

---

# Semantics of Business Vocabulary and Business Rules Specification

---

This OMG document replaces the submission document (bei/05-08-01), the Draft Adopted specification (dtc/05-11-01), and the Final Adopted Specification (dtc/06-03-01). This is the first Interim Specification.

Comments on the content of this document are welcomed, and should be directed to *issues@omg.org* by July 24, 2006.

You may view the pending issues for this specification from the OMG revision issues web page <http://www.omg.org/issues/>.

The FTF Recommendation and Report for this specification will be published on October 6, 2006. If you are reading this after that date, please download the available specification from the OMG Specifications Catalog.

---

# Semantics of Business Vocabulary and Business Rules (SBVR)

Interim Convenience Document  
dtc/06-03-02



OBJECT MANAGEMENT GROUP

Copyright © 2005 Business Rule Solutions, LLC  
Copyright © 2005 Business Semantics Ltd  
Copyright © 2005 Fujitsu Ltd  
Copyright © 2005 Hendryx & Associates  
Copyright © 2005 LibRT  
Copyright © 2005 KnowGravity Inc  
Copyright © 2005 Model Systems  
Copyright © 2005 Neumont University  
Copyright © 1997-2005 Object Management Group.  
Copyright © 2005 Unisys Corporation

## USE OF SPECIFICATION - TERMS, CONDITIONS & NOTICES

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

## LICENSES

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

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

## PATENTS

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

## GENERAL USE RESTRICTIONS

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

## DISCLAIMER OF WARRANTY

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

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

## RESTRICTED RIGHTS LEGEND

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

## TRADEMARKS

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

## COMPLIANCE

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

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

## **OMG's Issue Reporting Procedure**

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





# Table of Contents

Preface .....	v
Part I - Introduction .....	1
1 Scope .....	3
2 Conformance .....	3
2.1 Logical Formulation of Semantics .....	3
2.2 Business Vocabulary .....	3
2.3 Business Vocabulary and Business Rules .....	3
2.4 MOF 2 Generation from Vocabulary .....	4
2.5 XMI Generation from Vocabulary .....	4
3 Normative References .....	4
4 Terms and Definitions .....	4
5 Symbols .....	5
6 Additional Information .....	5
6.1 Changes to Adopted OMG Specifications .....	5
6.2 How to Read this Specification .....	5
6.2.1 About the Annexes .....	5
6.2.2 About the Normative Specification .....	6
6.3 Acknowledgements .....	7
Part II - Business Vocabulary+Rules for Business Vocabulary+Rules.....	9
7 Vocabulary Registration Vocabulary .....	11
7.1 Vocabulary Registration Vocabulary .....	11
7.1.1 Vocabularies Presented in this Document .....	11
7.1.2 Other Namespaces and Rule Sets Presented in this Document .....	12
7.1.3 External Vocabularies and Namespaces .....	12
8 Meaning and Representation Vocabulary .....	13
8.1 Meanings.....	15
8.1.1 Concepts .....	15
8.1.2 Propositions .....	19

8.1.3 Questions .....	20
8.2 Expressions .....	20
8.3 Representations .....	22
8.3.1 Designations .....	22
8.3.2 Definitions .....	23
8.3.3 Statements .....	23
8.3.4 Forms of Expression .....	24
8.3.5 Namespaces .....	27
8.4 Reference Schemes .....	29
8.5 Conceptual Schemas and Models .....	31
8.6 Extensions .....	33
8.6.1 Relating Meaning to Extension .....	34
8.6.2 Necessities Concerning Extension .....	34
8.7 Elementary Concepts .....	35
<b>9 Logical Formulation of Semantics Vocabulary .....</b>	<b>37</b>
9.1 Semantic Formulations.....	40
9.1.1 Logical Formulations .....	41
9.1.2 Projections .....	67
<b>10 Providing Semantic and Logical Foundations for Business Vocabulary and Rules .....</b>	<b>71</b>
10.1 Logical Foundations for SBVR .....	71
10.1.1 SBVR Formal Grounding Model Interpretation .....	71
10.1.2 Formal Logic & Mathematics in General .....	94
10.2 Formal Logic Interpretation Placed on SBVR Terms .....	101
10.3 Mapping of SBVR Business Terms to Formal Logic .....	101
10.3.1 Concepts with Different Mappings for 'First-Order' and 'Restricted Higher-order' .....	102
10.3.2 Concepts with a Single Mapping for 'First-Order' and 'Restricted Higher-order' .....	102
10.3.3 Concepts that Inherit their Formal Logic Mapping from their Generalization ....	105
10.3.4 Concepts whose Formal Logic Mapping is specified in their Specializations ...	107
<b>11 Business Vocabulary .....</b>	<b>109</b>
11.1 Business Meaning.....	110
11.1.1 Communities, Meanings & Vocabularies .....	110
11.1.2 Concepts & Characteristics .....	114
11.1.3 Kinds of Definition .....	118
11.1.4 Conceptualization Decisions .....	121
11.1.5 Fact Type Templating .....	123
11.2 Business Representation .....	128
11.2.1 Symbolization .....	128
11.2.2 Forms of Business Representation .....	132

