# SBVR’s Use of MOF and XMI

## General

This clause defines a transformation:

* from the *business* vocabulary text in the terminological entries in SBVR Clauses 7-21 which is created for use by the business people operating an organization
* to a *data model*, i.e. the SBVR XMI XML Schema *(see Clause 25.3)* for SBVR Content Model Exchange Documents *(see Clause 2)*.

The SBVR XMI Metamodel *(see Clause 25.2)* is a serialization of the SBVR MOF Metamodel that specifies a CMOF representation of SBVR semantics in the SBVR XMI XML Schema *(see Clause 25.3)* data model for exchanging the content of SBVR Terminological Dictionaries and Rulebooks in SBVR Content Model Exchange XML Documents.

SBVR’s use of MOF and how the SBVR MOF Metamodel handles certain semantic modeling challenges using MOF 2.n are described below. The SBVR MOF Metamodel is available in serialized form as the XML document, SBVR XMI Metamodel *(see Clause 25.2)*. The SBVR MOF Metamodel is generated by the transformation specified in this Clause from the text of the Clauses 7 through 21 terminological entries. An example SBVR Content Model that instantiates the SBVR XMI Metamodel is then shown and explained (see Clause 25.5). Finally, the SBVR Content Model for SBVR (see Clause 25.6) is explained.

Models of business concepts, business vocabularies and business guidance can be communicated in terms of SBVR semantics using XML documents, i.e. “SBVR Content Model Exchange Documents”, *(see Clause 2)* that conform to the “SBVR XMI XML Schema” (see Clause 25.3) created from the “SBVR XMI Metamodel” (see Clause 25.2). The Clause 25.4 “SBVR Content Model for SBVR” XML document is an example of an “SBVR Content Model Exchange Document” containing the terminological entry content of SBVR Clauses 7-21.

## SBVR's Use of MOF

### SBVR uses MOF as a syntax to represent SBVR semantics

###### SBVR’s use of MOF imposes no MOF or UML semantics on SBVR content

The SBVR MOF Metamodel, its SBVR XMI Metamodel serialization and the SBVR XMI XML Schema represent only SBVR semantics as defined in Clauses 7-21 with the MOF representations mapped to those definitions generally or specifically in Clause 23.3. That SBVR Terminological Dictionary and Rulebook content can be exchanged using the SBVR XMI XML Schema must not be read to imply that any MOF or UML semantics accrue to the SBVR content.

###### SBVR Terminological Dictionary and Rulebook content is not “meta”

SBVR Terminological Dictionary and Rulebook content documents the meaning of terms and other representations that business authors intend when they use them in their business communications, as evidenced in their written documentation, such as contracts, product/service specifications, and governance and regulatory compliance documents. Such documents are the authoritative source for the SBVR content.

SBVR content is not a model of anything external to people, whether physical or partially intangible such as an organization as a social system. Such content is a documentation of the *meaning*, i.e. concepts and propositions, *about* those things external to people and the *representations* used to express the *meanings* in natural language. SBVR content is best understood by analogy with business dictionaries and policy manuals.

One would never think of using a business dictionary, even if its meaning were 100% unambiguous, directly without a transformation as a data model or any kind of meta data. A dictionary simply documents the meanings people intend when they use the terms it defines to communicate. SBVR Clauses 1.4 and 1.5 are very clear on this.

###### MOF Reflection is prohibited in SBVR’s use of MOF to represent SBVR semantics

Because, by its very nature, SBVR Terminological Dictionary and Rulebook content cannot be used directly in any “meta” kind of way, the normative SBVR use of MOF prohibits the use of MOF Reflection on any Instance Model created by using the SBVR XMI Metamodel as a metamodel in any tool.

In other words, MOF models and/or UML models containing SBVR Terminological Dictionary or Rulebook content can not be imported into any tool as a metamodel and that use be claimed to be conformant with the SBVR specification. SBVR Clause 1.5 is very clear that a transformation is required to bridge from SBVR Terminological Dictionary and Rulebook content to a data model, message model, model for business information, model for reasoning over business information or any other IT system model.

###### Definition of MOF Terms

The following terms used in this clause are not words defined by SBVR. Their meanings come from MOF 2.4.2 or UML 2.4.2.



alias

association

association end

attribute

class

data type

data value

element

link

metamodel

model

package

How each of these is used with respect to SBVR is explained in Clauses 23.2 and 23.3.

### SBVR Content Models

An SBVR Content Model contains SBVR Terminological Dictionary and Rulebook content as data in an XML file that uses the SBVR XMI XML Schema as its XSD. The SBVR XMI XML Schema describes the structure and language by which the SBVR content is represented in an SBVR Content Model XML document.

SBVR Content Models contain representations of facts that document the meanings, representations and supporting information in SBVR Terminological Dictionaries and Rulebooks. For example, facts about EU-Rent’s concepts, rules, their representations and their semantic formulations can be represented in an SBVR Content Model. An example SBVR Content Model is shown in Using MOF to Represent Semantics below. SBVR Content Models are often incomplete representations of an SBVR Terminological Dictionary or Rulebook. The size of a model depends on what facts are being exchanged, which can be as little as a single fact.

One particular SBVR Content Model is the SBVR Content Model for SBVR (see Clause 25.4), which is the SBVR Terminological Dictionary for communicating about business vocabulary and business rules. It is described in Clause Example SBVR Content Model below.

An SBVR Content Model instantiates the SBVR XMI XML Schema. It is exchanged as a fact model, which combines a conceptual schema , i.e. the SBVR XMI XML Schema (Clause 25.3) and a set of facts, e.g. the SBVR Content Model for SBVR (Clause 25.4) that contains the terminological entries in Clauses 7-21.

All uses of the terms “conceptual schema” and “fact model” in this clause are as defined in Clause 24.2.2.1.

### 23.2.3 Transformation that creates SBVR machine-readable files

###### Source of the Transformation

All SBVR machine-readable files are generated from the natural language text in the SBVR terminological entries in the SBVR Vocabulary in Clauses 7-21. The SBVR Structured English styling of the SBVR terminological entries, as defined in Annex A, is used to map the content of the terminological entries to the SBVR concepts defined in Clauses 7-21.

###### Targets of the Transformation

The key target of the transformation, which is specified in Clause 23.3, is the SBVR MOF Metamodel in a UML modeling tool.

