Object Management Group

109 Highland Avenue
Needham, MA 02494
USA

Telephone: +1-781-444-0404
Facsimile: +1-781-444-0320
rfp@omg.org

Systems Modeling Language (SysML®) v2
Request For Proposal (RFP)

OMG Document: ad/2017-12-02

Letters of Intent due: 24 September, 2018
Submissions due: 4 November, 2019

Objective of this RFP

This RFP specifies the requirements for the next generation of the OMG Systems Modeling Language (OMG SysML® v2) that are intended to address many of the limitations of the current version of OMG SysML® to enable the more effective application of model-based systems engineering (MBSE). In particular, the emphasis for SysML v2 is to improve the precision, expressiveness, interoperability, and the consistency and integration of the language concepts relative to SysML v1. SysML v2 will express the core concepts required to precisely specify a system, its elements, and its environment (i.e., the system model). The language will be specified as both a SysML profile of UML and as a SysML metamodel. A complementary SysML v2 API and Services RFP is intended to further enhance interoperability by specifying standard services to access SysML v2 models. (The specific requirements for this RFP are contained in Section 6.)

1 Introduction

1.1 Goals of OMG

The Object Management Group (OMG) is a software consortium with an international membership of vendors, developers, and end users. Established in 1989, its mission is to help computer users solve enterprise integration problems by supplying open, vendor-neutral portability, interoperability and reusability specifications based on Model Driven Architecture (MDA). MDA defines an approach to IT system specification that separates the specification of system functionality from the specification of the implementation of that functionality on a specific technology platform, and provides a set of guidelines for structuring specifications expressed as models. OMG has published many widely-used specifications such as UML [UML], BPMN [BPMN], MOF [MOF], XMI [XMI], DDS [DDS] and CORBA [CORBA], to name but a few significant ones.

1.2 Organization of this document

The remainder of this document is organized as follows:

Section 2 – *Architectural Context*. Background information on OMG's Model Driven Architecture.

Section 3 – *Adoption Process*. Background information on the OMG specification adoption process.

Section 4 – *Instructions for Submitters*. Explanation of how to make a submission to this RFP.

Section 5 – *General Requirements on Proposals*. Requirements and evaluation criteria that apply to all proposals submitted to OMG.

Section 6 – *Specific Requirements on Proposals*. Problem statement, scope of proposals sought, mandatory requirements, non-mandatory features, issues to be discussed, evaluation criteria, and timetable that apply specifically to this RFP.

Appendix A – References and Glossary Specific to this RFP

Appendix B – General References and Glossary

1.3 Conventions

The key words **"shall"**, **"shall not"**, **"should"**, **"should not"**, **"may"** and **"need not"** in this document should be interpreted as described in Part 2 of the ISO/IEC Directives [ISO2]. These ISO terms are compatible with the same terms in IETF RFC 2119 [RFC2119].

1.4 Contact Information

Questions related to OMG's technology adoption process and any questions about this RFP should be directed to *rfp@omg.org*.

OMG documents and information about the OMG in general can be obtained from the OMG's web site: *http://www.omg.org*. Templates for RFPs (like this document) and other standard OMG documents can be found on the Template Downloads Page: *http://www.omg.org/technology/template_download.htm*

2 Architectural Context

MDA provides a set of guidelines for structuring specifications expressed as models and the mappings between those models. The MDA initiative and the standards that support it allow the same model, specifying business system or application functionality and behavior, to be realized on multiple platforms. MDA enables different applications to be integrated by explicitly relating their models; this facilitates integration and interoperability, and supports system evolution (deployment choices) as platform technologies change. The three primary goals of MDA are portability, interoperability and reusability.

Portability of any subsystem is relative to the subsystems on which it depends. The collection of subsystems that a given subsystem depends upon is often

loosely called the *platform*, which supports that subsystem. Portability – and reusability – of such a subsystem is enabled if all the subsystems that it depends upon use standardized interfaces (APIs) and usage patterns.

MDA provides a pattern comprising a portable subsystem that is able to use any one of multiple specific implementations of a platform. This pattern is repeatedly usable in the specification of systems. The five important concepts related to this pattern are:

1. *Model* – A model is a representation of a part of the function, structure and/or behavior of an application or system. A representation is said to be formal when it is based on a language that has a well-defined form ("syntax"), meaning ("semantics"), and possibly rules of analysis, inference, or proof for its constructs. The syntax may be graphical or textual. The semantics might be defined, more or less formally, in terms of things observed in the world being described (e.g. message sends and replies, object states and state changes, etc.), or by translating higher-level language constructs into other constructs that have a well-defined meaning. The (non-mandatory) rules of inference define what unstated properties can be deduced from explicit statements in the model. In MDA, a representation that is not formal in this sense is not a model. Thus, a diagram with boxes and lines and arrows that is not supported by a definition of the meaning of a box, and the meaning of a line and of an arrow is not a model – it is just an informal diagram.

2. *Platform* – A set of subsystems/technologies that provide a coherent set of functionality through interfaces and specified usage patterns that any subsystem that depends on the platform can use without concern for the details of how the functionality provided by the platform is implemented.

3. *Platform Independent Model (PIM)* – A model of a subsystem that contains no information specific to the platform, or the technology that is used to realize it.

4. *Platform Specific Model (PSM)* – A model of a subsystem that includes information about the specific technology that is used in the realization of that subsystem on a specific platform, and hence possibly contains elements that are specific to the platform.

5. *Mapping* – Specification of a mechanism for transforming the elements of a model conforming to a particular metamodel into elements of another model that conforms to another (possibly the same) metamodel. A mapping may be expressed as associations, constraints, rules or templates with parameters that to be assigned during the mapping, or other forms yet to be determined.

OMG adopts standard specifications of models that exploit the MDA pattern to facilitate portability, interoperability and reusability, either through *ab initio* development of standards or by reference to existing standards. Some examples of OMG adopted specifications are:

1. *Languages* – e.g. IDL for interface specification [IDL], UML for model specification [UML], BPMN for Business Process specification [BPMN], etc.

2. *Mappings* – e.g. Mapping of OMG IDL to specific implementation languages (CORBA PIM to Implementation Language PSMs), UML Profile for EDOC (PIM) to CCM (CORBA PSM) and EJB (Java PSM), CORBA (PSM) to COM (PSM) etc.

3. *Services* – e.g. Naming Service [NS], Transaction Service [OTS], Security Service [SEC], Trading Object Service [TOS] etc.

4. *Platforms* – e.g. CORBA [CORBA], DDS [DDS]

5. *Protocols* – e.g. GIOP/IIOP [CORBA] (both structure and exchange protocol), DDS Interoperability Protocol [DDSI].

6. *Domain Specific Standards* – e.g. Model for Performance-Driven Government [MPG], Single Nucleotide Polymorphisms specification [SNP], TACSIT Controller Interface specification [TACSIT].

For an introduction to MDA, see [MDAa]. For a discourse on the details of MDA please refer to [MDAc]. To see an example of the application of MDA see [MDAb]. For general information on MDA, see [MDAd].

Object Management Architecture (OMA) is a distributed object computing platform architecture within MDA that is related to ISO's Reference Model of Open Distributed Processing RM-ODP [RM-ODP]. CORBA and any extensions to it are based on OMA. For information on OMA see [OMA].

3 Adoption Process

3.1 Introduction

OMG decides which specifications to adopt via votes of its Membership. The specifications selected should satisfy the architectural vision of MDA. OMG bases its decisions on both business and technical considerations. Once a specification is adopted by OMG, it is made available for use by both OMG members and non-members alike, at no charge.

This section 3 provides an extended summary of the RFP process. For more detailed information, see the *Policies and Procedures of the OMG Technical Process* [P&P], specifically Section 4.2, and the *OMG Hitchhiker's Guide* [Guide]. In case of any inconsistency between this document or the Hitchhiker's Guide and the Policies and Procedures, the P&P is always authoritative. All IPR-related matters are governed by OMG's *Intellectual Property Rights Policy* [IPR].

3.2 The Adoption Process in detail

3.2.1 Development and Issuance of RFP

RFPs, such as this one, are drafted by OMG Members who are interested in the adoption of an OMG specification in a particular area. The draft RFP is presented to the appropriate TF, discussed and refined, and when ready is recommended for issuance. If endorsed by the Architecture Board, the RFP may then be issued as an OMG RFP by a TC vote.

Under the terms of OMG's Intellectual Property Rights Policy [IPR], every RFP shall include a statement of the IPR Mode under which any resulting specification will be published. To achieve this, RFP authors choose one of the three allowable IPR modes specified in [IPR] and include it in the RFP – see section 6.10.

3.2.2 Letter of Intent (LOI)

Each OMG Member organisation that intends to make a Submission in response to any RFP (including this one) shall submit a Letter of Intent (LOI) signed by an officer on or before the deadline specified in the RFP's timetable (see section 6.11). The LOI provides public notice that the organisation may make a submission, but does not oblige it to do so.

3.2.3 Voter Registration

Any interested OMG Members, other than Trial, Press and Analyst members, may participate in Task Force voting related to this RFP. If the RFP timetable includes a date for closing the voting list (see section 6.11), or if the Task Force separately decides to close the voting list, then only OMG Member that have registered by the given date and those that have made an Initial Submission may vote on Task Force motions related to this RFP.

Member organizations that have submitted an LOI are automatically registered to vote in the Task Force. Technical Committee votes are not affected by the Task Force voting list – all Contributing and Domain Members are eligible to vote in DTC polls relating to DTC RFPs, and all Contributing and Platform Members are eligible to vote in PTC polls on PTC RFPs.

3.2.4 Initial Submissions

Initial Submissions shall be made electronically on or before the Initial Submission deadline, which is specified in the RFP timetable (see section 6.11), or may later be adjusted by the Task Force. Submissions shall use the OMG specification template [TMPL], with the structure set out in section 4.9. Initial Submissions shall be written specifications capable of full evaluation, and not just a summary or outline. Submitters normally present their proposals to the Task Force at the first TF meeting after the submission deadline. Making a submission incurs obligations under OMG's IPR policy – see [IPR] for details.

An Initial Submission shall not be altered once the Initial Submission deadline has passed. The Task Force may choose to recommend an Initial Submission, unchanged, for adoption by OMG; however, instead Task Force members usually offer comments and feedback on the Initial Submissions, which submitters can address (if they choose) by making a later Revised Submission.

The goals of the Task Force's Submission evaluation are:

- Provide a fair and open process
- Facilitate critical review of the submissions by OMG Members
- Provide feedback to submitters enabling them to address concerns in their revised submissions
- Build consensus on acceptable solutions
- Enable voting members to make an informed selection decision

Submitters are expected to actively contribute to the evaluation process.

3.2.5 Revised Submissions

Revised Submissions are due by the specified deadline. Revised Submissions cannot be altered once their submission deadline has passed. Submitters again normally present their proposals at the next meeting of the TF after the deadline. If necessary, the Task Force may set a succession of Revised Submission deadlines. Submitters choose whether or not to make Revised Submissions - if they decide not to, their most recent Submission is carried forward, unless the Submitter explicitly withdraws from the RFP process.

The evaluation of Revised Submissions has the same goals listed above.

3.2.6 Selection Votes

When the Task Force's voters believe that they sufficiently understand the relative merits of the available Submissions, a vote is taken to recommend a submission to the Task Force's parent Technical Committee. The Architecture Board reviews the recommended Submission for MDA compliance and technical merit. Once the AB has endorsed it, members of the relevant TC vote on the recommended Submission by email. Successful completion of this vote moves the recommendation to OMG's Board of Directors (BoD).

3.2.7 Business Committee Questionnaire

Before the BoD makes its final decision on turning a Technical Committee recommendation into an OMG published specification, it asks its Business Committee to evaluate whether implementations of the specification will be publicly available. To do this, the Business Committee will send a Questionnaire [BCQ] to every OMG Member listed as a Submitter on the recommended Submission. Members that are not Submitters can also complete a Business Committee Questionnaire for the Submission if they choose.

If no organization commits to make use of the specification, then the BoD will typically not act on the recommendation to adopt it – so it is very important that submitters respond to the BCQ.

Once the Business Committee has received satisfactory BCQ responses, the Board takes the final publication vote. A Submission that has been adopted by the Board is termed an *Alpha Specification*.

At this point the RFP process is complete.

3.2.8 Finalization & Revision

Any specification adopted by OMG by any mechanism, whether RFP or otherwise, is subject to Finalisation. A Finalization Task Force (FTF) is chartered by the TC that recommended the Specification; its task is to correct any problems reported by early users of the published specification. The FTF first collaborates with OMG's Technical Editor to prepare a cleaned-up version of the Alpha Specification with submission-specific material removed. This is the Beta1 specification, and is made publicly available via OMG's web site. The FTF then works through the list of bug reports ("issues") reported by users of the Beta1 specification, to produce a Finalisation Report and another Beta specification (usually Beta2), which is a candidate for Formal publication. Once endorsed by the AB and adopted by the relevant TC and BoD, this is published as the final, Formal Specification.

Long-term maintenance of OMG specifications is handled by a sequence of Revision Task Forces (RTFs), each one chartered to rectify any residual problems in the most-recently published specification version. For full details, see P&P section 4.4 [P&P].

4 Instructions for Submitters

4.1 OMG Membership

To submit to an RFP issued by the Platform Technology Committee an organisation shall maintain either Platform or Contributing OMG Membership from the date of the initial submission deadline, while to submit to a Domain RFP an organisation shall maintain either a Contributing or Domain membership.

4.2 Intellectual Property Rights

By making a Submission, an organisation is deemed to have granted to OMG a perpetual, nonexclusive, irrevocable, royalty-free, paid up, worldwide license to copy and distribute the document and to modify the document and distribute copies of the modified version, and to allow others to do the same. Submitter(s) shall be the copyright owners of the text they submit, or have sufficient copyright and patent rights from the copyright owners to make the Submission under the terms of OMG's IPR Policy. Each Submitter shall disclose the identities of all copyright owners in its Submission.

Each OMG Member that makes a written Submission in response to this RFP shall identify patents containing Essential Claims that it believes will be infringed if that Submission is included in an OMG Formal Specification and implemented.

By making a written Submission to this RFP, an OMG Member also agrees to comply with the Patent Licensing terms set out in section 6.10.

This section 4.2 is neither a complete nor an authoritative statement of a submitter's IPR obligations – see [IPR] for the governing document for all OMG's IPR policies.

4.3 Submission Effort

An RFP submission may require significant effort in terms of document preparation, presentations to the issuing TF, and participation in the TF evaluation process. OMG is unable to reimburse submitters for any costs in conjunction with their submissions to this RFP.

4.4 Letter of Intent

Every organisation intending to make a Submission against this RFP shall submit a Letter of Intent (LOI) signed by an officer on or before the deadline listed in section 6.11, or as later varied by the issuing Task Force.

The LOI should designate a single contact point within the submitting organization for receipt of all subsequent information regarding this RFP and the submission. The name of this contact will be made available to all OMG members. LOIs shall be sent by email, fax or paper mail to the "RFP Submissions Desk" at the OMG address shown on the first page of this RFP.

A suggested template for the Letter of Intent is available at http://doc.omg.org/loi [LOI].

4.5 Business Committee terms

This section contains the text of the Business Committee RFP attachment concerning commercial availability requirements placed on submissions. This attachment is available separately as OMG document omg/12-12-03.

4.5.1 *Introduction*

OMG wishes to encourage rapid commercial adoption of the specifications it publishes. To this end, there must be neither technical, legal nor commercial obstacles to their implementation. Freedom from the first is largely judged through technical review by the relevant OMG Technology Committees; the second two are the responsibility of the OMG Business Committee. The BC also looks for evidence of a commitment by a submitter to the commercial success of products based on the submission.

4.5.2 Business Committee evaluation criteria

4.5.2.1 Viable to implement across platforms

While it is understood that final candidate OMG submissions often combine technologies before they have all been implemented in one system, the Business Committee nevertheless wishes to see evidence that each major feature has been implemented, preferably more than once, and by separate organisations. Pre-product implementations are acceptable. Since use of OMG specifications should not be dependent on any one platform, cross-platform availability and interoperability of implementations should be also be demonstrated.

4.5.2.2 Commercial availability

In addition to demonstrating the existence of implementations of the specification, the submitter must also show that products based on the specification are commercially available, or will be within 12 months of the date when the specification was recommended for adoption by the appropriate Task Force. Proof of intent to ship product within 12 months might include:

- A public product announcement with a shipping date within the time limit.
- Demonstration of a prototype implementation and accompanying draft user documentation.

Alternatively, and at the Business Committee's discretion, submissions may be adopted where the submitter is not a commercial software provider, and therefore will not make implementations commercially available. However, in this case the BC will require concrete evidence of two or more independent implementations of the specification being used by end-user organisations as part of their businesses.

Regardless of which requirement is in use, the submitter must inform the OMG of completion of the implementations when commercially available.

4.5.2.3 Access to Intellectual Property Rights

OMG will not adopt a specification if OMG is aware of any submitter, member or third party which holds a patent, copyright or other intellectual property right (collectively referred to in this policy statement as "IPR") which might be infringed by implementation or recommendation of such specification, unless OMG believes that such IPR owner will grant an appropriate license to organizations (whether OMG members or not) which wish to make use of the specification. It is the goal of the OMG to make all of its technology available with as few impediments and disincentives to adoption as possible, and therefore OMG strongly encourages the submission of technology as to which royalty-free licenses will be available.

The governing document for all intellectual property rights (“IPR”) policies of Object Management Group is the Intellectual Property Rights statement, available at: http://doc.omg.org/ipr. It should be consulted for the authoritative statement of the submitter's patent disclosure and licensing obligations.

4.5.2.4 Publication of the specification

Should the submission be adopted, the submitter must grant OMG (and its sublicensees) a worldwide, royalty-free licence to edit, store, duplicate and distribute both the specification and works derived from it (such as revisions and teaching materials). This requirement applies only to the written specification, not to any implementation of it. Please consult the Intellectual Property Rights statement (http://doc.omg.org/ipr) for the authoritative statement of the submitter's copyright licensing obligations.

4.5.2.5 Continuing support

The submitter must show a commitment to continue supporting the technology underlying the specification after OMG adoption, for instance by showing the BC development plans for future revisions, enhancement or maintenance.

4.6 Responding to RFP items

4.6.1 Complete proposals

Submissions should propose full specifications for all of the relevant requirements detailed in Section 6 of this RFP. Submissions that do not present complete proposals may be at a disadvantage.

Submitters are encouraged to include any non-mandatory features listed in Section 6.

4.6.2 Additional specifications

Submissions may include additional specifications for items not covered by the RFP and which they believe to be necessary. Information on these additional items should be clearly distinguished. Submitters shall give a detailed rationale for why any such additional specifications should also be considered for adoption. Submitters should note that a TF is unlikely to consider additional items that are already on the roadmap of an OMG TF, since this would pre-empt the normal adoption process.

4.6.3 Alternative approaches

Submitters may provide alternative RFP item definitions, categorizations, and groupings so long as the rationale for doing so is clearly stated. Equally, submitters may provide alternative models for how items are provided if there are compelling technological reasons for a different approach.

4.7 Confidential and Proprietary Information

The OMG specification adoption process is an open process. Responses to this RFP become public documents of the OMG and are available to members and non-members alike for perusal. No confidential or proprietary information of any kind will be accepted in a submission to this RFP.

4.8 Proof of Concept

Submissions shall include a “proof of concept” statement, explaining how the submitted specifications have been demonstrated to be technically viable. The technical viability has to do with the state of development and maturity of the technology on which a submission is based. This is not the same as commercial availability. Proof of concept statements can contain any information deemed relevant by the submitter; for example:

> “This specification has completed the design phase and is in the process of being prototyped.”
>
> “An implementation of this specification has been in beta-test for 4 months.”
>
> “A named product (with a specified customer base) is a realization of this specification.”

It is incumbent upon submitters to demonstrate the technical viability of their proposal to the satisfaction of the TF managing the evaluation process. OMG will favor proposals based on technology for which sufficient relevant experience has been gained.

4.9 Submission Format

4.9.1 General

- Submissions that are concise and easy to read will inevitably receive more consideration.
- Submitted documentation should be confined to that directly relevant to the items requested in the RFP.
- To the greatest extent possible, the submission should follow the document structure set out in "ISO/IEC Directives, Part 2 – Rules for the structure and drafting of International Standards" [ISO2]. An OMG specification template is available to make it easier to follow these guidelines.
- The key words **"shall"**, **"shall not"**, **"should"**, **"should not"**, **"may"** and **"need not"** shall be used as described in Part 2 of the ISO/IEC Directives

[ISO2]. These ISO terms are compatible with the same terms in IETF RFC 2119 [RFC2119]. However, the RFC 2119 terms **"must"**, **"must not"**, **"optional"**, **"required"**, **"recommended"** and **"not recommended"** shall not be used (even though they are permitted under RFC2119).

4.9.2 Mandatory Outline

All submissions shall use the following structure, based on the OMG Specification template [TEMPL]:

Section 0 of the submission shall be used to provide all non-normative supporting material relevant to the evaluation of the proposed specification, including:

- The full name of the submission
- A complete list of all OMG Member(s) making the submission, with a named contact individual for each
- The acronym proposed for the specification (e.g. UML, CORBA)
- The name and OMG document number of the RFP to which this is a response
- The OMG document number of the main submission document
- Overview or guide to the material in the submission
- Statement of proof of concept (see 4.8)
- If the proposal does not satisfy any of the general requirements stated in Section 5, a detailed rationale explaining why
- Discussion of each of the “Issues To Be Discussed” identified in Section 6.
- An explanation of how the proposal satisfies the specific requirements and (if applicable) requests stated in Section 6.
- If adopting the submission requires making changes to already-adopted OMG specifications, include a list of those changes in a clearly-labelled subsection in Section 0. Identify exactly which version(s) of which OMG specification(s) shall be amended, and include the list of precise wording changes that shall be made to that specification.

Section 1 and subsequent sections of the submission shall contain the normative specification that the Submitter(s) is/are proposing for adoption by OMG, including:

- Scope of the proposed specification
- Overall design rationale
- Conformance criteria for implementations of the proposed specification, clearly stating the features that all conformant implementations shall support, and any features that implementations may support, but which are not mandatory.
- A list of the normative references that are used by the proposed specification
- A list of terms that are used in the proposed specification, with their definitions
- A list of any special symbols that are used in the proposed specification, together with their significance
- The proposed specification itself

Section 0 will be deleted from any specification that OMG adopts and publishes. Therefore Section 0 of the submission shall contain no normative material (other than any instructions to change existing specifications; ensuring that these are implemented is the responsibility of the FTF that finalises the specification, before it deletes section 0). Any non-normative material outside section 0 shall be explicitly identified.

The main submission document and any models or other machine-interpretable files accompanying it shall be listed in an inventory file conforming to the inventory template [INVENT].

The submission shall include a copyright waiver in a form acceptable to OMG. One acceptable form is:

> "Each of the entities listed above: (i) grants 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, and (ii) grants to each member of the OMG a nonexclusive, royalty-free, paid up, worldwide license to make up to fifty (50) copies of this document for internal review purposes only and not for distribution, and (iii) 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 any OMG specification that may be based hereon or having conformed any computer software to such specification."

Other forms of copyright waiver may only be used if approved by OMG legal counsel beforehand.

4.10 How to Submit

Submitters should send an electronic version of their submission to the *RFP Submissions Desk (rfp@omg.org)* at OMG Headquarters by 5:00 PM U.S. Eastern Standard Time (22:00 GMT) on the day of the Initial and Revised Submission deadlines. Acceptable formats are Adobe FrameMaker source, ISO/IEC 26300:2006 (OpenDoc 1.1), OASIS DocBook 4.x (or later) and ISO/IEC 29500:2008 (OOXML, .docx).

Submitters should ensure that they receive confirmation of receipt of their submission.

5 General Requirements on Proposals

5.1 Requirements

5.1.1 Use of modelling languages

Submitters are encouraged to express models using OMG modelling languages such as UML, MOF, CWM and SPEM (subject to any further constraints on the types of the models and modeling technologies specified in Section 6 of this RFP). Submissions containing models expressed using OMG modeling languages shall be accompanied by an OMG XMI [XMI] representation of the models (including a machine-readable copy). A best effort should be made to provide an OMG XMI representation even in those cases where models are expressed via non-OMG modeling languages.

5.1.2 PIMs & PSMs

Section 6 of this RFP specifies whether PIM(s), PSM(s), or both are being solicited. If proposals specify a PIM and corresponding PSM(s), then the rules specifying the mapping(s) between the PIM and PSM(s) shall either be identified by reference to a standard mapping or specified in the proposal. In order to allow possible inconsistencies in a proposal to be resolved later, proposals shall identify whether it's the mapping technique or the resulting PSM(s) that shall be considered normative.

5.1.3 Complete submissions

Proposals shall be *precise* and *functionally complete*. Any relevant assumptions and context necessary to implement the specification shall be provided.

5.1.4 Reuse

Proposals shall *reuse* existing OMG and other standard specifications in preference to defining new models to specify similar functionality.

5.1.5 Changes to existing specifications

Each proposal shall justify and fully specify any *changes or extensions* to existing OMG specifications necessitated by adopting that proposal. In general, OMG favors proposals that are *upwards compatible* with existing standards and that minimize changes and extensions to existing specifications.

5.1.6 Minimalism

Proposals shall factor out functionality that could be used in different contexts and specify their models, interfaces, etc. separately. Such *minimalism* fosters re-use and avoids functional duplication.

5.1.7 Independence

Proposals shall use or depend on other specifications only where it is actually necessary. While re-use of existing specifications to avoid duplication will be encouraged, proposals should avoid gratuitous use.

5.1.8 Compatibility

Proposals shall be *compatible* with and *usable* with existing specifications from OMG and other standards bodies, as appropriate. Separate specifications offering distinct functionality should be usable together where it makes sense to do so.

5.1.9 Implementation flexibility

Proposals shall preserve maximum *implementation flexibility*. Implementation descriptions should not be included and proposals shall not constrain implementations any more than is necessary to promote interoperability.

5.1.10 Encapsulation

Proposals shall allow *independent implementations* that are *substitutable* and *interoperable*. An implementation should be replaceable by an alternative implementation without requiring changes to any client.

5.1.11 Security

In order to demonstrate that the specification proposed in response to this RFP can be made secure in environments that require security, answers to the following questions shall be provided:

- What, if any, security-sensitive elements are introduced by the proposal?
- Which accesses to security-sensitive elements should be subject to security policy control?
- Does the proposed service or facility need to be security aware?
- What default policies (e.g., for authentication, audit, authorization, message protection etc.) should be applied to the security sensitive elements introduced by the proposal? Of what security considerations should the implementers of your proposal be aware?

The OMG has adopted several specifications, which cover different aspects of security and provide useful resources in formulating responses. [SEC] [RAD].

5.1.12 Internationalization

Proposals shall specify the degree of internationalization support that they provide. The degrees of support are as follows:

a) Uncategorized: Internationalization has not been considered.

b) Specific to <region name>: The proposal supports the customs of the specified region only, and is not guaranteed to support the customs of any other region. Any fault or error caused by requesting the services outside of a context in which the customs of the specified region are being consistently followed is the responsibility of the requester.

c) Specific to <multiple region names>: The proposal supports the customs of the specified regions only, and is not guaranteed to support the customs of any other regions. Any fault or error caused by requesting the services outside of a context in which the customs of at least one of the specified regions are being consistently followed is the responsibility of the requester.

d) Explicitly not specific to <region(s) name>: The proposal does not support the customs of the specified region(s). Any fault or error caused by requesting the services in a context in which the customs of the specified region(s) are being followed is the responsibility of the requester.