<b>12 Business Rules .....</b>	<b>137</b>
12.1 Categories of Guidance .....	137
12.1.1 Guidance .....	138
12.1.2 Rules .....	139
12.1.3 Enforcement .....	139
12.1.4 Admonitions and Affirmations .....	140
12.2 Statements of Guidance .....	142
12.2.1 Categories of Business Statement .....	143
12.2.2 Business Statements .....	144
<b>13 Vocabulary-Driven Interchange Using MOF and XMI .....</b>	<b>147</b>
13.1 Vocabulary-to-MOF/XMI Vocabulary .....	147
13.1.1 Mapping .....	148
13.1.2 Relevant Concepts from UML 2 .....	149
13.1.3 XMI for MOF 2 .....	151
13.2 Essential SBVR .....	152
13.3 Vocabulary-to-MOF/XMI Mapping Rule Set .....	154
13.3.1 Namespace Mapping Rules .....	154
13.3.2 Designation Mapping Rules .....	155
13.3.3 Sentential Form Mapping Rules .....	155
13.3.4 Placeholder Mapping Rules .....	155
13.3.5 Notes and Limitations .....	156
<b>14 Index of Vocabulary Entries .....</b>	<b>157</b>
<b>15 Supporting Documents .....</b>	<b>169</b>
15.1 SBVR Metamodel .....	169
15.1.1 SBVR Metamodel Document.....	169
15.1.2 SBVR XML Schema .....	169
15.2 Business Vocabulary .....	170
15.2.1 Business Vocabulary XML Document .....	170
15.2.2 Describing Business Vocabulary MOF .....	170
15.2.3 Describing Business Vocabulary XML Schema .....	170
15.3 Business Vocabulary and Rules .....	170
15.3.1 Business Rules XML Document .....	170
15.4 Vocabulary-to-MOF/XMI .....	171
15.4.1 Vocabulary-to-MOF/XMI Vocabulary and Mapping Rule Set XML Document ..	171
15.4.2 Vocabulary-to-MOF/XMI MOF .....	171
15.4.3 Vocabulary-to-MOF/XMI XML Schema .....	171
15.4.4 Essential SBVR MOF .....	171
15.4.5 Essential SBVR XML Schema .....	171
15.5 XMI XML Schema .....	171

Part III - Annexes .....	173
A - Overview of the Approach .....	175
B - The Business Rules Approach .....	189
C - SBVR Structured English .....	193
D - SBVR Structured English Patterns .....	211
E - EU-Rent Example .....	221
F - The RuleSpeak® Business Rule Notation .....	293
G - Concept Diagram Graphic Notation .....	309
H - Use of UML Notation in a Business Context to Represent SBVR-Style Vocabularies .....	315
I - The ORM Notation for Verbalizing Facts and Business Rules .....	325
J - ORM Examples Related to the Logical Foundations for SBVR .....	329
K - Design Rationale Details for the Use of MOF and XMI .....	335
L - Examples of SBVR's Use of MOF .....	351
M - Mappings and Relationships to Other Initiatives ....	359
N - Additional References .....	371

# Preface

## About the Object Management Group

### OMG

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

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

More information on the OMG is available at <http://www.omg.org/>.