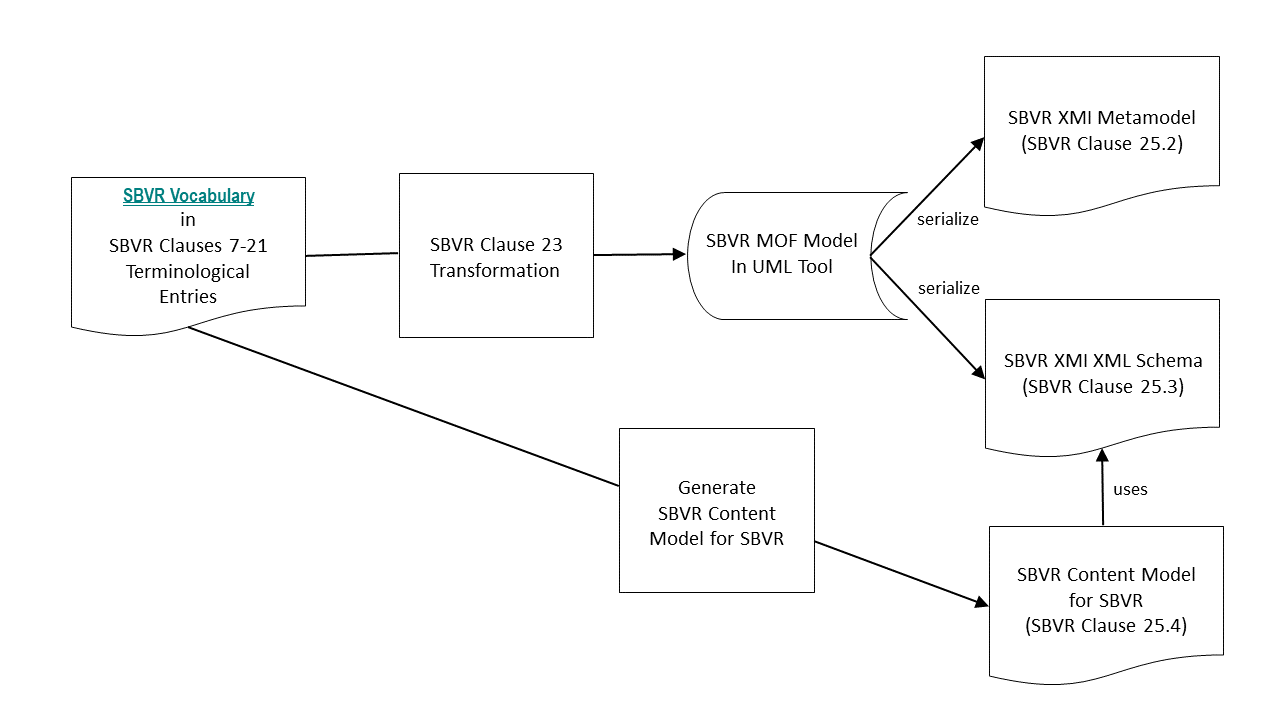
The SBVR MOF Metamodel does not include definitions, rules, notes, examples or semantic formulations. Rather, it mirrors the SBVR namespaces for the SBVR Vocabulary in Clauses 7-21. It provides a MOF means of expression (classes and associations) where the SBVR Vocabulary namespaces identify an English language means of expression (designations and verb concept wordings). Both use the same signifiers and express the same SBVR semantics. A result of this alignment of the SBVR MOF Metamodel with the SBVR Vocabulary is that the SBVR Vocabulary defines the SBVR meanings expressed by the SBVR MOF Metamodel. The SBVR MOF Metamodel is serialized as an XML document, SBVR XMI Metamodel (see 25.2).

From the SBVR MOF Metamodel in a UML modeling tool, two of the machine-readable files are generated:

* SBVR XMI Metamodel (Clause 25.2)
* SBVR XMI XML Schema (XSD) (Clause 25.3)

The SBVR Content Model for SBVR (Clause 25.4) is generated directly from the Clauses 7-21 terminological entries *(see “Source of the Transformation”)* into the XML data structure specified by the SBVR XMI XML Schema (Clause 25.3)

###### Transformation Process



**Figure 23.1 - SBVR Machine-Readable File Transformation**

## MOF Model Elements for SBVR

The SBVR Vocabulary is mapped to MOF elements that make up the SBVR MOF Metamodel. It should not be construed from this one-way mapping that a MOF class is the same thing as an SBVR concept or that there is any semantic equivalence between MOF and SBVR.

Both the mapping of the SBVR Vocabulary to MOF and the representation of SBVR model content using MOF are described below, divided using the following headings.

|  |  |
| --- | --- |
| Heading | Purpose |
| *MOF Elements of the SBVR MOF Metamodel* | Prescriptive specification of the mapping of the SBVR Vocabulary into a MOF-based metamodel |
| *Elements of SBVR Content Models* | Prescriptive specification of how facts are represented within an SBVR Content Model |
| *Rationale* | Design rationale explaining aspects of SBVR or MOF that led to the MOF representations described here |

The MOF diagram patterns shown in the Figures in Clause 23.3 are the target patterns for the SBVR content represented using the SBVR MOF Metamodel as serialized in the SBVR XMI Metamodel *(see 25.2)* that is generated from the terminological entry text in Clauses 7-21 according to the transformations specified in Clause 23 (this clause).

### MOF Packages for SBVR Vocabulary Namespaces

###### MOF Elements of the SBVR MOF Metamodel

The SBVR Vocabulary is mapped to the SBVR MOF Metamodel, which is made up of one package, which is a MOF-based representation of the SBVR Vocabulary namespace.

###### Elements of SBVR Content Models

The package that makes up the SBVR MOF Metamodel contain classes and associations.

###### Rationale

The SBVR MOF Metamodel package can be imported or merged into other MOF-based metamodels. For example, a metamodel of organizational structure can import the SBVR MOF Metamodel package as a starting point for modeling organization types and organizational roles. Similarly, a metamodel of business process can import the SBVR MOF Metamodel package in order to relate processes to rules, or for modeling semantic formulations of rules that govern processes.

### MOF Classes for SBVR Noun Concepts

###### MOF Elements of the SBVR MOF Metamodel

Each designation in a vocabulary namespace for a noun concept that is not a role is mirrored in the SBVR MOF Metamodel as a class. The signifier of the designation is the name of the class. The signifier of each synonym of the designation is an alias for the class.

The metamodel includes generalizations between classes reflecting generalizations between the represented noun concepts. Each SBVR concept besides ‘thing’ specializes ‘thing’, so the classes have the class ‘**thing**’ as a superclass either directly or indirectly.

The classes in the metamodel that mirror the following concepts are abstract (isAbstract = true):

actuality

binary logical operation

bindable target

closed semantic formulation

community

concept

expression

fact

logical formulation

logical operation

meaning

modal formulation

projecting formulation

quantification

res

semantic formulation

set

situation

state of affairs

thing

Example Vocabulary:

**characteristic**

General Concept: verb concept

Synonym: unary verb concept

Figure:



SBVR MOF Metamodel:



###### Elements of SBVR Content Models

Note that it is possible that two elements in an SBVR Content Model represent the same actual thing. Clause 24.4.1 explains situations where this is likely and tells how to relate the two elements within the SBVR Content Model. Also, a lack of an element in an SBVR Content Model implies nothing - it does not imply that something does not exist (see Clause 23.4.2 Open World Assumption).

An element of an abstract class exists in a MOF-based model only by instantiating a nonabstract subclass of that abstract class.

###### Rationale

Use of aliasing, though not common in MOF-based metamodels, keeps a strong alignment of the SBVR MOF Metamodel with the SBVR Vocabulary .

The SBVR MOF metamodel is intended to provide for representing meanings and their representations. It is not intended for representing things in general. Making some classes abstract simplifies interpretation of SBVR Content Models by limiting them to SBVR’s scope.

Each case of an objectification involves the verb concept objectified, the noun concept objectifying it along with terminological content that specifies the connection. Each of the two concepts is represented separately in the SBVR MOF Metamodel.

### MOF Boolean Attributes for SBVR Characteristics

###### MOF Elements of the SBVR MOF Metamodel

A characteristic is represented in MOF as an optional Boolean attribute as shown below.

Example Vocabulary:

variable is unitary

Figure:



SBVR MOF Metamodel:



###### Elements of SBVR Content Models

For an element in an SBVR Content Model, the meaning of the value TRUE is that the characteristic is attributed to the thing represented by the element. A meaning of FALSE is that the thing represented by the element does not have the characteristic. A meaning of the attribute being null is the same as the attribute being unspecified for the element.

###### Rationale

The attribute is optional in support of the Open World Assumption, explained in Clause 23.4.2 Open World Assumption.