5.2 Evaluation criteria

Although the OMG adopts model-based specifications and not implementations of those specifications, the technical viability of implementations will be taken into account during the evaluation process. The following criteria will be used:

5.2.1 Performance

Potential implementation trade-offs for performance will be considered.

5.2.2 Portability

The ease of implementation on a variety of systems and software platforms will be considered.

5.2.3 Securability

The answer to questions in section 5.1.11 shall be taken into consideration to ascertain that an implementation of the proposal is securable in an environment requiring security.

5.2.4 Conformance: Inspectability and Testability

The adequacy of proposed specifications for the purposes of conformance inspection and testing will be considered. Specifications should provide sufficient constraints on interfaces and implementation characteristics to ensure that conformance can be unambiguously assessed through both manual inspection and automated testing.

5.2.5 Standardized Metadata

Where proposals incorporate metadata specifications, OMG standard XMI metadata [XMI] representations should be provided.

6 Specific Requirements on Proposals

6.1 Problem Statement

The transition to a model-based systems engineering (MBSE) approach is essential for systems engineering to meet the demands of increasing system complexity, productivity and quality, and shorter design cycles. Many other engineering disciplines, such as mechanical, electrical, and controls engineering, utilize models as an integral part of their practice. Models have long been important for systems engineering to support systems analysis and design, but MBSE emphasizes the need to create a coherent model of the system architecture that helps integrate other aspects of the design, including electrical,

mechanical, software, and other cross-cutting considerations such-as reliability, safety, and security.

The system model provides a shared view of the system that can enhance communication and coordination across the system development lifecycle. This model represents an authoritative source of information that is maintained to ensure consistency and traceability between requirements, design, analysis, and verification. The model-based approach contrasts with the traditional document-based approach in which information about the system is captured independently in many different documents using common applications such as Word, Visio, Excel, and PowerPoint. To take full advantage of a model-based approach, the system model must be maintained as part of the technical baseline, and integrated with other engineering models and tools.

The capability to express system concepts in the form of models can result in quality improvements by reducing downstream design errors, and in productivity improvements through reuse of models throughout the lifecycle and across projects. The system model can result in other benefits, such as the ability to automate tasks like change impact analysis, and auto-generation of reports and documentation with increased confidence that the information is valid, complete, and consistent.

A system modeling language is an essential capability to specify and architect increasingly complex systems. A systems modeling language enables the expression of fundamental concepts about the system such as system composition, interconnection and interfaces, functional and state-based behavior, parametric aspects, and traceability relationships between requirements, design, analysis, and verification. A standard systems modeling language can help overcome the informational "Tower of Babel" by providing a means to express these concepts in a standard and precise way that enables communications between engineers and tools.

SysML v1 was adopted in 2006 as a general-purpose graphical modeling language for specifying, analyzing, designing, and verifying complex systems that may include hardware, software, information, personnel, procedures, and facilities. The language provides graphical representations with a semantic foundation for modeling system requirements, behavior, structure, and constraints.

Since its adoption, SysML enabled broad recognition and increased adoption of model-based systems engineering practices across industry. Systems engineers, tool vendors, and academia have learned much from this experience, including both the strengths and weaknesses of SysML as a language, and the benefits and challenges of adopting and applying MBSE with SysML. The SysML

specification has continued to evolve through the SysML Revision Task Force since its adoption. However, the scope of the RTF limits how much change can be introduced to address the needs.

Based on the industry experiences with the adoption and use of MBSE with SysML over the preceding 10-plus years, it was determined that a more comprehensive update to the language is needed beyond what can be accomplished through the SysML RTF alone. This RFP is intended to enable a re-architecting of the SysML profile of UML, and provide a SysML v2 metamodel that is not constrained by UML, to address some of the more fundamental issues associated with the language, including the need for additional expressiveness, increased precision, interoperability, and improved consistency and integration of the concepts, such as those related to behavior and structure. Section 6.2 describes the scope of the proposals to address these needs.

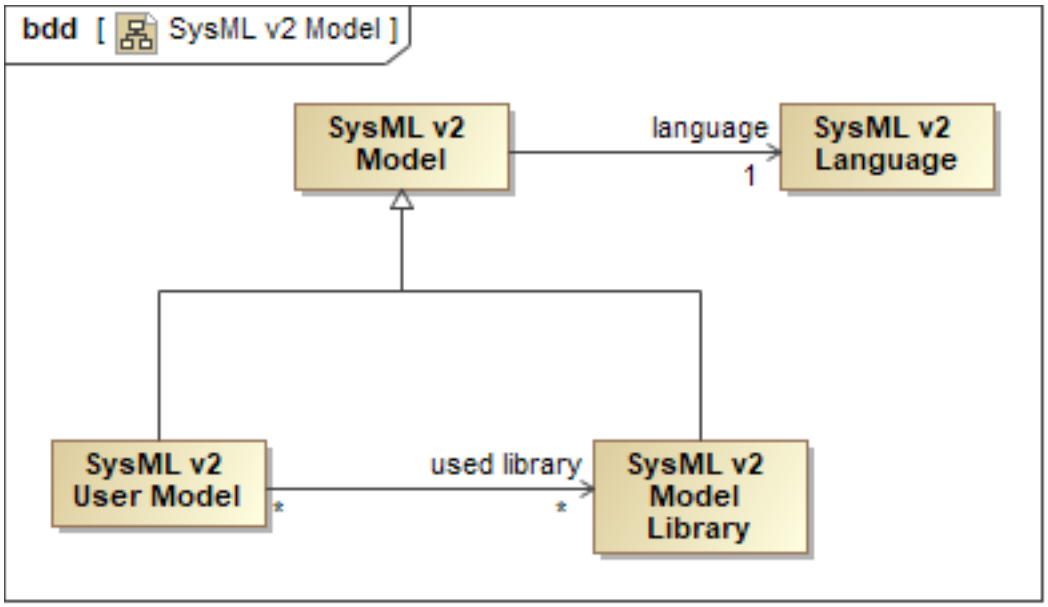
6.2 Scope of Proposals Sought

6.2.1 Language Architecture and Formalism

6.2.1.1 Language Architecture

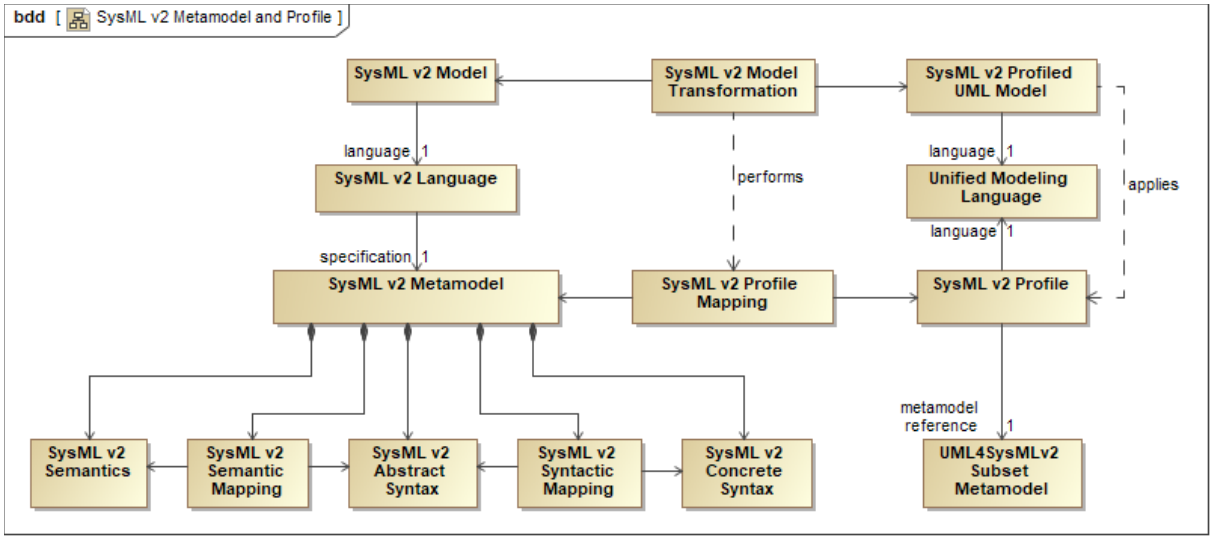
The SysML v2 modeling language will be used to represent SysML v2 models. As shown in Figure 1, SysML v2 models will include both models created by SysML v2 end users and model libraries containing reusable modeling components that may be used in the creation of user models. In particular, some of the SysML v2 language requirements may be implemented in the SysML v2 specification as user-level model libraries.

Figure 1. SysML v2 Models and Model Libraries



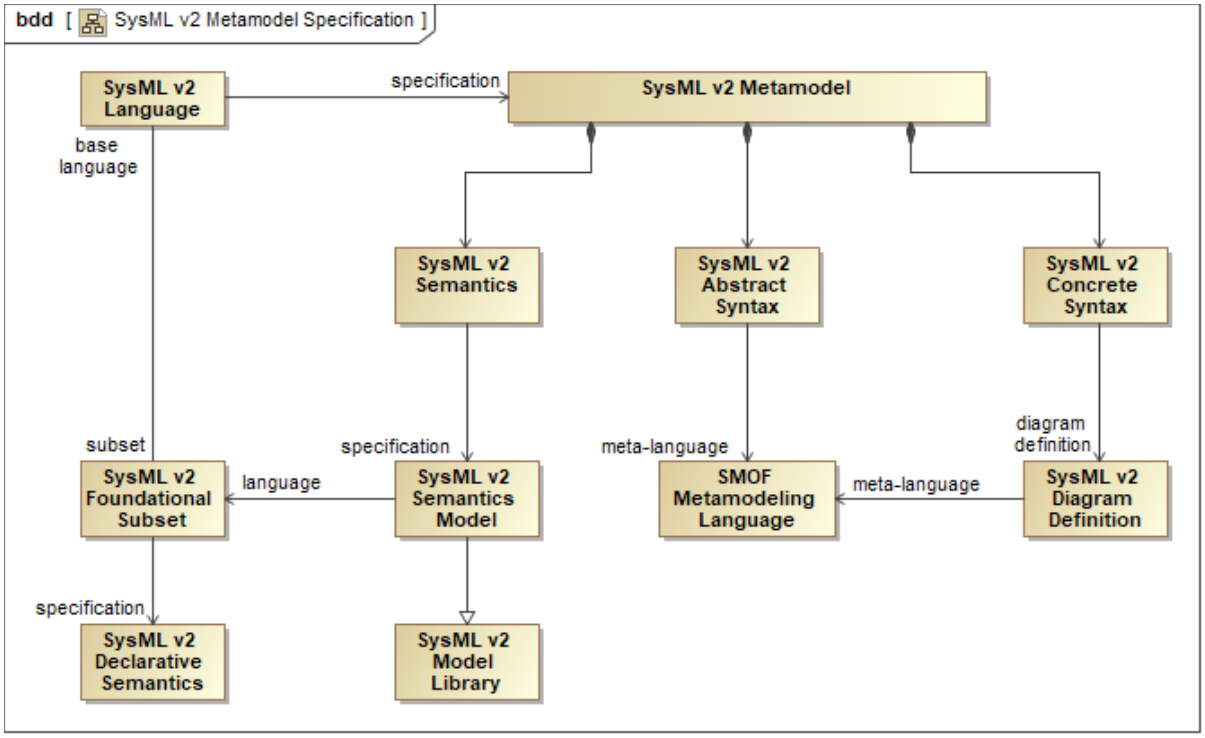
The SysML v2 language will be specified using a SysML v2 metamodel that defines the language's semantics, abstract syntax and concrete syntax, and the relationships between them, as shown in the Figure 2. The language also provides mechanisms to support further customization to reflect domain specific concepts. In addition, the SysML v2 metamodel will be mapped to a SysML v2 profile. This will allow a SysML v2 model to also be represented as an extension of a UML model, using the profile to adapt UML syntax and semantics to those of SysML v2. The combination of a metamodel and a profile will enable a broader range of vendor implementations. The metamodel supports implementation of the system concepts without some of the constraints imposed by UML, while the profile supports implementation of the system concepts in a way that is more closely aligned with SysML v1 implementations.

Figure 2. SysML v2 Metamodel and Profile



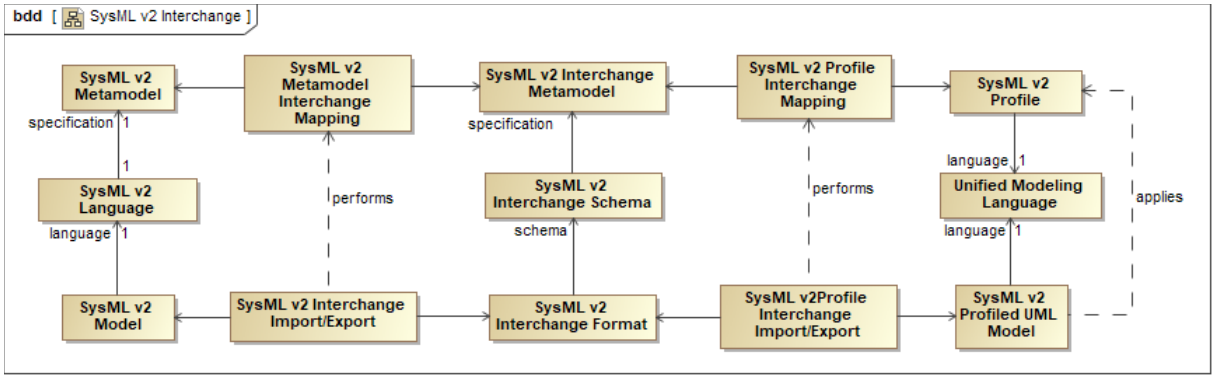
As shown in the Figure 3, the SysML v2 abstract syntax may be specified using the OMG-standard Meta-Object Facility (MOF) or its extension, Semantic MOF (SMOF), which provides the basis for the proposed Metamodel Extension Facility (MEF). The SysML v2 concrete syntax may be specified using common Backus-Naur Form (BNF) productions for textual notations and OMG-standard MOF-based Diagram Definition for graphical notations. These MOF-based standards, however, only provide support for structural modeling of the (concrete and abstract) syntactic constructs of the language. The SysML v2 semantics can be specified using a foundational subset of the language for which the semantics are specified using a separate declarative formalism (as described further in the section on Formalism below).

Figure 3. SysML v2 Language Metamodel Specification



Having the SysML v2 metamodel aligned with the SysML v2 profile will also allow for the specification of a SysML v2 format for interchanging models between tools, usable by both metamodel and profile-based tools. As shown in the Figure 4, this will be done by defining mappings from both the SysML v2 metamodel and the SysML v2 profile to a common SysML v2 interchange metamodel, consistent with the mapping between the SysML v2 metamodel and the SysML v2 profile. That is, a SysML v2 model based on the metamodel and profile are related by the SysML v2 metamodel-to-profile mapping and represented in the same way for the purposes of model interchange. This will allow a SysML v2 model exported from a metamodel-based tool to be imported into a profile-based tool, and vice versa. Furthermore, the SysML v2 interchange format enables interchange of both the abstract syntax representation of a SysML v2 model and the concrete syntax representation of views of that model.

Figure 4. SysML v2 Model Interchange



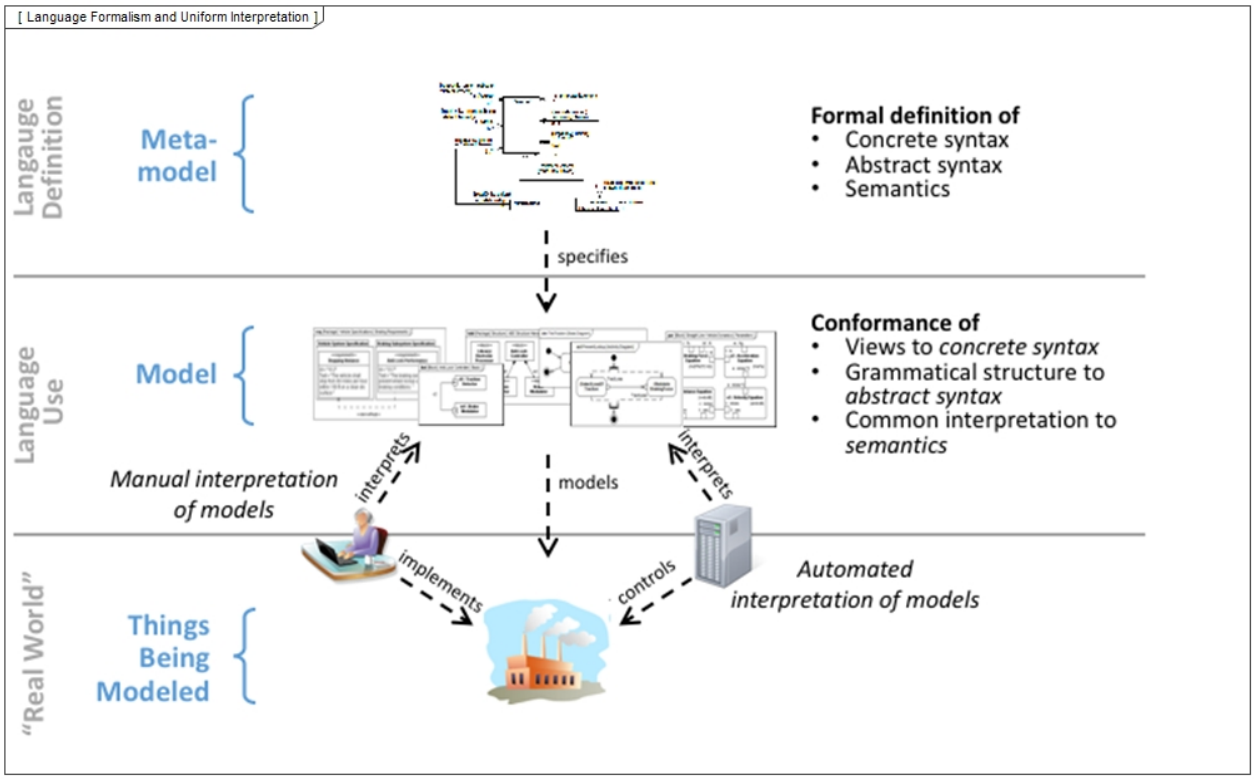
6.2.1.2 Formalism

In mathematics and logic, formalism has to do with how something is structured and expressed, as opposed to the actual content of what is being expressed. Formalism aims to express content in a well-defined form, such that this expression can be given a uniform interpretation. A formalism may also extend to rules for the consistent manipulation of the form of expression, such as the ability to construct formal proofs using deduction rules based on given axioms and to reason about the system being represented.

For SysML, the formalism defines how the language itself is specified in terms of its syntax and semantics, as opposed to what is in the language. This includes: the abstract syntax that specifies the grammar of the language, including the basic constructs of the language analogous to verbs and nouns, and the rules for constructing legal sentences (i.e., statements); the concrete syntax that specifies the symbols (textual, graphical and diagrammatic) that define how grammatical constructs in the language can be presented; and the semantics that specify the meaning of the constructs so that they can be interpreted in the domain that the model is intended to represent. The specification of how models are interchanged can also be considered part of the formalism. The rigorous specification of the abstract syntax, concrete syntax, semantics, and interchange format is intended to ensure the precision and integrity of the language.

Figure 5 shows the relationship between the language formalism and the things being modeled. The goal of a formal specification for SysML is to provide a uniform syntactic and semantic interpretation for the language. That is, a SysML model should be interpreted in a consistent way and subject to an objective evaluation as to whether it conforms to the SysML v2 Specification, whether this interpretation is done by a human that interprets a view of the model, or a machine that interprets the model.

Figure 5. Language Formalism and Uniform Interpretation



SysML v1 is specified using the formalism of UML 2 profiles. A UML profile is an extended subset of the UML metamodel, which is itself specified using the formalism provided by the Meta Object Facility (MOF). The SysML v1 specification defines the abstract syntax of SysML as a profile of UML that extends the subset of UML abstract syntax using stereotypes, extends the concrete syntax of UML diagrams, and adopts and adapts UML semantics as appropriate.

There are some significant limitations of the formalism used for SysML v1 that result in ambiguities of interpretation. For example, SysML v1 does not include a complete formal mapping between the concrete syntax and the abstract syntax, which can result in ambiguity in how a SysML diagram conforms to the rules of the grammar. In addition, the semantics of SysML v1 are often defined in English rather than a more precise formal representation, which can result in ambiguity of meaning.

In contrast, SysML v2 will have a more formal specification of its abstract syntax, concrete syntax and semantics, and the mappings between them. To maximize the flexibility of this specification, the required approach is to specify a small set of foundational concepts and their base semantics using a mathematical *declarative* semantics. Then, model libraries written in SysML itself, grounded in the base semantics, are used to further extend the concepts of

the language and their associated semantics. These extensions will represent the core domain concepts for the SysML v2 language. SysML v2 is also intended to include additional user-level model libraries that extend these core concepts, and provide a mechanism to further customize the language.

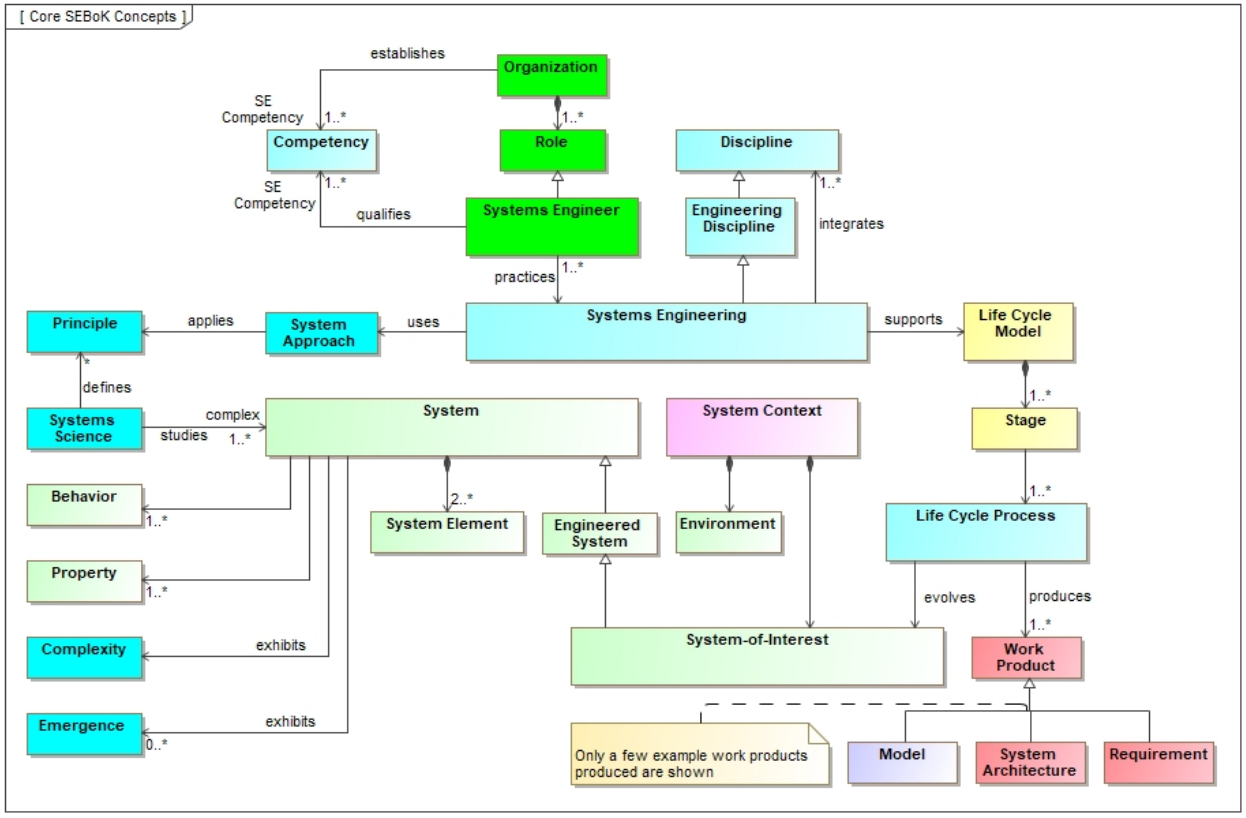
The advantage of grounding SysML semantics in a declarative approach is that well-known techniques of mathematical logic can then be used to make formal deductions based on the assertions made in a model, in order to prove things that are true or not about the system or domain that is being modeled. Declarative semantics contrast with the operational semantics which specify how a model executes, such that the execution results are evaluated to determine how the system will behave. It is expected that the full semantics for SysML v2 will include both declarative and operational components.

As an example of how the semantics of SysML v2 could be built up from a declarative base, consider the case of the semantics of control nodes used in activity diagrams. Currently (in UML and, so, SysML v1), each type of control node such as a fork node, join node, decision node, or merge node is defined with its own unique semantics. In SysML v2, the general concept of a control node might be specified along with its base semantics. The specific semantics for fork, join, decision and merge nodes could then be specified in the core model library, specializing the base control node semantics. The language formalism would include rules for how this could be done in an unambiguous, rigorous way. A formal mathematically-based language does not have to be difficult to use. The usability of the language will be emphasized using graphical, textual and tabular notations appropriate for practicing system engineers.

6.2.2 Data Model

The requirements for SysML v2 are intended to be consistent with industry standards for systems engineering that include the Systems Engineering Body of Knowledge (SEBoK), the ISO standard for Systems and Software Engineering -- System lifecycle processes (ISO/IEC/IEEE 15288:2015), and the INCOSE Systems Engineering Handbook v4. These sources and others provide high-level concepts that serve as input to the requirements for SysML v2. Figure 6 shows some of the core concepts that are defined in the SEBoK. There are many other concepts in the industry reference model beyond what is shown in this figure.

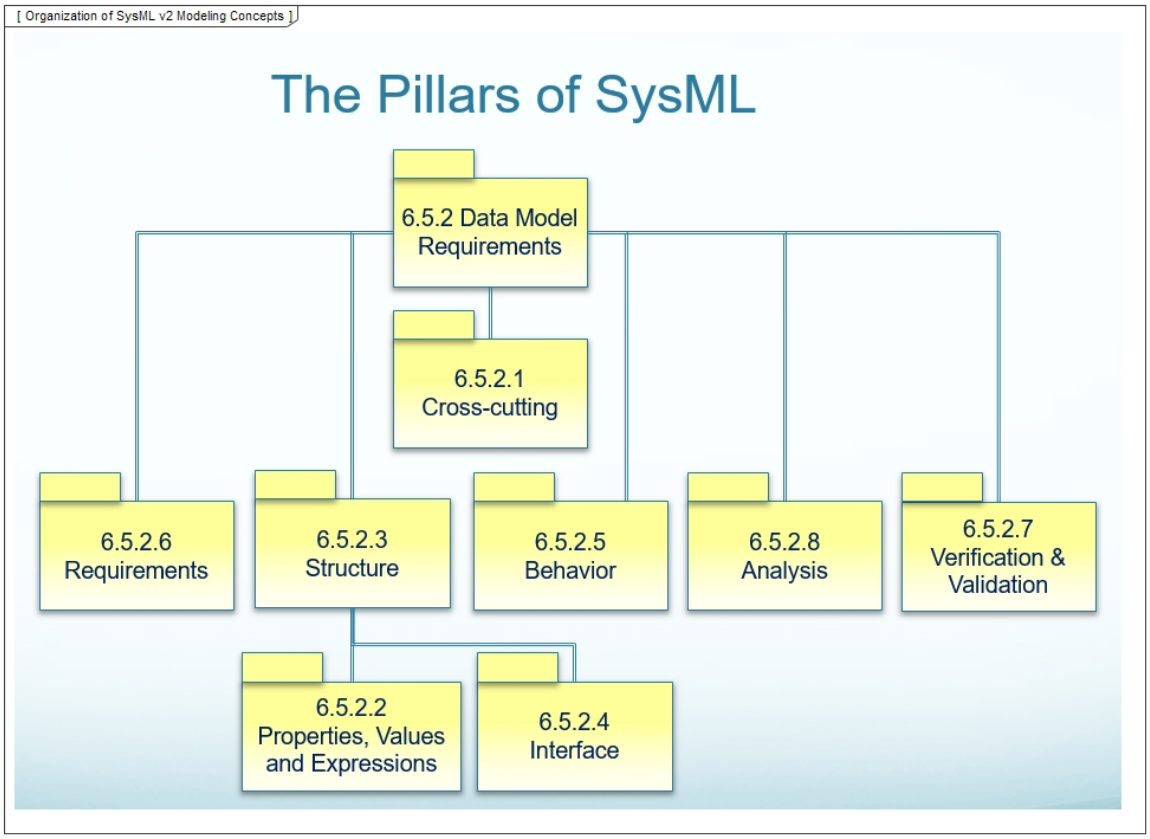
Figure 6. Core SEBoK Concepts (Source: SEBoK Part 1: Introduction to Core Concepts)



The scope of SysML v2 is not intended to address the full scope of these concepts, but focuses on those concepts directly related to the specification, design, analysis, and verification of systems. At the same time, the requirements for SysML v2 include additional concepts that are not explicitly referred to in these concepts but have been identified through the requirements elicitation and development process.