### OMG Specifications

As noted, OMG specifications address middleware, modeling and vertical domain frameworks. A catalog of all OMG Specifications Catalog is available from the OMG website at:

[http://www.omg.org/technology/documents/spec\\_catalog.htm](http://www.omg.org/technology/documents/spec_catalog.htm)

Specifications within the Catalog are organized by the following categories:

#### OMG Modeling Specifications

- UML
- MOF
- XMI
- CWM
- Profile specifications.

#### OMG Middleware Specifications

- CORBA/IIOP
- IDL/Language Mappings
- Specialized CORBA specifications
- CORBA Component Model (CCM).

#### Platform Specific Model and Interface Specifications

- CORBA services
- CORBA facilities

- OMG Domain specifications
- OMG Embedded Intelligence specifications
- OMG Security specifications.

All of OMG's formal specifications may be downloaded without charge from our website. (Products implementing OMG specifications are available from individual suppliers.) Copies of specifications, available in PostScript and PDF format, may be obtained from the Specifications Catalog cited above or by contacting the Object Management Group, Inc. (as of January 16, 2006) at:

OMG Headquarters  
140 Kendrick Street  
Building A, Suite 300  
Needham, MA 02494  
USA  
Tel: +1-781-444-0404  
Fax: +1-781-444-0320  
Email: [pubs@omg.org](mailto:pubs@omg.org)

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

## Issues

The reader is encouraged to report any technical or editing issues/problems with this specification to <http://www.omg.org/technology/agreement.htm>.

# Part I - Introduction

This part includes Scope, Conformance, Normative References, Terms and Definitions, Symbols, and Additional Information.





# 1 Scope

This specification defines the vocabulary and rules for documenting the semantics of business vocabulary, business facts, and business rules; as well as an XMI schema for the interchange of business vocabularies and business rules among organizations and between software tools.

This specification is interpretable in predicate logic with a small extension in modal logic. This specification supports linguistic analysis of text for business vocabulary and rules, with the linguistic analysis itself being outside the scope of this specification.

This specification is applicable to the domain of business vocabulary and business rules of all kinds of business activities of all kinds of organizations. It is conceptualized optimally for business people rather than automated rules processing, and is designed to be used for business purposes, independent of information systems designs.

This specification is applicable as input to transformations by IT staff into information system designs, using a combination of decisions from system architects and Platform Independent Model designers together with software tool function.

## 2 Conformance

Software that conforms to this specification will be able to import and export XMI documents that conform with the XMI rules applied to the normative metamodel contained in documents listed below under each conformance point. To be conformant, software must import and export.

There are five conformance points listed below. Software can be conformant with one or more conformance points and not with others, but conformance with any conformance point requires complete satisfaction of all of requirements of that conformance point. Any statement of conformance should specify the conformance points in which the implementation is conformant.

### 2.1 Logical Formulation of Semantics

A conformant software correctly consumes and produces XML documents that conform to the SBVR Logical Formulation of Semantics XML Schema. The conformant software also detects and reports when XML input violates necessary conditions stated by this specification.

### 2.2 Business Vocabulary

The software correctly consumes and produces XML documents conveying vocabulary information conforming to the SBVR Business Vocabulary XML Schema. The conformant software also detects and reports when XML input violates necessary conditions stated by this specification.

### 2.3 Business Vocabulary and Business Rules

The software correctly consumes and produces XML documents conveying information conforming to the SBVR Business Vocabulary and Business Rules XML Schema. The conformant software also detects and reports when XML input violates necessary conditions stated by this specification.

## 2.4 MOF 2 Generation from Vocabulary

The software correctly generates an XML document conforming to OMG's MOF 2 XML Schema from any XML document that conforms to the SBVR Logical Formulation of Semantics XML Schema. Production follows the rules of the Vocabulary-to-MOF/XMI Mapping Rule Set.

## 2.5 XMI Generation from Vocabulary

The software correctly generates an XML schema following OMG's XMI for MOF 2 Specification from any XML document that conforms to the SBVR Logical Formulation of Semantics XML Schema. Production follows the rules of the Vocabulary-to-MOF/XMI Mapping Rule Set.

