spacial) arrangement of replaceable units, such as a locked cabinet. |

## 7.5 Artifacts Table

Artifacts (together with Traits and Characteristics) provide a shared vocabulary to make assertions about the nature of the elements of SOI. Artifacts describe the common assertions related to the nature of the replaceable elements of the SOI, their role in missions and capabilities supported by the SOI, and their place in the supply chain enabling the SOI. Artifacts address assertions related to the unique technologies of the replaceable elements.

Artifact - Artifact represents a unique attack opportunity and is defined as a combination of a characteristic and a reference to a unique suppliable artifact type called "Artifact image". Characteristics are defined as concrete subclasses of Artifact. From the supply chain perspective (BOM/SBOM), an artifact is something owned by an attackable unit (hardware, firmware, custom software, operating system, media, etc). Artifact is associated with a uniquely identified suppliable element called "artifact image". An artifact image can be associated with multiple replaceable units. An artifact image is further associated with a supplier and a CPE code. Note that several other important elements associated with a replaceable unit can be derived from the model, such as facility, operational procedures, personnel, policy, controls, etc.  Specific artifact subclasses represent individual attack opportunities as well as frequently used common suppliable element categories.

**Table Identification**
SPECTRA Artifacts for system <SOI name>

**Table Columns**

| Column title | Column data | Description |
|---|---|---|
| Name | Identifier | Specifies the identifying name of the element |
| Purpose | Text | Provides the description of the element |
| Category | ArtifactCategoryEnum | Specifies the meaning of the element by referencing a specific enumeration literal |
| Unit | Reference | Specifies the unit that deploys this artifact |
| Supplier | Reference | Provides a reference to supplier of this artifact |
| CPE | String | Specifies the MITRE Common Platform enumeration (CPE) code for the element |
| isVirtualized | Boolean | Specifies whether the element is virtualized |
| Host | Reference | Specifies the cloud infrastructure (for virtualized artifacts) |

**Constraints**

- Names in the table shall be unique
- Reference to a unit shall match exactly one name in the units table
- Reference to a supplier shall match exactly one name in the agents table, where AgentCategoryEnum=Supplier
- CPE shall be a valid MITRE Common Platform Enumeration (CPE) string
- Reference to a host shall match exactly one name in the artifacts table, where ArtifactCategoryEnum=CloudInfrastructure

## 7.5.1 Artifact Category Enumeration

Traits (together with Artifacts and Characteristics) provide a shared vocabulary to make assertions about the nature of the elements of SOI. Traits describe the common assertions related to the nature of the replaceable elements of the SOI and their role in the missions and capabilities supported by the SOI. The vocabulary of Traits is coordinated with the vocabulary of Operational Data (hence this package is named "Traits and Data").

### Computing Element

Computing element is an abstract element that represents computing hardware. Detailed semantics is provided by concrete subclasses.

**Table 5 Enumeration Literals for Computing Elements**

| Enum Value | Definition |
|---|---|
| DSP | Dsp - digital signal processing unit. A digital signal processor (DSP) is a specialized microprocessor chip, with its architecture optimized for the operational needs of digital signal processing. The digital signals processed in this manner are a sequence of numbers that represent samples of a continuous variable in a domain such as time, space, or frequency. |
| Hardware | Hardware - Computer hardware includes the physical parts of a computer, such as the central processing unit (CPU), random access memory (RAM), motherboard, computer data storage, graphics card, sound card, and computer case. It includes external devices such as a monitor, mouse, keyboard, and speakers. Aligned with Cyclone DX component type “device” |
| Firmware | Firmware - firmware is software that provides low-level control of computing device hardware. For a relatively simple device, firmware may perform all control, monitoring and data manipulation functionality. For a more complex device, firmware may provide relatively low-level control as well as hardware abstraction services to higher-level software such as an operating system.<br>Firmware is found in a wide range of computing devices including personal computers, phones, home appliances, vehicles, computer peripherals and in many of the digital chips inside each of these larger systems.<br>Firmware is stored in non-volatile memory – either read-only memory (ROM) or programmable memory such as EPROM, EEPROM, or flash. Changing a device's firmware stored in ROM requires physically replacing the memory chip – although some chips are not designed to be removed after manufacture. Programmable firmware memory can be reprogrammed via a procedure sometimes called flashing. Aligned with Cyclone DX component type “firmware” |
| MobileDevice | Mobile Device (NIST-800-53) is a portable computing device that has a small form factor such that it can easily be carried by a single individual; is designed to operate without a physical connection (e.g., wirelessly transmit or receive information); possesses local, non-removable data storage; and is powered on for extended periods of time with a self-contained power source. Mobile devices may also include voice communication capabilities, on-board sensors that allow the device to capture (e.g., photograph, video, record, or determine location) information, and/or built-in features for synchronizing local data with remote locations. Examples include smartphones, tablets, and e-readers.<br>Mobile devices often emphasize wireless networking to both the local area networks and Internet and to other devices in their vicinity. |

### Cloud Infrastructure

**Table 6 Enumeration Literals for Cloud Infrastructure**

| Enum Value | Definition |
|---|---|
| CloudInfrastructure | Cloud Infrastructure is an element that represents the hosting environment for virtualized infrastructures. |

### Software

Software is an abstract element that represents computer programs that instruct the execution of a computing device. Detailed semantics relevant to cyber and cyber-physical systems are provided by the subclasses.

**Table 7 Enumeration Literals for Software**

| Enum Value | Definition |
|---|---|
| OperatingSystem | Operating system - An operating system (OS) is system software that manages computer hardware and software resources and provides common services for custom computer programs and other system software. Aligned with Cyclone DX component type “operating-system” |
| Application | Application- application/custom software. artifact comprising only instructions for computer hardware, excluding operating system software, database and other common utility systems. Aligned with Cyclone DX component type “application” |
| Database | Database - database, SQL or no-SQL relational, key-valued, graph, etc. e.g. Oracle, Postgres, Redis, etc. |
| NetworkController | Network controller - telecommunication network technologies based on physically wired, optical, and wireless radio-frequency methods that may be arranged in a variety of network topologies. The transmission media (often referred to in the literature as the physical medium) used to link devices to form a computer network include electrical cable, optical fiber, and free space. A network interface controller (NIC) is computer hardware that connects the computer to the network media and has the ability to process low-level network information. For example, the NIC may have a connector for plugging in a cable, or an aerial for wireless transmission and reception, and the associated circuitry. |
| Framework | Framework - A software framework. Aligned with Cyclone DX component type "framework" |
| Platform | Platform - A runtime environment which interprets or executes software. This may include runtimes such as those that execute bytecode or low-code/no-code application platforms. Aligned with Cyclone DX component type "platform" |
| Hypervisor | Hypervisor element represents a type of computer software that creates and runs virtual machines. |
| ContainerManager | Container manager - containerization is operating system-level virtualization or application-level virtualization over multiple network resources so that software applications can run in isolated user spaces called containers in any cloud or non-cloud environment, regardless of type or vendor. containerization technology has been widely adopted by cloud computing platforms like Amazon Web Services, Microsoft Azure, Google Cloud Platform, and IBM Cloud. Container orchestration or container management is mostly used in the context of application containers. Implementations providing such orchestration include Kubernetes and Docker swarm. |
| Library | Library – A software library. Aligned with Cyclone DX component type "library" |

### Network Element

The nodes of a computer network can include personal computers, servers, networking hardware, or other specialized or general-purpose hosts. Computer networks may be classified by many criteria, including the transmission medium used to carry signals, bandwidth, communications protocols to organize network traffic, the network size, the topology, traffic control mechanisms, and organizational intent.