The scope of SysML v2 will encompass the scope of SysML v1, which includes support for modeling structure, behavior, parametric, and requirements, often referred to as the 4 pillars of SysML. SysML v2 will also support modeling concepts related to verification, analysis, and other concepts beyond what is in SysML v1. The organization of the system modeling concepts are indicated in Figure 7.

Figure 7. Organization of SysML v2 Modeling Concepts



In addition to extending the SysML v1 concepts, a major emphasis for SysML v2 will be to ensure integration and consistency of these concepts across the language..

A preliminary data model, also referred to as a concept model, is included in Appendix C. *This model does not constitute part of the requirements, but is provided to help better understand the intent of the requirements.*

6.3 Relationship to other OMG Specifications and activities

6.3.1 Relationship to OMG specifications

Proposals may reference and build upon any of the OMG specifications identified in this section. In each case, the most recent version is applicable, unless the most recent version was adopted less than six months before the final submission to this specification, in which case the previous version may be used. Proposals should identify the specific dependencies they have on any of these specifications including their specific version.

Submitters are required to consider the most up-to-date versions of the following OMG specifications, as specified in 6.5.

[DD] Diagram Definition ™ (DD™)

[MOF] Meta Object Facility ™ (MOF™) Core

[SMOF] MOF Support for Semantic Structures ™ (SMOF™)

[SysML] OMG Systems Modeling Language ™ (SysML®)

[UML] Unified Modeling Language ™ (UML®)

[XMI] XML Metadata Interchange ™ (XMI®)

Submitters could consider the most up-to-date versions of the following OMG specifications.

[ALF] Action Language for Foundational UML™ (Alf ™)

[BMM] Business Motivation Metamodel (BMM™)

[DMN] Decision Model and Notation ™ (DMN™)

[FUML] Semantics of a Foundational Subset for Executable UML Models (fUML™)

[MARTE] UML Profile for MARTE™

[OCL] Object Constraint Language ™ (OCL™)

[ODM] Ontology Definition Metamodel ™ (ODM™)

[PSCS] Precise Semantics of UML Composite Structures ™ (PSCS™)

[PSSM] Precise Semantics of UML State Machines (PSSM)

[ReqIF] Requirements Interchange Format (ReqIF ™)

[SysPISF] SysML Extension for Physical Interaction and Signal Flow Simulation (SysPISF)

[UAF] Unified Architecture Framework (UAF) previously UPDM [UTP] UML Testing Profile ™ (UTP™)

6.3.2 Relationship to other OMG Documents and work in progress

Submissions for the following related OMG RFPs are currently in progress at the time of issuance of this RFP:

[MEF] Metamodel Extension Facility
[S&R] Profile for Safety and Reliability

[MVF] Multiple Vocabulary Facility RFP

[SIMF] Semantic Information Modeling RFP

[UOTR] UML Operational Threat & Risk Model

6.4 Related non-OMG Activities, Documents and Standards

Proposals may reference and build upon any of the non-OMG Activities, Documents and Standards identified in this section. In each case, the most recent version is applicable, unless the most recent version was adopted less than six months before the final submission to this specification, in which case the previous version may be used.

[ISO 80000] Quantities and units -- Part 1: General: ISO 80000-1:2009
[FMI] Functional Mock-Up Interface (FMI)

[STEP] ISO 10303-233:2012 (STEP)

[ArchDes] ISO 42010/IEC/IEEE:2011 - Systems and software engineering - Architecture description

[ISO 15288] ISO/IEC 15288:2015 - Systems and software engineering - System lifecycle processes

[ISO 15704] Industrial automation systems - Requirements for enterprise-reference architectures and methodologies

[ISO 26550] ISO/IEC 26550:2015 - Software and Systems Engineering - Reference model for product line engineering and management

[ISO 80000] Quantities and units -- Part 1: General: ISO 80000-1:2009

[HCD] ISO/DIS 9241-220.2(en) Ergonomics of human-system interaction - Part 220: Processes for enabling, executing and assessing human-centered design within organizations

[SE Handbook] INCOSE Systems Engineering Handbook

[SEBoK] Systems Engineering Body of Knowledge (SEBoK)

6.5 Mandatory Requirements

In the following section, some of the requirements include additional information identified as **Supporting Information** and **SysML v1.x Constructs**. This information is intended to be informative only to aid in the understanding of the mandatory requirement. The SysML v1.x construct refers to the corresponding construct in SysML v1.x that partially or fully addresses the requirement. In addition, the glossary defines many of the terms used in the requirements, and should be referred to for additional clarification.

The requirements in the following paragrpahs have numbers that reflect their logical grouping, which are not constrained by the paragraph that they appear in.

6.5.1 Language Architecture and Formalism Requirements

LNG 1: Language Architecture and Formalism Requirements Group

This group specifies how the language is structured and defined.

Supporting Information: Some concepts may be implemented as user-level model libraries.

LNG 1.1: Metamodel and Profile Group

LNG 1.1.1: SysML Metamodel

Proposals for SysML v2 shall be specified using a metamodel that includes abstract syntax, concrete syntax, semantics, and the relationships between them.

LNG 1.1.2: Metamodel Specification

Proposals for the SysML v2 metamodel shall be specified in MOF or SMOF.

Supporting Information: MOF is a subset of SMOF. SMOF provides support for the Metamodel Extension Facility (MEF).

LNG 1.1.3: SysML Profile

Proposals for SysML v2 shall be specified as a SysML v2 profile of UML that includes, as a minimum, the functional capabilities of the SysML v1.x profile, and a mapping to the SysML v2 metamodel.

Supporting information: Equivalent functional capability can be demonstrated by mapping the UML metaclasses and SysML stereotypes between SysML v2 and SysML v1.

SysML v1.X Constructs: SysML v1.x Profile

LNG 1.3: Abstract Syntax Group

LNG 1.3.1: Syntax Specification

Proposals for SysML v2 abstract and concrete syntax shall be specified using MOF or SMOF (including constraints on syntactic structure).

Supporting Information:

Expressing the syntax formally using a single consistent language which is more understandable to the user.

LNG 1.3.2: View Independent Abstract Syntax

Proposals for the SysML v2 abstract syntax representation of SysML v2 models shall be independent of all views of the models.

Supporting Information: Rationale

This is intended to define the concept independent of how it is presented. This enables a consistent representation of concepts with common semantics across a diverse range of views, including graphical, tabular, and other textual representations.

LNG 1.4: Concrete Syntax Group

LNG 1.4.1: Concrete Syntax to Abstract Syntax Mapping

Proposals for the SysML v2 concrete syntax representationof all views of a SysML model shall be separate from, and mapped to the abstract syntax representation of that model.The concrete syntax representation can include one or more images or snippets of images,

Supporting Information:

Enables views to provide unambiguous concrete representation of the abstract syntax of the model.

Enables views to be rendered in a consistent way across tools.

SysML v1.X Constructs: Diagram Definition

LNG 1.4.2: Graphical Concrete Syntax

Proposals for SysML v2 shall provide a standard graphical concrete syntax.

SysML v1.X Constructs: Graphical syntax

LNG 1.4.3: Syntax Examples

All examples of model views in the proposals for the SysML v2 specification shall include the concrete syntax of the view, and the mapping to the abstract syntax representation of the parts of the models being viewed.

Supporting Information:

Experience has shown that the mapping of examples to the concrete and abstract syntax is not always obvious. Making these mappings explicit helps clarify their formal specification.

LNG 1.5: Extensibility Group

LNG 1.5.1: Extension Mechanisms

Proposals for SysML v2 syntax and semantics shall include mechanisms to subset and extend the language.

Supporting Information: This is essential to enable further customization of the language. SysML v1 includes a stereotype and profile mechanism to extend the language.

SysML v1.X Constructs: Stereotype, Profile

LNG 1.6: Model Interchange, Mapping, and Transformations Group

LNG 1.6.3: UML Interoperability

Proposals for SysML v2 shall provide the capability to map shared concepts between SysML and UML.

SysML v1.X Constructs: SysML Profile of UML

6.5.2 Data Model Requirements

6.5.2.1 Cross-cutting Requirements

CRC 1: Cross-cutting Requirements Group

The following specify the requirements that apply to all model elements.

CRC 1.1: Model and Model Library Group

CRC 1.1.1: Model

Proposals for SysML v2 shall include a capability to represent a Model (aka system model) that contains a set of uniquely identifiable model elements.

Supporting Information: This is intended to be a kind of Container or Namespace.

SysML v1.X Constructs: Model

CRC 1.1.2: Model Library

Proposals for SysML v2 shall include a capability to represent a Model Library that contains a set of model elements that are intended to support reuse.

Supporting Information: This is intended to be a kind of Container or Namespace.

SysML v1.X Constructs: Model Library

CRC 1.1.3: Container

Proposals for SysML v2 shall include the capability to represent a Container that is a model element that contains other model elements. Model elements within a container shall be distinguishable from one another.

Supporting Information: This provides a way to organize the model. Containers can contain other containers.

SysML v1.X Constructs: Package

CRC 1.2: Model Element Group

CRC 1.2.2: Unique Identifier

Proposals for SysML v2 shall include a capability to represent a single universally unique identifier for each model element that cannot be changed.

Supporting Information: The unique identifier should enable assignment of URIs.

SysML v1.X Constructs: UUID is part of the XMI specification

CRC 1.2.3: Name and Aliases

Proposals for SysML v2 shall include a capability to represent a name and one or more aliases for any named model element.

Supporting Information:

Selected kinds of model elements may not require a name (e.g. dependency), or the name may be optional, but still should be distinguishable within a namespace.

Aliases enable users to assign more than one name for the same element, such as a shortened name. A common use of aliases is the use of an abbreviated or shortened name.

SysML v1.X Constructs: Named Element

CRC 1.2.4: Definition / Description

Proposals for SysML v2 shall include a capability to represent one or more definitions and/or descriptions for each model element.

SysML v1.X Constructs: Owned Comment

CRC 1.2.5: Annotation

Proposals for SysML v2 shall include a capability to represent an annotation of one or more model elements that includes a text string. The text string can include a link that refers to a Navigation relationship (refer to CRC 1.3.10), and a classification field to identify the kind of annotation.

Supporting Information: Annotations should be able to be related to other elements.

SysML v1.X Constructs: Comment

CRC 1.2.6: Element Group

Proposals for SysML v2 shall include a capability to represent a group of model elements that can satisfy user-defined criteria for membership in the group.

Supporting Information:

1. A member of an element group is not intended to impose ownership constraints on the members.

2. Element group can be specialized for different kinds of members, such as groups that contain requirements, functions, and structural elements, which may impose additional constraints on its members.

SysML v1.X Constructs: Element Group

CRC 1.2.7: Additional Cross-Cutting Concepts Group

The requirements in this group include additional concepts that can be associated with any model element.

CRC 1.2.7.1: Problem

Proposals for SysML v2 shall include a capability to represent a problem that causes an undesired affect.

Supporting Information: A problem is often represented as a cause in a cause-effect relationship.

SysML v1.X Constructs: Problem

CRC 1.2.7.2: Risk

Proposals for SysML v2 shall include a capability to represent a Risk that identifies the kind of risk (e.g., cost, schedule, technical), and the likelihood of occurrence, and the potential impact.

CRC 1.3: Model Element Relationships Requirements Group

CRC 1.3.01: Relationship

Proposals for SysML v2 shall include a capability to represent a Relationship between any two model elements, which may have a name and direction.

SysML v1.X Constructs: Relationship

CRC 1.3.02: Derived Relationship

Proposals for SysML v2 shall include a capability to represent a relationship that is derived from other relationships.

Supporting Information:

An example is a derived relationship from a transitive relationship where B relates to A and C relates to B, then C relates to A.

Another example is a connector between two composite parts that is derived from a connector between their nested parts.

CRC 1.3.03: Dependency Relationship

Proposals for SysML v2 shall include a capability to represent a Dependency Relationship where one side of the relationship refers to the independent element and the other side of the relationship refers to the dependent element.

SysML v1.X Constructs: Dependency

CRC 1.3.04: Cause-Effect Relationship

Proposals for SysML v2 shall include a capability to represent a Cause-Effect Relationship where one side of the relationship refers to the cause and the other side of the relationship refers to the effect.

CRC 1.3.05: Explanation Relationship

Proposals for SysML v2 shall include a capability to represent an Explanation Relationship where one side of the relationship refers to the rationale and the other side of the relationship refers to the element being explained.

SysML v1.X Constructs: Anchor on a rationale

CRC 1.3.06: Conform Relationship

Proposals for SysML v2 shall include a capability to represent a Conform Relationship where the conforming element is constrained by the element on the other side of the relationship.

CRC 1.3.07: Refine Relationship

Proposals for SysML v2 shall include a capability to represent a Refine Relationship where the refined side of the relationships refers to the more precisely specified element.

SysML v1.X Constructs: RefineReqt

CRC 1.3.08: Allocation Relationship

Proposals for SysML v2 shall include a capability to represent an Allocation Relationship where one side of the relationship refers to the allocated from, and the other side of the relationship refers to the allocated to.

SysML v1.X Constructs: Allocate

CRC 1.3.09: Element Group Relationship

Proposals for SysML v2 shall include a capability to represent an Element Group Relationship where one side of the relationship refers to the member, and the other side of the relationship refers to the Element Group.

SysML v1.X Constructs: Anchor

CRC 1.3.10: Navigation Relationship

Proposals for SysML v2 shall include a capability to represent a Navigation Relationship between a model element and another model element or an external element, similar to a hyperlink, where one side of the relationship refers to the linked to, and the other side of the relationship refers to the linked from. The external element can be a data element, a file, and/or an element of an external model.

Supporting information:

This is a navigation aid that standardizes what many tools already do.

The navigation can specify the ability to navigate from either end of the relationship.

SysML v1.X Constructs: Some tools support navigation links, but not in a standard way.

CRC 1.4: Variability Modeling Group

The requirements in this group should accommodate approaches to model variants as choices among design options. The modeling approaches may include a separate variability model to identify the design choices. Additional variability modeling concepts may be included.

Supporting information: refer to ISO/IEC 26550:2015

CRC 1.4.1: Variation Point

Proposals for SysML v2 shall include a capability to model variation points that identify features that can vary across a set of variants (e.g., vehicles with manual or automatic transmission, variable number of axles, or variable wheel size). A variation point may be dependent on another variant selection. (e.g., number of axles and wheel size is dependent on selection of load size).

CRC 1.4.2: Variant

Proposals for SysML v2 shall include a capability to model variants that correspond to particular selections that are associated with a variation point.

CRC 1.4.3: Variability Expression and Constraints

Proposals for SysML v2 shall include a capability to model variability expressions that constrain possible variant choices (e.g., 3 axles plus large wheel size or 2 axles plus small wheel size).

CRC 1.4.4: Variant Binding

Proposals for SysML v2 shall include a capability to model the binding between a variant and the model elements that vary.

Supporting Information: The binding is intended to enable the use of a separate variability model that defines variation that may span multiple kinds of models such as a SysML model, simulation model, and a CAD model.

CRC 1.5: View and Viewpoint Group

The following specify the requirements associated with View and Viewpoint.

CRC 1.5.1: View Definition

Proposals for SysML v2 shall include a capability to define a class of artifacts that can be presented to a stakeholder.

Supporting Information: The View Definition for a document can be thought of as its table of contents along with the list of figures and tables. The View Definition can be specialized, and decomposed into sub-views that can be ordered.

An individual View is intended to be a specific artifact, such as a document, diagram, or table that is presented to a stakeholder. The individual View conforms to a View Definition that defines construction methods to create an individual View. The execution of the construction methods involves querying a particular model (or more generally one or more data sources) to select the kinds of model elements, and then presenting the information in a specified format.

SysML v1.X Constructs: View

CRC 1.5.2: Viewpoint

Proposals for SysML v2 shall include a capability to represent a Viewpoint that frames a set of stakeholders and their concerns. It specifies the requirements a View must satisfy.

Supporting Information:

The stakeholder and their concerns should be represented in the model.

The concern represents aspects of the domain of interest that the stakeholder has an interest in.

The intent is to align the view and viewpoint concepts with the update to ISO 42010.

SysML v1.X Constructs: Viewpoint

CRC 1.6: Metadata Group

The requirements in this group identify metadata is a kind of model element that can apply to other model elements or to other elements external to the model that refer to a model element (e.g., a model configuration item). Also, refer to the requirement for Analysis Metadata in the Analysis requirements section.

CRC 1.6.1: Version

Proposals for SysML v2 shall include a capability to represent the version of one or more model elements, or of an element external to the model that refers to one or more model elements.

CRC 1.6.2: Time Stamp

Proposals for SysML v2 shall include a capability to represent a model management time stamp for one or more elements, or of another element that refers to one or more model elements.

CRC 1.6.3: Data Protection Controls

Proposals for SysML v2 shall include a capability to represent Data Protection Controls for one or more model elements, or of another element that refers to one or more elements.

Supporting Information: This can include markings such as ITAR, proprietary or security classifications

6.5.2.2 Properties, Values & Expressions Requirements

PRP 1: Properties, Values and Expressions Requirements Group

The requirements in this group provide a unified representation of the type of properties, variables, constants, operation parameters and return types as well as literal values and value expressions. This includes types to represent variable size collections, compound value types, and measurement units and scales.

PRP 1.01: Unified Representation of Values

Proposals for SysML v2 shall include a capability to represent any value-based characteristic in a unified way, called a value property, which shall include representation of a constant, a variable in an expression or a constraint, state variable, as well as any formal parameter and the return type of an operation.

Supporting Information:

A classification of "invariant" can be attached to a value property to assert that is does not vary over time. A constant is an invariant value property of some higher-level context (ultimately the "universe" in case of fundamental physics constants).

Provisions should be made to distinguish between a fundamental physical or mathematical constant (i.e., Pi) from a constant value within the context of a particular model or model execution (i.e., amplifier gain).

SysML v1.X Constructs: Value Property, Formal Parameter of an Operation, Default Value, Static Value, Initial Value

PRP 1.02: Value Type

Proposals for SysML v2 shall include a capability to represent a Value Type as a named definition of the essential semantics and structure of the set of allowable values of a value-based characteristic.

SysML v1.X Constructs: Value Type

PRP 1.03: Value Expression

Proposals for SysML v2 shall include a capability to represent a value as a literal or through a reusable Value Expression that is stated in an expression language. A Value Expression shall include the capability to represent opaque expressions.

SysML v1.X Constructs: Opaque and OCL expressions, Value Specification

PRP 1.05: Unification of Expression and Constraint Definition

Proposals for SysML v2 shall include a capability to represent a reusable constraint definition in the form of an equality or inequality of value expressions which can be evaluated to true or false.

SysML v1.X Constructs: Constraint Block

PRP 1.06: System of Quantities

Proposals for SysML v2 shall include a capability to represent a named system of quantities that support definition of numerical Value Types in accordance with the ISO/IEC 80000 standard.

Supporting Information: The typical Systems of Quantities is the ISO/IEC 80000 International System of Quantities (ISQ) with seven base quantities: length, mass, time, electric current, thermodynamic temperature, amount of substance and luminous intensity.

SysML v1.X Constructs: SystemOfQuantities in Annex E.5 QUDV

PRP 1.07: System of Units and Scales

Proposals for SysML v2 shall include a capability to represent a named system of measurement units and scales to define the precise semantics of numerical Value Types in accordance with the [ISO/IEC 80000] standard.

Supporting Information: Similar to SysML v1 QUDV, SysML v2 should include model libraries representing the [ISO/IEC 80000] units, as well as the conversion to US Customary Units defined in [NIST SP 811] Appendix B.

SysML v1.X Constructs: SystemOfUnits in Annex E.5 QUDV

PRP 1.08: Range Restriction for Numerical Values

Proposals for SysML v2 shall include a capability to represent a value range restriction for any numerical Value Type.

Supporting Information: This requirement allows further restriction of the range of values beyond what is specified by its type. A simple example is a

planar angle typed by a real number Value Type and a degree measurement scale. However, the value range may be further restricted from 0 to 360 degrees for positioning a rotational knob. This can also include the definition of optional lower and upper bounds on an associated measurement scale.

PRP 1.10: Primitive Data Types

Proposals for SysML v2 shall include a capability to represent the following primitive data types as a minimum: signed and unsigned integer, signed and unsigned real, string, Boolean, enumeration type, ISO 8601 date and time, and complex.

Supporting Information: These are intended to be represented in a Value Type Library as they are in SysML v1.

SysML v1.X Constructs: Primitive ValueType Library

PRP 1.11: Variable Length Collection Value Types

Proposals for SysML v2 shall include a capability to represent variable length value collections where each member of the collection is typed by a particular Value Type and is referable by index, and where the collection may be one of the established collection types: sequence (ordered, non-unique), set (unordered, unique), ordered set (ordered, unique) or bag (unordered, non-unique).

PRP 1.12: Compound Value Type

Proposals for SysML v2 shall include a capability to represent both scalar and compound Value Types, where a scalar Value Type represents elements with a single value, and compound Value Type represents elements with a fixed number of component values, where each component value is typed in turn by a scalar Value Type or another compound Value Type.

Supporting Information: Such compound Value Types are needed to support the representation of vector, matrix, higher order tensor, complex number, quaternion, and other richer Value Types.

SysML v1.X Constructs: ValueType

PRP 1.15: Probabilistic Value Distributions

Proposals for SysML v2 shall include a capability to represent the value of a quantity with a probabilistic value distribution, including an extensible mechanism to detail the kind of distribution, i.e. the probability density function for continuous random variables, or the probability mass function for discrete random variables.

SysML v1.X Constructs: Annex E.7 Distribution Extensions

PRP 1.19: Materials with Properties

Proposals for SysML v2 shall include a capability to represent named materials with their material properties in a model library and assignment of such materials to physical elements such as hardware components.

Supporting information: This requirement is intended to specify a model library with a generic material kind that has generic material properties that can be further specialized. Examples of generic material properties include density, hardness, and tensile yield strength.

6.5.2.3 Structure Requirements

STC 1: Structure Requirements Group

This group of requirements is intended to represent composable, deeply nested, connectible structure that supports definition of a family of configurations, specific configurations, and individual elements that are uniquely identified.

Supporting Information:

These requirements refer to definition elements and usage elements analogous to structured classifiers and classifier features in UML. A particular specialization of these concepts in SysML v1 is used to represent blocks and parts,

The requirements also refer to configuration elements and individual elements. Configuration elements are used to unambiguously represent deeply nested structures as a tree of configuration elements. Individual elements are used to represent a particular element that can be uniquely identified, which is not to be interpreted as a UML or SysML instance. A particular system, such as a system with a serial number on the manufacturing floor, can be represented by an individual element which in turn can be represented as a tree of individual elements.

The terms Component Definition and Component Usage refer to a particular kind of Definition Element and Usage Element that are analogous to a Block and Part in SysML v1. The terms Item Definition and Item Usage are also used to refer to a particular kind of Definition Element and Usage Element that correspond to something that flows through a system, such as Water. Component and Item are introduced in the Interface requirements section.

STC 1.01: Modular Unit of Structure

Proposals for SysML v2 shall include a capability to represent a modular unit of structure that defines its characteristics through value properties, interface ends (ports), constraints, and other structural and behavioral features.

Supporting Information: The term used in this RFP to refer to a modular unit of structure is Definition Element. Such modular units of structure can be regarded as the fundamental named building blocks from which system representations, i.e. architectures, can be constructed. The capability enables modeling multiple levels of a hierarchy (e.g., system-of systems, system,

subsystem and components) that can include logical and physical representations of hardware, software, information, people, facilities, and natural objects.

The concept model refers many specializations of Definition Element. One example is the Component Definition which is intended to represent any level of a product structure. The concept model refers to an Item Definition as a specialized Definition Element to represent an element that flows through a system, such as water or a message. As noted above, the decomposition of Definition Elements may include variability that may be represented by multiplicity, subclasses, and/or a range of property values, which is removed when selecting a specific design configuration.

SysML v1.X Constructs: Block

STC 1.02: Usage Element

Proposals for SysML v2 shall include a capability to represent the usage of a Definition Element, called a Usage Element, in order to support reuse in different contexts.

SysML v1.X Constructs: Structural Feature, Behavioral Feature, ElementPropertyPath, NestedConnectorEnd

STC 1.03: Generic Hierarchical Structure

Proposals for SysML v2 shall include a capability to represent hierarchical composition structure of Definition Elements.

SysML v1.X Constructs: Composite Association, Part Property

STC 1.04: Reference Element

Proposals for SysML v2 shall include a capability to represent a reference from one element to any other element within a shared scope.

SysML v1.X Constructs: Reference Property, Reference Association

STC 1.05: Multiplicity of Usage

Proposals for SysML v2 shall include a capability to define the multiplicity of any particular Usage Element or Reference Element as an integer range (i.e., lower bound and upper bound).

Supporting Information:

Multiplicity refers to the number of Individual Elements.

SysML v1.X Constructs: Multiplicity on properties.

STC 1.06: Definition Element Specialization

Proposals for SysML v2 shall include a capability to represent a specialization from a more general Definition Element into a more specific Definition Element,

where the more specific element inherits all features of the more general element.

SysML v1.X Constructs: Generalization/Specialization

STC 1.07: Unambiguous Deeply Nested Structure

Proposals for SysML v2 shall support a capability to unambiguously represent Usage Elements at any level of nesting.

SysML v1.X Constructs: ElementPropertyPath, NestedConnectorEnd, Redefinition, Subsetting

STC 1.08: Structure With Variability

Proposals for SysML v2 shall include a capability to represent multiple possible variant configurations of a system-of-interest with a single collection of Definition Elements and Usage Elements, where at each usage level in the (de)composition, a variant from different possible variant choices can be selected.

Supporting Information: A Structure With Variability enables the definition of a product line architecture, see e.g. ISO 26550. Some common variant choices are defined by multiplicity range. sub-classes, and different values of a value property.

SysML v1.X Constructs: Multiplicity of property, specialization of classifiers Redefined property, Subsetted property

STC 1.10: Structure of an Individual

Proposals for SysML v2 shall include a capability to represent a (de)composition of an Individual Element that is uniquely identifiable, and that can conform to an associated Structure resolved to a Single Variant and/or a Structure with Variability.

Supporting Information: Such a digital representation of a real-world system is sometimes called a 'digital twin'. The elements in a Structure of an Individual are typically designated by a unique serial number, a batch number or an effectivity code.