# 3 Normative References

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

- Berners-Lee, T., R. Fielding, L. Masinter. IETF RFC 2396: *Uniform Resource Identifiers (URI): Generic Syntax*, August 1998.
- International Organization for Standardization (ISO) : ISO 639-2. *Codes for the Representation of Names of Languages, Part 2: Alpha-3 Code*. Library of Congress, 2002.
- International Organization for Standardization (ISO) : 1087-1. *Terminology work — Vocabulary — Part 1: Theory and Application*
- *OMG UML 2 Infrastructure* (<http://www.omg.org/cgi-bin/doc?ptc/2004-10-14>).
- *The Cambridge Dictionary of Philosophy*, 2nd ed. Cambridge University Press, 1999.
- *The New Oxford Dictionary of English*.
- *Unicode 4.0.0 specification* : Glossary (<http://www.unicode.org/versions/Unicode4.0.0/b1.pdf>).
- XMI 2.1 Tags

# 4 Terms and Definitions

For the purposes of this specification, the terms and definitions given in the normative reference and the following apply.

### **SBVR**

shorthand for Semantics of Business Vocabulary and Business Rules.

### **SBVR Vocabularies**

the vocabularies that make up SBVR itself, for talking about semantics, vocabulary, and rules.

### **Business Vocabulary**

a structured set of terms and other symbols together with their meanings and relationships among them, for use by a business community.

**Business Rule**

a rule that is under business jurisdiction.

**Business Vocabulary+Rules**

a business vocabulary plus a set of business rules specified in terms of that business vocabulary.

**SBVR Metamodel**

the MOF model created from the combination of SBVR's Logical Formulation of Semantics Vocabulary, Vocabulary for Describing Business Vocabularies, and Vocabulary for Describing Business Rules guided by the Vocabulary-to-MOF/XMI Mapping Rule Set.

## 5 Symbols

**FL** The indicated term is to be interpreted in formal logic. Terms without this symbol are not interpreted in formal logic.

The non-normative notation used in Part II and Annex E is specified in Annexes C, F, and H.

## 6 Additional Information

### 6.1 Changes to Adopted OMG Specifications

This specification does not require or request any change to any other OMG specification.

### 6.2 How to Read this Specification

SBVR is a vocabulary, or actually a set of sub-vocabularies, each consisting of a set of terminological entries. Each entry includes a definition, along with other specifications such as notes and examples. Often, the entries include rules (necessities) about the particular item being defined.

The sequencing of the sub-vocabularies in this specification reflects the inherent logical order of the subject matter itself. Later sub-vocabularies build semantically on the earlier ones. The initial sub-vocabularies are therefore rather 'deep' in terms of SBVR's grounding in formal logics and linguistics. Only after these sub-vocabularies are presented do sub-vocabularies more relevant to day-to-day business communication and business rules emerge.

This overall form of presentation, essential for a vocabulary standard, unfortunately means the material is rather difficult to approach. A non-normative diagram presented for each sub-vocabulary does help illustrate its structure; however, no continuous 'narrative' or explanation is appropriate.

#### 6.2.1 About the Annexes

For that reason, the first-time general reader is urged to start with some of the non-normative Annexes, which do provide full explanation of the material, as well as context and purpose.

- Annex A, Overview of the Approach, is strongly recommended in that regard. It provides a general introduction to the fundamental concepts and approach of SBVR.
- Annex B, The Business Rules Approach, explains the core ideas and principles of business rules, which underpin

SBVR's origin and focus. This short Annex is strongly recommended for readers who are unfamiliar with this area.

Good preparation for reading the specification is becoming familiar with the notation (non-normative) used to present the entries.

- Annex C, SBVR Structured English, provides comprehensive explanation in that regard.
- Annex D, SBVR Structured English Patterns, explains how to verbalize vocabulary structure.

General practitioners will find the following sections of significant interest.

- Annex E, EU-Rent Example, provides a comprehensive case study, with a robust vocabulary and set of business rules fully worked through. Examples from EU-Rent are used widely in the both the specification and Annexes to provide on-going commonality.
- Annex F, The RuleSpeak<sup>R</sup> Business Rule Notation, presents a widely-used, business-friendly syntax for expressing business rules.
- Annex G, Concept Diagram Graphic Notation, offers suggestions for how an SBVR vocabulary can be diagrammed.
- Annex H, Use of UML Notation in a Business Context to Represent SBVR-style Vocabularies, is of special interest to practitioners familiar with UML diagramming.