### MOF Associations for SBVR Binary Verb Concepts

###### MOF Elements of the SBVR MOF Metamodel

Each binary verb concept is represented in MOF terms as an association. Association names match verb concept wordings. If a verb concept has only one verb concept wording, the association's name is the expression of that verb concept wording, but with subscripts raised to normal text. The names of the association's ends are the placeholder expressions from the verb concept wording. The ends are owned by the association so that individual links can be serialized using XMI.

In cases of more than one verb concept wording (synonymous forms), one is chosen to name the association that does not imply a designation in an attributive namespace. Then there is an alias for the association for each other verb concept wording that has matching placeholder expressions (which implies matching association end names).

Including the names of an association's ends in the association’s name makes the association's name unique within a package, as required by MOF.

In cases where an association’s ends both connect to the same class, subscripts are used on placeholders to distinguish them. In the association name and its ends’ names the subscripts are raised to normal text and serve to distinguish the ends.

Example Vocabulary:

concept1 specializes concept2

Synonymous Form: concept2 generalizes concept1

Figure:



SBVR MOF Metamodel:



Some definitional rules impose multiplicity constraints for binary verb concepts. These are formally-stated restrictions on multiplicity in a definition or a definitional rule in the SBVR terminological entries (Clauses 7-21) and are included in the SBVR MOF Metamodel. If no multiplicity restriction is specified in the SBVR terminological entries, the implied multiplicity of “no minimum and/or maximum restriction” in the SBVR terminological entries is made explicit in the SBVR MOF Metamodel using the multiplicity “0..\*” .

Annex A.2.1.1 documents the SBVR Structured English keywords that are used for the expression of multiplicities in SBVR terminological entries. The following table shows those keywords along with the corresponding multiplicity in the SBVR MOF Metamodel.

|  |  |  |
| --- | --- | --- |
| ***SBVR Structured English Keyword*** | ***Kind of Logical Formulation*** | ***UML Multiplicity*** |
| at least one | existential quantification | 1..\* |
| at least *n* | at-least-n quantification | *n..\** |
| at most one | at-most-one quantification | 0..1 |
| at most *n* | at-most-n quantification | 0..*n* |
| exactly one | exactly-one quantification | 1..1 |
| exactly *n* | exactly-n quantification | *n..n* |
| at least *n* and at most *m* | numeric range quantification | *n..m* |
| more than one | at-least-n quantification with *n* = 2 | 2..\* |

###### Elements of SBVR Content Models

Where an association represents a binary verb concept, a link of the association within an SBVR Content Model represents a fact of that binary verb concept. The absence of a link implies nothing. There are no defaults.

###### Rationale

All association ends in the SBVR MOF metamodel are non-composite.

### MOF Attributes for SBVR Roles of Verb Concepts

###### MOF Elements of the SBVR MOF Metamodel

A role of a binary verb concept that has a designation in an attributive namespace is understood in MOF terms as an attribute owned by the subject class. Such designations appear as names on association ends. In the example below, ‘element’ is in an attributive namespace for the concept ‘set,’ so it is mirrored in the SBVR MOF Metamodel as an attribute.

Example Vocabulary:

thing is in set

Synonymous Form: set includes thing

Synonymous Form: set has element

Figure:



SBVR MOF Metamodel:



In each case where an attribute and an association end represent the same role, the SBVR MOF Metamodel includes a tag that tags both the attribute and the association end. The tag connects them to show their correlation. The tag’s name is “org.omg.sbvr.sameRole,” its value is "" (the empty string), and its elements are the attribute and the association end.

Where structural rules impose multiplicity constraints, they are included in the SBVR MOF Metamodel for association ends and for attributes.

###### Elements of SBVR Content Models

Where a role of a binary verb concept is understood in MOF terms as an attribute, specification of the attribute for an element in an SBVR Content Model represents the entire extension of that verb concept for the element. There are no defaults. If the attribute is unspecified for an element, it is simply unspecified; it is not presumed by default to have no value. If anything is specified, all values of the attribute are specified. Specification that the attribute is null means there is no instance of the verb concept for the element.

###### Rationale

The attributes described in the clause are in addition to the associations that represent the binary verb concepts - the reason for the distinction is explained below.

To preserve ‘set’ semantics, any two values of the same attribute of the same element in an SBVR Content Model represent two different things. Where an attribute has two or more values, it can be concluded that each of the values represents a thing that is distinct from the others.

### MOF Classes for SBVR Ternary Verb Concepts

###### MOF Elements of the SBVR MOF Metamodel

Each ternary verb concept is represented in MOF terms as a class with one single-valued, required attribute for each role of the verb concept. The class’s name takes the same form as the name of an association for a binary verb concept. If there are multiple verb concept wordings for a ternary verb concept, aliases are used.

Example Vocabulary:

**state of affairs involves thing in role**

Figure:



SBVR MOF Metamodel:



###### Elements of SBVR Content Models

In an SBVR Content Model, an element of such a class represents a fact of the ternary verb concept.

### Data Values

###### MOF Elements of the SBVR MOF Metamodel

The classes ‘text’ and ‘integer,’ representing ‘text’ and ‘integer,’ have data attributes shown below.

SBVR MOF Metamodel:



###### Elements of SBVR Content Models

If one of these attributes is specified in an SBVR Content Model, the represented text or integer is the specified value. Specification of null is equivalent to not specifying anything. There are no defaults.

The concepts ‘text’, ‘integer’, and ‘number’ are SBVR noun concepts, so their instances can be represented like instances of other noun concepts (see 23.2.2 MOF classes for SBVR Noun Concepts) without using the ‘value’ attributes shown above. A specific number can be identified by a designation. The ISO 6093 Number Namespace includes designations of all integers and of numbers with decimal places. Each designation in the ISO 6093 Number Namespace shall be interpreted according to [ISO 6093].

Each text value is a Unicode string of undefined length and is considered without regard to markup.

###### Rationale

The attributes are optional because SBVR allows that texts and integers, like other kinds of things, can be described by facts without necessarily being identified. Also, the data types ‘String’ and ‘Integer’ in MOF have size limitations, so the attributes cannot be used for all cases. To refer to a string or integer that is beyond the MOF limitations, a model can identify the string or integer using facts about it that satisfy a reference scheme. For example, the number 999999999999 can be identified as having a designation in the ISO 6093 Number Namespace with the signifier “999999999999”.

### XMI Names

###### MOF Elements of the SBVR MOF Metamodel

A named element is tagged with an ‘org.omg.xmi.xmiName’ tag if its XMI name differs from its MOF name. XMI names are determined from MOF names by upcasing each character that follows a blank and then removing the blank. The names, because they come from the SBVR vocabularies, do not contain any characters that are invalid in XML identifiers.

## Using MOF to Represent Semantics

The SBVR MOF Metamodel is a direct representation of the meanings in the SBVR Vocabulary, which also represents SBVR meanings, but this direct representation of SBVR meanings requires two semantic modeling capabilities not directly provided by MOF 2.n. The two following clauses explain how the two capabilities, multiclassification and the Open World Assumption, are supported by the SBVR MOF Metamodel.

### Multiclassification

MOF 2.n requires that each element is described by one class (its “metaClass”). Sometimes a thing cannot be represented by an element of a single class. This happens when a thing is an instance of multiple concepts, neither one specializing the other. To represent this case, multiple elements are used, one per concept. A link of the association ‘**thing1 is thing2**’ (representing the verb concept ‘thing1 is thing2’) is used to indicate that the multiple elements represent the same thing. A consumer of a model in which two elements represent the same thing should assume that a fact represented in reference to either element applies to both elements (since they both represent the same thing).

As an example, consider the noun concepts ‘closed logical formulation’ and ‘obligation formulation.’ Neither specializes the other. Where an obligation formulation is a closed formulation that formulates a proposition, a model uses one element of type ‘**closed logical formulation**’ and a separate element of type ‘**obligation formulation**’ along with a ‘**thing1 is thing2**’ link that says the two elements represent the same thing.