**Table 8 Enumeration Literals for Network Elements**

| Enum Value | Definition |
|---|---|
| Switch | Network bridges and network switches are distinct from a hub in that they only forward frames to the ports involved in the communication whereas a hub forwards to all ports. Bridges only have two ports,Abut a switch can be thought of as a multi-port bridge. Switches normally have numerous ports, facilitating a star topology for devices, and for cascading additional switches. |
| Firewall | A firewall is a network device or software for controlling network security and access rules. Firewalls are inserted in connections between secure internal networks and potentially insecure external networks such as the Internet. Firewalls are typically configured to reject access requests from unrecognized sources while allowing actions from recognized ones. |
| Router | A router is an internetworking device that forwards packets between networks by processing the addressing or routing information included in the packet. The routing information is often processed in conjunction with the routing table. A router uses its routing table to determine where to forward packets and does not require broadcasting packets which is inefficient for very big networks. |
| Bridge | Network bridges and network switches are distinct from a hub in that they only forward frames to the ports involved in the communication whereas a hub forwards to all ports. Bridges only have two ports but a switch can be thought of as a multi-port bridge. Switches normally have numerous ports, facilitating a star topology for devices, and for cascading additional switches. |
| Repeater | A repeater is an electronic device that receives a network signal, cleans it of unnecessary noise and regenerates it. The signal is retransmitted at a higher power level, or to the other side of obstruction so that the signal can cover longer distances without degradation. |
| Hub | An Ethernet repeater with multiple ports is known as an Ethernet hub. In addition to reconditioning and distributing network signals, a repeater hub assists with collision detection and fault isolation for the network. Hubs and repeaters in LANs have been largely obsoleted by modern network switches. |
| Modem | Modems (modulator-demodulator) are used to connect network nodes via wire not originally designed for digital network traffic, or for wireless. To do accomplish this, one or more carrier signals are modulated by the digital signal to produce an analog signal that can be tailored to give the required properties for transmission. |

### Storage Media

**Table 9 Enumeration Literals for storage Media**

| Enum Value | Definition |
|---|---|
| StorageMedia | Storage Media - Computer data non-volatile secondary storage refers to a technology consisting of computer components and recording media that are used to retain digital data. Secondary storage (also known as external memory or auxiliary storage) differs from primary storage in that it is not directly accessible by the CPU. The computer usually uses its input/output channels to access secondary storage and transfer the desired data to primary storage (part of hardware). Secondary storage is non-volatile (retaining data when its power is shut off). In modern computers, hard disk drives (HDDs) or solid-state drives (SSDs), rotating optical storage devices, such as CD and DVD drives are usually used as secondary storage. Other examples of secondary storage technologies include USB flash drives, floppy disks, magnetic tape, paper tape, punched cards, and RAM disks. |

## 7.6 Datatypes Table

Conveyable Element - any type of thing that is conveyed between replaceable units that allows a flow of information or other items. Information Type – a digital information item that can flow between replaceable units and/or be processed by replaceable units and/or be stored at one or more replaceable units.

While the Information Pathways package describes the Core Assertions related to the information bearers of the SOI, the Conveyable Elements addresses the elements of Digital Information (as well as some physical items) involved in the missions and capabilities supported by the SOI.

This package addresses the “WHAT/To WHAT” clauses in assertions about the SOI. For example, “Failure of the Encrypted VHF radio causes exposure of the position information”. Here “position information” is the “WHAT/To WHAT” clause, the “Encrypted VHF Radio” is the “WHERE” clause, and the “Failure of…” is the “Impacting WHAT” clause. “exposure of …” is a “consequence cause”.

**Table Identification**
SPECTRA Datatypes for system <SOI name>

**Table Columns**

| Column Title | Column Data | Description |
|---|---|---|
| Name | Identifier | Specifies the identifying name of the element |
| Purpose | Text | Provides the description of the element |
| Category | DatatypeCategoryEnum | Specifies the meaning of the element by referencing a specific enumeration literal |
| Domain | DomainEnum | Specifies the domain of the element |
| Characteristics | List of CharacteristicEnum | Specifies the set of unique characteristics of the element |
| Traits | List of DataTraitEnum | Specifies the set of unique traits of the element |
| Inverse | Reference | Specifies the inverse datatype, if applicable (e.g a response to a request) |
| Datatypes | List of Reference | Specifies the set of unique datatype from which the current datatype is composed, if applicable |
| Classification | ClassificationEnum | Specifies the classification markup of the datatype |
| Confidentiality | ImpactLevelEnum | Specifies the sensitivity to loss of confidentiality, if applicable |
| Integrity | ImpactLevelEnum | Specifies the sensitivity to loss of integrity, if applicable |

| | | |
|---|---|---|
| Availability | ImpactLevelEnum | Specifies the sensitivity to loss of availability, if applicable |
| Privacy | ImpactLevelEnum | Specifies the sensitivity to loss of privacy, if applicable |
| Safety | ImpactLevelEnum | Specifies the sensitivity to loss of safety, if applicable |
| isInput | Boolean | Specifies whether the element is an input to SOI from its environment |
| isOutput | Boolean | Specifies whether the element is an output from SOI to its environment |
| isProcessed | Boolean | Specifies whether the element is processed by the SOI |
| isStored | Boolean | Specifies whether the element is stored by the SOI |
| Notes | Text | Informal notes to the element |

**Constraints**

- Names in the table shall be unique
- Inverse reference, when present, shall match exactly one name in the datatypes table
- A reference to child datatype in Datatypes column shall match exactly one name in the datatypes table
- A reference to child datatype shall not match the name in the current row
- Reference in the Datatypes list shall be unique

**Validation**

- The value of isInput is used for validation. A datatype is considered input (isInput=true) if there exists at least one exchange where producer is external and consumer is internal and the data of the exchange matches the name of the datatype. The value in the isInput column shall match the value derived for the rest of the assertions, based on the isInternal values of the units.
- The value of isOutput is used for validation. A datatype is considered output (isOutput=true) if there exists at least one exchange where consumer is external and producer is internal and the data of the exchange matches the name of the datatype. The value in the isOutput column shall match the value derived for the rest of the assertions, based on the isInternal values of the units.
- The value of isProcessed is used for validation. A datatype is considered processed (isProcessed=true) if there exists at least one flow (of a functional thread) where isProcessed=true and the data of the exchange matches the name of the datatype. The value in the isProcessed column shall match the value derived for the rest of the assertions, based on the definitions of the functional threads.
- The value of isStored is used for validation. A datatype is considered output (isStored=true) if there exists at least one datastore where the data of the datastore matches the name of the datatype. The value in the isStored column shall match the value derived for the rest of the assertions, based on the definitions of the datastores.

**Semantics**

## 7.6.1 Datatype Category Enumeration

**Table 10 Enumeration Literals for Datatype Category**

| Enum Value | Definition |
|---|---|
| Information | |
| Message | |
| Resource | Resource - value or block or port type, activity pin; something that is consumed by attackable units as they perform their functions (and accomplish missions as part of a system, and a system-of-systems), e.g. fuel, money, time-to-overhaul the engine, etc. Depletion of resources may be an objective of an attacker, often not well-accounted for by models. Various Physical characteristics can be applied to a resource. A resource should not be confused with cyber digital information, which is a distinct subclass of the Conveyable element, central to the descriptions of cyber systems. |

| | |
|---|---|
| Signal | |
| ControlFlow | |

## 7.6.2 Datatype Trait Enumeration

Traits (together with Artifacts and Characteristics) provide a shared vocabulary to make assertions about the nature of the elements of SOI. Traits describe the common assertions related to the nature of the replaceable elements of the SOI and their role in the missions and capabilities supported by the SOI. The vocabulary of Traits is coordinated with the vocabulary of Operational Data (hence this package is named “Traits and Data”).