Object-Role Modeling (ORM)-related Annexes:

- Annex I, The ORM Notation for Verbalizing Facts and Business Rules, provides an introduction to the ORM approach. ORM contributes heavily to the theoretical underpinnings of SBVR, and represents some of the best practices in fact-based vocabulary and rule development.
- Annex J, ORM Discussion and Diagrams Related to the Logical Foundations for SBVR, provides supplemental ORM material further clarifying the normative material, Logical Foundations for SBVR.

For specialists in software engineering and tooling, especially regarding MOF and XMI, the following Annexes are of particular interest.

- Annex K, Design Rational Details for the Use of MOF and XMI, explains the SBVR approach to implementation using MOF and XMI.
- Annex L, Examples of SBVR's Use of MOF, illustrates the SBVR Approach described in Annex A.

For those specialists and researchers interested in standards and/or in the formal logics underpinning of SBVR, the following material is of special interest.

- Annex M, Mappings and Relationships to Other Initiatives, addresses where and how SBVR fits with other software and standards initiatives.
- Annex N, Additional References, provides supplemental sources relevant to the formal logics underpinnings of SBVR.

## 6.2.2 About the Normative Specification

The rest of this document contains the technical content of this specification. As background for this specification, readers are encouraged to first read:

Chapters 7-15 contain chapters for the SBVR vocabularies and rules that are the foundation for the SBVR Metamodel as well as a definition of the Essential SBVR Package, which is part of all MOF models created following the Vocabulary-to-MOF/XMI Rule Set.

Chapters 7-15 address different audiences. Four of the chapters are directly tied to conformance points, which are listed in Chapter 2. Chapter 7 gives names to the SBVR Vocabularies and to some other vocabularies and namespaces used by SBVR. Chapter 8 provides the Meaning and Representation Vocabulary, which covers different kinds of meaning and representations. It is the foundation for the rest of the specification. Chapter 9 provides the Logical Formulation of Semantics Vocabulary, which is the SBVR way to formulate semantics. It is not a vocabulary for business people but, rather, for detailed descriptions of the meanings of business words and statements. Chapter 10 shows the formal logics and mathematical underpinnings of SBVR. Numerous concepts in chapters 8 and 9 are marked with the symbol 'FL' indicating that they are mapped to formal logics concepts in 10.

Chapters 11 and 12 are vocabularies for use in business to describe vocabularies (11) and business rules (12).

Chapter 13 specifies how SBVR uses MOF and XMI. Chapter 14 is an index of vocabulary entries in Chapters 7-13. Chapter 15 lists supporting documents, such as an XMI-based XML schema for the SBVR Metamodel.

Chapters 7-15 use SBVR Structured English to define the SBVR vocabularies and rules. Annex C describes how the Structured English is interpreted such that SBVR is specified in terms of itself.

Much of the material in Part II is illustrated by examples in the annexes, especially Annex E.

Although the chapters are organized in a logical manner and can be read sequentially, this is a reference specification and is intended to be read in a non-sequential manner. Consequently, extensive cross-references are provided to facilitate browsing and search.

## 6.3 Acknowledgements

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

- Adaptive
- Automated Reasoning Corporation
- Business Rule Solutions, LLC
- Business Rules Group
- Business Semantics Ltd
- Fujitsu Ltd
- Hendryx & Associates
- Hewlett-Packard Company
- InConcept
- LibRT
- KnowGravity Inc
- MEGA
- Model Systems
- Neumont University
- Perpetual Data Systems
- Sandia National Laboratories
- The Rule Markup Initiative
- Unisys Corporation
- X-Change Technologies Group



## Part II - Business Vocabulary+Rules for Business Vocabulary+Rules

This part contains sections for the SBVR vocabularies and rules that are the foundation for the SBVR Metamodel as well as a definition of the Essential SBVR Package, which is part of all MOF models created following the Vocabulary-to-MOF/XMI Rule Set.