### Open World Assumption

See Clause 24.2.1.3 Open/Closed World Semantics for the SBVR interpretation of this topic in formal logic. This clause specifies that SBVR Content Model exchange documents always have an Open World Assumption.

The open world assumption is that representation of facts in a model does not imply that those are the only facts of a particular type nor that they are the only facts of a particular type about a subject thing - there are no implications to be taken from what is not represented in a model. For example, consider facts about a set S. The two facts, “1 is in S” and “2 is in S,” do not convey the same meaning as “S = {1, 2}” because the two facts do not imply anything about whether other things are in S.

In general, SBVR Content Models represent facts with an open world assumption. But some reference schemes use roles of binary verb concepts extensionally (see Clause 11.3 reference scheme *extensionally uses*  verb concept role), so models represent a complete extension with respect to a subject thing being identified.

MOF supports the open world assumption about instantiation of classifiers (classes and associations). MOF’s attributes support representation of an entire extension of an attribute with respect to a given subject. In order to enable a clear distinction in a model between individual facts and complete extensions with respect to a subject, association links are used to represent individual facts of a binary verb concept while attributes are used when identifying a complete extension of a binary verb concept with respect to a particular subject. This means that a fact can in one model be represented by a link, and in another by a value of an attribute of an element. The fact is represented using an attribute only when the complete extension of the verb concept is being represented for the subject. Examples of both cases appear in the example below. SBVR has a designation in an attributive namespace for every role that is extensionally used by a reference scheme such that the SBVR MOF Metamodel has the required attributes to satisfy all of SBVR’s reference schemes.

## Example SBVR Content Model

Consider the following example, which includes a small portion of a vocabulary and a rule statement.

company

officer

company appoints officer

EU-Rent

General Concept: company

EU-Rent must appoint at least 3 officers.

The following figure is a UML instance diagram showing an SBVR Content Model of the example. Some end names are elided where they are obvious from the class names or for ‘**thing1 is thing2**’ (where it makes no difference). For elements of the vocabulary, the three layers of expression, representation, and meaning are apparent in the diagram. The rule, shown at the bottom, connects to the meanings of the elements of the vocabulary though its logical formulation.



The example SBVR Content Model is expressed below in XML based on the SBVR XMI XML Schema. The xmi:id values are arbitrary and have no special meaning, but they build on the related signifiers to help readability. The XML tags, which include the namespace prefix ‘sbvr’, are the XMI names for model elements of the SBVR XMI Metamodel.

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

<xmi:XMI xmi:version="2.1" xmlns:xmi="http://schema.omg.org/spec/XMI/2.1"   
 xmlns:sbvr="http://www.omg.org/spec/SBVR20190601/SBVR.xml">

For ‘company’:

<sbvr:designation xmi:id="company" signifier="company-t" meaning="company-c"/>

<sbvr:generalConcept xmi:id="company-c"/>

<sbvr:text xmi:id="company-t" value="company"/>

For ‘officer’:

<sbvr:designation xmi:id="officer" signifier="officer-t" meaning="officer-c"/>

<sbvr:generalConcept xmi:id="officer-c"/>

<sbvr:text xmi:id="officer-t" value="officer"/>

For ‘company appoints officer’:

<sbvr:sententialForm xmi:id="companyAppointsOfficer" expression="cao-t" meaning="cao-c" placeholder="cao-p1 cao-p2"/>

<sbvr:binaryVerbConcept xmi:id="cao-c" role="cao-r1 cao-r2"/>

<sbvr:verbConceptWordingIncorporatesVerbSymbol verbConceptWording="companyAppointsOfficer" verbSymbol="appoints"/>

<sbvr:designation xmi:id="appoints" signifier="appoints-t" meaning="cao-c"/>

<sbvr:text xmi:id="cao-t" value="company appoints officer"/>

<sbvr:text xmi:id="appoints-t" value="appoints"/>

<sbvr:placeholder xmi:id="cao-p1" expression="company-t" startingCharacterPosition="i1" meaning="cao-r1"/>

<sbvr:placeholderUsesDesignation placeholder="cao-p1" designation="company"/>

<sbvr:roleRangesOverObjectType role="cao-r1" generalConcept="company-c"/>

<sbvr:verbConceptRole xmi:id="cao-r1"/>

<sbvr:positiveInteger xmi:id="i1" value="1"/>

<sbvr:placeholder xmi:id="cao-p2" expression="officer-t" startingCharacterPosition="i18" meaning="cao-r2"/>

<sbvr:placeholderUsesDesignation placeholder="cao-p2" designation="officer"/>

<sbvr:roleRangesOverObjectType role="cao-r2" generalConcept="officer-c"/>

<sbvr:verbConceptRole xmi:id="cao-r2"/>

<sbvr:positiveInteger xmi:id="i18" value="18"/>

For ‘EU-Rent’ with “General Concept: company”:

<sbvr:designation xmi:id="EU-Rent" signifier="EU-Rent-t" meaning="EU-Rent-c"/>

<sbvr:individualConcept xmi:id="EU-Rent-c"/>

<sbvr:text xmi:id="EU-Rent-t" value="EU-Rent"/>

<sbvr:concept1SpecializesConcept2 concept1="EU-Rent-c" concept2="company-c"/>

For “EU-Rent must appoint at least 3 officers”:

<sbvr:statement xmi:id="stmt" expression="stmt-t" meaning="stmt-p"/>

<sbvr:text xmi:id="stmt-t" value="EU-Rent must appoint at least 3 officers"./>

<sbvr:proposition xmi:id="stmt-p"/>

<sbvr:closedLogicalFormulationFormalizesStatement closedLogicalFormulation="ob2" statement="stmt"/>

<sbvr:closedLogicalFormulationMeansProposition closedLogicalFormulation="ob2" proposition="stmt-p"/>

<sbvr:obligationFormulation xmi:id="ob"/>

<sbvr:closedLogicalFormulation xmi:id="ob2"/>

<sbvr:thing1IsThing2 thing1="ob" thing2="ob2"/>

<sbvr:modalFormulationEmbedsLogicalFormulation modalFormulation="ob" logicalFormulation="am3"/>

<sbvr:at-least-nQuantification xmi:id="am3" scopeFormulation="atom" minimumCardinality="i3"/>

<sbvr:quantificationIntroducesVariable quantification="am3" variable="v"/>

<sbvr:variable xmi:id="v" ranged-overConcept="officer-c" restrictingFormulation="" isUnitary="false"/>

<sbvr:atomicFormulation xmi:id="atom" roleBinding="bind1 bind2"/>

<sbvr:atomicFormulationIsBasedOnverbConcept atomicFormulation="atom" verbConcept="cao-c"/>

<sbvr:roleBinding xmi:id="bind1"/>

<sbvr:roleBindingBindsToBindableTarget roleBinding="bind1" bindableTarget="EU-Rent-c"/>

<sbvr:verbConceptRoleHasRoleBinding verbConceptRole="cao-r1" roleBinding="bind1"/>

<sbvr:roleBinding xmi:id="bind2"/>

<sbvr:roleBindingBindsToBindableTarget roleBinding="bind2" bindableTarget="v"/>

<sbvr:verbConceptRoleHasRoleBinding verbConceptRole="cao-r2" roleBinding="bind2"/>

<sbvr:positiveInteger xmi:id="i3" value="3"/>

</xmi:XMI>

The example shows some of the points explained previously about SBVR Content Models.