### Application Data

**Table 11 Enumeration Literals for Application Data Traits**

| Enum Value | Definition |
|---|---|
| Command | Command – digital information that is interpreted by the recipient as a command. Often an element of a cyber-physical system, identified as a Command Control, or a Controller sends Command data to a Controller or an Actuator. The application data going in the opposite direction is often identified as Status. |
| Status | Status – digital information that is interpreted by the receiver as a status report. Often an element of a cyber-physical system, identified as a Command Control, or a Controller sends Command data to a Controller or an Actuator and would expect a Status. |
| Request | Request – digital information that is interpreted by the recipient as a request/query. Often an element of a cyber or a cyber-physical system, identified as a Processing Element or Command Control, or a Controller sends Request data to another Processing Element, and would expect a Content (and possibly a Status) back. |
| Content | Content – digital information that is interpreted by the receiver as an application-specific content (e.g. in response to a request). Often an element of a cyber or a cyber-physical system, identified as a Processing Element or Command Control, or a Controller sends Request data to another Processing Element, and would expect a Content (and possibly a Status) back. |
| Alert | Alert – digital information that is interpreted by the receiver as a notification of an event. |
| SensorData | Sensor Data – digital information that is interpreted by the recipient as a digital description of the physical environment of the SOI, often produced directly by some Sensor. Often an element of a cyber-physical system, identified as a Sensor, sends Sensor data to some Processing Element, Controller or Command Control, possibly as the result of a Request. |

### Security Data

**Table 12 Enumeration Literals for Security Data Traits**

| Enum Value | Definition |
|---|---|
| UserIdentifier | Identifier (FIPS 201-2, NIST-800-53) is a unique data used to represent a person's identity and associated attributes. A name or a card number are examples of identifiers. A unique label used by a system to indicate a specific entity, object, or group. Identifier is a form of digital identity. A digital identity is data stored on computer systems relating to an individual, organization, application, or device. For individuals, it involves the collection of personal data that is essential for facilitating automated access to digital services, confirming one's identity on the internet, and allowing digital systems to manage interactions between different parties. |
| Authenticator | Authenticator (NIST-800-53) is something that the claimant possesses and controls (typically a cryptographic module or password) that is used to authenticate the claimant's identity. |
| Credential | Credential (NIST SP 800-634-3, NIST-800-53) is an object or data structure that authoritatively binds an identity, via an identifier or identifiers, and (optionally) additional attributes, to at least one authenticator possessed and controlled by a subscriber. |
| AuditRecord | Audit record is security related information generated from certain selected event types related to the SOI. Audit records are produced by the Audit mechanism. Audit records are often compiled into a system-wide audit trail that is time-correlated within organizational tolerances. Protection of the audit records may involve various cryptographic mechanisms. |
| CryptoKey | A key in cryptography is a piece of information, usually a string of numbers or letters that are stored in a file, which, when processed through a cryptographic algorithm, can encode or decode cryptographic data. Based on the used method, the key can be different sizes and varieties, but in all cases, the strength of the encryption relies on the security of the key being maintained.<br>The security of a key is dependent on how a key is exchanged between parties. Establishing a secured communication channel is necessary so that outsiders cannot obtain the key. A key establishment scheme (or key exchange) is used to transfer an encryption key among entities. Key agreement and key transport are the two types of a key exchange scheme that are used to be remotely exchanged between entities. |
| Certificate | In cryptography, a public key certificate, also known as a digital certificate or identity certificate, is an electronic document used to prove the validity of a public key. The certificate includes the public key and information about it, information about the identity of its owner (called the subject), and the digital signature of an entity that has verified the certificate's contents (called the issuer). If the device examining the certificate trusts the issuer and finds the signature to be a valid signature of that issuer, then it can use the included public key to communicate securely with the certificate's subject. |
| MonitoringData | Data generated by a monitoring mechanism. |
| UserAccount | A user is an Individual, or (system) process acting on behalf of an individual, authorized to access a system. A user account is the information about the user, often involving the identifier (such as a user name), the authenticator (e.g. a password), the access permissions, etc. |

| | |
|---|---|
| Backup | Data that is generated by a backup mechanism and that is intended for a subsequent restore of the system. |
| AccessToken | In computer systems, an access token contains the security credentials for a login session and identifies the user, the user's groups, the user's privileges, and, in some cases, a particular application. An access token is an object encapsulating the security identity of a process or thread. A token is used to make security decisions and to store tamper-proof information about some system entity.<br>An access token is usually generated by the logon service when a user logs on to the system and the credentials provided by the user are authenticated against the authentication database. The authentication database contains credential information required to construct the initial token for the logon session, including its user id, primary group id, all other groups it is part of, and other information. |

### Maintenance Data

**Table 13 Enumeration Literals for Maintenance Data Traits**

| Enum Value | Definition |
|---|---|
| MaintenanceRecord | The Maintenance Record element is similar to the Monitoring Data element in the Security domain. Monitoring Data is usually generated by a Data Recorder component (as the Maintenance trait). |
| ArtifactImage | Artifact image is data that is intended to be used to modify an existing software resource such as a program or a file, often to fix bugs and security vulnerabilities (referred to as a patch). An artifact image may represent a hotfix (or a hot-swap image), a patch, a major or a minor release of the software or firmware. |
| Configuration | Configuration (usually a configuration file) is data used to configure the parameters and initial settings for some computer programs or applications, server processes and operating system settings. Some computer programs only read their configuration files at startup. Others periodically check the configuration files for changes. Managing, distributing and updating configurations is part of the maintenance domain for SOI. |

## 7.6.3 Classification Enumeration

**Table 14 Enumeration Literals for Classification**

| Enum Value | Definition |
|---|---|
| TS | Datatype is classified as “top secret” |
| S | Datatype is classified as “secret” |
| FOUO | Datatype is classified as “for official use only” |
| U | Datatype is “unclassified” |

## 7.7 Datastores Table

A Datastore is a repository for persistently storing and managing collections of data, which include not just repositories such as databases, but also simpler store types such as simple files, emails, logs, etc. As part of the Core Assertions, A Datastore represents “data at rest”.

**Table Identification**
SPECTRA Datastores for system <SOI name>

**Table Columns**

| Column Title | Column Data | Description |
|---|---|---|
| Name | Identifier | Specifies the identifying name of the element |
| Purpose | Text | Provides the description of the element |
| Unit | Reference | Specifies the unit that owns the element |
| Datatype | Reference | Specifies the datatype that is associated with the element |
| isInternal | Boolean | Specifies whether the element in internal to SOI |
| isEncrypted | Boolean | Specifies whether the datastore is encrypted |
| Notes | Text | Informal notes to the element |

**Constraints**

- Names in the table shall be unique
- Reference to a unit shall match exactly one name in the units table
- Reference to a datatype shall match exactly one name in the datatypes table

**Validation**

- The value of isInternal is used for validation. A datastore is considered internal (isInternal=true) if the unit that owns the datastore is internal (isInternal=true). The value in the isInternal column shall match the value derived for the definitions of the units.

**Semantics**

Encryption is the process of encoding information in a way that, ideally, only authorized parties can decode. This process converts the original representation of the information, known as plaintext, into an alternative form known as ciphertext. Applies to Channel, Flow, Datastore (however encryption is achieved by the producer and consumer, not by the channel), but one of the protocols of the carrier interface may support encryption, which also entails exchange of crypto keys).

## 7.8 Channels Table

Channel - any connector or an association between replaceable units that allows a flow of information or other items; conveying element has at least one producer and at least one consumer; a special case is a "local link" where one replaceable unit is both the producer and the consumer; a conveying element supports a certain protocol stack that may involve multiple levels; conveying element is usually attackable, defendable and configurable. Together with replaceable units conveying elements describe the information pathways in the SOI (tangible/attackable connectors between tangible/attackable places). In a SysML model channels may be multi-segmented (multi-leg) connections through ports and parts.

SPECTRA has a two-fold complementary perspective on the SOI: 1) a set of parts (replaceable units, processing elements), plugged into unique places within the communication fabric, and 2) a communication fabric (constituted by Conveying Elements) that defines the “sockets” for the replaceable units. Conveying Elements and Replaceable Units define the “technical surface” of the SOI.

This technical surface identifies the exact set of parts under assessment, helps identify the end-to-end data flows through the system, especially those that originate outside of the SOI and thread through the parts of the SOI and those that originate inside of the SOI and thread outside of the SOI. From the CRA perspective, this information is essential to identify possible attack paths for the SOI.