The chapters of Part II address different audiences. Chapter 7 gives names to the SBVR Vocabularies and to some other vocabularies and namespaces used by SBVR. Chapter 8 provides the [Meaning and Representation Vocabulary](#), which covers different kinds of meaning and representations. It is the foundation for the rest of the specification. Chapter 9 provides the [Logical Formulation of Semantics Vocabulary](#), which is the SBVR way to formulate semantics. It is not a vocabulary for business people, but rather, for detailed descriptions of the meanings of business words and statements. Chapter 10 shows the formal logics and mathematical underpinnings of SBVR. Numerous concepts in chapters 8 and 9 are marked with the symbol 'FL' indicating that they are mapped to formal logics concepts in Chapter 10.

Chapters 11 and 12 are vocabularies for use in business to describe vocabularies (11) and business rules (12).

Chapter 13 specifies how SBVR uses MOF and XMI. Chapter 14 is an index of vocabulary entries in Part II. Chapter 15 lists supporting documents, such as an XMI-based XML schema for the SBVR Metamodel.

Part II uses SBVR Structured English to define the SBVR vocabularies and rules. Annex C describes how the Structured English is interpreted such that SBVR is specified in terms of itself.

Much of the material in Part II is illustrated by examples in the annexes, especially Annex E.





# 7 Vocabulary Registration Vocabulary

## 7.1 Vocabulary Registration Vocabulary

This section gives names of vocabularies, rule sets and namespaces. Each one is either provided by SBVR or is external to SBVR but formally referenced.

---

### Vocabulary Registration Vocabulary

Language: [English](#)

---

#### 7.1.1 Vocabularies Presented in this Document

##### Vocabulary Registration Vocabulary

General Concept: [vocabulary](#)

Note: This section.

##### Meaning and Representation Vocabulary

General Concept: [vocabulary](#)

Note: See Chapter 8 - Meaning and Representation Vocabulary.

##### Logical Formulation of Semantics Vocabulary

General Concept: [vocabulary](#)

Note: See Chapter 9 - Logical Formulation of Semantics Vocabulary.

##### Formal Logic & Mathematics Vocabulary

General Concept: [vocabulary](#)

Note: See Chapter 10 - Providing Semantic and Logical Foundations for Business Vocabulary and Rules.

##### Vocabulary for Describing Business Vocabularies

General Concept: [vocabulary](#)

Note: See Chapter 11 - Business Vocabulary.

##### Vocabulary for Describing Business Rules

General Concept: [vocabulary](#)

Note: See Chapter 12 - Business Rules.

##### Vocabulary-to-MOF/XMI Vocabulary

Definition: [the vocabulary](#) that is used to describe transformation of any vocabulary defined in terms of SBVR into a MOF/XMI implementation that supports repository services and data interchange of facts in terms of atomic formulations using the SBVR vocabulary

Note: See Section 13.1 - Vocabulary-to-MOF/XMI Vocabulary.

### Essential SBVR Vocabulary

General Concept: [vocabulary](#)

Note: See Section 13.2 - Essential SBVR.

## 7.1.2 Other Namespaces and Rule Sets Presented in this Document

### Vocabulary-to-MOF/XMI Mapping Rule Set

General Concept: [set](#)

Note: See Section 13.3 - Vocabulary-to-MOF/XMI Mapping Rule Set.

### Integer Namespace

Definition: [the vocabulary namespace](#) that has designations for all integers, each designation representing an individual concept of a particular integer using a sequence of one or more decimal numerals, optionally preceded by a minus sign (“-”)

Note: The [Integer Namespace](#) includes designations using every possible sequence of decimal numerals, with and without a leading minus sign.

## 7.1.3 External Vocabularies and Namespaces

### ISO 1087-1 (English)

Definition: [the vocabulary](#) for the English language specified in [ISO1087-1]

### ISO 639-2 (English)

Definition: [the vocabulary](#) of English language names of languages specified in [ISO639-2]. Available at <http://www.loc.gov/standards/iso639-2/englangn.html>

### ISO 639-2 (Alpha-3 Code)

Definition: [the vocabulary](#) of 3-letter codes for languages specified in [ISO639-2]. Available at <http://www.loc.gov/standards/iso639-2/englangn.html>

### UML 2 Infrastructure

Definition: [the namespace](#) of designations for UML 2 Infrastructure concepts as defined by [UML2infr].

### Unicode Glossary

Definition: [the vocabulary](#) presented in [Unicode4].

### Uniform Resource Identifiers Vocabulary

Definition: [the vocabulary](#) presented in [IETF RFC 2396].