* Fact Model - the entire XML content represents a fact model, which is a combination of a conceptual schema and a set of facts. The conceptual schema of the fact model is identified in the heading where it says, xmlns:sbvr=”http://www.omg.org/spec/SBVR20190601/SBVR.xml.” The URL identifies a document that serializes the SBVR Content Model for SBVR, which describes the concepts and rules that make up the conceptual schema (see 23.4 and 25.4). The elements of the XML content represent the set of facts of the fact model.
* Multiclassification - There is an occurrence of ‘thing1IsThing2’ which is used to connect a pair of elements that represent the same thing. There is an element of type ‘obligationFormulation’ (xmi:id="ob") and another element of type ‘closedLogicalFormulation’ (xmi:id="ob2"). Neither type specializes the other so there is one element of each type and a ‘thing1IsThing2’ link indicates that the two elements represent the same thing.
* Open World Assumption - Links, rather than attributes, are always used where there is an open world assumption, such as for the fact that the individual noun concept ‘EU-Rent’ specializes the concept ‘company’ - there is no indication that these concepts are not involved in other specializations.
* Attributes giving Complete Extensions for a Subject - Each specification of an attribute occurs where the entire extension of the attribute is being specified for a subject thing, such as for identifying the two placeholders of the verb concept wording ‘company appoints officer’ or the two roles of the verb concept. The one ‘variable’ in the example is serialized with ‘restrictingFormulation=””’ representing that it has no restricting formulation. In a number of cases, attributes are unspecified because the entire extension of the attribute for an element is not being specified. For example, the attribute ‘representation’ is unspecified for the elements representing meanings (e.g., ‘company-c’ and ‘officer-c’ - there can be any number of representations of a meaning, and the example model does not specify them all. However, each representation has exactly one meaning, so the ‘meaning’ attribute is specified for each representation to identify its one meaning.

## The SBVR Content Model for SBVR

The SBVR Content Model for SBVR represents facts concerning all of the formally captioned contents of Clauses 7 through 12. In general, this includes all of the information given in the SBVR specification about its concepts that can be represented in terms of the SBVR MOF Metamodel. This includes:

* noun concepts and their designations
* verb concepts and their verb concept wordings
* specializations/generalizations
* concept types
* definitions and, where formal, their semantic formulations
* necessity statements and, where formal, their semantic formulations
* vocabularies, language, namespaces and their URIs
* notes, examples, sources, descriptions

The SBVR Content Model for SBVR is like the example in Clause 23.3 above except that it is about SBVR’s vocabulary and meanings, not EU-Rent’s. The complete SBVR Content Model for SBVR is serialized as XML documents listed in Clause 25.4. It can be used and extended by other SBVR Content Models that build on SBVR’s concepts.

## XMI for the SBVR Model of SBVR

XML patterns are shown below for the various parts of vocabulary descriptions and vocabulary entries used in Clauses 7 through 12. These patterns are used to create the XML documents that serialize the SBVR Content Model for SBVR. Each pattern is shown for a corresponding SBVR Structured English entry (see Annex A for entry descriptions).

The XML patterns provide a normative definition of which SBVR concepts are represented by each use of SBVR Structured English in the vocabulary descriptions and entries contained in Clauses 7 through 21.

The general principles used for the patterns are these: First, the facts of what is presented using SBVR Structured English are represented using XML. Second, for the objects referenced by those facts, further facts are represented to satisfy reference schemes for those objects wherever sufficient detail is given. The principles are applicable to SBVR-based communication in general. The XML files specified in Clause 25, which are created based on these principles following the patterns below, are examples of XML serializations of SBVR Content Models.

The xmi:id values used in the patterns below are replaced by different values in the actual XML documents because the multitude of repetitions of the patterns need their own unique xmi:id values. But the xmi:id values shown below consistently and correctly show relationships within the patterns. Most xmi:id values are referenced only locally within the XML elements for the same Structured English entry, but some are referenced beyond that scope and are shown in bold blue (e.g., "**vocabulary**") so that references to them are easily followed. The different types of vocabulary entries (term, name and verb concept wording) are mutually exclusive. They each introduce an xmi:id value "**meaning**" which is referenced in other patterns.

Made-up names (e.g., “Xyz Vocabulary”), terms (e.g., “example term”) and verb concept wordings (e.g., “example *is seen*”) are used to show the patterns and to show how signifiers and other expressions appear in XML. Certain assumptions are made by the patterns based on the way the vocabularies in Clauses 7 through 12 are interrelated. The patterns assume that a vocabulary being described has a name in the Vocabulary Registration Vocabulary (of Clause 7). The patterns assume that where a term or name is used with a formal interpretation in SBVR Structured English, that term or name is found by way of the vocabulary namespace derived from the vocabulary being described. These assumptions are correct regarding Clauses 7 through 12, but they cannot necessarily be assumed about all vocabulary descriptions.

Each pattern has a part that remains unchanged for the kind of entry or caption shown (except for differences in xmi:id values as described above) and a part that varies based on the content of the entry. The part that varies is shown in ***bold italics***. It can be a text or integer value, a quoted xmi:id of an object introduced elsewhere, or an XML tag.

The final XML documents created from the vocabulary clauses can differ slightly from what is exactly produced from the templates, but the represented meaning does not differ. In cases where two objects are created and then connected by a ‘thing1IsThing2’ link, the objects can be combined into one if they are of the same class or if one class specializes the other. In cases where the patterns would create two identical XML elements, only one is actually created. For example, all uses of an element for the integer 1 can use the same element.

### XML Patterns for Vocabularies

**Xyz Vocabulary**

<sbvr:vocabulary xmi:id="**vocabulary**"/>

<sbvr:nameReferencesThing thing="vocabulary" name="XyzVocabulary"/>

<sbvr:name xmi:id="XyzVocabulary" signifier="v-s" meaning="vocabulary-concept"/>

<sbvr:individualConcept xmi:id="**vocabulary-concept**" instance="vocabulary"/>

<sbvr:text xmi:id="v-s" value="***Xyz Vocabulary***"/>

<sbvr:designationIsInNamespace designation="XyzVocabulary" namespace="vocabularyRegistrationNamespace"/>

<sbvr:vocabularyNamespace xmi:id="**vocabularyNamespace**"/>

<sbvr:vocabularyNamespaceIsDerivedFromVocabulary vocabularyNamespace="vocabularyNamespace" vocabulary="vocabulary"/>

The pattern above assumes the Vocabulary Registration Vocabulary has a vocabulary namespace like this:

<sbvr:vocabularyNamespace xmi:id="**vocabularyRegistrationNamespace**"/>

Included Vocabulary: Abc Vocabulary

<sbvr:vocabulary1IncorporatesVocabulary2 vocabulary1="vocabulary" vocabulary2="***Abc***"/>

<sbvr:namespace1IncorporatesNamespace2 namespace1="vocabularyNamespace" namespace2="*Abc-ns*"/>

The pattern above assumes there is a vocabulary named Abc Vocabulary like this:

<sbvr:vocabulary xmi:id="Abc"/>

<sbvr:vocabularyNamespace xmi:id="Abc-ns"/>

Language: English

<sbvr:language xmi:id="language"/>

<sbvr:vocabularyNamespaceIsForLanguage vocabularyNamespace="vocabularyNamespace" language="language"/>

<sbvr:nameReferencesThing thing="language" name="English"/>

<sbvr:name xmi:id="English" signifier="l-s" meaning="l-c"/>

<sbvr:individualConcept xmi:id="l-c" instance="language"/>

<sbvr:text xmi:id="l-s" value="***English***"/>

<sbvr:designationIsInNamespace designation="English" namespace="ISO639-2English"/>