**Table Identification**
SPECTRA Channels for system <SOI name>

**Table Columns**

| Column Title | Column Data | Description |
|---|---|---|
| Name | Identifier | Specifies the identifying name of the element |
| Purpose | Text | Provides the description of the element |
| Category | ChannelCategoryEnum | Specifies the meaning of the element by referencing a specific enumeration literal |
| Domain | DomainEnum | Specifies the domain of the element |
| Characteristics | List of CharacteristicEnum | Specifies the list of unique characteristics of the element |
| isBoundary | Boolean | Specifies whether the element is crossing the boundary of SOI |
| isWireless | Boolean | Specifies whether the channel is wireless |
| Units | List of unique Reference | Specifies the unique list of units connected by the channel |
| Transport | Identifier | Specifies the transport protocol of the channel |
| Network | Identifier | Specifies the network protocol of the channel |
| Network Interface | Identifier | Specifies the network interface of the channel |
| Notes | Text | Informal notes to the element |

**Constraints**

- The names in the table shall be unique.
- A reference to a unit shall match exactly one name in the units table

**Validation**

- The value of isBoundary is used for validation. A channel is considered boundary (isBoundary=true) if there exist at least two units in the list of units connected by the channel, that have opposite isInternal values. The value in the isBoundary column shall match the value derived for the definitions of the units.

**Semantics**

A Carrier Interface is a representation of the essential protocols supported by a Conveying Element. A protocol is a system of rules that allows two or more entities to transmit information via any variation of a physical quantity. There are two reference frameworks for describing network protocols: the OSI Reference Model and the TCP/IP Conceptual Layers. The Carrier Interface follows some industry best practices, and focuses at the Transport, Network and Network Interface layers of the TCP/IP stack, which are aligned with the levels 4,3 and a combination of layers 2 and 1 in the OSI Reference Model, respectively.

A transport protocol can be identified as e.g. “TCP”, “UDP”,
A network protocol can be identified as e.g. “MIL-STD-188-164A”, “CANopen”, “TCP/IP”, “PWM”, “IPV4”, “IPv6”, “IPv4/IPv6”, “RTP”, “MIL-STD-1553B”, “RS-232”, “NMEA 0183”, “ARINC 429”
A network interface can be identified as e.g. “RF for SATCOM”, “CAN Bus”, “Ethernet”, “MIL-STD-1553”, “SONET/SDH”, “IEEE 1394”, “Wi-Fi”

### 7.8.1 Channel Category Enumeration

**Table 15 Enumeration Literals for Channel Category**

| Enum Value | Definition |
|---|---|
| Link | Link - a channel between exactly one producing and one consuming Replaceable Element. |
| Bus | Bus - a channel among one or more producing and/or one or more consuming Replaceable Elements. For example, a bus may have a single producer and multiple consumers, or multiple producers and a single consumer, or multiple producers and multiple consumers. One Replaceable Element can be both a producer and a consumer for the same bus. |
| Network | A Network (as defined in NIST-800-53) is a system implemented with a collection of connected components. Such components may include routers, hubs, cabling, telecommunications controllers, key distribution centers, and technical control devices.<br>A Network involves a communication medium connecting one or more computing devices and involving common communication protocols over digital interconnections using telecommunications technologies based on physically wired, optical or wireless radio-frequency methods and arranged in a variety of network topologies. For example, an Ethernet local area network is a Network. One Replaceable Element can be both a producer and a consumer for the same network. |

## 7.9 Exchanges Table

Exchange - a representation of a type of item that flows between a single producer and a single consumer, involving an information type or some other (physical) kind of a Conveyable Element, a Conveying Element and a specific Carrier Interface supported by the Conveying Element. While cyber systems involve "information flows", in a cyber physical system one may find a broader range of "item flows" where items conveyed from the producer to the consumer may be physical. A flow is described as a combination of a specific channel (network, bus or link), a unique (logical) information type and a unique (physical) protocol. A channel may support multiple communication protocols.

The Conveyable Element of the Flow shall be marked by a specialization of the Operational Data stereotype. For a cyber-physical system, it can also be a Resource. SysML models of cyber and cyber-physical systems shall avoid using Carrier Data as Conveyable Element in Flows. The Carrier Interface for the Conveying Element (channel) shall be used instead.

**Table Identification**
SPECTRA Exchanges for system <SOI name>

**Table Columns**

| Column Title | Column Data | Description |
|---|---|---|
| From | Reference | Specifies the unit that produces the exchange |
| To | Reference | Specifies the unit that consumes the exchange |
| Datatype | Reference | Specifies the datatype conveyed by the exchange |
| Channel | Reference | Specifies the channel used by the exchange |
| Purpose | Text | Provides the description of the element |
| Domain | DomainEnum | Specifies the domain of the element |
| Characteristics | List of CharacteristicEnum | Specifies the set of unique characteristics of the element |
| isInternal | Boolean | Specifies whether the exchange is internal to the SOI |
| isBoundary | Boolean | Specifies whether the exchange crosses the boundary of the SOI |
| isInput | Boolean | Specifies whether the exchange provides an input into the SOI from its environment |

| | | |
|---|---|---|
| isEncrypted | Boolean | Specifies whether the exchange is encrypted |
| Notes | Text | Informal notes to the element |

**Constraints**

- A reference to a unit in column from, to shall match exactly one name in units table
- A reference to datatype shall match exactly one name in the datatypes table

**Validation**

- The value of isInternal is used for validation. An exchange is considered internal (isInternal=true) if both the produce and the consumer units are internal (isInternal=true). The value in the isInternal column shall match the value derived for the definitions of the units.
- The value of isBoundary is used for validation. An exchange is considered boundary (isBoundary=true) if producer and consumer units have opposite isInternal values. The value in the isBoundary column shall match the value derived for the definitions of the units.
- The value of isInput is used for validation. An exchange is considered input (isInput=true) if both the producer unit is external (isInternal=false) and the consumer unit is internal (isInternal=true). The value in the isInput column shall match the value derived for the definitions of the units.

**Semantics**

Encryption is the process of encoding information in a way that, ideally, only authorized parties can decode. This process converts the original representation of the information, known as plaintext, into an alternative form known as ciphertext. Applies to Channel, Flow, Datastore (however encryption is achieved by the producer and consumer, not by the channel), but one of the protocols of the carrier interface may support encryption, which also entails exchange of crypto keys).

## 7.10 Capabilities Table

A Capability is a strategic element that refers to an enterprise's ability to achieve a desired effect realized through a combination of ways and means (e.g. capability configuration that involves a set of collaborating performers and channels) along with specified measures. In a SysML model, a capability can be represented as an activity, a block or a use case. A capability may depend on other capabilities, and usually involves collaboration and flows among multiple replaceable units

**Table Identification**
SPECTRA Capabilities for system <SOI name>

**Table Columns**

| Column Title | Column Data | Description |
|---|---|---|
| Name | Identifier | Specifies the identifying name of the element |
| Purpose | Text | Provides the description of the element |
| Parent | Reference | Specifies the parent capability, if applicable |
| DependsOn | List of Reference | Specifies the capabilities that the current one depends on, if applicable |
| Functions | List of Reference | Specifies the list of unique function that implement this capability |
| Confidentiality | ImpactLevelEnum | |
| Integrity | ImpactLevelEnum | |
| Availability | ImpactLevelEnum | |
| Privacy | ImpactLevelEnum | |
| Safety | ImpactLevelEnum | |
| Notes | Text | |

**Constraints**

- Names in the tables shall be unique
- Reference to parent capability shall match exactly one name in this table.
- Reference to the parent capability shall not match the name of the current row
- Reference in DependsOn column shall match exactly one name in this table
- All references in a list in DependsOn column shall be unique
- No name in the list in DependsOn column shall match the name of the current row
- Reference in Functions column shall match exactly one name in the functions able
- All references in a list in Functions column shall be unique
- No name in the list in Functions column shall match the name of the current row

**Semantics**

## 7.11 Threads Table

While the Information Pathways package describes the Core Assertions related to the information bearers of the SOI, and the Conveyable Elements addresses the elements of Digital Information (as well as some physical items) conveyed by the parts of the SOI through Conveying Elements, this packages the relative timing of activities performed by the SOI as it supports missions and capabilities. This package identifies the Core Assertions in terms of functional threads through the system, as collections of Flows ordered by "happens before".