SysML v1.X Constructs: Instance Specification

STC 1.11: Usage Specific Localized Type

Proposals for SysML v2 shall include a capability to represent local override, redefinition, or addition of features with respect to the features defined by its more general type at any level of nesting.

Supporting Information: The more-general to more-specific type chain is: Definition Element - direct Usage Feature - deeply nested Usage Feature - Configuration Element - Individual Element.

The localized usage should support capabilities equivalent to redefinition and sub-setting for usage elements at any level of nesting.

SysML v1.X Constructs: PropertySpecificType Redefinition, Subsetting

6.5.2.4 Interface Requirements

INF 1: Interface Requirements Group

SysML v2 is intended to provide a robust capability to model interfaces that constrain the physical and functional interaction between structural elements. An interface in SysML v2 includes two (2) interface ends, the connection between them, and any constraints on the interaction.

Supporting Information:

An interface should support the following:

1. Different levels of abstraction that include logical, functional, and physical interfaces, nested interfaces, and interface layers;
2. Diverse domains that include a combination of electrical, mechanical, software, and user interfaces;
3. Reuse of interfaces in different contexts;
4. Generation of interface control documents and interface specifications

A Port is also used to refer to an Interface End.

INF 1.01: Interface Definition and Reuse

Proposals for SysML v2 shall provide the capability to define an interface that can be used in different contexts that includes the definition of the interface ends, the interface connections, and the constraints on the interaction.

Supporting Information:

Interfaces must conform to the structural concepts of definition and usage. The constraints can constraint properties, such as conservation laws that can apply to a physical interface, and/or constraints on exchanged items such as protocol constraints that can apply to message exchange, and/or geometric constraints that can apply to a physical interface such as between a plug and socket.

SysML v1.X Constructs: Port Definitions including Interface Blocks and Blocks, Association and Association Blocks used to type Connectors, Item Flows, Constraints

INF 1.02: Interface Usage

Proposals for SysML v2 shall provide the capability to represent a usage of an interface that constrains the interaction between any two (2) structural elements.

SysML v1.X Constructs: Ports, Connectors, Parts

INF 1.03: Interface Decomposition

Proposals for SysML v2 shall provide the capability to represent nested interfaces, such as when modeling two electrical connectors with pin to pin connections.

SysML v1.X Constructs: Nested ports

INF 1.04: Interface End Definitions

Proposals for SysML v2 shall provide the capability to represent the definition of an Interface End whose features constrain the interaction of the end, including items that can be exchanged and their direction, behavioral features, and constraints on properties.

Supporting Information:

Interface End Definitions are also referred to as Port Definitions and Interface End Usages are referred to as Port Usages or Ports for short.

SysML v1.X Constructs: Interface Blocks with flow properties, value properties, and behavioral features

INF 1.05: Conjugate Interface Ends

Proposals for SysML v2 shall provide the capability to reverse the direction of the items that are exchanged in an Interface End.

SysML v1.X Constructs: Conjugate Ports

INF 1.06: Item Definition

Proposals for SysML v2 shall provide the capability to represent the kind of items that can be exchanged between Interface Ends.

Supporting Information: The items represent the type of things that are exchanged, such as water or electrical signals. The items may have physical characteristics such as mass, energy, charge, and force, and logical characteristics such as information that is encoded in the physical exchange. In addition to being exchanged, these items may also be stored.

An item that is input to a component should become a stored item usage that can be transformed by function usages. An item, such as an engine that is an input and output of an assembly process, may also have the role as a component, when it is assembled into a vehicle. Item Definitions must conform to the structural concepts of definition and usage. The rate at which a usage of an Item Definition is updated may be marked with an update rate that is continuous or discrete valued. (Refer to Behavior Requirement called "Discrete and Continuous Time Behavior")

SysML v1.X Constructs: Blocks, Signal

INF 1.07: Interface Agreement Group

INF 1.07.1: Item Exchange Constraints

Proposals for SysML v2 shall provide the capability to constrain the interaction between the interface ends that includes constraints on the items to be exchanged, the allowable sequences and directions of those items, timing of the exchange and other characteristics. The items exchanged shall be consistent with the type and direction of the items specified in the connected Interface Ends.

SysML v1.X Constructs: Activities, state machines and sequence diagrams in an association block with participant properties

INF 1.07.2: Property Constraints

Proposals for SysML v2 shall provide the capability to constrain the interaction between the interface ends that include mathematical constraints on the properties exposed by the Interface Ends.

Supporting Information: The value properties may further be identified as Across or Through variables consistent with standard usage of the terms (e.g. specify properties that are constrained by conservation laws).

SysML v1.X Constructs: Parametric diagrams that specify constraints on the ends of an association block with participant properties.

INF 1.08: Interface Medium

Proposals for SysML v2 shall include a capability to represent an Interface Medium that enable 2 or more components to interact.

Supporting Information: The Interface Medium may represent either an abstract or physical element that connects elements to enable interactions. Examples of an interface medium included an electrical harness, a communications network, a fluid pipe, the atmosphere, or even empty space. The interface medium may connect one to many components, which include support for peer-to-peer, multi-cast, and broadcast communications.

Consider replacing the term Interface Medium with Transport Medium.

SysML v1.X Constructs: A block with a user-defined stereotype indicating its special function.

Also, a connector typed by an Association Block that has part properties, e.g. hotwater:Pipe and coldwater:Pipe.

INF 1.09: Interface Layers

Proposals for SysML v2 shall provide the capability to represent interfaces between layers of an interface stack.

Supporting Information:

A layer of a stack can be represented as a component. A layer in a stack transforms the data to match the input to the adjacent layer. For example, an application layer may correspond to a component that transforms packets to

match the TCP layer, and the TCP layer may correspond to a component that transforms the data to match the IP layer.

SysML v1.X Constructs: Complex combination of the ports and flow concepts.

INF 1.10: Allocating Functional Exchange to Interfaces

Proposals for SysML v2 shall provide the capability to allocate or bind the outputs and inputs of a function to interface ends (or nested interface ends)',

Supporting Information:

It is expected that there are validation rules to ensure consistency between the inputs and outputs of a function and the interface ends.

This allocate or binding should be inherited by the Component subclasses.

SysML v1.X Constructs: On Port metaproperty.

6.5.2.5 Behavior Requirements

BHV 1: Behavior Requirements Group

BHV 1.01: Behavior

Proposals for SysML v2 shall include the capability to model a Behavior that represents the interaction between individual structural elements and their change of state over time.

SysML v1.X Constructs: Activity, State Machine, Interaction, Simple Time

BHV 1.02: Behavior Decomposition

Proposals for SysML v2 shall include the capability to decompose a behavior to any level of decomposition, and to define localized usages of behavior at nested levels of decomposition.

Supporting Information:

The decomposition of behavior should conform to a similar pattern as the decomposition of structure, and include capabilities for specialization, redefinition, and sub-setting.

The decomposition should also include the equivalent capability to decompose a SysML v1 activity on a BDD, and the ability to decompose actions using a structured activity node.

SysML v1.X Constructs: Composited Association of Behavior Classifiers with Adjunct Properties

BHV 1.03: Function-based Behavior Group

BHV 1.03.1: Function-based Behavior

Proposals for SysML v2 shall include the capability to represent a controlled sequence of actions (or functions) that can transform a set of input items to a set of output items.

Supporting Information:

SysML v2 should provide an integrated approach to specify behavior that reflects similar capabilities to SysML v1 activities and sequence diagrams, which are expected to be different views of the same underlying model.

The input items and output items correspond to item usages and their associated value properties whose values can vary over time. Item flows connect an output item usage to an input item usage.

The start and stop events should be represented explicitly (e.g., control pins). Event flows connect a stop event to a start event.

The specific features of activities and sequence diagrams to be included in SysML v2 beyond what is specified in this section should be defined in the proposal.

SysML v1.X Constructs: Activity, Interaction, Object Flow, Control Flow

lBHV 1.03.3: Function-based Behavior Constraints

Proposals for SysML v2 shall include the capability to model constraints on a function-based behavior that includes the ability to represent a declarative specification in terms of its pre-conditions and post-conditions, and any constraints that apply throughout execution of the behavior.

SysML v1.X Constructs: Pre and post conditions

BHV 1.03.4: Opaque Behavior

Proposals for SysML v2 shall include the capability to represent a behavior that embeds the definition in a language such as a programming language.

SysML v1.X Constructs: Opaque Behavior

BHV 1.03.6: Structure Modification Behavior

Proposals for SysML v2 shall include the capability to represent behaviors that can modify the structure of an element over time, such as the creation and destruction of interconnections and composition.

Supporting Information:

An example is the behavior associated with the separation of a first stage rocket, or the assembly or disassembly of a product.

SysML v1.X Constructs: Primitive Actions

BHV 1.04: State-based Behavior Group

BHV 1.04.1: Regions, States, and Transitions

Proposals for SysML v2 shall include the capability to represent the state behavior of a structural element in terms of its concurrent regions with mutually exclusive finite states, and transitions between finite states.

Supporting Information: A state change can result from a change in structure.

SysML v1.X Constructs: State Machine

BHV 1.04.2: Integration of Function-based Behavior with Finite State Behavior

Proposals for SysML v2 shall include the capability to model function-based behavior both on transitions between finite states, and upon entry, exit, and while in a finite state.

SysML v1.X Constructs: Entry, Exit, Do Behavior and Transition effect

BHV 1.04.3: Integration of Constraints with Finite State Behavior

Proposals for SysML v2 shall include the capability to model constraints both on transitions between finite states, and upon entry, exit, and while in a finite state.

SysML v1.X Constructs: State invariants

BHV 1.05: Discrete and Continuous Time Behavior

Proposals for SysML v2 shall include the capability to model behaviors whose inputs and outputs vary continuously as a function of time, or discretely as a function of time.

SysML v1.X Constructs: Continuous, streaming

BHV 1.06: Events

Proposals for SysML v2 shall include the capability to model signal events, time events, and change events and their ordering.

Supporting Information: The ordering of actions (i.e., functions) is accomplished through ordering of their start and completion events. Events can trigger a change from one finite-state to another. Events should be able to be explicitly represented in both function-based behavior and finite-state behavior. Events can be defined and used in different contexts.

SysML v1.X Constructs: Triggering events on state machines, accept event actions, send signal actions

BHV 1.07: Control Nodes

Proposals for SysML v2 shall include the capability to model control nodes that specify a logical expression of conditions and events to enable a flow.

Supporting Information: For Example: {Inputs A < a1 AND B>=b2 OR C AND NOT D} must be true).

SysML v1.X Constructs: Join, Fork, Merge, Decision, Join Specification

BHV 1.08: Time Constraints

Proposals for SysML v2 shall include the capability to specify the absolute or relative time associated with an event that includes start events, stop events, and duration constraints between events to represent the time-line associated with a behavior.

Supporting Information: Time is a property typed by a Value Type whose quantity kind and units are specified as part of QUDV.

SysML v1.X Constructs: Simple Time

BHV 1.10: Behavior Execution

Proposals for SysML v2 shall include the capability to execute function-based and state-based behavior to specify the state history of individual elements and their interactions with other individual elements.

Supporting Information: The behavior of a Definition Element or Configuration Element represent the default behavior of the conforming Individual Elements.

SysML v1.X Constructs: fUML

BHV 1.11: Integration between Structure and Behavior

BHV 1.11.1: Allocation of Behavior to Structure

Proposals for SysML v2 shall include the capability to represent the behavior of one or more structural elements.

Supporting Information:

This should support the ability to define a state machine of a structural element, with finite states that enable actions (i.e., functions) and constraints. In addition, this should support the ability to specify the functions performed by a component, and the applicable constraints, without specifying the finite state that enables them. The representation should allow more than one structural element to perform a single function, such as when two people carry a load. This is analogous to a reference interaction in a SysML v1 sequence diagram that spans multiple lifelines and displays the participating lifelines. The reference interaction refers to another sequence diagram.

SysML v1.X Constructs: Allocate, Allocated Activity Partition, Structured Activity Node, Reference Interaction

BHV 1.11.2: Integration of Control Flow and Input/Output Flow

Proposals for SysML v2 shall ensure that inputs, outputs, and events can be represented consistently across behavior and structure.

Supporting Information:

In SysML v1, it is often difficult to ensure consistent representation of control flow and input/output flow. Examples include potential inconsistencies between:

- Flows on activity diagrams and messages on sequence diagrams.
- Flows on activity diagrams and item flows on ibd
- Inputs and outputs on activity diagram and corresponding inputs and outputs on activity decomposition on a bdd
- Inability to represent input/output of activities on do behaviors of state machines

SysML v1.X Constructs: Adjunct properties,

On Port

BHV 1.12: Case

Proposals for SysML v2 shall include the capability to represent a case that can be specialized into a use case, verification case, analysis case, and domain specific cases, such as safety case and assurance case.

Supporting Information: A case is a series of steps with an associated objective that produce a result or conclusion. An analysis case and assurance case correspond to a set of steps to implement a study or investigation. Refer to the Structured Assurance Case Metamodel (SACM).

6.5.2.6 Requirements for Requirements

RQT 1: Requirement Group

The requirements in this group are used to represent requirements and their relationships.

RQT 1.1: Requirement Definition Group

RQT 1.1.1: Requirement Definition Name

Proposals for SysML v2 shall include a capability to represent a requirement definition that can be used to constrain a solution.

SysML v1.X Constructs: Requirement Name

RQT 1.1.2: Requirement Identifier

Proposals for SysML v2 shall include a capability to represent an identifier for each requirement that is adaptable to a user defined numbering scheme, and can be set to not change.

SysML v1.X Constructs: Requirement ID

RQT 1.1.3: Requirement Attributes

Proposals for SysML v2 shall include a capability to represent the following optional requirement attributes for a requirement definition.

- Requirement Status
- Priority
- Risk
- Originator/Author
- Owner
- User-defined Attributes (e.g., confidence level, uncertainty status, etc.)

Supporting Information: These attributes are derived from commonly used attributes as defined in the INCOSE Handbook and ReqIF, and should be reconciled with other model element metadata and model element attributes that apply more generally.

SysML v1.X Constructs: Non Normative extensions

RQT 1.1.4: Textual Requirement Statement

Proposals for SysML v2 shall include a capability to represent a requirement definition that contains an optional textual requirement statement.

SysML v1.X Constructs: Requirement text statement

RQT 1.1.5: Restricted Requirement Statement Group

Supporting Information: Refer to Restricted Use Case Modeling (RUCM) [36] as an example of a restricted requirement statement.

RQT 1.1.5.1: Restricted Requirement Statement

Proposals for SysML v2 shall include a capability to represent a requirement definition that contains an optional restricted requirement statement which may include predefined key words and sentence structures.

RQT 1.1.5.2: Restricted Requirement Statement Extensibility

Proposals for SysML v2 shall include a capability to extend a restricted requirement statement with additional key words and sentence structures. RQT 1.1.5.3: Restricted Requirement Statement Transformation

SysML v2 will include a capability to maintain traceability between the restricted requirement statement and the textual requirement statement and/or the formal requirement statement.

RQT 1.1.6: Formal Requirement Statement Group

RQT 1.1.6.1: Formal Requirement Statement

Proposals for SysML v2 shall include a capability to represent a requirement definition that contains an optional formal requirement statement that includes one or more constraints that an acceptable solution must satisfy.

Supporting Information: It is desired to also enable the element that is intended to satisfy the requirement to contain the formal requirement statement. This can provide a more lightweight modeling style.

SysML v1.X Constructs: Non-normative extension for a property based requirement

RQT 1.1.6.2: Assumptions

Proposals for SysML v2 shall include a capability to represent a formal requirement statement that includes one or more expressions to specify the assumptions and conditions for acceptable solutions (e.g., the weight of a car includes the fuel weight)

Supporting Information: This should be consistent with the concept of Assumption that is applied in other parts of the model.

RQT 1.2: Groups of Requirements

RQT 1.2.1: Requirement Group

Proposals for SysML v2 shall provide the capability to model a group of requirements that are used to constrain a solution.

Supporting Information: This is intended to be a sub-class of Element Group.

SysML v1.X Constructs: Requirement

RQT 1.2.2: Requirement Usage (localized)

Proposals for SysML v2 shall include a capability to represent localized values of a requirement usage that can over-ride the values of its requirement definition.

Supporting Information: The structural concepts of definition, usage, configuration, and individuals are intended to support reuse of requirement definitions, and unambiguously define a tree of requirements that specify a design configuration or an individual element.

RQT 1.2.3: Requirement Usage Identifier

Proposals for SysML v2 shall include a capability to represent each requirement in a requirement group with an identifier that is adaptable to a user defined numbering scheme, and that the user can specify whether the identifier can change or not.

SysML v1.X Constructs: Requirement ID

RQT 1.2.4: Requirement Usage Ordering

Proposals for SysML v2 shall include a capability to represent the order of each requirement in a requirement group that is not constrained by its requirement identifier.

Supporting Information: This primarily allows the user to further organize the requirements, but it does not impact the meaning of the requirements. For example, there may be a requirement group with one requirement to open a valve and another requirement to close a valve. The user may want to order the open requirement as the first requirement in the group.

RQT 1.3: Requirement Relationships Group

RQT 1.3.1: Requirement Specialization

Proposals for SysML v2 shall include a capability to represent a generalization relationship that relates a specialized requirement definition to a more general requirement definition.

RQT 1.3.2: Requirement Satisfaction

Proposals for SysML v2 shall include a capability to represent a satisfy relationship that relates a requirement to a model element that is asserted to satisfy it.

Supporting Information: This is intended to be a specialization of the Conform Relationship.

SysML v1.X Constructs: Satisfy

RQT 1.3.3: Requirement Verification

Proposals for SysML v2 shall include a capability to represent a verify relationship that relates a verification case to the requirement it is intended to verify.

SysML v1.X Constructs: Verify

RQT 1.3.4: Requirement Derivation

Proposals for SysML v2 shall include a capability to represent a derive relationship that relates a derived requirement to a source requirement.

SysML v1.X Constructs: DerivedRequirement

RQT 1.3.5: Requirement Group Relationship

Proposals for SysML v2 shall include a capability to represent a relationship between a requirement group and the members of the group that can include either a requirement or another requirement group.

Supporting Information: This relationship groups requirements into a shared context.

RQT 1.3.6: Relationships to a Requirement Group

Proposals for SysML v2 shall specify the meaning of relationships with a requirement group on each member of the requirement group.

Supporting Information: This applies more generally to element groups.

RQT 1.4: Requirement Supporting Information

Proposals for SysML v2 shall include a capability to represent supporting information for a requirement, requirement definition, and a requirement group.

Supporting Information: This is a kind of annotation that applies more generally to any model element.

RQT 1.5: Goals, Objectives, and Evaluation Criteria

Proposals for SysML v2 shall include a capability to represent goals, objectives, and evaluation criteria.

Supporting Information:

Criteria can be viewed as a superclass of a requirement that is used as a basis for evaluation, but does not specify specific values. For example, a cost requirement may be to require the cost to be less than a particular value, where-as a cost criterion may be to select a design with the lowest cost. Goals can be a type of criteria. For example, a goal of the system is to minimize the cost. An objective represents a desired end state. For example, the mission objective is to land a person on the moon and safely return them to earth. An objective can be thought of as a kind of requirement.

Refer to Business Motivation Metamodel (BMM).

6.5.2.7 Verification Requirements

VRF 1: Verification and Validation Requirements Group

The requirements in this group represent how to evaluate whether systems satisfy their requirements using verification methods.

Supporting Information: The requirements for validation are not called out explicitly, but are intended to be supported in a similar way as the requirements for verification.

VRF 1.1: Verification Context

Proposals for SysML v2 shall include the capability to model a Verification Context that includes the unit-under-verification, the verification case, and the verification system and associated environment that performs the verification.

VRF 1.2: Verification Case Group

VRF 1.2.1: Verification Case

Proposals for SysML v2 shall include the capability to model a verification case to evaluate whether one or more requirements are satisfied by a unit under verification.

Supporting Information: This is intended to be a specialization of Case.

SysML v1.X Constructs: Test Case

VRF 1.2.2: Verification Objectives

The verification case shall include verification objectives to be implemented by the verification activities.

VRF 1.2.3: Verification Success Criteria

The verification case shall include the criteria used to evaluate whether the verification objectives are met and the requirements are satisfied.

VRF 1.2.4: Verification Methods

The verification case shall include the methods used to verify the requirements. The methods, including inspection, analysis, demonstration, test, external verification, engineering reviews, and similarity, shall be included in a library. More than one method can be applied to verify a requirement.

Supporting information:

A verification method may include additional classification such as qualification test and acceptance test.

An external verification is a method used in some industries, such as an Underwriters Labs.

VRF 1.3: Verification System

Proposals for SysML v2 shall include the capability to model the system and associated environment that is used to verify the unit under verification. (Note: the verification system may include verification elements that are combinations of operational and simulated hardware, software, people, and facilities.)

VRF 1.4: Verification Relationships Group

VRF 1.4.1: Verification Objectives to Verification Cases

Proposals for SysML v2 shall include the capability to model relationship between the verification cases and their verification objectives.

VRF 1.4.2: Validate Relationship

Proposals for SysML v2 shall include the capability to model the relationship between the validation case and the model element being validated.

Supporting Information: An element being validated may represent a requirement, design, as-built system, model, etc.

The Verify Relationship is included in the requirements section.

6.5.2.8 Analysis Requirements

ANL 1: Analysis Requirements Group

The requirements in this group are used to specify an analysis, along with other requirements such as Properties, Values, and Expressions.

ANL 1.01: Analysis

Proposals for SysML v2 shall include the capability to specify an Analysis, including the subject of analysis (e.g., system), the analysis case, and the analysis models and related infrastructure to perform the analysis.

ANL 1.02: Subject of the Analysis

Proposals for SysML v2 shall include the capability to model the relationship between the analysis and the subject of the analysis (system being analyzed).

ANL 1.03: Parameters of Interest

Proposals for SysML v2 shall include the capability to identify the key parameters of interest including measures-of-effectiveness (MoE) and other key measures of performance (MoP).

SysML v1.X Constructs: Value Property, MoE

ANL 1.04: Analysis Case

Proposals for SysML v2 shall include the capability to model the analysis case to specify the analysis scenarios and associated analysis methods needed to produce an analysis result that achieves the analysis objectives.

Supporting Information: This is intended to be a specialization of Case.

ANL 1.05: Analysis Objectives

Proposals for SysML v2 shall include the capability to model the objective of the analysis being performed in text or as a mathematical formalism, e.g. math expression, so that it can be evaluated.

ANL 1.06: Analysis Scenarios

Proposals for SysML v2 shall include the capability to model the scenarios that identify the analysis models to be executed, the conditions and assumptions, and the configurations of the subject of the analysis and the related infrastructure to perform the analysis.

ANL 1.07: Analysis Assumption

Proposals for SysML v2 shall include the capability to model the assumptions of the analyses in a text or mathematical form, e.g. constraints and boundary conditions.

ANL 1.08: Analysis Decomposition

Proposals for SysML v2 shall include the capability to decompose an analysis into constituent analyses.

ANL 1.09: Analysis Model

Proposals for SysML v2 shall include the capability to specify an analysis model.

Supporting Information: Analysis models can be defined natively in SysML (e.g. parametric model or behavior model) or externally (e.g. equation-based math models, finite element analysis models, or computational fluid dynamics models). The level of fidelity of the specification of the analysis model can vary from an abstract specification that defines the intent of the analysis including its input and output parameters, to a detailed specification that a particular solver can execute.

ANL 1.11: Analysis Result

Proposals for SysML v2 shall include the capability to relate the results of executing analysis models to the analysis.

Supporting Information: The results may be stored in the SysML v2 model itself or an in an external store (e.g. CSV file or database). The results can be used to evaluate how well the analysis objectives are satisfied, and to obtain the supporting rationale for decisions taken based on the analysis.

ANL 1.13: Analysis Metadata

Proposals for SysML v2 shall include the capability to represent the metadata relevant to specifying the analysis.

ANL 1.14: Decision Group

The requirements in this group support trade-off analysis among alternatives. This typically involves making decisions during the design process to evaluate alternative designs based on a set of criteria, and selecting a preferred design.

4.2: Alternative

Proposals for SysML v2 shall include a capability to represent a set of alternatives.

ANL 1.14.4: Decision

Proposals for SysML v2 shall include a capability to represent a decision as one or more selections among alternatives.

Supporting Information: This Decision and Rationale can be related through an Explanation relationship. The Rationale can refer to the supporting analysis.

ANL 1.14.5: Criteria

Proposals for SysML v2 shall include a capability to represent criteria that is used as a basis for a decision or evaluation.

ANL 1.14.6: Rationale

Proposals for SysML v2 shall include a capability to represent rationale for a decision or other conclusion.

SysML v1.X Constructs: Rationale

6.5.3 Example Model Requirements

RML 1: Example Model and Model Libraries Group

RML 1.1: Example Model

Proposals for SysML v2 shall include an example model that demonstrates the application of the SysML v2 language concepts to a commonly understood domain.

6.5.4 Conformance Requirements

CNF 1: Conformance Requirements Group

These requirements specify that the proposals provide a suite of test cases that a conformant SysML v2 implementation must satisfy. The test cases can more generally be verification cases.

The SysML v2 specification will specify the conformance levels for each conformance area below. Vendors are expected to identify specific levels of conformance within each of the sub-section of groupings in this document so that a cross functional compliance matrix can be developed for each tool implementation. This enables the ecosystem of potential SysML tool vendors who only wish to partially implement the SysML specification to expand, (i.e. only the requirements or test aspects for example).

CNF 1.1: Metamodel Conformance Group

Proposals for SysML v2 shall provide test cases to assess conformance of a SysML v2 implementation with the SysML v2 metamodel specification (abstract syntax, concrete syntax, and semantics).

CNF 1.2: Profile Conformance

Proposals for SysML v2 shall provide test cases to assess conformance of a SysML v2 implementation with the SysML v2 profile specification.

CNF 1.3: Model Interoperability Conformance

Proposals for SysML v2 shall provide test cases to assess conformance of a SysML v2 implementation with the SysML v2 model interoperability specification.

CNF 1.4: Traceability Matrix

Proposals for SysML v2 shall include a traceability matrix (include reference) that demonstrates how each language feature is verified by the conformance test suite.

6.6 Non-mandatory features

6.6.1 Non-mandatory Language Architecture and Formalism Features

LNG 1: Language Architecture and Formalism Requirements Group

LNG 1.2: Semantics Group

LNG 1.2.1: Semantic Model Libraries

Proposals for SysML v2 semantics shall be modeled with SysML v2 model libraries.

Supporting Information:

1. Simplifies the language when model libraries are used to extend the base declarative semantics without additional abstract syntax.

2. Enables SysML to be improved and extended more easily by changes and additions to model libraries, rather than always through abstract syntax.

LNG 1.2.2: Declarative Semantics

Proposals for SysML v2 models may be grounded in a declarative semantics expressed using mathematical logic.

Supporting Information:

Semantics are defined formally to reduce ambiguity. Declarative semantics enable reasoning with mathematical proofs. This contrasts with operational semantics that requires execution in order to determine correctness.

The semantics provide the meaning to the concepts defined in the language, and enable the ability to reason about the entity being represented by the models.

SysML v1.X Constructs: Semantics of UML and SysML

LNG 1.2.3: Reasoning Capability

Proposals for SysML v2 may provide a subset of its semantics that is complete and decidable.

Supporting Information: This enables the ability to reason about the entity being modeled by querying the model, and returning results that satisfy the specified set of constraints.

As an example, a query could return valid vehicle configurations that have a vehicle mass<2000kg AND vehicles that have a sunroof.

LNG 1.4: Concrete Syntax Group

LNG 1.4.4: Textual Concrete Syntax

Proposals for SysML v2 may provide a standard human readable textual concrete syntax.