### XMI 2.1 Tags

Definition: [the vocabulary namespace](#) of tagged values of [XMI2.1].

## 8 Meaning and Representation Vocabulary

The primary subjects of the [Meaning and Representation Vocabulary](#) fit between two other relevant subject areas described below.

1. **Expression** – things used to communicate (e.g., sounds, text, diagrams, gestures), but apart from their meaning — one expression can have many meanings
2. **Representation** – the connection between expression and a meaning. Each representation ties one expression to one meaning
3. **Meaning** – what is meant by a word (a concept) or by a statement (a proposition) – how we think about things
4. **Extension** – the things to which meanings refer, which can be anything (even expressions, representations, and meanings when they are the subjects of our discourse)

Following are examples of how some things, like “driver,” cross through each subject area.

Extension	Meaning	Representation	Expression
The actual drivers of motor vehicles	Concept ‘driver’ — how we think of drivers, what characterizes them	Designation of the concept ‘driver’ by the signifier “driver”	The character sequence “driver”
		Definition of the concept ‘driver’ as “operator of a motor vehicle”	The character sequence “operator of a motor vehicle”
The actual City of Los Angeles, California – a real place	Individual concept ‘Los Angeles’ — how we think of that city, what distinguishes it from other places	‘Los Angeles’ as a designation for the individual concept of ‘Los Angeles’	The character sequence “Los Angeles”
For each car that is out of service, its actually being out of service	Characteristic applicable to a car, what is meant by a car being out of service	Form of expression ‘ <u>car</u> is out of service’ as a template for the characteristic with ‘ <u>car</u> ’ being a placeholder	The text “ <u>car</u> is out of service”
The actual state of affairs of it being obligatory in the EU-Rent business that it not rent to a barred driver	Proposition — the meaning of the statement “EU-Rent must not rent to a barred driver”	The statement, “EU-Rent must not rent to a barred driver,” having the proposition as its meaning	The character sequence “EU-Rent must not rent to a barred driver”

Another subject area of this vocabulary is reference schemes, which are ways people use information about something to identify it. For example, a city in the United States is identified by a name combined with the state it is in. The state is identified by its name or by a two-letter state code.

Representations provide a reference scheme for concepts and propositions because they are always tied to exactly one expression and to exactly one meaning. On the other hand, a single expression can have multiple meanings, a concept can have multiple expressions, a thing can be an instance of many concepts, and a proposition can be meant by many equivalent expressions.

A single representation can be tied to many speech acts, or to a single speech act, depending on how its expression is identified. For example, if the expression is a text or a sequence of words independent of any particular act of writing or speaking, the representation is independent in the same way. Conversely, if the expression is identified as belonging to a specific speech act, then the representation is tied to that speech act also.

The [Meaning and Representation Vocabulary](#) is not presented alphabetically. It is organized by subjects presented in the following order.

1. Meanings
  - a. Concepts
  - b. Propositions
  - c. Questions
2. Expressions
3. Representations
4. Reference Schemes
5. Conceptual Schemas and Models
6. Extension
7. Elementary Concepts

---

### [Meaning and Representation Vocabulary](#)

Language: [English](#)