This package addresses the "WHEN" clause.

**Table Identification**
SPECTRA Threads for system <SOI name>

**Table Columns**

| Column Title | Column Data | Description |
|---|---|---|
| Name | Identifier | Specifies the identifying name of the element |
| Purpose | Text | Provides the description of the element |
| Domain | DomainEnum | Specifies the domain of the element |
| Notes | Text | Informal notes to the element |

**Constraints**

- Names in the table shall be unique
- This table defines threads. A reference to an thread in other tables shall match exactly one name in this table

**Semantics**

## 7.12. Flows Table

**Table Identification**
SPECTRA Flows for system <SOI name>

**Table Columns**

| Column Title | Column Data | Description |
|---|---|---|
| Thread | Reference | Specifies the functional thread that is the context for this flow/step |
| Seq | Integer | Specifies the sequence number of the flow in the thread |
| From | Reference | Specifies the producer unit for the flow |
| Sender | Reference | Specifies the specific function that sends the data |

| | | |
|---|---|---|
| To | Reference | Specifies the consumer unit of the flow |
| Receiver | Reference | Specifies the specific function that receives the data |
| Datatype | Reference | Specifies the datatype associates with the flow |
| Authentication | Boolean | Specifies whether the flow involves authentication |
| AccessControl | Boolean | Specifies whether the flow involves access control |
| isProcessed | Boolean | Specifies whether the data is processed by the flow |
| isStored | Boolean | Specifies whether the flow involves stored data |
| Notes | Text | Informal notes to the element |

**Constraints**

- A reference to a unit is columns From, To shall match exactly one name in the units table
- A reference to sender, receiver shall match exactly one name in the functions table
- A reference to a datatype shall match exactly one name in the datatypes table

**Semantics**

## 7.13 Functions Tables

Information Pathways package through its emphasis on how digital information types (and some material items) are conveyed between the parts of the SOI, already implies certain "activities" performed by the replaceable units of the SOI namely:

- send an information item (or a material item)
- receive an information item (or a material item)
- store an information item
- retrieve an information item

Material items imply some kind of storage, once a material item is transferred to a replaceable unit, it remains stored in that unit.

Behavior package identifies Core Assertion related to the custom language of other "activities", performed by the SOI (either directly by one of the replaceable units or through collaboration of several units). Such activities are usually related to processing, converting, transforming various items, producing items, disposing of items, performing computations, making decisions, etc.

The behavior viewpoint of SPECTRA is optional; attacks can be adequately predicted at the purely structural viewpoint offered by information pathways alone; SPECTRA allows identification of function units (and emergent functions) as refinement of the replaceable units. SPECTRA also allows identification of Mission Tasks (as they are introduced by the Mission Engineering language and are distinct from the Function Units and Emergent Functions of the SOI.

SysML activity, action or block that represent something that an individual replaceable unit does; function happens in a certain place (some functions are performed by an individual replaceable unit. A function unit is the most specific producer/consumer of a flow, and an agent that stores/retrieves data from datastores.

**Table Identification**
SPECTRA Functions for system <SOI name>

**Table Columns**

| Column Title | Column Data | Description |
|---|---|---|
| Name | Identifier | Specifies the identifying name of the element |
| Purpose | Text | Provides the description of the element |
| Domain | DomainEnum | Specifies the domain of the element |

| | | |
|---|---|---|
| Characteristics | List of CharacteristicEnum | Specifies the set of unique characteristics of the element |
| Parent | Reference | Specifies the parent of the element, if applicable |
| isSystem | Boolean | Specifies whether this is a system function |
| isCollaborative | Boolean | Specifies whether this is a collaborative function |
| Notes | Text | Informal notes to the element |

**Constraints**

- Names in the table shall be unique
- Reference to a parent shall match exactly one name in this table
- Reference to a parent shall not match the name in the current row

**Semantics**

## 7.14 Missions Table

Mission - Strategic element that refers to an important task that people must accomplish. In a SysML model, a mission can be represented by an activity, a block, or a use case. Mission is an end-to-end "sequence" of flows usually corresponding to achieving a certain operational objective. Usually a mission involves multiple "phases", possibly involving different configurations of nodes, and with different functions enabled/disabled.

**Table Identification**
SPECTRA Missions for system <SOI name>

**Table Columns**

| Column Title | Column Data | Description |
|---|---|---|
| Name | Identifier | Specifies the identifying name of the element |
| Purpose | Text | Provides the description of the element |
| Domain | DomainEnum | Specifies the domain of the element |
| Capability | List of Reference | Specifies the set of capabilities that this mission depends on, if applicable |
| Notes | Text | Informal notes to the element |

**Constraints**

**Semantics**

## 7.15 Mission Phases Table

Mission phase - Specific configuration involved in a mission. A Mission usually involves an ordered collection of "phases" with possibly different configurations of nodes involved, and different functions enabled/disabled.

MissionPhase element represents a snapshot of the SOI's architecture at a certain point in a Mission. Identification of such configuration is essential for assessing SOI with dynamically changing architecture, which is then represented as a series of static snapshots associated with various mission phases. Each MissionPhase configuration then has a static digital technical surface that can be assessed independently.

**Table Identification**
SPECTRA MissionPhases for system <SOI name>

**Table Columns**

| Column Title | Column Data | Description |
|---|---|---|
| Name | Identifier | Specifies the identifying name of the element |
| Purpose | Text | Provides the description of the element |
| Successors | List of Reference | Specifies the set of successor phases for the current one |
| Nodes | List of Reference | Specifies the set of nodes that a part of the configuration for this phase |
| Tasks | List of Reference | Specifies the set of tasks for this phase |
| Notes | Text | Informal notes to the element |

**Constraints**

- Names in the table shall be unique
- A reference in the successors column shall match exactly one name in this table
- All references in the list of successors column shall be unique
- A reference in the nodes column shall match exactly one name in the units table or in the assemblies table
- All references in the list of nodes column shall be unique
- A reference in the tasks column shall match exactly one name in the tasks table
- All references in the list of tasks column shall be unique

**Semantics**

## 7.16 Mission Tasks Table

Mission Taks is an activity or a task, defined in the context of Mission Engineering. Mission Task is involved in a Mission Thread. A Mission Task is different from the Function Units or Emergent Functions of the SOI (at least uses a different vocabulary, one of Mission Engineering), and depends on some of Capabilities of the SOI (and transitively on some Function Units, possibly through some Emergent Functions).

See section Kernel::Impactful Elements::Mission for more detail.

**Table Identification**
SPECTRA Tasks for system <SOI name>

**Table Columns**

| Column Title | Column Data | Description |
|---|---|---|
| Name | Identifier | Specifies the identifying name of the element |
| Purpose | Text | Provides the description of the element |
| Parent | Reference | Specifies the parent tasks for which  the current task is part of a decomposition, if applicable |
| Functions | List of Reference | Specifies the set of functions that implement the task |
| Notes | Text | Informal notes to the element |

**Constraints**

**Semantics**

## 7.17 Agents Table

The Access package addresses the “BY WHOM?” clauses related to SOI (in addition to “WHERE” and “To WHAT” clauses introduced in the Kernel Package). Assertions related to access are important for understanding the trust zones of the SOI, its internal attack surface as well as the external attack surface in the context of advanced persistent threats.

Stereotype is the Access package that allows identifying certain blocks as the “agents” who are trusted to access some capabilities of the system. Access assertions are made in the form of agent handles data, agent accesses unit, or agent performs function. In addition, this package also identifies assertions related to the supply chain of the SOI in the form supplier supplies artifact.

Agent - block or part that represents a human being (as a part in a complex information processing enterprise); e.g. an operator; a human can be vulnerable to subversion, deceit, can be clueless, careless or malicious; or can be vulnerable to some health and safety hazard.