Supporting information: Graphical and textual concrete syntax representations can be used in combination to more efficiently and effectively present the model. Refer to Alf as an example of a textual notation.

LNG 1.5: Extensibility Group

LNG 1.5.2: Extensibility Consistency

Proposals for all SysML v2 extension mechanisms may be applicable to SysML v2 syntax (concrete and abstract) and semantics, and be consistent with how these are specified in SysML v2.

Supporting Information:

The SysML v2 Specification includes syntax, semantics, and vocabulary, so extending the language requires all of these to be extensible.

LNG 1.6: Model Interchange, Model Mapping, and Transformations Group

LNG 1.6.1: Model Interchange

Proposals for SysML v2 may provide a format for unambiguously interchanging the abstract syntax representation of a model and the concrete syntax representation of views of the model, which supports exchange of models that are created using either the metamodel or the profile.

Supporting Information: The interchange should facilitate long term retention, file exchange, and version upgrades.

Consider consistency with related interchange standards, such as AP233. For the concrete syntax, consider consistency with Diagram Definition and Diagram Interchange.

SysML v1.X Constructs: XMI

LNG 1.6.2: Model Mappings and Transformations

Proposals for SysML v2 may provide a capability to specify model mappings and transformations.

Supporting Information: SysML may be used to represent the metamodel of other languages and data sources to enable transformation between SysML models, other data sources, and models in other languages. These languages include languages for queries, validation rules, expressions, viewpoint methods, and transformations.

A common need is to map elements between SysML and Excel that supports import of Excel data into a SysML model, and export of SysML model elements to Excel. Another example is a mapping between SysML models and Simulink models.

SysML v1.X Constructs: QVT

6.6.2 Non-mandatory Data Model Features

6.6.2.1 Non-mandatory Cross-cutting Features

CRC 1: Cross-cutting Requirements Group

CRC 1.2: Model Element Group

CRC 1.2.1: Model Element

Proposals for SysML v2 shall include a root element that contains features that apply to all other kinds of elements in the model.

SysML v1.X Constructs: Model Element

CRC 1.3: Model Element Relationships Requirements Group

CRC 1.3.11: Copy Relationship

Proposals for SysML v2 may include a capability to represent a Copy Relationship where one side of the relationship refers to the element (or elements) being copied and the other side of the relationship refers to the copy (or copies).

Supporting Information:

The primary goals for this relationship are to establish provenance to support traceability, and to enable reuse of catalog items. This relationship provides the ability to copy elements such as a Container (e.g., package) and its contents,

within a model and from one model to another. Additional constraints can be defined to specify the rules for what part of the element being copied can be modified in the copy. It is assumed that updates to the copied element are not propagated, unless there is a rule to support this.

CRC 1.4: Variability Modeling Group

6.6.2.2 Non-mandatory Properties, Values & Expressions Features

PRP 1: Properties, Values and Expressions Requirements Group

PRP 1.04: Logical Expressions

Proposals for SysML v2 may include a capability to represent, as part of the Expression language, logical expressions that support as a minimum the standard boolean operators AND, OR, XOR, NOT, and conditional expressions like IF-THEN-ELSE and IF-AND-ONLY-IF, in which symbols bound to any characteristics (e.g. value properties or usage features) may be used.

PRP 1.09: Automated Quantity Value Conversion

Proposals for SysML v2 may include a capability to represent all information necessary to perform automated conversion of the value of a quantity (typed by a numerical Value Type) expressed in one measurement scale to the value expressed in another compatible measurement scale with the same quantity kind.

Supporting Information: This capability is needed to rebase a set of (smaller) system models coming from various contributors on a single coherent set of measurement scales, so that an integrated (larger) system model can be consistently constructed and analyzed.

SysML v1.X Constructs: Most concepts are defined in Annex E.5 QUDV, but measurement scales are lacking detail to fully automate value conversions.

PRP 1.13: Discretely Sampled Function Value Type

Proposals for SysML v2 may include a capability to represent variable length sets of values that constitute discrete time series data, frequency spectra, temperature dependent material properties, and any other datasets that can be represented through a discretely sampled mathematical function.

Supporting Information: Such a discretely sampled function can be defined by a tuple of one or more Value Types that prescribe the type of the domain (independent) variables, and a tuple of one or more Value Types that prescribe the range (dependent) variables, as well as a variable length sequence of tuples that represent the actual set of sampled values.

PRP 1.14: Discretely Sampled Function Interpolation

Proposals for SysML v2 may include a capability to represent an interpolation scheme for a Discretely Sampled Function Value Type for derivation of the function's range values for domain values that are in-between sampled values.

PRP 1.16: System Simulation Models

Proposals for SysML v2 may include a capability to represent signal flow graph models and lumped parameter models as well as combinations thereof.

Supporting Information: See [SysPISF] for details.

This requirement is augmented by the analysis requirements.

PRP 1.17: Across and Through Value Properties

Proposals for SysML v2 shall include a capability to define across and through properties of flows on Interface Ends that participate in representing physical interactions in lumped parameter models.

Supporting Information: Typically, the across and through properties are defined together as a pair, where the across property does not conserve energy and the through property does. For example, in a lumped parameter model of an electric circuit, the across and through properties are voltage and current respectively. See [SysPISF] for details.

PRP 1.18: Basic Geometry

Proposals for SysML v2 may include a capability to represent basic two- and three-dimensional geometry of a structural element, including a base coordinate frame as well as relative orientation and placement of shapes through nested coordinate frame transformations, where the basic shape definitions are provided in a model library.

Supporting Information: These capabilities are intended to provide basic geometry and coordinate frame representations to support specification of physical envelopes. The intent is that each block or equivalent will have its own reference coordinate system, and transformations can be applied between coordinate systems of different blocks. The shape of a block is defined as a property (e.g., 3-dimensional rectangular shape with length, height, and depth) whose values can be defined in its reference coordinate system. Consider references to standard formats (e.g., ISO 10303 (STEP), IGES)

6.6.2.3 Non-mandatory Structure Features

STC 1: Structure Requirements Group

STC 1.09: Structure Resolved to a Single Variant

Proposals for SysML v2 may include a capability to represent a single variant of a system-of-interest as a tree of Configuration Elements that establishes a fully expanded hierarchical (de)composition that can conform to an associated

Structure with Variability where a single selection is made for each variability choice (aka variation point).

Supporting Information: A SysML v2 implementation should support auto-generation of a tree of configuration elements from a decomposition of definition elements with variability based on a set of rules. A SysML v2 implementation should ideally also provide a capability to semi-automatically generate the reverse transformation from a tree of configuration elements to a decomposition of definition elements.

SysML v1.X Constructs: Sub-class with Redefinition, Subsetting, Property Specific Type

6.6.2.4 Non-mandatory Interface Requirements

INF 1: Interface Requirements Group

INF 1.07: Interface Agreement Group

INF 1.07.3: Geometric Constraints

Proposals for SysML v2 may provide the capability to constrain the interaction between the interface ends that include geometrical constraints on either Interface End.

Supporting Information: An example are the geometric constraints associated with connecting a plug and socket.

6.6.2.5 Non-mandatory Behavior Features

BHV 1: Behavior Requirements Group

BHV 1.03: Function-based Behavior Group

BHV 1.03.5: Composite Input and Output

Proposals for SysML v2 may include the capability to model composite inputs and outputs of function-based behavior with separate flows defined for the constituent inputs and outputs.

Supporting Information:

Refer to a Simulink Bus Object and a Modelica Expandable Connector

BHV 1.03.6: Behavior Library

Proposals for SysML v2 may include a library that can be populated with commonly used behaviors to support execution that includes functions to store items, such as data and energy.

SysML v1.X Constructs: fUML actions library

BHV 1.09: State History

Proposals for SysML v2 shall provide the capability to represent a state history of an individual element as a sequence of snapshots to describe how the individual element changes over time. The state history shall contain a reference time scale consistent with QUDV, and can include a start time, end time, and time step.

Supporting Information:

A snapshot represents the state of an individual element at a point in time by capturing the values of each of its value properties. An example is a snapshot of a vehicle that may include the value of its position, velocity, and acceleration at a point in time, and the snapshot of its engine that may include the value of its power-out and temperature at the same point in time. The value properties that vary with time are also called state variables.

The state history of a configuration element represents the default state history for each of its conforming individual elements.

6.6.2.6 Non-mandatory Requirements Features

RQT 1: Requirement Group

RQT 1.3: Requirement Relationships Group

RQT 1.3.7: Relationship Logical Constraint

Proposals for SysML v2 may include a capability to represent a logical expression (e.g. AND, OR, XOR, NOT, and conditional expressions like IF-THEN-ELSE and IF-AND-ONLY-IF) to one or more requirement relationships of the same kind, with an associated completeness property (e.g., complete satisfaction or partial satisfaction) and with a default expression of "And" for the logical expression.

Supporting Information: As an example, two blocks that have a satisfy relationship with the same requirement are asserted to completely satisfy the requirement by default

6.6.2.7 Non-mandatory Verification Features

VRF 1: Verification and Validation Requirements Group

VRF 1.2: Verification Case Group

VRF 1.2.5: Verification Activity

Proposals for SysML v2 may include a verification method that includes activities to collect the verification data, and include the ability to reference this data.

Supporting Information: The data may be extensive and not captured directly in the model.

VRF 1.2.6: Verification Evaluation Activity

Proposals for SysML v2 may include a verification method that includes activities to evaluate the verification data and the verification success criteria and generate a verification result of how well the requirements are satisfied (e.g., pass/fail/unverified).

6.6.2.8 Non-mandatory Analysis Features

ANL 1: Analysis Requirements Group

ANL 1.12: Decision Group

ANL 1.12.5: Trade-off

Proposals for SysML v2 may include a capability to represent an evaluation among a set of alternatives that can result in a decision based on a set of criteria. A trade-off may be dependent on other decisions.

ANL 1.12.6: Decision Expression

Proposals for SysML v2 may include a capability to model decision expressions that constrain the possible decisions (e.g., alternative A OR (alternative B and alternative C)).

ANL 1.13: Analysis Model - System Model Transformation

Proposals for SysML v2 may include the capability to represent the transformation and the mapping between the analysis model and the system model.

Supporting Information:

This transformation will represent the algorithm or derivation process, if used, for generating analysis models from system model (or vice versa), and the mapping will provide a mechanism to verify and synchronize analysis models when the system model changes (or vice versa). Refer to the requirement for Model Mappings and Transformations under the Language Architecture and Formalism Requirements.

ANL 1.14: Analysis Infrastructure

Proposals for SysML v2 may include the capability to represent the hardware, software, and the personnel (analysis experts) required for performing the analysis.

6.6.3 Non-mandatory Model Libraries Features

RML 1: Example Model and Model Libraries Group

RML 1.2: Model Libraries

Proposals for SysML v2 may include Model Libraries that contain generic elements that can be further specialized to define domain specific libraries in the following domain areas:

- Primitive Value Types
- Units and Quantity Kinds
- Components
- Natural environments
- Interfaces
- Behaviors
- Requirements
- Verification methods
- Analyses
- Basic geometric shapes
- Basic material kinds
- Viewpoint methods
- View definitions (i.e. different kinds of documents and other artifacts)
- Domain-specific symbols

Supporting information: The generic elements provide a common starting point for development of domain specific model libraries that can be elicited in future RFPs and/or the open source community.

6.7 Issues to be discussed

6.7.1 Proposals shall describe a proof of concept implementation that can successfully execute the test cases that are required in 6.5.4.

6.7.2 Proposals shall provide a requirements traceability matrix that demonstrates how each requirement in the RFP is satisfied. It is recognized that the requirements will be evaluated in more detail as part of the submission process. Rationale

should be included in the matrix to support any proposed changes to these requirements.

6.7.3 Proposals shall include a description of how OMG technologies are leveraged and what proposed changes to these technologies are needed to support the specification.

These issues will be considered during submission evaluation. They should not be part of the proposed normative specification. The responses to these Issues should be placed in Section 0 of the submission.

6.8 Evaluation Criteria

The following criteria will be used to evaluate how effectively the language supports the model-based systems engineering (MBSE) needs. Some of these criteria are difficult to quantify. The submission teams can propose more quantifiable criteria that support the intent.

6.8.1 Proposals will be evaluated for clarity of the proposed specification for the purpose of implementing conforming tools.

6.8.2 Proposals will be preferred if they demonstrate the ability to satisfy the conformance test cases.

6.8.3 Proposals will be evaluated based on their effectiveness in satisfying the following criteria:

- **Expressiveness:** Ability to express the concepts needed to describe systems
- **Precision:** Ability to represent the concepts in a concise way that enables unambiguous human and computer interpretation
- **Consistency/integrity**: Level of integration of the concepts to ensure the consistency and integrity of the language, such as integration between structure and behavior, between different behavior concepts (e.g., state-based, function-based), and consistency of decomposition semantics across different kinds of elements.
- **Presentation/communication**: Ability to effectively support communications with diverse stakeholders that includes presentation and generation of technical baseline information related to specification, design, analysis, and verification of the system and their relationships
- **Usability:** Ability for stakeholders, that include novice and experienced modelers, to efficiently and intuitively create, maintain, interpret, and use the model
- **Interoperability:** Ability to exchange data with other SysML models, other engineering models and tools, and other structured data sources
- **Adaptability/Customizability:** Ability to extend models to support domain-specific customizations
- **Scalability:** Ability to scale from small to large models

6.9 Other information unique to this RFP

The SysML v2 Requirement Support Document (syseng/2017-11-01) contains more detailed context information for each of the requirement sections provided in section 6, Specific Requirements on Proposals.

6.10 IPR Mode

Every OMG Member that makes any written Submission in response to this RFP shall provide the Non-Assertion Covenant found in Appendix A of the OMG IPR Policy [IPR].

6.11 RFP Timetable

The timetable for this RFP is given below. Note that the TF or its parent TC may, in certain circumstances, extend deadlines while the RFP is running, or may elect to have more than one Revised Submission step. The latest timetable can always be found at the OMG *Work In Progress* page at http://www.omg.org/schedules under the item identified by the name of this RFP.

Event or Activity	Date
Letter of Intent (LOI) deadline	*24 September, 2018*
Initial Submission deadline	*4 November, 2019*
Voter registration closes	*25 November, 2019*
Initial Submission presentations	*2 December, 2019*
Revised Submission deadline	*9 November, 2020*
Revised Submission presentations	*7 December, 2020*

Appendix A References & Glossary Specific to this RFP

A.1 References Specific to this RFP

A.1.1 Bibliographic Citation List

The following documents are referenced in this document:

[1] Created for SECM - This citation indicates that this text was created specifically for the SECM. SECM refers to the modeling effort that was used to derive the requirements for SysML v2. Enter [1, created for SECM] at the end of the text field.

[2] INCOSE. 2011. INCOSE Systems Engineering Handbook, Version 3.2.2. San Diego, CA, USA: International Council on Systems Engineering (INCOSE), INCOSE-TP-2003-002-03.2.2.

[3] BKCASE Editorial Board. 2015. The Guide to the Systems Engineering Body of Knowledge (SEBoK), v. 1.5. R.D. Adcock (EIC). Hoboken, NJ: The Trustees of the Stevens Institute of Technology. Accessed DATE. www.sebokwiki.org. BKCASE is managed and maintained by the Stevens Institute of Technology Systems Engineering Research Center, the

International Council on Systems Engineering, and the Institute of Electrical and Electronics Engineers Computer Society.

[5] ISO/IEC 2015. Systems and Software Engineering -- System Life Cycle Processes. Geneva, Switzerland: International Organization for Standardization / International Electromechanical Commissions. ISO/IEC/IEEE 15288:2015 (E).

[6] Wikipedia: Safety: Mar 31, 2015: http://en.wikipedia.org/wiki/Safety#Safety_measures

[8] Wikipedia. Main Page. Mar 31, 2015. http://en.wikipedia.org

[10] INCOSE (2015). Systems Engineering Handbook: A Guide for System Life Cycle Process and Activities (4th ed.) D. D. Walden, G. J. Roedler. K. J. Forsberg, R.D. Hamelin, and, T. M. Shortell (Eds.). San Diego, CA: International Council on Systems Engineering. Published by John Wiley & Sons, Inc.

[11] Merriam-Webster on-line dictionary

[12] UML 4SE RFP. SE Definitions List, April 01 2003: http://syseng.omg.org/UML%20for%20SE%20Definitions%20030401.xls

[14] INCOSE. 2015. Guide for Writing Requirements. Version 2, San Diego, CA, USA: International Council on Systems Engineering (INCOSE), INCOSE-TP-2010-006-02.

[15] OMG Unified Modeling Language (OMG UML), Version 2.5, March 2015, OMG Document Number - formal/2015-03-01

[17] ISO Online Browsing Platform (OBP), Terms and Definitions, https://www.iso.org/obp/ui/#home

[18] Weilkiens, Tim: Variant Modeling with SysML, MBSE4U - Tim Weilkiens, Apr 12 2016, ISBN 978-3-9817875-4-2

[19] Hilbert, D., & Ackerman, W. (1950) Mathematical Logic (L. Hammond, G. Leckie, & F. Steinhardt, Trans.). New York, NY: Chelsea Publishing Company

[20] Friedenthal, Sanford, Moore, Alan, Steiner, Rick. A Practical Guide to SysML: the systems modeling language. New York, NY: Elsevier, 2015. Third Edition

[21] Friedenthal, S. 2016. "Evolving SysML and the System Modeling Environment to Support MBSE, Part 2" INSIGHT (December Volume 19 Issue 4, Pg. 76-80)

[22] Torroni, P., Yolum, P., Singh, M., Alberti, M., Chesani, F., Gavanelli, M., Lamma, E., & Mello, P., (2009). Modeling Interactions via Commitments and Expectations. In Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models (p. 263-284) Hershey, PA: IGI Global.

[23] Winskel, D., & Pitts, A. (2005) Lecture Notes on Denotational Semantics for Part II of the Computer Science Tripos. Retrieved at: http://www.cl.cam.ac.uk/~gw104/dens.pdf

[24] (2017) Oxford Living Dictionaries. Retrieved at https://en.oxforddictionaries.com

[25] ISO 9241-210:2010. Ergonomics of human-system interaction - Part 210: Human-centered design for interactive systems. Geneva, Switzerland: International Organization for Standardization.

[26] OMG Systems Modeling Language (OMG SysML), Version 1.5, May 2017, OMG Document Number: formal/2017-05-01

[29] Introduction to SLIM - www.intercax.com/slim

[30] MOF Support for Semantic Structures (SMOF), Version 1.0, April 2013, OMG Document Number - formal/2013-04-02

[31] Meta Object Facility (MOF), Version 2.5.1, November 2016, OMG Document Number - formal/2016-11-01

[32] Matthews, P.H. (2014). The Concise Oxford Dictionary of Linguistics: Oxford University Press. Retrieved at: http://www.oxfordreference.com/

[33] Object Management Group. RFP Template, June 2015, OMG Document Number ab/15-06-01

[35] Friedenthal, S., Burkhart, R. 2015. "Evolving SysML and the System Modeling Environment to Support MBSE" INSIGHT (August 2015 Volume 18 Issue 2, Pg. 39-42)

[36] Yue, t., Ali, s Feb 2017. SECM - Requirements Concepts, Proposal on restricted textual requirements, Restricted Use Case Modeling (RUCM)

A.1.2 OMG Standards List

The following documents are referenced in this document: [ALF] Action Language for Foundational UML™ (Alf ™)

http://www.omg.org/spec/ALF

[BMM] Business Motivation Metamodel (BMM™)

http://www.omg.org/spec/BMM/[DMN] Decision Model and Notation ™ (DMN™)

http://www.omg.org/spec/DMN

[DD] Diagram Definition ™ (DD™)

http://www.omg.org/spec/DD

[FUML] Semantics of a Foundational Subset for Executable UML Models (fUML™)

http://www.omg.org/spec/FUML

[MARTE] UML Profile for MARTE

http://www.omg.org/spec/MARTE/

[MEF] Metamodel Extension Facility RFP

ad/2011-06-02

[MOF] Meta Object Facility ™ (MOF™) Core

http://www.omg.org/spec/MOF/

[MOFVD] Versioning and Development Lifecycle ™ (MOFVD™)

http://www.omg.org/spec/MOFVD

[MVF] Multiple Vocabulary Facility RFP

ad/2016-03-04

[OCL] Object Constraint Language ™ (OCL™)

http://www.omg.org/spec/OCL

[ODM] Ontology Definition Metamodel ™ (ODM™)

http://www.omg.org/spec/ODM/

[PST] Precise Semantics of Time

In progress

[PSCS] Precise Semantics of UML Composite Structures ™ (PSCS™)

http://www.omg.org/spec/PSCS

[PSSM] Precise Semantics of UML State Machines (PSSM)

http://www.omg.org/spec/PSSM

[ReqIF] Requirements Interchange Format (ReqIF ™)

http://www.omg.org/spec/ReqIF

[S&R] Profile for Safety and Reliability RFP

ad/2017-03-05

[SMOF] MOF Support for Semantic Structures ™ (SMOF™)

http://www.omg.org/spec/SMOF/

[SACM] Structured Assurance Case Metamodel (SACM™)

http://www.omg.org/spec/SACM

[SIMF] Semantic Information Modeling RFP

ad/2011-12-10

[SysPISF] SysML Extension for Physical Interaction and Signal Flow Simulation (SysPISF)

http://www.omg.org/spec/SysPISF/1.0/Beta1/

[SysML] OMG Systems Modeling Language Version ™ (SysML®)

http://www.omg.org/spec/SysML/

[UAF] Unified Architecture Framework (UAF) previously UPDM

http://www.omg.org/spec/UAF

[UML] Unified Modeling Language ™ (UML®)

http://www.omg.org/spec/UML

[UTP] UML Testing Profile ™ (UTP™)

http://www.omg.org/spec/UTP

[XMI] XML Metadata Interchange ™ (XMI®)

http://www.omg.org/spec/XMI

A.1.3 Other Standards List

The following documents are referenced in this document:

[FMI] Functional Mock-Up Interface (FMI)

http://fmi-standard.org/

[STEP] ISO 10303-233:2012 (STEP)

https://www.iso.org/standard/55257.html

[ArchDes] ISO 42010 - Systems and software engineering - Architecture description

http://cabibbo.dia.uniroma3.it/asw/altrui/iso-iec-ieee-42010-2011.pdf

[ISO 15288] ISO/IEC 15288:2015 - Systems and software engineering - System lifecycle processes

https://www.iso.org/obp/ui/#iso:std:iso-iec-ieee:15288:ed-1:v1:en

[ISO 15704] Industrial automation systems - Requirements for enterprise-reference architectures and methodologies

https://www.iso.org/obp/ui/#iso:std:iso:15704:ed-1:v1:en

[ISO 26550] ISO/IEC 26550:2015 - Software and Systems Engineering - Reference model for product line engineering and management

https://www.iso.org/obp/ui/#iso:std:iso-iec:26550:ed-2:v1:en

[ISO 80000] Quantities and units -- Part 1: General: ISO 80000-1:2009

https://www.iso.org/standard/30669.html

[ISO-TC184] Interoperability, integration, and architectures for enterprise systems and automation applications

https://www.iso.org/committee/54192.html

[HCD] ISO/DIS 9241-220.2(en) Ergonomics of human-system interaction - Part 220: Processes for enabling, executing and assessing human-centered design within organizations

https://www.iso.org/obp/ui/#iso:std:iso:9241:-220:dis:ed-1:v2:en

[SE Handbook] INCOSE Systems Engineering Handbook

http://www.incose.org/ProductsPublications/sehandbook

[SEBoK] Systems Engineering Body of Knowledge (SEBoK)
www.sebokwiki.org

A.2 Glossary Specific to this RFP

Abstract Relationship **-** A relationship between a more abstract element and a more concrete element. [1, created for SECM]

Abstract Syntax **-** The set of modeling concepts, their attributes and their relationships, as well as the rules for combining these concepts to construct partial or complete models. Based on [15, UML spec]

Alias **-** An assumed or additional name. [11, Merriam-Webster on-line dictionary]

An additional name that can include an acronym, an abbreviated name, a less formal name, or any other name intended to convey the same meaning. [1, created for SECM]

Allocation Relationship **-** A mapping relationship between model elements that is often intended to assign responsibility of one element to another. An example is the allocation of a function to a component which means the component is assigned the responsibility to perform the function. [1, created for SECM]

Allocate relationship provides a mechanism for associating elements of different types, or in different hierarchies, at an abstract level. Allocate is used for assessing user model consistency and directing future design activity. It is expected that an allocate relationship between model elements is a precursor to a more concrete relationship between the elements, their properties, operations, attributes, or sub-classes. [26, derived from SysML specification]

Alternative **-** A choice that is available to a decision maker. [1, created for SECM]

Analysis **-** A systematic investigation of a real or planned system to determine the information requirements and processes of the system and how these relate to each other and to any other system. (ISO/IEC/IEEE 2009) [3, SEBoK Glossary]

An investigation of some domain of interest that is intended to further understanding. The scope of the analysis includes the domain of interest that is the subject of the analysis, the models and supporting infrastructure used to perform the analysis, and the analysis case that defines how the analysis will be performed. [1, created for SECM]

The systematic investigation of a real or planned system to compare, evaluate, and select candidate system architectures, and/or determine causes & resolutions of failures and exceptions. [3, SEBoK].

Analysis Case - A process or set of steps intended to achieve an analysis objective. [1, created for SECM]

Analysis Infrastructure - All items needed to conduct the analysis effort, including the analysis models, modeling tools, measurement devices, people, procedures, documentation and information. [1, created for SECM]

Analysis Model - The computation model used to calculate the system properties (relevant to the analysis) to meet the analysis objectives. An Analysis model could be computer-based executable model (e.g. Mathematica/MATLAB code or FEA/CFD model), or a model representing physical measurement on a prototype or actual system. An analysis model includes the following characteristics:

1. Language in which the model is formulated
2. Software used to formulate the model
3. Type of model
4. Result data from executing the model
5. Relationship to the system representation, e.g. design model. This relationship embodies the model transformations required to generate or update the analysis model from the system representation

[1, created for SECM]

Analysis Objective - Represents the objective of the analysis, and can often be specified in terms of questions to be answered. The objective can be specified in the form of textual description and/or as an expression. When the objective is modeled using a set of math expressions their formal evaluation can be automated. The objective of an analysis is met if the expressions can be successfully evaluated by the information generated during the analysis. [1, created for SECM]

Analysis Result - Represents the evaluations of all the expressions in the Analysis Objective. An analysis is successful if its objective has been met. [1, created for SECM]

Analysis Subject - Also Known as "Subject of the Analysis" - The entities being analyzed in order to satisfy the analysis objectives. Since the scope of system analysis spans across the lifecycle, the subject of the analysis could be either of the following:

- Design representation of the system, such as a digital mock-up (computer model) of a spacecraft being developed
- Prototype of the system, such as a scaled or real prototype of the spacecraft
- Deployed system, such as the actual spacecraft deployed in orbit.