<sbvr:vocabularyNamespace xmi:id="ISO639-2English"/>

<sbvr:namespaceHasURI namespace="ISO639-2English" URI="lm-u"/>

<sbvr:URI xmi:id="lm-u"  
 value="http://www.loc.gov/standards/iso639-2/php/English\_list.php"/>

Namespace URI: http://some.uri

<sbvr:namespaceHasURI namespace="vocabularyNamespace" URI="vn-uri"/>

<sbvr:URI xmi:id="vn-uri" value="***http://some.uri***"/>

Speech Community: English Mechanics

<sbvr:speechCommunityOwnsVocabulary speechCommunity="em" vocabulary="vocabulary"/>

<sbvr:conceptHasInstance concept="***em-concept***" instance="em"/>

<sbvr:speechCommunity xmi:id="em"/>

It is assumed for this entry that there is a name ‘English Mechanics’ for an individual noun concept like this:

<sbvr:name xmi:id="em-name" signifier="em-s" meaning="em-concept"/>

<sbvr:individualConcept xmi:id="em-concept"/>

<sbvr:text xmi:id="em-s" value="English Mechanics"/>

The captions “Description:”, “Note:” and “Source:” are handled for a vocabulary in the same way as for terms within a vocabulary, as shown below, except that the related meaning is given as meaning="vocabulary-concept".

### XML Patterns for General Concepts

example term

<sbvr:term xmi:id="exampleTerm" signifier="et-s" meaning="meaning"/>

<sbvr:generalConcept xmi:id="**meaning**"/>

<sbvr:text xmi:id="et-s" value="***example term***"/>

<sbvr:thingIsInSet set="vocabulary" thing="exampleTerm"/>

<sbvr:designationIsInNamespace designation="exampleTerm" namespace="vocabularyNamespace"/>

If there is no “See:” caption, then the following is included:

<sbvr:preferredDesignation xmi:id="exampleTermPreferred"/>

<sbvr:thing1IsThing2 thing1="exampleTermPreferred" thing2="exampleTerm"/>

Concept Type: role

<sbvr:*role* xmi:id="meaningAsRole"/>

<sbvr:thing1IsThing2 thing1="meaningAsRole" thing2="meaning"/>

The pattern above is used if the concept type is an SBVR concept. The pattern below is used if the concept type is  
 not an SBVR concept.

Concept Type: example type

<sbvr:conceptHasInstance concept="***exampleType-c***" instance="meaning"/>

There is assumed to be a term ‘example type’ for a general concept like this:

<sbvr:term xmi:id="exampleType" signifier="exampleType-s" meaning="exampleType-c"/>

<sbvr:generalConcept xmi:id="exampleType-c"/>

<sbvr:text xmi:id="exampleType-s" value="example type"/>

Definition: example that is seen

<sbvr:definition xmi:id="def-formal" expression="def-formal-e" meaning="meaning"/>

<sbvr:text xmi:id="def-formal-e" value="***example that is seen***"/>

<sbvr:concept1SpecializesConcept2 concept1="meaning" concept2="***example-concept***" />

<sbvr:closedProjectionFormalizesDefinition closedProjection="***def-formal-projection***" definition="def-formal"/>

<sbvr:closedProjectionDefinesNounConcept closedProjection="***def-formal-projection*"** nounConcept="meaning"/>

The closed projection of the definition (not shown) has xmi:id="def-formal-projection". It is assumed for this entry   
 and several others that there is a term ‘example’ for a general concept like this:

<sbvr:term xmi:id="example" signifier="example-s" meaning="example-concept"/>

<sbvr:generalConcept xmi:id="**example-concept**"/>

<sbvr:text xmi:id="**example-s**" value="example"/>

Definition: example that shows something

<sbvr:definition xmi:id="def-semiformal" expression="def-semiformal-e" meaning="meaning"/>

<sbvr:text xmi:id="def-semiformal-e" value="***example that shows something***"/>

<sbvr:concept1SpecializesConcept2 concept1="meaning" concept2="***example-concept***" />

Definition: whatever demonstrates

<sbvr:definition xmi:id="def-informal" expression="def-informal-e" meaning="meaning"/>

<sbvr:text xmi:id="def-informal-e" value="***whatever demonstrates***"/>

Description: A description of something

<sbvr:descriptionPortraysMeaning description="desc" meaning="meaning"/>

<sbvr:description xmi:id="desc" expression="desc-e"/>

<sbvr:text xmi:id="desc-e" value="***A description of something***"./>

Example: An example of an example

<sbvr:descriptiveExampleIllustratesMeaning descriptiveExample="de" meaning="meaning"/>

<sbvr:descriptiveExample xmi:id="de" expression="de-e"/>

<sbvr:text xmi:id="de-e" value="***An example of an example***"/>

General Concept: example

<sbvr:concept1SpecializesConcept2 concept1="meaning" concept2="***example-concept***" />

Necessity: Each example is seen.

<sbvr:statement xmi:id="nec-stmt" expression="nec-e" meaning="nec"/>

<sbvr:text xmi:id="nec-e" value="***Each example is seen***"./>

<sbvr:proposition xmi:id="nec" isNecessarilyTrue="true"/>

<sbvr:closedLogicalFormulationFormalizesStatement closedLogicalFormulation="***nec-formulation***" statement="nec-stmt"/>

<sbvr:closedLogicalFormulationMeansProposition closedLogicalFormulation="***nec-formulation***" proposition="nec"/>

A closed logical formulation of the statement (not shown) has xmi:id="nec-formulation".

Note: This note says little.

<sbvr:noteCommentsOnMeaning note="note" meaning="meaning"/>

<sbvr:note xmi:id="note" expression="note-e"/>

<sbvr:text xmi:id="note-e" value="***This note says little***"./>

Possibility: Some example is seen.

<sbvr:statement xmi:id="pos-stmt" expression="pos-e" meaning="pos"/>

<sbvr:text xmi:id="pos-e" value="***Some example is seen***"./>

<sbvr:proposition xmi:id="pos" isPossiblyTrue ="true"/>

<sbvr:closedLogicalFormulationFormalizesStatement closedLogicalFormulation="***pos-formulation***" statement="pos-stmt"/>

<sbvr:closedLogicalFormulationMeansProposition closedLogicalFormulation="***pos-formulation***" proposition="pos"/>

A closed logical formulation of the statement (not shown) has xmi:id="pos-formulation".

Reference Scheme: An id of the example term and the set of authors of the example term

<sbvr:referenceScheme xmi:id="refScheme" simplyUsedRole="***ethi-r2***" extensionallyUsedRole="***etha-r2***"   
 identifyingCharacteristic=""/>

It is assumed for this entry that there is a binary verb concept ‘example term *has* id’ whose ‘id’ role has   
 xmi:id="ethi-r2".

It is assumed for this entry that there is a binary verb concept ‘example term *has* author’ whose ‘author’ role has  
 xmi:id="etha-r2".

See: example general concept designation

Same as “Synonym: example general concept designation”.