**Table Identification**
SPECTRA Agents for system <SOI name>

**Table Columns**

| | | |
|---|---|---|
| Name | Identifier | |
| Purpose | Text | |
| Category | AgentCategoryEnum | |
| Domain | DomainEnum | |
| Notes | Text | |

**Constraints**

**Semantics**

### 7.17.1 Agent Category Enumeration

**Table 16 Enumeration Literals for Agent Category**

| Enum Value | Definition |
|---|---|
| Operator | An operator is a human can be vulnerable to subversion, deceit, can be clueless, careless or malicious; or can be vulnerable to some a health and safety hazard |
| Maintainer | An agent performing maintenance. Maintenance operations can be performed at various mission phases, such as during the operational phases (hot patches, dynamic reconfiguration), during dedicated maintenance phases (planned patching or an upgrade), or during the major evolutionary upgrade phases. |
| Supplier | Supplier - Representation of a supplier of an artifact type (optional, can be uniquely derived in the form of "supplier of an artifact type xxx"), but if a model provides this information, it is important to annotate accordingly, so that this element is not mis-interpreted, and so that fidelity is not compromised |

## 7.18 Attackers Table

The Access package addresses the “BY WHOM?” clauses related to SOI (in addition to “WHERE” and “To WHAT” clauses introduced in the Kernel Package). Assertions related to access are important for understanding the trust zones of the SOI, its internal attack surface as well as the external attack surface in the context of advanced persistent threats. Stereotype is the Access package that allows identifying certain blocks as the “agents” who are trusted to access some capabilities of the system. Access assertions are made in the form of agent handles data, agent accesses unit, or agent performs function. In addition, this package also identifies assertions related to the supply chain of the SOI in the form supplier supplies artifact.

**Table Identification**
SPECTRA Attackers for system <SOI name>

**Table Columns**

| Column Title | Column Data | Description |
|---|---|---|
| Name | Identifier | |
| Purpose | Text | |
| Category | AttackerCategoryEnum | |
| Accesses | List of Reference | |
| Performs | List of Reference | |
| Impersonates | Reference | |
| MitigatedBy | List of Reference | |
| Notes | Text | |

**Constraints**

**Semantics**

### 7.18.1 Attacker Category Enumeration

**Table 17 Enumeration Literals for Attacker Category**

| Enum Value | Definition |
|---|---|
| Attacker | |
| Hazard | Hazard - A hazard is a potential source of harm. Substances, events, or circumstances can constitute hazards when their nature would potentially allow them to cause damage to health, life, property, or any other interest of value. In physics terms, a common theme across many forms of hazards is the presence of energy that can cause damage, as it can happen with chemical energy, mechanical energy or thermal energy. |

## 7.19 Mitigations Table

Allocated Control - representation of a statement, that a control type x from a certain control catalog, has been allocated to a certain node, or channel. By allocating controls to attackable units (as well as related channels, protocols and configurations) we can consider various related functional threads and understand segments of threads that are mitigated; by considering threads we can see how our protection covers other attackable targets; we can also understand control by-passes.

**Table Identification**
SPECTRA Mitigations for system <SOI name> [variant <variant name>]

**Table Columns**

| Column Title | Column Data | Description |
|---|---|---|
| MitigationOption | Identifier | |
| Control Name | Reference | |
| Kind | MitigatedElementEnum | |
| Mitigated Element | Reference | |
| Notes | Text | |

**Constraints**

**Semantics**

### 7.19.1 MitigatedElement Enumeration

**Table 18 Enumerated Literals for Mitigated Element Category**

| Enum Value | Definition |
|---|---|
| Unit | The mitigated element reference is from replaceable element namespace |
| Section | The mitigated element reference is from section namespace |
| Subsystem | The mitigated element reference is from subsystem namespace |
| Assembly | The mitigated element reference is from assembly namespace |
| Datastore | The mitigated element reference is from datastore namespace |
| Mission | The mitigated element reference is from mission namespace |
| MissionPhase | The mitigated element reference is from mission phase namespace |

## 7.20 Characteristics

### 7.20.1 Domain Enumeration

Domain Characteristic is an abstract element that identifies the domain to which the annotated element belongs to. A domain is a specific subject area in which the SOI is involved, a field of study that defines a set of common requirements, terminology, and functionality for any system or a software program constructed to solve a problem in a given field. For the purposes of risk and cybersecurity assessments of cyber and cyber-physical systems, the following domains are important:

- Application Domain
- Security Domain
- Maintenance Domain
- Carrier Domain (or networking domain)

**Table 19 Enumeration Literals for Domain**

| Enum Value | Definition |
|---|---|
| Application | This stereotype identifies the element as belonging to the application domain of the SOI (without further specifying what that domain is, e.g. a medical device, an electrical substation, an avionics system, a weapons system). Marking an element as belonging to the application domain distinguishes it from other elements that are marked as belonging to other domains, relevant to SPECTRA (security, maintenance, carrier). |
| Security | This stereotype identifies the element as belonging to the security domain. Cyber Security is protection of digital information, as well as the integrity of the infrastructure housing and |

| | |
|---|---|
| | transmitting digital information. More specifically, cyber security includes the body of technologies, processes, practices and response and mitigation measures designed to protect networks, computers, programs and data from attack, damage or unauthorized access to ensure confidentiality, integrity and availability. Security Domain is common to systems in various Application Domains.<br>Marking an element as belonging to the application domain distinguishes it from other elements that are marked as belonging to other domains, relevant to SPECTRA (application, maintenance, carrier). |
| Maintenance | This stereotype identifies the element as belonging to the Maintenance Domain. Maintenance Domain is about the capabilities that support the evolution of the system (e.g. configuration, upgrades, hot-swap, patching).<br>Maintenance involves functional checks, servicing, repairing or replacing of necessary devices, equipment, software, building infrastructure and supporting utilities in system installations. Maintenance domain is involves certain capabilities of SOI, such as logging, fault management, alerting and software uploads (configuration, patches, hot-swap updates or upgrades). Some data communications within SOI may be related to maintenance, such as delivering an image for a hot-swap without interrupting the regular operations of the system. Maintenance Domain is common to systems in various Application Domains.<br>Marking an element as belonging to the maintenance domain distinguishes it from other elements that are marked as belonging to other domains, relevant to SPECTRA (application, security, carrier). |
| Carrier | This stereotype identifies the element as belonging to the Carrier (Networking) Domain. Computer networks are a key part of the infrastructure for cyber and cyber-physical systems. Computing devices use common communication protocols over digital interconnections to communicate with each other. These interconnections are made up of telecommunication network technologies based on physically wired, optical, and wireless radio-frequency methods that may be arranged in a variety of network topologies. Carrier Domain is common to systems in various Application Domains.<br>Marking an element as belonging to the carrier domain distinguishes it from other elements that are marked as belonging to other domains, relevant to SPECTRA (application, security, maintenance). Absence of a clear distinction between the application and carrier data is a common anti-pattern in SysML models that is a barrier for effective understanding of the core assertions made about the SOI. |
| Virtualization | |

### 7.20.2 Characteristic Enumeration

Characteristics (together with Artifacts and Traits) provide a shared vocabulary to make assertions about the nature of the elements of SOI. Characteristics describe the common assertions related to the nature of various elements of the SOI.

Characteristics apply to nodes, channels, flows, information types, resources; The characteristics are common shared meanings that can be used to select appropriate rules and axioms in a knowledgebase and match them to specific elements of the SOI as represented by the model being ingested

#### Cyber Characteristics

Cyber - element that is involved with digital information processing. The characteristics are common shared meanings that can be used to select appropriate rules and axioms in a knowledgebase and match them to specific elements of the SOI as represented by the model being ingested. For a cyber system the elements are assumed to be cyber, however in a cyber-physical system it is important to distinguish cyber elements from physical elements.