[1, created for SECM]

Annotation **-** A note added by way of comment or explanation. [11, Merriam Webster on-line dictionary]

A text statement that can also contain one or more navigational links to reference information that provides additional explanatory information about the annotated model element(s). [1, created for SECM]

API **-** In computer programming, an application programming interface (API) is a set of subroutine definitions, protocols, and tools for building application software. A good API makes it easier to develop a computer program by providing all the building blocks, which are then put together by the programmer. [8, Wiki]

Assumption **-** A statement that is asserted to be true. [1, created for SECM]

Basic 2D/3D Library **-** A digital library containing a collection of predefined model elements representing a set of reusable basic two and three dimensional geometric shapes that can be copied or referenced while constructing a model. [1, created for SECM]

Behavior **-** The interaction between individual elements and their change of state over time. Note: the state of an element refers to the values of its state variables at a point in time. [1, created for SECM]

Case **-** A process or set of steps intended to achieve an objective. The process often includes collecting and examining data or evidence about some subject in a context to produce a result or conclusion. [1, created for SECM]

Cause-effect Relationship **-** Relates a cause to an effect. The cause and effect may be represented by any model element, such as a state. [1, Created for SECM]

Component Definition **-** A type of definition element that can perform functions and present interface ends (i.e., ports) that connect to other components... Components can also contain sub-components that can connect with each other. [1, created for SECM]

Concept **-** An abstraction; a general idea inferred or derived from specific instances. [32, Oxford Dictionaries Online 2012] [3, SEBoK]

Concrete Syntax **-** The definition of the textual, graphical or other notations by which a model may be directly represented to users. [1, created for SECM]

Configuration Element **-** A relationship where one element (conforming element) is asserted to meet the constraints imposed by another element (i.e., the constraining element). An example is a view definition that conforms to a viewpoint where the viewpoint specifies the requirements that a view definition must satisfy.

Note: Other relationships may be specializations of the conform relationship. For example, a satisfy relationship is a conform relationship when a requirement is the constraining element. Other kinds of relationships such as realization, elaboration, and abstraction may be considered subclasses of this relationship.

[1, created for SECM]

Configuration Model **-** A model that represents a fully expanded hierarchical composition structure for a particular configuration of a system and its interfaces.

The Configuration Model can be used in two ways:

1. As an explicit representation that is generated from a Definition Model with a set of configuration parameters for any Variability Choice, e.g. chosen multiplicity or usage expression, as well as possible filtering or pruning.

2. Direct use of the Configuration Model as a simple 'loosely' typed hierarchical composition structure.

For the case (2) it is in principle possible to generate a (best effort) modular / typed Definition Model with some heuristic algorithm that detects elements of the same type.

Note: It is possible that a Configuration Model is over specified or out of sync w.r.t. to its associated Definition Model, and vice versa. Tool implementations will need to handle this situation with rules, refactoring and synchronization functionality.

[1, created for SECM]

Conform Relationship **-** A relationship where the element on one side of the relationship imposes constraints on the element on the other side of the relationship. [1, created for SECM]

Connector Definition **-** Relates two component definitions so the component usages can be connected and interact. [1, created for SECM]

Constraint Definition **-** A specification of a constraint expression and the parameters of the expression that can be used in different contexts. [1, created for SECM]

Constraint Expression **-** An expression that can be evaluated to true or false. [1, created for SECM]

Container **-** A kind of model element that contains other model elements and applies scoping rules such as namespace rules to the contained elements.

Containers can contain other containers providing a mechanism to organize the model. [1, created for SECM]

Context **-** A model element that establishes a scope for representing usages (or roles) of defined concepts in a particular situation. An example is a vehicle context where the same type of tire can be a front or rear tire, depending on how it is used in its vehicle context, or the same type of tire can be a swing when it is used in the context of a swing-set. [1, created for SECM]

Control Node **-** A kind of function used to control sequencing of item flows and event flows by specifying a logical expression of output flows in terms of its input flows. Note: SysML v1 examples include join specification, join, fork, decision, and merge nodes in activities, and junction and choice pseudo-states in state machines, and alt, par, and opt interaction operators in sequence diagrams. [1, created for SECM]

Criteria **-** An expression that specifies the characteristics of interest and their relative weighting and or objective function that can be used as a basis for an evaluation. [1, Created for SECM]

Data Model **-** A Data Model is a model that organizes elements of data and standardizes how they relate to one another. [1, created for SECM]

A data model explicitly determines the structure of data. Data models are specified in a data modeling notation, which is often graphical in form. [8, Wiki]

Decision **-** A selection among alternatives based on some criteria. [1, created for SECM]

a determination arrived at after consideration: conclusion made the decision to attend graduate school [11, Merriam Webster on-line, def b]

Declarative Semantics **-** Association of meaning that specifies what rather than how. Communication with declarative semantics specifies what actions should be brought out in an interaction, rather than how they are brought out. [22, Modeling Interactions via Commitments and Expectations]

Definition Element **-** An element that defines a class of individual elements with shared features. [1, created for SECM]

Definition Model - A Definition Model represents a strongly typed, modular, hierarchical composite structure. It allows for inclusion of variation points so that it can represent a set of possible system variants. Variation points can include multiplicity ranges, subtrees, alternative composition with or without constraints, etc.

The Definition Model contains a bag of building blocks represented by Definition Elements that may directly use (i.e. one level deep) zero or more Direct Usage Features of other Definition Elements.

In many cases there is a need to unambiguously identify and reference more deeply nested usage elements, e.g. to introduce local override of values for a particular usage two or more levels down from a Definition Element or to define an interface connector between nested interface ends. For this purpose, an element path enables unambiguous traversal from a top element to a deeply nested usage feature.

If no features need to be overridden, redefined nor added at a usage more than one level deep, then a Definition Model constitutes a complete implicit definition of the decomposition structure provided that a single top element (i.e. context element) is identified. With that, it is possible to automatically generate a corresponding Configuration Model that represents the full and explicit (deeply nested) expansion of the decomposition structure.

It is important to note that Deeply Nested Usage Features need only be created (and persisted) if there is a need to override feature values, (re)define features or reference usages at a deeply nested level. Otherwise their representation can be automatically derived 'on the fly' as their existence is fully implied by Definition Elements and Direct Usage Features only.

In case there are Variation Points present in the Definition Model, choices must be made within the range of possible variabilities in order to transform the Definition Model into a Configuration Model that represents a single actual variant that complies with the implicit definition represented by the Definition Model.

Using some heuristics, and perhaps with some human assistance, it is in principle also possible to devise an algorithm that can automatically derive a Definition Model from a given Configuration Model. This is attractive since it would allow a beginning system modeler to start with the simpler to understand Configuration Model and transform it to the more powerful and generalized Definition Model. Such a capability would also support use cases for reverse engineering of existing system architectures.

[1, created for SECM]

Dependency Relationship - A relationship between two elements where a change in the independent element may impact the dependent element. [1, created for SECM]

A Dependency is a Relationship that signifies that a single model Element or a set of model Elements requires other model Elements for their specification or implementation. This means that the complete semantics of the client Element(s) are either semantically or structurally dependent on the definition of the supplier Element(s). [15, UML Specification]

Derived Relationship - A relationship that is derived from other relationships. Note: This is not to be confused with a Requirement Derivation Relationship.

An example of a derived relationship is a transitive relationship between C and A, where B relates to A and C relates to B.

Another example of a derived relationship is a connector between two composite parts that is derived from a connector between their nested parts. [1, created for SECM]

Element - An entity that can be assigned an identifier and may be contained in or external to the subject model. [1, created for SECM]

Element Group - A mechanism for grouping various model elements without imposing ownership constraints or impacting the model element that are members of the group. Criteria is defined to determine which members belong to the group. Examples of Element Groups include model elements that are associated with a particular release of the model, have a certain risk level, or model elements from a legacy design that are part of a new design. [1, created for SECM]

Element Group Relationship - Relates an element group to a member of the group. In addition to membership criteria, logical expressions can be applied to membership of the group, such as AND, OR, XOR, NOT, and conditional expressions like IF-THEN-ELSE and IF-AND-ONLY-IF. [1, created for SECM]

Environment - Any entity that is external to a system of interest that affects the system of interest or is affected by the system of interest system through direct or indirect interactions between the system of interest and the external entities. [1, created for SECM]

(1) Anything affecting a subject system or affected by a subject system through interactions with it, or anything sharing an interpretation of interactions with a subject system. (IEEE 1175.1-2002 (R2007), 3.6) (2) The surroundings (natural or man-made) in which the system-of-interest is utilized and supported; or in which the system is being developed, produced or retired. (INCOSE 2010) [3, SEBoK]

Event **-** Occurrence defined by a change in conditions that may trigger a response (e.g., a change in state or start of a function). [1, created for SECM]

1. Occurrence of a particular set of circumstances. ISO/IEC 16085:2006 (IEEE Std. 16085-2006), Systems and software engineering - Lifecycle processes - Risk management.3.2.

2. An external or internal stimulus used for synchronization purposes

[17, ISO OBP Definitions]

Explanation Relationship **-** A relationship between an element being rationalized, i.e. the conclusion, and the element justifying the conclusion, i.e. the rationale.

A conclusion that is explained can be represented by any type of element including elements such as blocks, requirements, or relationships, such as a satisfy relationship between a requirement and design element.

The rationale can refer to the supporting information, such as reference to one or more analysis.

[1, created for SECM]

Expression **-** In mathematics, an expression or mathematical expression is a finite combination of symbols that is well-formed according to rules that depend on the context. Mathematical symbols can designate numbers (constants), variables, operations, functions, brackets, punctuation, and grouping to help determine order of operations, and other aspects of logical syntax. [8, Wiki]

External Element **-** An entity external to the subject model. In the context of model management, this often refers to items such as a file, web page, or a model element in another model [1, created for SECM]

Finite State **-** The condition of an individual element for a period of time that constrains its structure and behavior. [1, created for SECM]

Formal Requirement Statement **-** A formal requirement captures all aspects of a requirement in a machine-readable form, vs. text in a Textual requirement. This enables requirements to be stated more precisely, and to be evaluated in an automated way in support of verification and validation.

[1, created for SECM]

Formalism **-** A description of something in formal mathematical or logical terms. [24, Oxford Living Dictionaries]

Function Definition **-** A transformation from inputs to outputs through a controlled sequence of actions (e.g., function usages). [1, created for SECM]

(4) A function is defined by the transformation of input flows to output flows, with defined performance. [3, SEBoK]

Generalization Relationship **-** A taxonomic relationship between a more general Classifier and a more specific Classifier. Each instance of the specific Classifier is also an instance of the general Classifier. The specific Classifier inherits the features of the more general Classifier. A Generalization is owned by the specific Classifier. [15, UML Specification]

Hardware **-** 1. Physical equipment used to process, store, or transmit computer programs or data. 2. All or part of the physical components of an information system. ISO/IEC 2382-1:1993, Information technology - Vocabulary - Part 1: Fundamental terms.01.010.07 cf. software [17, ISO OBP Definitions]

Hyperlink **-** A hyperlink is a reference to data that the reader can directly follow either by clicking, tapping, or hovering. A hyperlink points to a whole document or to a specific element within a document. Hypertext is text with hyperlinks. [8, Wiki]

Individual Element **-** A representation of an identifiable object that may or may not exist in the physical world. An example is an individual element that represents a specific automobile with a vehicle identification number, which in turn is composed of individual elements that represent each of its components with their respective serial numbers. An individual element may conform to a configuration element that corresponds to the design configuration that may be realized by multiple individual elements. The individual element can be thought of as a "digital twin". [1, created for SECM]

Interaction **-** A dynamic exchange between 2 or more individual elements. [1, created for SECM]

Interface Agreement Definition **-** Specifies the rules that constrain an interaction.

Two examples of interface agreements are the constraints on a physical interaction that are specified by conservation laws, and a communication protocol: [1, created for SECM]

In telecommunications, a communications protocol is a system of rules that allow two or more entities of a communications system to transmit information via any kind of variation of a physical quantity. These are the rules or standard that defines the syntax, semantics and synchronization of communication and possible error recovery methods. Protocols may be implemented by hardware, software, or a combination of both. [8, Wiki, Communications Protocol]

Interface Definition **-** 1. A relationship that enables the physical and functional interaction between elements that includes two (2) interface ends, the connection between them, and the constraints on the interaction. [1, created for SECM]

A shared boundary between two functional units, defined by various characteristics pertaining to the functions, physical signal exchanges, and other characteristics. (ISO/IEC 1993) [3, SEBoK, 2]

A hardware or software component that connects two or more other components for the purpose of passing information from one to the other. (ISO/IEC 1993) [3, SEBoK, 3]

To connect two or more components for the purpose of passing information from one to the other. (ISO/IEC/IEEE 200) [3, SEBoK, 4]

Interface Medium Definition - A transmission medium between components. It presents interface ends that can connect to other components via interface agreements. Examples of interface medium include a wire, a cable harness, a computer bus, a network, a pipe that enables the flow of fluid, and the atmosphere. [1, created for SECM]

Layered Interface - An interface that is decomposed into functional units, called layers that can accept inputs from an adjacent layer, transform the inputs, and provide the outputs to its other adjacent layer. Each layer has its own interface with its adjacent layers that are governed by interface agreements. Each layer typically addresses a distinct set of concerns. A fundamental principle of an interface layer is that the layer below is independent of the layer above. A Protocol Stack is a set of layers that transforms items to enable their exchange, such as for purposes of communication. [1, created for SECM]

Logical Expression - An expression that supports as a minimum the standard boolean operators AND, OR, XOR, NOT, and conditional expressions like IF-THEN-ELSE and IF-AND-ONLY-IF, in which symbols bound to any characteristics (e.g. value properties or usage features) may be used. [1, created for SECM]

Mapping Rules - A set of rules that define how the elements of a model conforming to a particular metamodel are transformed into elements of another model or data source that conforms to another (possibly the same) metamodel. [1, created for SECM]

Mathematical Logic - An extension of the formal method of mathematics to the field of logic. [19, Mathematical Logic, Hilbert & Ackerman]

Meta Object Facility - Provides the basis for metamodel definition in OMG's family of MDA languages and is based on a simplification of UML2's class modeling capabilities. In addition to providing the means for metamodel definition it adds core capabilities for model management in general, including Identifiers, a simple generic Tag capability and Reflective operations that are defined generically and can be applied regardless of metamodel. [31, MOF Specification]

An OMG standard, closely related to UML, that enables metadata management and language definition. [33, OMG RFP Template]

Metadata - Data that provides information about other data that is used to summarize basic information about data to make tracking and working with data easier. Some examples include:

- Means of creation of the data
- Purpose of the data
- Time and date of creation
- Creator or author of the data
- Location on a computer network where the data was created
- Standards used
- File size

[8, derived from Wiki, Metadata]

Metamodel - A metamodel is a model of a modeling language. [1, created for SECM]

Model - A representation of one or more concepts that may be realized in the physical world. [20, A Practical guide to SysML]

Model Element - A constituent of a model. A model element in a system model typically is used to represent some aspect of the system or its environment that may include representations of structure, behavior, requirements, and their relationships at varying levels of granularity. [1, created for SECM]

Model Library - A model element that contains other model elements that are designated for reuse.

The contained elements typically can be copied, sub-classed, or referenced for use in a model. [1, created for SECM]

Model Transformation - A mapping between two modeling languages or other data sources that enables a model expressed in one modeling language to be expressed in whole or in part in the other modeling language or data source. (Created for SEBoK) [3, SEBoK]

Model-Theoretic Semantics - An account of meaning in which sentences are interpreted in terms of a model of, or abstract formal structure representing, an actual or possible state of the world: compare possible world. Usually, at least, an account of truth conditions; i.e. sentences are interpreted as true or false in such a model. [32, Oxford Dictionary of Linguistics]

Modeling Language **-** A language used to represent models. [1, created for SECM]

MOF **-** Acronym for Meta Object Facility. See Meta Object Facility for the definition. [1, created for SECM]

Navigation Relationship **-** A navigable connection from a model element or text within a model element to another model element within the same model or an external element. [1, created for SECM]

The connections may have different behaviors, such as (1) reference connections for basic traceability, (2) data map connections for exchange for parameter values, (3) function wrap connections for wrapping executable code in system model elements, and (4) model transform connections for generating and synchronizing model structures bi-directionally. [29, Intro to SLIM]

An identifier attached to an element (as an index term) in a system in order to indicate or permit connection with other similarly identified elements; especially: one (as a hyper link) in a computer file [11, Merriam-Webster]

Objective **-** A desired or required state. An example is to achieve a certain level of performance within specified cost constraints. [1, created for SECM]

Operational Semantics **-** Meanings for program phrases defined in terms of the steps of computation they can take during program execution. [23, Denotational Semantics Lecture]

Platform Specific Binding **-** A general approach for mapping a Platform Independent Model to a corresponding Platform Specific Model in a specific technology platform, with an implementation that is directly usable in the target platform.

Platform Specific Model **-** (PSM) - A model of a system, subsystem, or component that includes information about the specific technology that is used in the realization of it on a specific platform, and hence possibly contains elements that are specific to the platform. [33, OMG RFP Template]

Port Definition **-** Also known as "Interface End Definition". The specification of a connection point that enables one element to be connected to another. [1, created for SECM]

Precondition Expression **-** A set of conditions that must exist prior to initiating a behavior. [1, created for SECM]

Something that must exist or happen before something else can exist or happen [11, Merriam-Webster on-line definition]

Problem **-** A Problem is a deficiency, limitation, or failure to satisfy a requirement or need or cause some other undesired outcome or effect from the

perspective of a stakeholder. It may be used during any lifecycle phase including design, verification, manufacture, and operations. [26, derived from SysML Specification]

Property - Any named, measurable or observable attribute, quality or characteristic of a system or system element. (OMG 2003) [3 SEBoK]

Rationale - Argument that provides the justification for the selection of an engineering element. (Faisandier 2012) [3, SEBoK]

A model element that refers to other data that provides justification for a decision or other conclusion. [1, created for SECM]

Reference Model - A reference model in systems, enterprise, and software engineering is an abstract framework or domain-specific ontology consisting of an interlinked set of clearly defined concepts produced by an expert or body of experts in order to encourage clear communication. A reference model can represent the component parts of any consistent idea, from business functions to system components, as long as it represents a complete set. This frame of reference can then be used to communicate ideas clearly among members of the same community. [8, Wiki]

Refine Relationship - A relationship between two elements where the element on one side provides a more precise representation than the other. An example is the relationships between two text statements, where one statement is verbose and ambiguous, and the other statement is concise and precise. [1, created for SECM]

Relationship - The way in which two or more things are connected. [11, Merriam-Webster on-line]

A representation of an association between model elements. SysML constrains relationships to be have two ends (i.e., a binary relationship). A relationship can have a name, and have a direction between its source end and target end. [1, created for SECM]

Requirement - Also known as Requirement Usage. A usage of a Requirement Definition. [1, created for SECM]

Requirement Attribute - Additional information included with a requirement definition, which is used to aid in the management of that requirement. [14, derived from Guide Writing Requirements, Definitions)

Requirement Definition - A definition of a constraint that is used in the context of a specification that a valid solution must satisfy. [1, created for SECM]

Statement that identifies a product* or process operational, functional, or design characteristic or constraint, which is unambiguous, testable or measurable, and necessary for product or process acceptability. (ISO/IEC 2007)

*includes product, service, or enterprise. [3, SEBoK]

Requirement Derivation Relationship - A derived relationship imposes constraints that are needed to satisfy other constraints (i.e., source requirements). The derived requirement can impose requirements at the same level of the design as the source requirement, or at a lower-level of design. [1, created for SECM]

Requirement Group - A model element that can group requirements to give them context. A specification is a top-level requirement group. [1, created for SECM]

Requirement Group Relationship - A relationship between a requirement group and a requirement to establish context for the requirement definition. [1, created for SECM]

Requirement Identifier - This is an identifier that uniquely identifies a requirement from other requirements. This identifier is not a paragraph number. It can be a separate identifier or automatically assigned by a Requirement Management Tool (RMT) the organization is using.

This identifier is not the same as the Unique Identifier that every model element contains. This identifier should be unique across all requirements and can be tailored to meet a specific organization's needs. This identifier typically includes some encoding to help humans relate to its context (for example CR_100 for a customer requirement, where CR_ is a user-defined prefix unique to a requirement specification, and 100 is tool generated). [1, created for SECM]

Requirement Type/Category - Each organization will define types or categories to which a requirement fits, based on how they may wish to organize their requirements. The Type/Category field is most useful because it allows the requirements database to be viewed by a large number of designers and stakeholders for a wide range of uses. [14, Guide for Writing Requirements, 5.3.25]

Restricted Requirement Statement - A specific type of Textual Requirement Statement, specified by using a restricted/controlled natural language that puts restrictions on grammar (which can be realized by templates and patterns) and vocabulary (by using e.g., pre-defined keywords). Restricted Requirement Statements (RRS) strikes a balance between practicality and level of automation, bridges the gap from informal requirements specifications in natural language to formal, precise, and analyzable specifications. [1, created for SECM]

Satisfy Relationship - A relationship between a requirement and the constrained element of the design solution, which asserts that the constraints imposed by the requirement on the design element are met.

Satisfy is intended to be a specialized kind of conform relationship where one side of the relationship must be a requirement.

[1, created for SECM]

Scenario Definition **-** A sequence of actions that are performed by interacting components. [1, created for SECM]

Semantics **-** The rules by which syntactic expressions and model elements are assigned meaning. (ISO 13537:2010, 3.2.3.14) [17, ISO OBP Definitions]

SMOF **-** MOF Support for Semantic Structures. This extension to MOF modifies MOF 2 to support dynamically mutable multiple classifications of elements and to declare the circumstances under which such multiple classifications are allowed, required, and prohibited. [30, OMG SMOF]

Snapshot **-** The state of each state variable of an individual item at a point of time. For example, a snapshot of a vehicle may include the value of its position, velocity, and acceleration at a point in time, and the snapshot of its engine may include the value of its power-out and temperature at the same point in time. [1, created for SECM]

Software **-** All or part of the programs, procedures, rules, and associated documentation of an information processing system. (ISO/IEC 2382-1:1993) [3, SEBoK]

A class of component which implements functionality that is specified by a computer program. [1, created for SECM]

Stakeholder **-** (1) Individual or organization having a right, share, claim, or interest in a system or in its possession of characteristics that meet their needs and expectations (ISO/IEC/IEEE 2015)

(2) Individual or organization having a right, share, claim, or interest in a system or in its possession of characteristics that meet their needs and expectations; N.B. Stakeholders include, but are not limited to end users, end user organizations, supporters, developers, producers, trainers, maintainers, disposers, acquires, customers, operators, supplier organizations and regulatory bodies. (ISO/IEC June 2010)

(3) An individual, team, or organization (or classes thereof) with interests in, or concerns relative to, a system or domain of interest. (Derived from ISO/IEC 2007)

(4) A stakeholder in an organization is (by definition) any group or individual who can affect or is affected by the achievement of the organization's objectives. (Freeman 1984)

[3, SEBoK]

State **-** The values of a state variable at a point in time. [1, created for SECM]

State History **-** A ordered series of snapshots of an individual element that define how the value of its value properties change over the elements lifetime. Note that an analysis is often used to predict a state history associated with selected value properties, or in some cases, reconstruct a previous state history, such as when analyzing a failure that has occurred. Measurements are used to measure the state history associated with selected value properties. A predicted or measured state history has uncertainty associated with it. [1, created for SECM]

State Machine **-** A representation of the finite states and the transitions between them for an individual element over its lifetime. In a particular finite state, or on transition between finite states, selected functions and constraints are enabled. A state machine may have multiple concurrent regions where only one finite state is active in each region at a given point in time.

Note: A lifetime of an individual element persists over the duration that it retains its identify. For example, a individual products lifetime may begin when it comes off the manufacturing line, and ends when it is dis-assembled for disposal. [1, created for SECM]

State Variable **-** A value property whose value varies with time. [1, created for SECM]

Supporting Information **-** Supporting Information provides additional information to help better understand the intent of a model element and specifically for a requirement or requirement group. This information can include items such as an introduction to a set of requirements, one or more goals, a reference to further readings, justification, rationales, examples, diagrams, pictures, graphs, tables, etc. In addition, it can include navigational links from this element or text within this element. [1, created for SECM]

Syntax **-** Structure of expressions in a language, and the rules governing the structure of a language, independent of their meanings or the manner of their interpretation and use. (derived from ISO/PAS 16917:2002(en), 3.2.68) [17, ISO OBP Definitions]

SysML **-** The OMG Systems Modeling Language (OMG SysML®) is a general-purpose language for representing systems.

SysML supports the specification, analysis, design, verification, and validation of a broad range of complex systems.