Source: ISO 1087-1 [‘example’]

<sbvr:referenceSupportsMeaning reference="ref" meaning="meaning"/>

<sbvr:reference xmi:id="ref" expression="source-e"/>  
 <sbvr:text xmi:id="source-e" value="***ISO 1087-1 [‘example’]***"/>

Subject Field: Philosophy

<sbvr:representationIsInSubjectField representation="exampleTerm" subjectField="philosophy"/>

<sbvr:conceptHasInstance concept="***philo-concept***" instance="philosophy"/>

<sbvr:subjectField xmi:id="philosophy"/>

It is assumed for this entry that there is a name ‘Philosophy’ for an individual noun concept like this:

<sbvr:name xmi:id="philo-name" signifier="philo-s" meaning="philo-concept"/>

<sbvr:individualConcept xmi:id=" philo-concept"/>

<sbvr:text xmi:id="philo-s" value="Philosophy"/>

Synonym: example general concept designation

<sbvr:term xmi:id="exampleObjectTypeDesignation" signifier="eotd-s" meaning="meaning"/>

<sbvr:text xmi:id="eotd-s" value="***example general concept designation***"/>

<sbvr:thingIsInSet set="vocabulary" thing="exampleObjectTypeDesignation"/>

<sbvr:designationIsInNamespace designation="exampleObjectTypeDesignation" namespace="vocabularyNamespace"/>

### XML Patterns for Individual Noun Concepts

**Example Name**

<sbvr:name xmi:id="exampleName" signifier="en-s" meaning="meaning"/>

<sbvr:individualConcept xmi:id="**meaning**"/>

<sbvr:text xmi:id="en-s" value="***Example Name***"/>

<sbvr:thingIsInSet set="vocabulary" thing="exampleName"/>

<sbvr:designationIsInNamespace designation="exampleName" namespace="vocabularyNamespace"/>

If there is no “See:” caption, then the following is included:

<sbvr:preferredDesignation xmi:id= "exampleNamePreferred"/>

<sbvr:thing1IsThing2 thing1="exampleNamePreferred" thing2="exampleName"/>

Definition: the example that is seen

<sbvr:definiteDescription xmi:id="defDesc-formal" expression="defDesc-formal-e" meaning="meaning"/>

<sbvr:text xmi:id="defDesc-formal-e" value="***the example that is seen***"/>

<sbvr:concept1SpecializesConcept2 concept1="meaning" concept2="example-concept" />

<sbvr:closedProjectionFormalizesDefinition closedProjection="***defDesc-formal-projection***" definition="defDesc-formal"/>

<sbvr:closedProjectionDefinesNounConcept closedProjection="***defDesc-formal-projection***" nounConcept="meaning"/>

The closed projection of the definition (not shown) has xmi:id="defDesc-formal-projection". Note that informal and  
 semiformal definitions of individual noun concepts follow the same pattern as shown for general concepts above   
 with the exception that they are rendered as sbvr:definiteDescription.

The captions “Concept Type:”, “Description:”, “Dictionary Basis:”, “Example:”, “General Concept:”, “Necessity:”, “Note:”, “Possibility:”, “See:”, “Source:”, “Subject Field:” and “Synonym:” are handled for a name in the same way as for terms as shown above.

### XML Patterns for Verb Concepts

example is seen

<sbvr:sententialForm xmi:id="exampleIsSeen" expression="eis-e" meaning="meaning" placeholder="eis-p"/>

<sbvr:verbSymbol xmi:id="example.isSeen" signifier="isSeen-s" meaning="meaning"/>

<sbvr:characteristic xmi:id="**meaning**" role="eis-r"/>

<sbvr:verbConceptWordingIncorporatesVerbSymbol verbConceptWording="exampleIsSeen" verbSymbol="example.isSeen"/>

<sbvr:text xmi:id="eis-e" value="***example is seen***"/>

<sbvr:text xmi:id="isSeen-s" value="***is seen***"/>

<sbvr:placeholder xmi:id="eis-p" expression="***example-s***" startingCharacterPosition="i1" meaning="eis-r"/>

<sbvr:placeholderUsesDesignation placeholder="eis-p" designation="***example***"/>

<sbvr:positiveInteger xmi:id="i1" value="***1***"/>

<sbvr:verbConceptRole xmi:id="**eis-r**"/>

<sbvr:roleRangesOverObjectType role="eis-r" generalConcept="***example-concept***"/>

<sbvr:thingIsInSet set="vocabulary" thing="exampleIsSeen"/>

<sbvr:thingIsInSet set="vocabulary" thing="example.isSeen"/>

<sbvr:verbConceptWordingIsInNamespace verbConceptWording="exampleIsSeen" namespace="vocabularyNamespace"/>

<sbvr:attributiveNamespaceIsWithinVocabularyNamespace attributiveNamespace="example-ans"   
 vocabularyNamespace="vocabularyNamespace"/>

<sbvr:attributiveNamespace xmi:id="example-ans"/>   
 <sbvr:attributiveNamespaceIsForSubjectConcept attributiveNamespace="example-ans"   
 subjectConcept="***example-concept***"/>

<sbvr:designationIsInNamespace designation="example.isSeen" namespace="example-ans"/>

example1 follows example2

<sbvr:sententialForm xmi:id="example1FollowsExample2" expression="efe-e" meaning="meaning" placeholder="efe-p1 efe-p2"/>

<sbvr:verbSymbol xmi:id="efe-follows" signifier="follows-s" meaning="meaning"/>

<sbvr:binaryVerbConcept xmi:id="**meaning**" role="efe-r1 efe-r2"/>

<sbvr:verbConceptWordingIncorporatesVerbSymbol verbConceptWording="example1FollowsExample2" verbSymbol="efe-follows"/>

<sbvr:text xmi:id="efe-e" value="***example1 follows example2***"/>

<sbvr:text xmi:id="follows-s" value="***follows***"/>

<sbvr:text xmi:id="**example1-s**" value="**example1**"/>

<sbvr:text xmi:id="example2-s" value="**example2**"/>

<sbvr:placeholder xmi:id="efe-p1" expression="example1-s" startingCharacterPosition="i1" meaning="efe-r1"/>

<sbvr:placeholder xmi:id="efe-p2" expression="example2-s" startingCharacterPosition="i18" meaning="efe-r2"/>

<sbvr:placeholderUsesDesignation placeholder="efe-p1" designation="***example***"/>

<sbvr:placeholderUsesDesignation placeholder="efe-p2" designation="***example***"/>

<sbvr:positiveInteger xmi:id="i1" value="***1***"/>

<sbvr:positiveInteger xmi:id="i18" value="***18***"/>

<sbvr:verbConceptRole xmi:id="**efe-r1**"/>

<sbvr:verbConceptRole xmi:id="**efe-r2**"/>

<sbvr:roleRangesOverObjectType role="efe-r1" generalConcept="***example-concept****"*/>

<sbvr:roleRangesOverObjectType role="efe-r2" generalConcept="***example-concept***"/>

<sbvr:thingIsInSet set="vocabulary" thing=" example1FollowsExample2"/>

<sbvr:thingIsInSet set="vocabulary" thing=" efe-follows"/>

<sbvr:verbConceptWordingIsInNamespace verbConceptWording="example1FollowsExample2" namespace="vocabularyNamespace"/>

Definition: the example1 comes after the example2 in a sequence

<sbvr:definition xmi:id="efe-def-formal" expression="efe-def-formal-e" meaning="meaning"/>

<sbvr:text xmi:id="efe-def-formal-e" value="***the example1 comes after the example2 in a sequence***"/>

<sbvr:closedProjectionFormalizesDefinition closedProjection="***efe-projection***" definition="efe-def-formal"/>

<sbvr:closedProjectionDefinesverbConcept closedProjection="***efe-projection***" verbConcept="meaning"/>

<sbvr:variableMapsToVerbConceptRole variable="***efe-var1***" verbConceptRole="***efe-r1***"/>

<sbvr:variableMapsToVerbConceptRole variable="***efe-var2***" verbConceptRole="***efe-r2***"/>

The definition formally defines ‘example1 followsexample2’ and has a closed projection (not shown) with   
 xmi:id="efe-projection" projectionVariable="efe-var1 efe-var2".