**Table 20 Enumeration Literals for Cyber Characteristics**

| Enum Value | Definition |
|---|---|
| Digital | Digital - An element involving automated processing of digital information (as opposed to human) |
| Analog | |
| Cloud | Cloud - any infrastructure-as-code element that is implemented in a public or private cloud |
| DataCenter | A data center is a set of computer systems, telecommunications and storage systems, often housed in a dedicated space such as a building or a group of buildings on the same site. While early data centers separated computing hardware from the remote terminal equipment, modern data centers usually host cloud-based virtualized infrastructure. |
| Human | Human - any element involving humans, subject to social engineering attacks |

### Physical Characteristics

Physical - "physical concerns for cyber-physical systems. "Physical" characteristics can be applied to a replaceable element, conveying element, or resource. Applying a physical characteristic to a conveying element refers to its physical communications medium and is a shortcut for describing the proper carrier interface. For the purposes of cybersecurity risk assessment, SPECTRA distinguishes several frequently used subclasses, usually the choice is straightforward. The characteristics are common shared meanings that can be used to select appropriate rules and axioms in a knowledgebase.

**Table 21 Enumeration Literals for Physical Characteristics**

| Enum Value | Definition |
|---|---|
| Electrical | Electrical - channel or node or flow or protocol or resource involving electrical current, usually involving some kind of conductors, e.g. house wiring for light fixtures, power substation, electronics, e.g. alarm triggers; flow or protocol involving electricity; DC or AC current. |
| Electronic | |
| Power | |
| Mechanical | Mechanical - channel or node or flow or protocol or resource involving physical forces or motion, e.g. brakes, wheel axle, antenna gimbal; not to be confused with computer hardware |
| Kinetic | While Kinetics is the branch of classical mechanics that is concerned with the relationship between the motion and its causes, specifically, forces and torques, the term Kinetic warfare is sometimes used as a term for military combat or other forms of directly-destructive warfare. In SPECTRA, kinetic characteristic is defined as producing a mechanical impact. It is somewhat overlapping with another SPECTRA characteristic “mechanical”. |
| Magnetic | |
| Electromagnetic | Electromagnetic - channel or node or flow or protocol or resource involving electro-magnetic spectrum, e.g HF, VHF, etc. |
| Acoustic | Acoustic - channel or node or flow or protocol or resource involving sound, acoustic element (using sound-waves) |

| | |
|---|---|
| Infrared | Infrared - channel or node or flow or protocol or resource involving infrared spectrum, technically, this is also EMF, but this is very specific and also short range |
| Optical | Optical - channel or node or flow or protocol or resource involving optical element (usually involving lasers); over fiber optic channels or any other appropriate media |
| Spacial | |

### Resource Characteristics

**Table 22 Enumeration Literals for Resource Characteristics**

| Enum Value | Definition |
|---|---|
| Document | Document - a physical document on some media, e.g. paper, optical disc, etc. This characteristic applies to resource elements. |
| Consumable | Consumable - a consumable physical resource (tangible or intangible), e.g. fuel, time-left-unit-maintenance. This characteristic applies to resource elements. |
| Financial | Financial - this characteristic applies to financial resources or information types. |
| Material | Material - a characteristic that can be applied to resources to describe a piece of equipment and supplied in a supply-chain management context. |

## 7.20.3 Impact Level Enumeration

**Table 23 Enumeration Literals for Impact Level Measure**

| Enum Value | Definition |
|---|---|
| VeryHigh | The loss of confidentiality, integrity, availability, privacy, or safety could be expected to have **multiple severe or catastrophic** adverse effects on organizational operations, organizational assets, or individuals. |
| High | The loss of confidentiality, integrity, availability, privacy or safety could be expected to have a **severe or catastrophic** adverse effect on organizational operations, organizational assets, or individuals. |
| Moderate | The loss of confidentiality, integrity, availability, privacy or safety could be expected to have a **serious** adverse effect on organizational operations, organizational assets, or individuals. |
| Low | The loss of confidentiality, integrity, availability, privacy or safety could be expected to have a **limited** adverse effect on organizational operations, organizational assets, or individuals. |
| VeryLow | The loss of confidentiality, integrity, availability, privacy or safety could be expected to have a **negligible** adverse effect on organizational operations, organizational assets, or individuals. |

# Annex A: SPECTRA Core Assertions

(informative)

## A.1 Overview

This section provides an overview of the SPECTRA Core Assertions. Core Assertions are defined by the SPECTRA Core Assertions Metamodel. This Appendix outlines the Core Assertions, as the foundation for defining the semantics of SPECTRA Comma-Separated Value tables. This is not a full specification of the Core Assertions Metamodel.
Prolog-like clauses were selected as the mechanism for describing the Core Assertions because of the fact that Prolog combines assertions to a database of facts, and logical inferences, and because of the abundance of practical tools. The definitions based on Prolog can be easily transferred to a multitude of other languages (relational database schemas, json, etc.) and other formalisms. Also, the choice of Prolog for describing Core Assertions emphasized the technical and pragmatic nature of these definitions, leaving a full SPECTRA Ontology as a separate specification in the SPECTRA family.

## A.2 Semantic Foundation

Core Assertions define some entities and relationships. The foundation for the Core Assertions has a simple triple organization.

| Meta-term | Description |
|---|---|
| entity(id,noun) | meta-term representing an assertion that a certain entity exists, and that it can be uniquely identified by an identifier *id*, and that it belongs to some class, identified by *noun*. |
| attribute(id, name, value) | meta-term representing an assertion that a certain entity identified by an identifier *id*, has an attribute with name *name*, and with value *value*. |
| relationship(verbphrase,fromid,toid) | meta-term representing an assertion that a certain entity identified by an identifier *fromid*, has a relationship to another entity identified by *toid*, and the class of the relationship is identified by *verbphrase*. |

Specific Core Assertions are defined as follows:

```
node( Id,Noun,Name) :- entity( Id,’Node’ ), attribute( Id,noun, Noun), attribute( Id,’name’,Name).
owner( Nid1, Nid2 ) :- relationship(‘nodeOwnsNode’,Nid1,Nid2).
```

Semantics of the SPECTRA metadata for SysML v2 defines how custom instances of core assertions for a specific SOI are constructed from a well-formed SysML v2 model with compliant SPECTRA metadata for selected elements.

Custom assertions are facts, for example:

```
node(‘n01’,’ReplaceableUnit’,’SATCOM Radio’).
node(‘n02’,’Subsystem’,’Communications Subsystem’).
owner( ‘n02’, ‘n01’ ).
```

The definition of the Core Assertion Schema involves the following terms:

| Term | Description |
|---|---|
| schemadef( sid, name, version ) | asserting existence of a Core Assertions Schema identified by *sid* and name *name*, also providing a *version* tag for this schema to manage the schema evolution. |
| entitydef( sid, noun ) | asserting that schema identified by *sid* involves an entity class identified by *noun* |
| attributedef( sid, noun, name, type ) | asserting that within the schema identified by *sid* an entity class identified by *noun* involves an attribute identified by *name*, with values belonging to type *type* |
| relationshipdef( sid, verbphrase, verb, noun1, noun2 ) | asserting that schema identified by *sid* involves a relationship class identified by *verbphrase*, that involves a verb *verb*, and two *nouns*. The *verbphrase* can be verbalized as "*noun1 verb noun2*" |
| typedef( sid, type ) | asserting that within the schema identified by *sid* involves a primitive type identified by *type* |
| subclassdef( sid, noun1, noun2 ) | asserting that within the schema identified by *sid* the class identified as *noun1* is a subclass of the class identified by *noun2* |
| abstractdef( sid, noun ) | asserting that within the schema identified by *sid* the involves an abstract class identified as *noun1* |

Definition of the schema constitutes additional facts, for example:

```
schemadef( ‘s1’, “SPECTRA Core Assertions’, ‘1.0’).
abstractdef( ‘s1’, ‘Node’ ).
attributedef( ‘s1’, ‘Node’, noun, ‘String’).
attributedef( ‘s1’, ‘Node’, ‘name’, ‘String’).
abstractdef( sid, ‘Node’ ).
abstractdef( ‘s1’, ‘ReplaceableElement’ ).
entitydef( ‘s1’, ‘ReplaceableUnit’ ).
subclassdef( sid, ‘ReplaceableUnit’ , ‘ReplaceableElement’).
subclassdef( sid, ‘ReplaceableElement’ , ‘Node’).
typedef( ‘s1’, ‘String’).
typedef( ‘s1’, ‘Boolean’).
relationshipdef( ‘s1’, nodeOwnsNode, ‘owns’, ‘Node’,’Node’).
```

## A.3 Core Assertions

This section enumerates the Core Assertions. This section focuses on the assertions related to SPECTRA Core entities and provides only a few examples of SPECTRA Core relationships. Full specification of the SPECTRA Core Assertions is provided in the SPECTRA Core Assertions Metamodel specification.