These systems may include hardware, software, information, processes, personnel, and facilities. [26, derived from SysML v1 specification]

SysML v2 Declarative Semantics **-** A declarative specification of the semantics of the SysML v2 Foundational Subset using a mathematical logic formalism. [1, created for SECM]

SysML v2 Diagram Definition **-** A specification of the standard diagrammatic views included in the SysML v2 Concrete Syntax, using a metamodel-based approach consistent with the OMG Diagram Definition standard. [1, created for SECM]

SysML v2 Foundational Subset **-** The foundational subset of the SysML v2 Language in which the SysML v2 Semantic Model (or at least the core of it) and whose semantics is also separately specified by the SysML v2 Declarative Semantics. [1, created for SECM]

SysML v2 Metamodel **-** A model of the SysML v2 modeling language that includes its abstract syntax, concrete syntax, semantics, and the mappings between them. [1, created for SECM]

SysML v2 Model **-** A top-level container of model elements represented in the SysML v2 modeling language, which is specified by the SysML v2 metamodel. [1, created for SECM]

SysML v2 Semantics Model **-** A SysML v2 Model Library that specifies the SysML v2 Semantics as a SysML v2 Model. [1, created for SECM]

System **-** A set of elements in interaction. (1, von Bertalanffy 1968)

Combination of interacting elements organized to achieve one or more stated purposes (ISO/IEC/IEEE 15288:2015) [3, SEBoK, 2]

"A set of elements that can interact with one another, and can be viewed as a whole that can interact with other external elements." [1, created for SECM]

System Context **-** (1) Describes the system relationships and environment, resolved around a selected system-of-interest. (Flood and Carson 1993) [3, SEBoK]

System Element **-** A member of a set of elements that constitutes a system. A system element is a discrete part of a system that can be implemented to fulfill specified requirements. A system element can be hardware, software, data, humans, processes (e.g., processes for providing service to users), procedures (e.g., operator instructions), facilities, materials, and naturally occurring entities (e.g., water, organisms, minerals), or any combination. (ISO/IEC 15288:2015) [3, SEBoK]

System Model **-** (3) A simplified representation of a system at some particular point in time or space intended to promote understanding of the real system. (Bellinger 2004)

(4) An abstraction of a system, aimed at understanding, communicating, explaining, or designing aspects of interest of that system (Dori 2002)

(5) A selective representation of some system whose form and content are chosen based on a specific set of concerns. The model is related to the system by an explicit or implicit mapping. (Object Management Group 2010)

[3, SEBoK]

System Modeling Environment **-** (SME) A part of the overall Model-Based Engineering (MBE) environment that systems engineers use to perform model-based systems engineering (MBSE) and interact with other members of the development team. The SME must implement the SME services to provide the functionality needed to enable systems engineers and others to evolve the system model throughout the lifecycle. [21, Insight Article Part 2]

Textual Requirement Statement **-** The traditional "shall" textual statement used to state a requirement. [1, created for SECM]

Time Property **-** A value property that represents the base quantity time. [1, created for SECM]

Timestamp **-** A sequence of characters or encoded information identifying when a certain event occurred, including the date and time of day. The timestamp refers to digital date and time information attached to digital data. For example, computer files contain timestamps that tell when the file was last modified. [8, Wiki]

A timestamp should be represented using a common, time zone independent format that includes resolution and context such as UTC. Format example: time=2009-06-15T13:45:30; context=last change [1, created for SECM]

Trade-off **-** A kind of analysis to evaluate a set of alternatives based on some criteria, and used to make a decision to select one or more of the alternatives. [1, created for SECM]

UML Profile **-** A standardized set of extensions and constraints that tailors UML to particular use. [33, OMG RFP Template]

Unified Modeling Language **-** (UML) The objective of the Unified Modeling Language (UML) is to provide system architects, software engineers, and software developers with tools for analysis, design, and implementation of software-based systems as well as for modeling business and similar processes. [15, UML Specification]

Unique Identifier **-** This unique identifier is assigned to every element. This identifier must be unique universally, that is within the containing model, within the SME and external to the SME. [1, created for SECM]

Unit Under Verification **-** A system or part of a system that is the subject of a verification procedure. [1, created for SECM]

Units Library **-** A digital library containing a collection of predefined model elements representing a set of reusable units and quantity kinds that can be copied or referenced by a model. [1, created for SECM]

URI **-** In information technology, a Uniform Resource Identifier (URI) is a string of characters used to identify a resource. Such identification enables interaction with representations of the resource over a network, typically the World Wide Web, using specific protocols. [8, Wiki]

Usability **-** The extent to which a system, product or service can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use. [25, ISO 9241-210:2010]

Usage Feature **-** Usage Feature - A generalization of a value property or a usage element that is contained by a Definition Element, and is typed by the applicable Value Type or Definition Element. [1, created for SECM]

UUID **-** Universally Unique identifier (UUID) - A unique identifier assigned to every model element. This identifier must be unique both within the SME and external to the SME. This UUID conforms to IETF RFC 4122 http://datatracker.ietf.org/doc/rfc4122/. See also http://en.wikipedia.org/wiki/Universally_unique_identifier for a practical introduction on a UUID. [1, created for SECM]

Value Expression **-** An expression that can be evaluated to yield a value typed by a Value Type. The expression is stated in an expression language that support all usual mathematical and logical operators. [1, created for SECM]

Value Property **-** Any named, quantifiable characteristic of a class of elements whose range of values are constrained by a Value Type. [1, created for SECM]

Value Type **-** Named definition of the set of possible values of a value-based characteristic. [1, created for SECM]

Variability Context **-** A model that captures the desired variabilities and constraints for a set of configurations. [1, Created for SECM]

Variability Expression **-** An expression that constrains the variant choices. [1, created for SECM]

Variant **-** A variant (or option) represents a choice among one or more features. A variant can include additional variation points. [1, Created for SECM]

Variant Binding **-** A variant binding binds a model element to an element in another model that specifies the variation. [1, Created for SECM]

Variant Selection **-** Defines the selected variant based on the options identified by a variation point. [1, created for SECM]

Variation Point - A definition of one or more choices of some characteristic feature of a definition element. The set of choices available may depend on other selections that have been made, such as the choice of entertainment systems that are available for a particular model of a vehicle. The addition of one or more choices allows for a compact and inherently consistent representation of options or alternatives at any level in the hierarchical composition that support modeling product lines. A configuration element reflects selection of a specific variant among the available choices. [1, created for SECM]

Verification - (1a) Confirmation, through the provision of objective evidence, that specified (system) requirements have been fulfilled. (ISO/IEC 2008, section 4.38)

(1b) Verification is a set of activities that compares a system or system element against the required characteristics. This includes, but is not limited to, specified requirements, design description and the system itself. The system was built right. (ISO/IEC/IEEE 2015, 1, Section 6.4.6)

(2) The evaluation of whether or not a product, service, or system complies with a regulation, requirement, specification, or imposed condition. It is often an internal process. Contrast with validation. (PMI 2013)

[3, SEBoK]

Verification Outcome - Describes the data and any other results from performing the Verification Activity. [1, created for SECM]

Verification Activity - An activity that specifies one or more steps of a verification case. [1, created for SECM]

Verification Case - A set of steps required to achieve a verification objective. [1, created for SECM]

Verification Context - A context for a unit under verification and the verification system that performs the verification as defined by the verification case. [1, created for SECM]

Verification Evaluation Activity - An activity that compares the verification outcome data produced by the verification activity with the verification success criteria. [1, created for SECM]

Verification Method - The verification method for each requirement simply states the planned method of verification (inspection, demonstration, test, analysis, and simulation). [10, INCOSE Handbook]

The Description property provides a textual description of the steps that will be taken in Verification Activity and Verification Evaluation Activity. [1, created for SECM]

The type of method may also include sampling and analogy. [2, SEBoK]

Verification Objective - The identification of the requirements that are to be verified. [1, created for SECM]

Verification Requirement - A requirement applied to the means of establishing compliance of an end item with its specification requirements. [1, created for SECM]

Verification Result - The result of the Verification Evaluation. [1, created for SECM]

Verification System - An aggregation of enabling elements needed to perform verification activities. This includes the equipment, users and facilities used to perform the activity. [1, created for SECM]

Verify Relationship - A relationship between a requirement and a verification case that can be elaborated to specify how verification of the requirement is accomplished and to produce the verification result. [1, created for SECM]

View - Also known as "View Individual" - A representation of a specific artifact that is constructed to address one or more stakeholder concerns specified by a viewpoint. This may represent an electronic file or hard copy of a diagram, table, document, or even a physical model. [1, created for SECM]

View Definition - The definition of the structure of a view in terms of its sub-views, and the methods needed to construct an individual view. The construction methods generally involve the querying of a model or other data source, and the presentation of the query results. A view definition is intended to conform to a viewpoint. [1, created for SECM]

View Element - A constituent element of a Concrete Syntax Metamodel that defines how a model element is presented. [1, created for SECM]

Viewpoint – Defines a set of stakeholder concerns and/or interest areas regarding a domain of interest. [1, created for SECM]

Appendix B General Reference and Glossary

B.1 General References

The following documents are referenced in this document:

[BCQ] OMG Board of Directors Business Committee Questionnaire

http://doc.omg.org/bcq

[CCM] CORBA Core Components Specification

http://www.omg.org/spec/CCM/

[CORBA] Common Object Request Broker Architecture (CORBA)

http://www.omg.org/spec/CORBA/

[CORP] UML Profile for CORBA

http://www.omg.org/spec/CORP

[CWM] Common Warehouse Metamodel Specification

http://www.omg.org/spec/CWM

[EDOC] UML Profile for EDOC Specification

http://www.omg.org/spec/EDOC/

[Guide] The OMG Hitchhiker's

http://doc.omg.org/hh

[IDL] Interface Definition Language Specification

http://www.omg.org/spec/IDL35

[INVENT] Inventory of Files for a Submission/Revision/Finalization

http://doc.omg.org/inventory

[IPR] IPR Policy

http://doc.omg.org/ipr

[ISO2] ISO/IEC Directives, Part 2 - Rules for the structure and drafting of International Standards

http://isotc.iso.org/livelink/livelink?func=ll&objId=4230456

[LOI] OMG RFP Letter of Intent template

http://doc.omg.org/loi

[MDAa] OMG Architecture Board, "Model Driven Architecture - A Technical Perspective"

http://www.omg.org/mda/papers.htm

[MDAb] Developing in OMG's Model Driven Architecture (MDA)

http://www.omg.org/mda/papers.htm

[MDAc] MDA Guide

http://www.omg.org/docs/omg/03-06-01.pdf

[MDAd] MDA "The Architecture of Choice for a Changing World

http://www.omg.org/mda

[MOF] Meta Object Facility Specification

http://www.omg.org/spec/MOF/

[NS] Naming Service

http://www.omg.org/spec/NAM

[OMA] Object Management Architecture

http://www.omg.org/oma/

[OTS] Transaction Service

http://www.omg.org/spec/OTS

[P&P] Policies and Procedures of the OMG Technical Process

http://doc.omg.org/pp

[RAD] Resource Access Decision Facility

http://www.omg.org/spec/RAD

[ISO2] ISO/IEC Directives, Part 2 - Rules for the structure and drafting of International Standards

http://isotc.iso.org/livelink/livelink?func=ll&objId=4230456

[RM-ODP]

ISO/IEC 10746

[SEC] CORBA Security Service

http://www.omg.org/spec/SEC

[TEMPL] Specification Template

http://doc.omg.org/submission-template

[TOS] Trading Object Service

http://www.omg.org/spec/TRADE

[UML] Unified Modeling Language Specification

http://www.omg.org/spec/UML

[XMI] XML Metadata Interchange Specification

http://www.omg.org/spec/XMI

B.2 General Glossary

Architecture Board (AB) **-** The OMG plenary that is responsible for ensuring the technical merit and MDA compliance of RFPs and their submissions. [33, OMG RFP Template]

Board of Directors (BoD) **-** The OMG body that is responsible for adopting technology. [33, OMG RFP Template]

Common Object Request Broker Architecture (CORBA) **-** An OMG distributed computing platform specification that is independent of implementation languages. [33, OMG RFP Template]

Common Warehouse Metamodel (CWM) **-** An OMG specification for data repository integration. [33, OMG RFP Template]

CORBA Component Model (CCM) **-** An OMG specification for an implementation language independent distributed component model. [33, OMG RFP Template]

Interface Definition Language (IDL) **-** An OMG and ISO standard language for specifying interfaces and associated data structures. [33, OMG RFP Template]

Letter of Intent (LOI) **-** A letter submitted to the OMG BoDs Business Committee signed by an officer of an organization signifying its intent to respond to the RFP and confirming the organizations willingness to comply with OMGs terms and conditions, and commercial availability requirements. [33, OMG RFP Template]

Model Driven Architecture (MDA) **-** An approach to IT system specification that separates the specification of functionality from the specification of the implementation of that functionality on a specific technology platform. [33, OMG RFP Template]

Normative Provisions **-** To which an implementation shall conform to in order to claim compliance with the standard (as opposed to non-normative or informative material, included only to assist in understanding the standard). [33, OMG RFP Template]

Normative Reference References **-** To documents that contain provisions to which an implementation shall conform to in order to claim compliance with the standard. [33, OMG RFP Template]

Platform **-** A set of subsystems/technologies that provide a coherent set of functionality through interfaces and specified usage patterns that any subsystem that depends on the platform can use without concern for the details of how the functionality provided by the platform is implemented. [33, OMG RFP Template]

Platform Independent Model (PIM) **-** A model of a subsystem that contains no information specific to the platform, or the technology that is used to realize it. [33, OMG RFP Template]

Request for Information (RFI) **-** A general request to industry, academia, and any other interested parties to submit information about a particular technology

area to one of the OMG's Technology Committee subgroups. [33, OMG RFP Template]

Request for Proposal (RFP) - A document requesting OMG members to submit proposals to an OMG Technology Committee. [33, OMG RFP Template]

Task Force (TF) - The OMG Technology Committee subgroup responsible for issuing a RFP and evaluating submission(s). [33, OMG RFP Template]

Technology Committee (TC) - The body responsible for recommending technologies for adoption to the BoD. There are two TCs in OMG the Platform TC (PTC) focuses on IT and modeling infrastructure related standards; while the Domain TC (DTC) focuses on domain specific standards. [33, OMG RFP Template]

XML Metadata Interchange (XMI) - An OMG standard that facilitates interchange of models via XML documents. [33, OMG RFP Template]

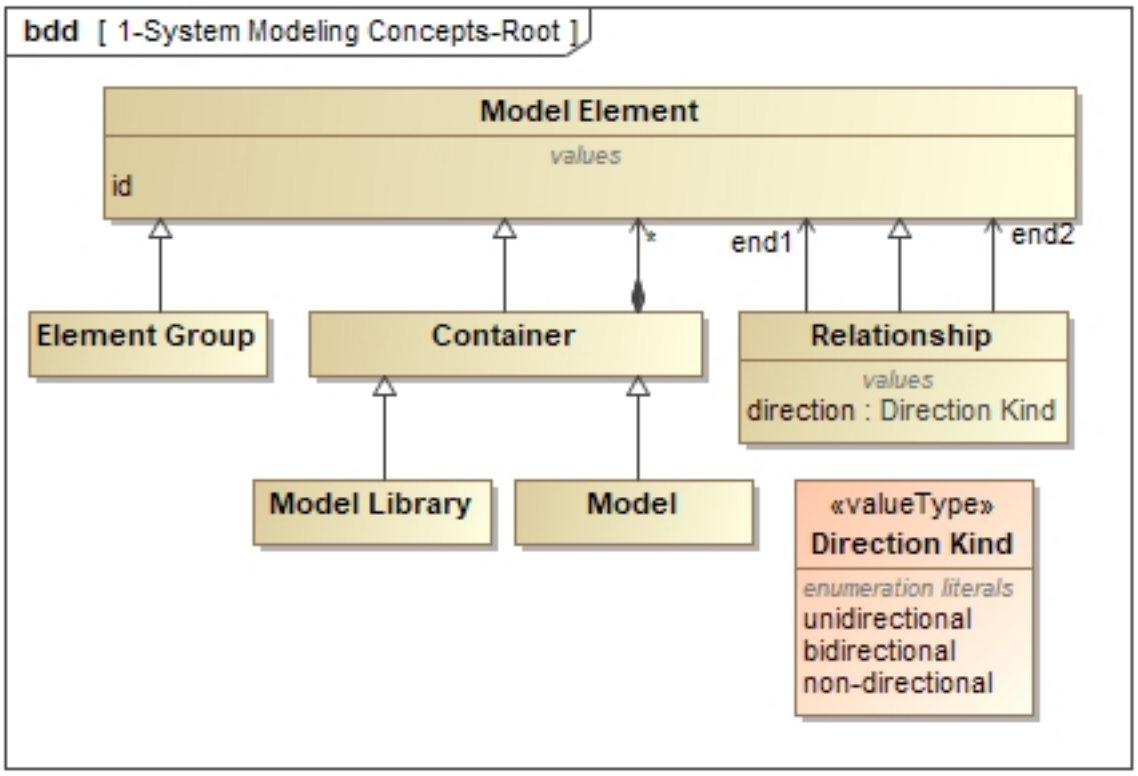
Appendix C Data Model

SysML v2 is intended to provide the capability to model systems with a precisely defined vocabulary. A concept model was used to capture the key concepts to represent systems, and help define the requirements for the SysML v2 metamodel, profile, and model libraries. *The concept model was used as part of the analysis to derive and integrate the SysML v2 requirements, but is not part of the mandatory requirements in the SysML v2 RFP*.

Although many of the concepts may be similar to those in UML and SysML v1.x, the terms are often different to avoid the implication of a particular solution. An effort is also made to apply consistent patterns to the names of the concepts. An example is the consistent naming of terms that reflect the definition and usage pattern, such as Component Definition and Component Usage, and Port Definition and Port Usage.

Root concepts. The root concepts that are reflected in the cross cutting requirements are included in Figure 8. A Model Element is the root element. A Container contains other model elements, and is analogous to a package in SysML. An Element Group is a grouping of Model Elements that establishes criteria to be a member of a group. Unlike a Container, the Element Group does not impose constraints such as deletion semantics on its members. Model and Model Library are kinds of Containers, where a Model is a top-level Container and a Model Library contains elements that are designated to be reused. Finally, the Relationship relates 2 model elements, and can be directed, non-directed, or both. All other relationships are specialized from this more general relationship.

Figure 8. Root Concepts

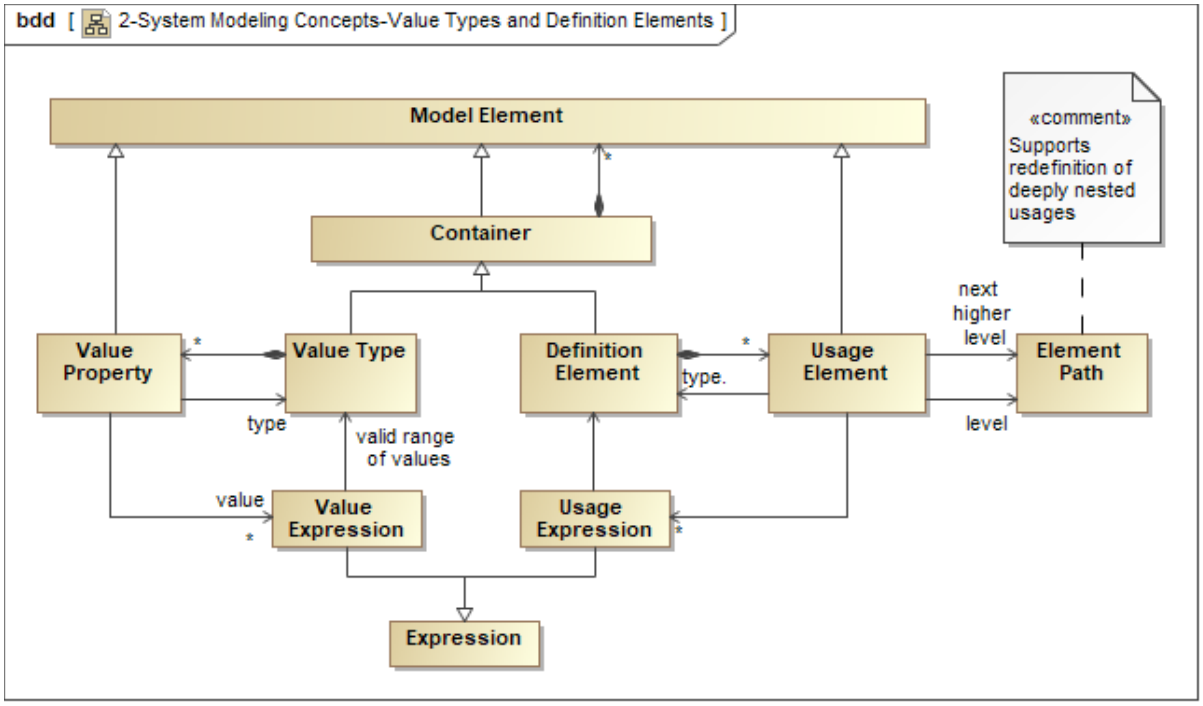


Value Type and Definition Element. The two important kinds of Containers shown in Figure 9are Value Types and Definition Elements. A Value Type is used to represent data structures with units and quantity kinds. A Value Property is typed by a Value Type to represent quantitative properties. The Value Type defines the valid range of values that a Value Property can have. A Value Expression establishes the specific value for a Value Property. A Value Type can contain other Value Properties. The kinds of Value Types have been significantly expanded beyond the primitive Value Types in SysML v1 to include vectors, collections, and other more complex data structures.

The concepts of definition and usage, such as block and part, are core concepts in SysML v1 that also apply to many of the SysML v2 language concepts. The Definition Element and Usage Element provide the ability to define a concept one time, and then reuse it in many different contexts. Usage Elements represent many concepts that are referred to as structural and behavioral features in UML and SysML. A Usage Element is typed by a Definition Element, and a Definition Element can contain other Usage Elements. An Element Path can unambiguously refer to a deeply nested usage element, and over-ride the definition for a particular localized usage (Note: analogous to SysML redefinition). This concept is further elaborated in the SECM. The Usage Expression allows the representation of specific usages and their logical

expressions such as {(Usage Element A AND Usage Element B) OR (Usage Element C AND Usage Element D)}.

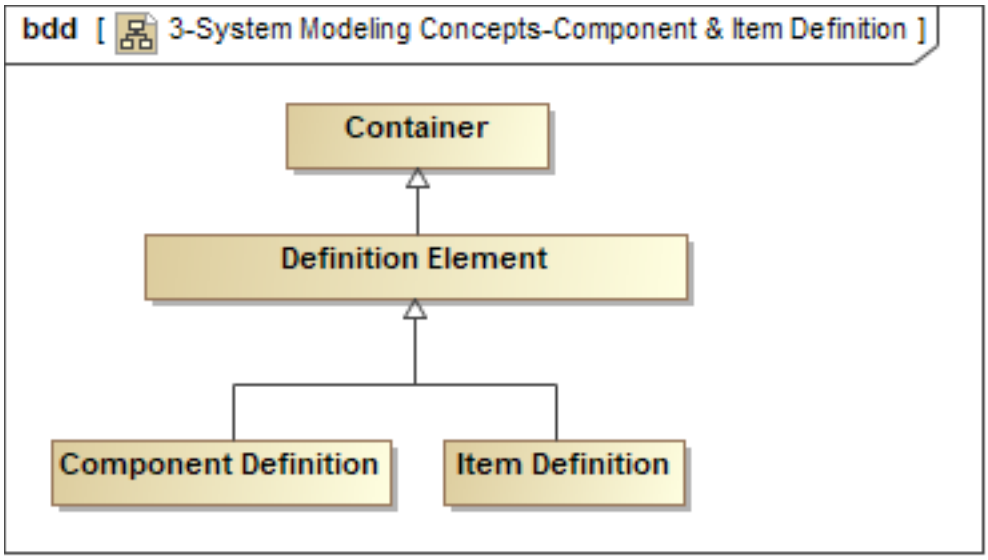
Figure 9. Value Type and Definition Element



Component Definition and Item Definition. Two particular types of Definition Elements are Component Definition and Item Definition as shown in Figure10. A Component Definition typically represents a system, subsystem, or other element that compose a system or other external entity. An Item Definition represent the kinds of things that flow through a system or between a system and other external entities. A simple example of a Component Definition is a Pump, and an example of an Item Definition is Water which can flow in and out of the Pump.

Both of these concepts can be represented in SysML v1 as a Block, which is defined as a modular unit of Structure. SysML v1 also includes more specific concepts to represent items that flow, such as flow properties and item properties. These would be referred to as item usages using the SysML v2 vocabulary.

Figure 10. Component Definition and Item Definition

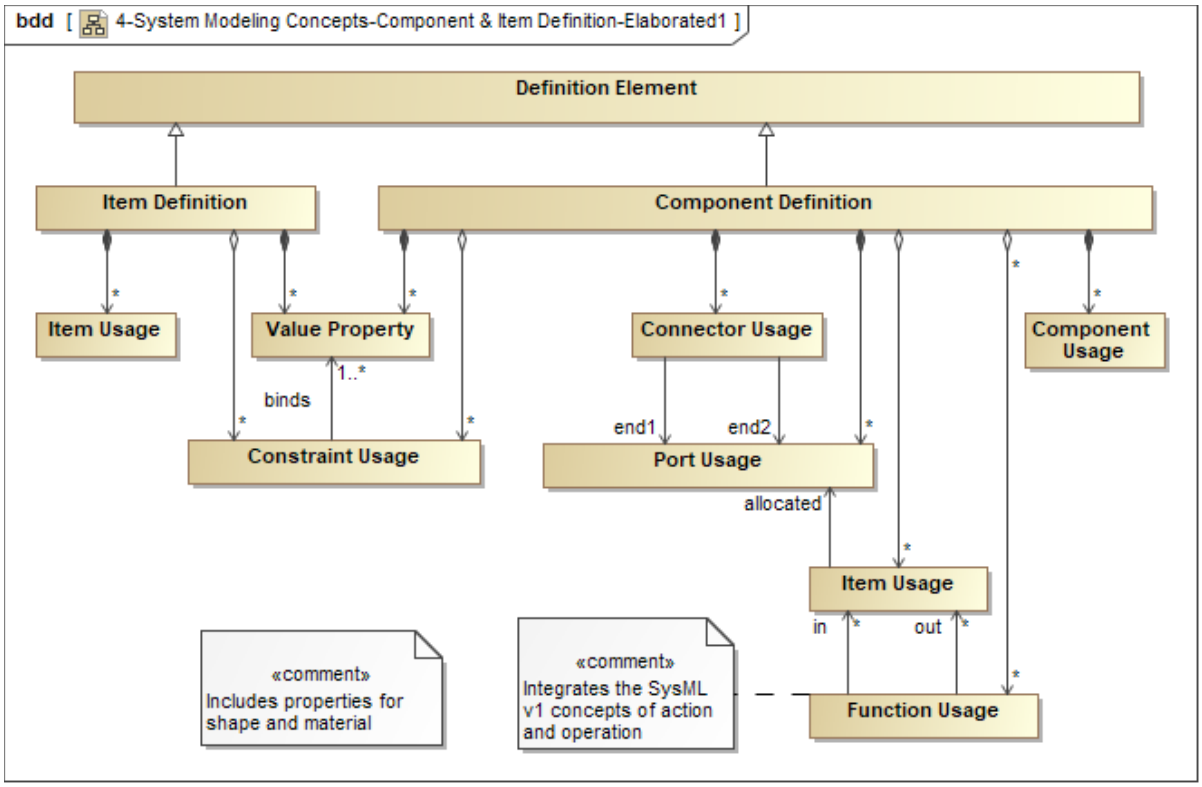


Component Definition and Item Definition-Elaborated. The Component Definition and Item Definition are further elaborated in Figure 11 to show the kinds of usage elements and value properties that they contain.

The Item Definition includes Value Properties and Constraint Usages to constrain its Value Properties, and includes Item Usages to create nested item structures. An example of a nested item structure may correspond to a message structure that includes a header and a body, which is further decomposed into specific fields that capture application data.

The Component Definition contains Value Properties and Constraint Usages, and Component Usages to define nested component structures. It also contains Port Usages and Connector Usages to connect Component Usages. Component Definition also contains Function Usages that are analogous to an Operation of a Block, but is also intended to represents an action that the Block performs to transform inputs to outputs, or to change the state of the owning Component. The input and output Item Usages are allocated to Port Usages. A Component Definition can also contain Connector Usages that connect Port Usages on it nested Component Usages. A Component Definition can also contain a shape property that specifies the simplified geometry and size of a Component in a reference coordinate system, which is intended to facilitate specification of physical envelopes.

Figure 11. Component Definition and Item Definition-Elaborated

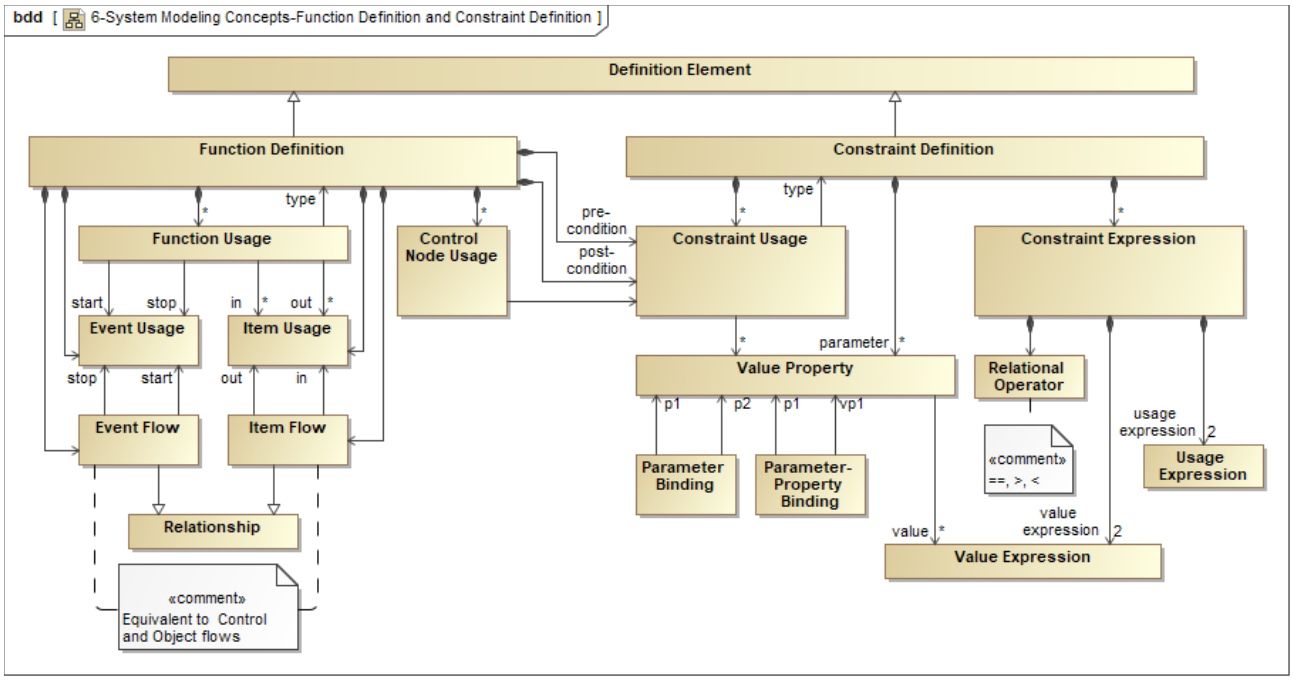


Function Definition and Constraint Definition. As noted in the Figure 11, both a Component Definition and Item Definition contain Constraint Usages, and a Component Definition can also contain Function Usages. The Constraint Definition and Function Definition are also Definition Elements as shown in Figure 12. They both use the standard pattern that enable the Definition Element to decompose into Usage Elements, and the Usage Elements are typed by a Definition Element, enabling a nested tree of usages.