Definition: the first example is after the second

<sbvr:definition xmi:id="efe-def-informal" expression="efe-def-informal-e" meaning="meaning"/>

<sbvr:text xmi:id="efe-def-informal-e" value="*the first example is after the second*"/>

See: example1 has prior example

Same as “Synonymous Form: example1 *has* prior example”.

Issue # 15684: Revise text

Synonymous Form: example1 has prior example

<sbvr:sententialForm xmi:id="example1HasPriorExample" expression="ehpe-e" meaning="meaning" placeholder="ehpe-p1   
 ehpe-p2"/>

<sbvr:verbSymbol xmi:id="ehpe-has" signifier="has-s" meaning="meaning"/>

<sbvr:verbConceptWordingIncorporatesVerbSymbol verbConceptWording="example1HasPriorExample" verbSymbol="ehpe-has"/>

<sbvr:verbConceptRoleDesignation xmi:id="example.priorExample" signifier="priorExample-s" meaning="efe-r2"/>

<sbvr:text xmi:id="ehpe-e" value="***example1 has prior example***"/>

<sbvr:text xmi:id="has-s" value="**has**"/>

<sbvr:text xmi:id="priorExample-s" value="**prior example**"/>

<sbvr:placeholder xmi:id="ehpe-p1" expression="***example1-s***" startingCharacterPosition="i1" meaning="efe-r1"/>

<sbvr:placeholder xmi:id="ehpe-p2" expression="priorExample-s" startingCharacterPosition="i14" meaning="efe-r2"/>

<sbvr:placeholderUsesDesignation placeholder="ehpe-p1" designation="***example***"/>

<sbvr:positiveInteger xmi:id="i1" value="***1***"/>

<sbvr:positiveInteger xmi:id="i14" value="***14***"/>

<sbvr:thingIsInSet set="vocabulary" thing="example1HasPriorExample"/>

<sbvr:verbConceptWordingIsInNamespace verbConceptWording="example1HasPriorExample" namespace="vocabularyNamespace"/>

<sbvr:attributiveNamespaceIsWithinVocabularyNamespace attributiveNamespace="example-ans"   
 vocabularyNamespace="vocabularyNamespace"/>

<sbvr:attributiveNamespace xmi:id="example-ans"/>   
 <sbvr:attributiveNamespaceIsForSubjectConcept attributiveNamespace="example-ans"   
 subjectConcept="***example-concept***"/>

<sbvr:designationIsInNamespace designation="example.priorExample" namespace="example-ans"/>

If there is a term ‘prior example’ for a general concept like this:

<sbvr:term xmi:id="priorExample" signifier="priorExample-s" meaning="priorExample-c"/>

then the following is included:

<sbvr:placeholderUsesDesignation placeholder="ehpe-p2" designation="***priorExample***"/>

<sbvr:roleRangesOverObjectType role="efe-r2" generalConcept="***priorExample-c***"/>

The captions “Concept Type:”, “Description:”, “Dictionary Basis:”, “Example:”, “General Concept:”, “Necessity:”, “Note:”, “Possibility:” and “Source:” are handled for a verb concept wording in the same way as for terms as shown above.

Issue # 10630: Revise text

### XML Patterns for Sets of Elements of Guidance (Rule Sets)

**Xyz Rules**

<sbvr:set xmi:id="**ruleSet**"/>

<sbvr:nameReferencesThing thing="ruleSet" name="XyzRules"/>

<sbvr:name xmi:id="XyzRules" signifier="XyzRules-s" meaning="ruleSet-concept"/>

<sbvr:individualConcept xmi:id="**ruleSet-concept**" instance="ruleSet"/>

<sbvr:text xmi:id="XyzRules-s" value="***Xyz Rules***"/>

<sbvr:thingIsInSet set="vocabulary" thing="XyzRules"/>

<sbvr:designationIsInNamespace designation=" XyzRules " namespace="vocabularyNamespace"/>

Vocabulary: Abc Vocabulary

None.

The captions “Description:”, “Note:”, and “Source:” are handled for a rule set in the same way as for terms within a vocabulary, as shown above, except that the related meaning is given as meaning="ruleSet-concept".

### XML Patterns for Guidance Statements

Each example must be seen.

<sbvr:guidanceStatement xmi:id="stmt-formal" expression="stmt-formal-e" meaning="meaning"/>

<sbvr:elementOfGuidance xmi:id="**meaning**"/>

<sbvr:text xmi:id="stmt-formal-e" value="***Each example must be seen***"./>

<sbvr:closedLogicalFormulationFormalizesStatement closedLogicalFormulation="***stmt-formal-formulation***"   
 statement="stmt-formal"/>

<sbvr:closedLogicalFormulationMeansProposition closedLogicalFormulation="stmt-formal-formulation" proposition="meaning"/>

<sbvr:thingIsInSet set="ruleSet" thing="meaning"/>

The closed logical formulation of the statement (not shown) has xmi:id="stmt-formal-formulation".

Guidance Type: operative business rule

In this case where the guidance type is an SBVR concept, the line above that says,  
 “<sbvr:elementOfGuidance xmi:id="meaning"/>”, is replaced with this:

<sbvr:***operativeBusinessRule*** xmi:id="meaning"/>

Guidance Type: exemplary rule

<sbvr:conceptHasInstance concept="***exemplaryRule-c***" instance="meaning"/>

This pattern is used if the concept type is not an SBVR concept. There is assumed to be a term ‘exemplary rule’ for  
 a general concept like this:

<sbvr:term xmi:id="exemplaryRule" signifier="exemplaryRule-s" meaning="exemplaryRule-c"/>

<sbvr:generalConcept xmi:id="exemplaryRule-c"/>

<sbvr:text xmi:id="exemplaryRule-s" value="exemplary rule"/>

Enforcement Level: strict

<sbvr:operativeBusinessRuleHasLevelOfEnforcement   
 operativeBusinessRule="meaning"   
 levelOfEnforcement="strict-instance"/>

<sbvr:conceptHasInstance concept="***strict-concept***" instance="strict-instance"/>

<sbvr:levelOfEnforcement xmi:id="strict-instance"/>

It is assumed that the name ‘strict’ represents an individual noun concept like this:

<sbvr:name xmi:id="strict" signifier="strict-s" meaning="strict-concept"/>

<sbvr:individualConcept xmi:id="strict-concept"/>

<sbvr:text xmi:id="strict-s" value="strict"/>

Name: Rule 25

<sbvr:nameReferencesThing thing="meaning" name="Rule25"/>

<sbvr:name xmi:id="Rule25" signifier="Rule25-s" meaning="rule25Meaning"/>

<sbvr:individualConcept xmi:id="rule25Meaning" instance="meaning"/>

<sbvr:text xmi:id="Rule25-s" value="***Rule 25***"/>

<sbvr:thingIsInSet set="vocabulary" thing="Rule25"/>

<sbvr:designationIsInNamespace designation="Rule25" namespace="vocabularyNamespace"/>

Synonymous Statement: It is obligatory that each rule be seen.

<sbvr:guidanceStatement xmi:id="synstmt-formal" expression="synstmt-formal-e" meaning="meaning"/>

<sbvr:text xmi:id="synstmt-formal-e" value="***It is obligatory that each rule be seen***"./>

<sbvr:closedLogicalFormulationFormalizesStatement closedLogicalFormulation="***synstmt-formal-formulation***"   
 statement="synstmt-formal"/>

<sbvr:closedLogicalFormulationMeansProposition closedLogicalFormulation="***synstmt-formal-formulation***" proposition="meaning"/>

The closed logical formulation of the statement (not shown) has xmi:id="synstmt-formal-formulation".

The captions “Description:”, “Example:”, “Note:” and “Source:” are handled for a guidance statement in the same way as for terms as shown above.