The objective of this section is to describe the key terms used in the semantic formulations in this specification, and to demonstrate the flexibility of the selected approach to describing the semantics of SPECTRA for SysML v2. This approach allows a multitude of "schemas" that can represent the facts extracted from a SysML v2 model annotated with SPECTRA metadata - ranging from a triple store format to a metamodel-driven format. This section demonstrates a hybrid format: instead of defining an individual term for each SPECTRA entity, the list below defines a common class for a selected abstract entity, e.g. "node", where the specific subclass of the entity is represented by its noun. The mapping of these terms is described by simple inference rules, as demonstrated in the Semantic Foundation section above.

**Table 24. Specific Core Assertions**

| Term | Description |
|---|---|
| node( id,noun,name) | foundation for assertions that an element with certain id and name in a SysML model is one of the subclasses of metadata Node; the noun is the literal representing of the concrete subclasses of Node, e.g. 'ReplaceableUnit' |
| channel( id, noun, name) | foundation for assertions that element with certain id and name in a SysML model is one of the subclasses of metadata ConveyableElement; the noun is the literal representing of the concrete subclasses of Node, e.g. 'Link' |
| endpoint( nid, cid ) | assertion that a node with id nid is an endpoint for the channel with id cid |
| owner( nid1, nid2 ) | assertion that node nid1 is the owner of node with id nid2 |
| member( nid1, nid2 ) | assertion that node nid1 belongs to group nid2 |
| datatype( id, noun, name ) | foundation for assertions involving stereotype InformationType; the noun is the literal representing of the concrete subclasses of OperationalData, e.g. 'ApplicationData' or 'Request' |
| exchange( id, fromid, toid, dataid ) | foundation for assertions that element with id is a Flow |
| carrierInterface( id, name, p1, p2, p3, isWireless) | foundation for assertions that element with id is a CarrierInterface |
| datastore(id, name) | foundation for assertions that element with id is a Datastore |
| agent(id, noun, name) | foundation for assertions that element with id is one of the subclasses of Agent; the noun is the literal representing of the concrete subclasses of Agent, e.g. 'Operator' |
| capability(id, name) | foundation for assertions that element with id is a Capability |
| mission(id, name) | foundation for assertions that element with id is a Mission |
| missionPhase(id, name) | foundation for assertions that element with id is a MissionPhase |
| function(id, noun, name) | foundation for assertions that element with id is one of the subclasses of  FunctionElement; the noun is the literal representing of the concrete subclasses of Node, e.g. 'EmergentFunction' |

| | |
|---|---|
| thread() | foundation for assertions that element with id is a Functional Thread |
| characteristic( ownerid, noun ) | foundation for assertions that element with id is one of the subclasses of Characteristic |
| artifact( id, noun, ownerid ) | foundation for assertions that element with id is one of the subclasses of Artifact; the noun is the literal representing of the concrete subclasses of Artifact, e.g. ‘Hardware’ |
| trait( id, noun, ownerid ) | foundation for assertions that element with id is one of the subclasses of Trait; the noun is the literal representing of the concrete subclasses of Trait, e.g. ‘Controller’ |
| internal( id, value ) | assertion that element with id is internal (true) or external (false) |
| boundaryCrossing( id ) | assertion that element with id is boundary-crossing |
| inbound(id) | assertion that element with id is inbound |
| outbound(id) | assertion that element with id is outbound |
| input( id ) | assertion that operational data with id is input |
| output(id) | assertion that operational data with is is output |
| stored(id) | assertion that operational data with id is stored |
| processed(id) | assertion that operational data with id is stored |
| impactLevel( id, objective, level ) | assertion that impactful element with id has Impact Level |
| note(id, ownerid, text) | assertion that element with id has a comment or a note |

# Annex B: References

1. Friedenthal S., Moore A., Steiner R., *A Practical Guide to SysML: The Systems Modeling Language* (2015). Morgan Kaufmann, OMG Press.
2. Friedenthal S., Oster C, *Architecting Spacecraft with SysML* (2017). AIAA.
3. Aleksandravicene A., Morkevicus A., *MagicGrid Book of Knowledge: A Practical Guide to Systems Modeling using MagicGrid from Dassault Systems*, (2021), Dassault Systems, Vitae Litera, Kaunas
4. McSweeny K., Finding a Single Source of Truth with Model-Based Systems Engineering, Northrop Grumman, https://www.northropgrumman.com/what-we-do/digital-transformation/finding-a-single-source-of-truth-with-model-based-systems-engineering
5. Mansourov N., Campara D., Systems Assurance: Beyond Detecting Vulnerabilities (2010), Morgan Kaufman, OMG Press.
6. Moir, I., *Military Avionics Systems* (2001) AIAA Education Series, CRC Press
7. INCOSE, Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, Wiley, 2015
8. [SysML v1] *OMG Systems Modeling Language (SysML),* Version 1.7 https://www.omg.org/spec/SysML/1.7
9. [UML] *Unified Modeling Language (UML),* Version 2.5.1 https://www.omg.org/spec/UML/2.5.1
10. [NIST SP-800-30] *NIST Special Publication 800-30, Guide for Conducting Risk Assessments. Information Security, 2012* https://doi.org/10.6028/NIST.SP.800-30r1
11. [NIST SP-800-37] *NIST Special Publication 800-37, Rev 2, Risk Management Framework for Information Systems and Organizations: A system Life Cycle Approach for Security and Privacy, 2018* https://doi.org/10.6028/NIST.SP.800-37r2
12. [NIST SP-800-53] *NIST Special Publication 800-53, Rev 5, Security and Privacy Controls for Information Systems and Organizations, 2020* https://doi.org/10.6028/NIST.SP.800-53r5
13. [NIST SP-800-53a] *NIST Special Publication 800-53A, Rev 5, Assessing Security and Privacy Controls in Information Systems and Organizations, 2022* https://doi.org/10.6028/NIST.SP.800-53Ar5
14. [FIPS-199] FIPS Pub 199, Federal Information Processing Standards Publication. Standards for Security Categorization of Federal Information and Information Systems, 2004, https://doi.org/10.6028/NIST.FIPS.199
15. [SPDX] ISO/IEC 5962:2021 Information technology – SPDX Specification V2.2.1, https://www.iso.org/standard/81870.html
16. [CycloneDX] CycloneDX Bill of materials specification, ECMA-424, 2024, https://ecma-international.org/publications-and-standards/standards/ecma-424
17. [ISO 42010] ISO/IEC/IEEE 42010:2022, Systems and software engineering – Architecture Description, https://www.iso.org/standard/74393.html
18. ISO/IEC/IEEE 15288:2023, Systems and software engineering – System life cycle processes, https://www.iso.org/standard/81702.html
19. ISO/IEC/IEEE 27005:2022, Information security, cybersecurity and privacy protection – Guidance on managing information security risks, https://www.iso.org/standard/74393.html
20. [KDM] ISO/IEC 19506:2012 Information technology Object Management Group Architecture-Driven Modernization (ADM) – Knowledge Discovery Metamodel (KDM), 2017, https://www.iso.org/standard/32625.html
21. [Prolog] ISO/IEC 13211:1:1995 Information technology — Programming languages — Prolog, Part 1: General Core, 2024, https://www.iso.org/standard/21413.html
22. [MITRE ATT&CK] https://attack.mitre.org

23. [MITRE D3FEND] https://d3fend.mitre.org
24. [CWE] Common Weakness Enumeration https://cwe.mitre.org
25. [CVE] Common Vulnerabilities and Exposures https://cve.mitre.org
26. [NVD] National Vulnerability Database https://nvd.nist.gov