The Function Definition contains Function Usages and Control Node Usages which are analogous to actions and control nodes in SysML v1. Function Usages can include both inputs and outputs (i.e., Item Usages), and start and stop events (i.e., Event Usages). An Output from one Function Usage is connected to the input of another Function Usage by an Item Flow, and the stop event of one Function Usage can be connected to the start event of another Function Usage by an Event Flow. Item Flow and Event Flow are analogous to Object Flow and Control Flow in SysML v1. Although not shown, these flows can also connect Control Node Usages that constrains the sequence of flow similar to a join specification in SysML v1. Finally, Function Definitions can include preconditions and post-conditions that must be satisfied prior to initiating and completing a function.

A Constraint Definition contains Constraint Usages and Constraint Expressions that constrain the parameters of expressions, similar to SysML v1 Constraint Blocks.

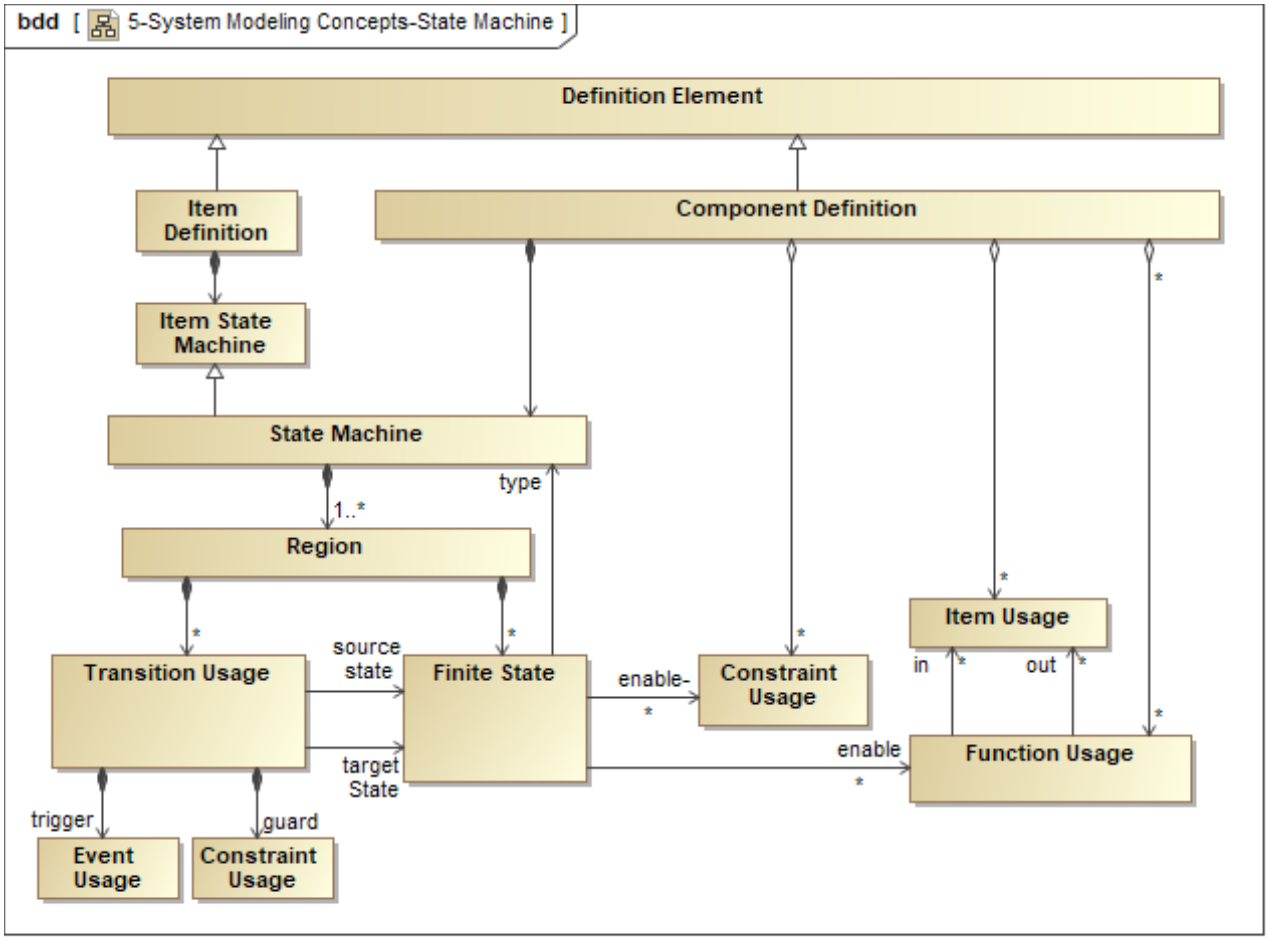
Figure 12. Function Definition and Constraint Definition



State Machine. The State Machine of a Component Definition and Item Definition specify its finite (i.e., discrete) states and the transitions between them as shown in Figure 13. It also contains Regions that enable each Region to have a single active finite state at any point in time.

The State Machine is a Definition Element which enables a Finite State to be typed by a State Machine. A Finite State can enable Constraint Usages and Function Usages in response to an event and guard condition. An Item State Machine for an Item Definition is a more generalized state machine that can define its discrete states and transitions, such as the transition between the solid, liquid, and gas state of H2O. A State Machine for an Item Definition can enable Constraint Usages, but does not enable Function Usages.

Figure 13. State Machine

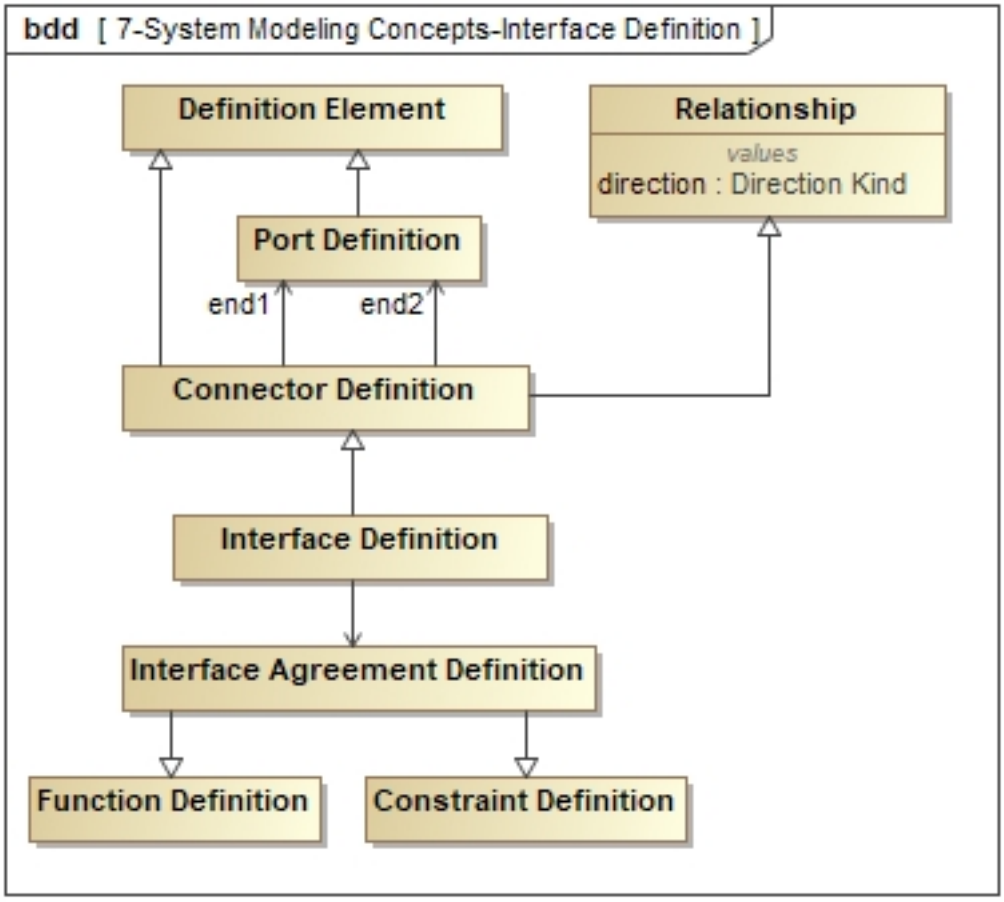


Interface Definition. An Interface definition in SysML v2 constrain the physical and functional interaction between structural elements. The Interface includes two ends, the connection between them, and the constraints on the connection. As shown in Figure 14, the Interface Definition is a subclass of a Connector Definition, which corresponds to a SysML v1 association block that can be used to type connectors.

The Connector Definition includes the definition of its ends as Port Definitions (aka Interface End Definitions), and includes an Interface Agreement which constrains the interaction across the connection. The two types of Interface Agreements include both a Function Definition and Constraint Definition. Function Definitions are generally used to constrain the exchange of Items, such as with a communication protocol, and Constraint Definitions are generally used to constrain physical interactions such as voltage and current (i.e., Across and Through Variables). Although not shown in Figure 14, a special type of component called an Interface Medium enables connection between other

components, such as a pipe, network, **or** cable. Interfaces also support nested ports and layered interfaces.

Figure 14. Interface Definition



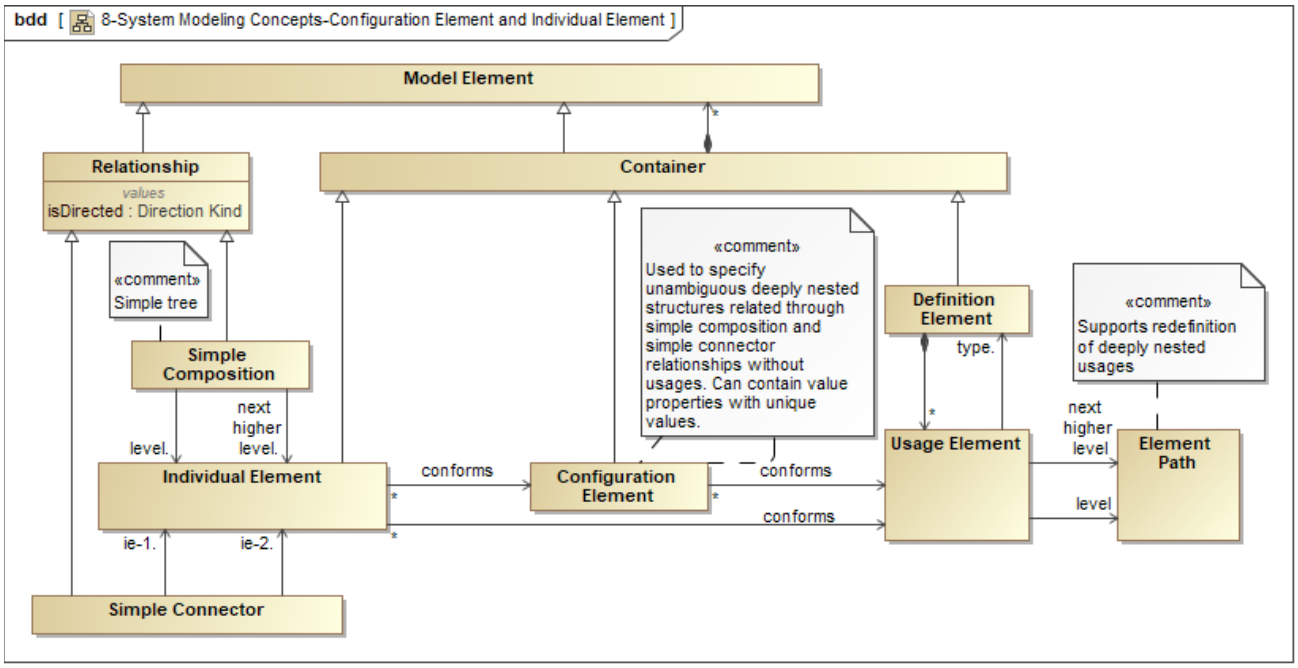
Configuration Element and Individual Element. A Definition Element can be decomposed into a tree of Usage Elements as noted previously. However, SysML v2 requires a mechanism to define an unambiguous deeply nested structure using Configuration Elements. This is intended to provide a straight forward way to specify a design configuration. A simple example is a vehicle that has 4 wheels, and each wheel has several lug bolts. The design configuration would enable the definition of an unambiguous product structure where each lug bolt on each wheel is clearly identified, and the torque value for each lug bolt can also be uniquely defined by its localized usage.

An Individual Element represents a model of a particular element that is uniquely identified, such as a model of a particular Vehicle on the factory floor with a Vehicle Identification Number (VIN). The structure of an Individual Element can be modified, such as replacing its wheels with a new kind of wheel and tire. A Simple Composition and Simple Connector is used to define a tree of Individual Elements and connect Individual Elements. The same Simple

Composition and Simple Connector can be used to compose and connect Configuration Elements.

A Configuration Element can conform to a Definition Element such as a Component Definition, and an Individual Element can conform to a Configuration Element aa shown in Figure 15. However, they are not required to conform to any particular element, which enables one to create models of Individual Elements and/or Configuration Elements independently. For example, one can model an Individual Element of the specific Vehicle on a factory floor without requiring a corresponding Configuration Element or a Definition Element.

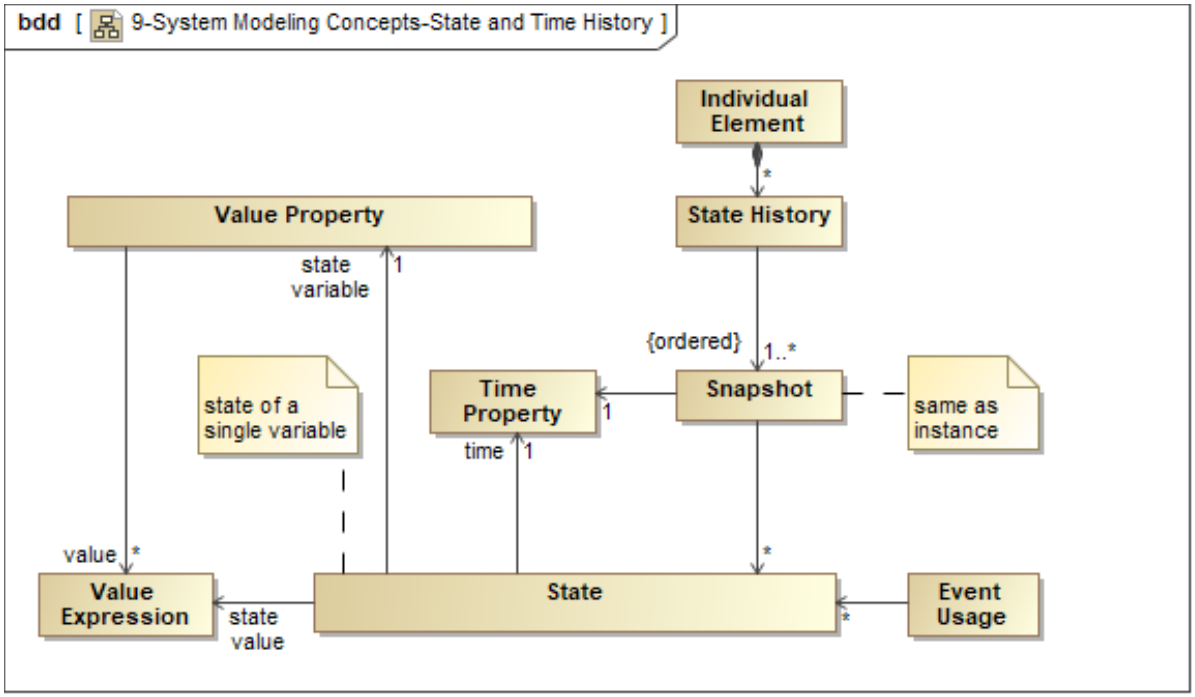
Figure 15. Configuration Element and Individual Element



State and Time History. An Individual Element can have a state history as shown in Figure 16. The state history is defined as a series of ordered Snapshots of an Individual Element, where each Snapshot represents the state of the Individual Element at a point in time. The Snapshot represents the values of each of its value properties at a particular point in time. As an example, the Snapshot of an Engine may include the values of its temperature and torque at a point in time. The value properties whose value can change over time are sometimes referred to as state variables.

Each Individual Element can contain multiple State Histories, where each State History can represent a particular estimate of its change in state over time. A State History for a Component Definition implies that each conforming Individual Element will have this State History.

Figure 16. State and Time History

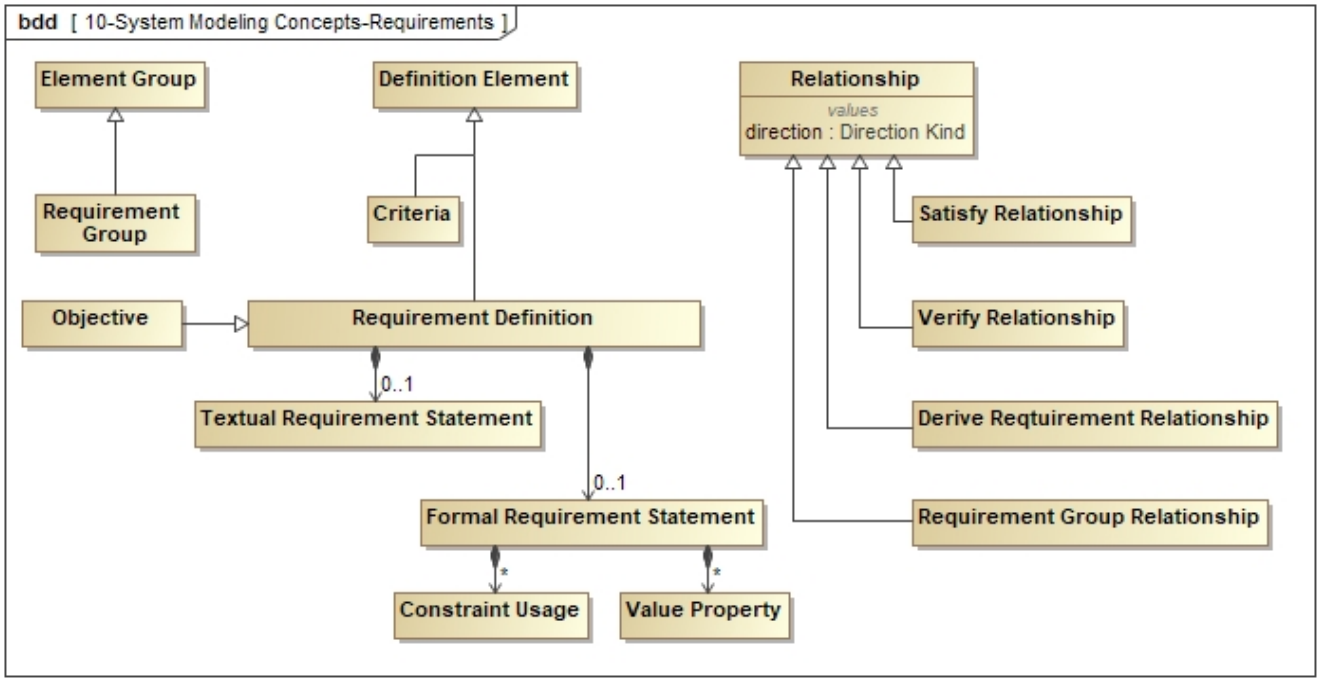


Requirements. A Requirement in SysML v2 will extend the SysML v1.5 Requirement which includes the ability to more precisely specify a requirement with a Formal Requirement Statement, in addition to a Text Requirement Statement as shown in Figure 17. The Formal Requirement Statement can be specified by constraints.

Requirements are grouped into Requirement Groups to provide context for the requirements. Requirement Groups can also contain other nested Requirements Groups that enable creation of a nested specification of requirements. Each requirement in a requirement group can be related to other elements using requirements relationships such as Satisfy, Verify, Derive, and others similar to SysML v1.

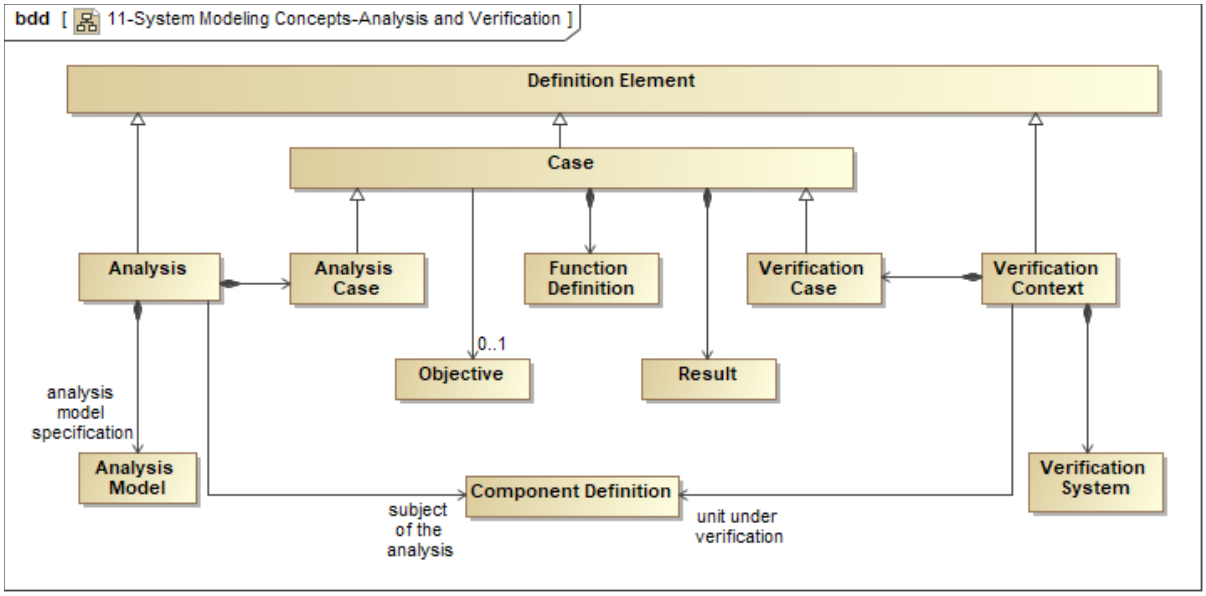
An Objective is considered a specialized requirement that reflects a desired or required end state. The Criteria define an expression that specifies the characteristics of interest and their relative weighting which can be used as a basis for an evaluation.

Figure 17. Requirements



Analysis and Verification. SysML v2 includes additional concepts to support Analysis and Verification. As shown in Figure 18, both Analysis and Verification can apply similar patterns to represent an Analysis or Verification Context that include the Component Definition, Configuration, or Individual being analyzed or verified, the analysis models or verification system used to perform the verification or analysis, and the Analysis Case or Verification Case used to define how the analysis or verification is performed. The concept of Case is a common concept that is specialized to define an Analysis Case and Verification Case.

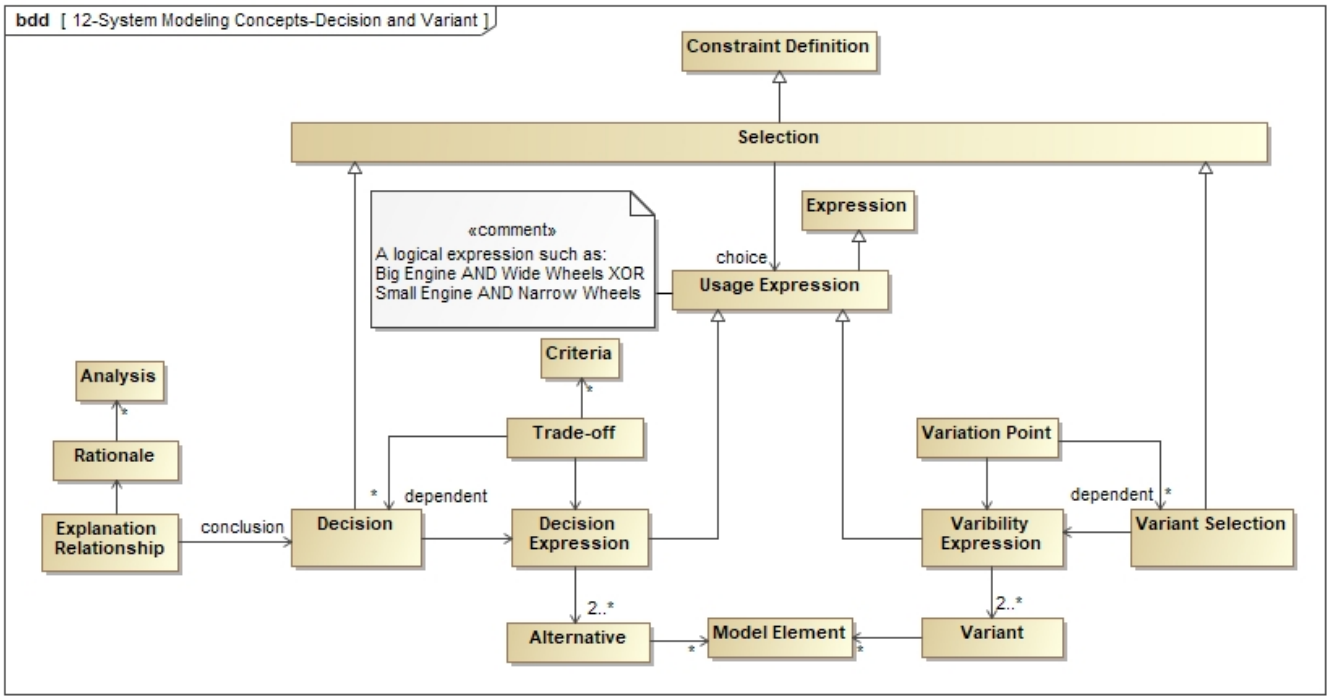
Figure 18. Analysis and Verification



Decision and Variant. SysML v2 requires additional concepts to support decisions analysis, such as trade studies, and variant modeling. Some common patterns for these concepts are noted in Figure19. In particular, both a Decision and Variant Selection involve a set of choices, called Alternative and Variant, respectively. An Expression can be used to define the choices such as A or B or C. The name for the set of choices is called a Trade-off and a Variation Point. A Selection is made among choices and called a Decision and a Variant Selection respectively. The available choices may be dependent on other Selections.

The Explanation Relationship relates the Decision to the Rationale, which in turn refers to the Supporting Analysis. The Rationale can be applied more generally to refer to the basis for any conclusion.

Figure 19. Decision and Variant



View and Viewpoint. SysML v2 includes concepts to enable the generation of Views of the system or Domain of Interest that address diverse Stakeholder Concerns. A View can represent a particular diagram, table, or complete document that is presented to Stakeholders to address their concerns. The Model is treated as a Data Source that is used to create the View.

A Viewpoint specifies the type of information and the format of the presentation that a View must provide. The View Definition defines the structure of the View in terms of its Sub-View Definitions, such as a Table of Contents for a document. The View Definition also includes methods to construct the View. The View is generated by executing the construction methods to query a particular model and present the results in a specific artifact. The concepts of View and Viewpoint are shown in Figure 20 and are intended to generally align with the proposed concepts from ISO 42010, Systems and Software Engineering, Architecture description [ArchDes].

Figure 20. View and Viewpoint