---

## 8.1 Meanings

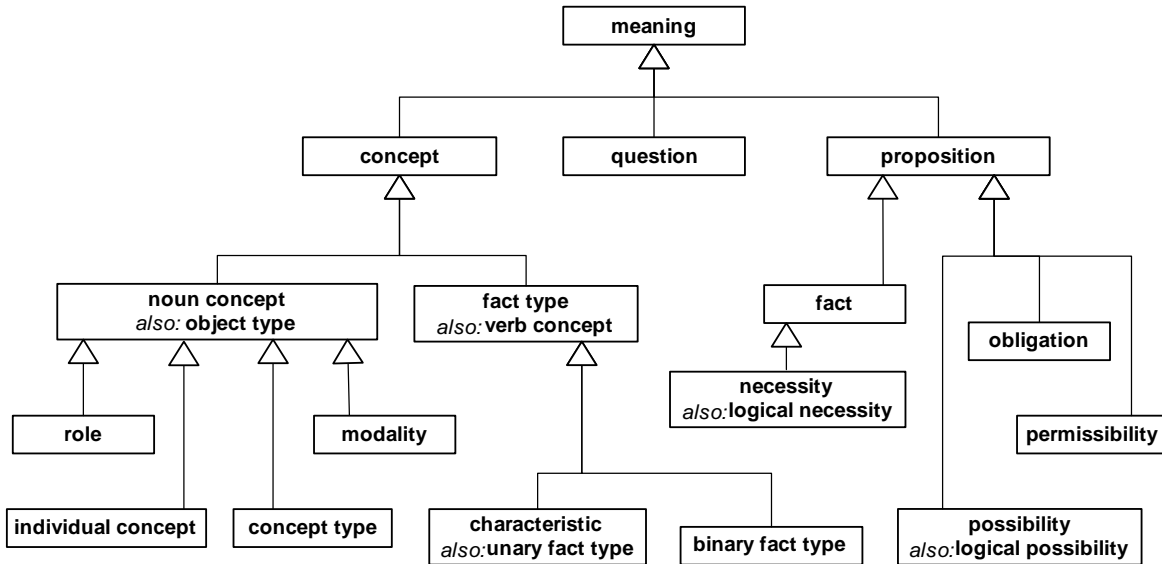


Figure 8.1

---

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

---

### meaning

Definition: what is meant by a word, sign, statement, or description; what someone intends to express or what someone understands

### 8.1.1 Concepts

#### concept

Source: [ISO 1087-1 \(English\)](#) (3.2.1) ['concept'] FL  
 Definition: unit of knowledge created by a unique combination of characteristics  
 General Concept: [meaning](#)  
 Reference Scheme: a [designation of the concept](#)

#### noun concept

Definition: [concept](#) that is not a [fact type](#) FL  
 Synonym: [object type](#)  
 Concept Type: [concept type](#)  
 Reference Scheme: a [closed projection](#) that *defines* the [noun concept](#)  
 Example: the [concept](#) 'car', the [concept](#) 'number', the [concept](#) 'person'

#### concept type

Definition: [noun concept](#) that *specializes* the [concept](#) 'concept' FL

Note: A [concept](#) is related to a [concept type](#) by being an [instance](#) of the [concept type](#).  
Example: [fact type](#), [role](#), [concept type](#)

## [role](#)

FL

Definition: [noun concept](#) that corresponds to things based on their playing a part, assuming a function or being used in some situation

Necessity: Each [role](#) is of at most one [fact type](#).

Concept Type: [concept type](#)

Reference Scheme: a [placeholder](#) that represents the [role](#)

Reference Scheme: a [variable](#) that maps to the [role](#)

Reference Scheme: a [characteristic](#) that has the [role](#)

Example: the [role](#) ‘[drop-off location](#)’ of the fact type ‘[shipment](#) has [drop-off location](#)’

Example: the [role](#) ‘[shipment](#)’ of the fact type ‘[shipment](#) has [drop-off location](#)’, which should not be confused with the general concept ‘[shipment](#)’ (which generalizes the role)

Example: the [role](#) ‘sum’ – a [role](#) of a number in relation to a set of numbers

Note: If a role is defined generally, either with respect to a very general fact type or without defining a corresponding fact type, then that role can be specialized with respect to specific fact types as in the example below.

Example: a [role](#) ‘pick-up date’ could be defined generally as a date for picking up something. The [role](#) ‘pick-up date’ of the [fact type](#) ‘[rental](#) has [pick-up date](#)’ is a separate [role](#) that specializes the generally defined one by being limited to pick-up dates of rentals and not other things. The [role](#) ‘pick-up date’ of the [fact type](#) ‘[shipment](#) has [pick-up date](#)’ is yet another separate [role](#) that specializes the generally defined one.

## [fact type](#)

FL

Definition: [concept](#) whose instances are all actualities and that is a basis for [atomic formulation](#), having at least one [role](#)

Synonym: [verb concept](#)

Note: For each instance of a [fact type](#), each [role](#) of the [fact type](#) is one point of involvement of something in that instance.

Concept Type: [concept type](#)

Necessity: Each [fact type](#) has at least one [role](#).

Reference Scheme: a [form of expression of the fact type](#)

Reference Scheme: a [closed projection](#) that defines the [fact type](#)

## [characteristic](#)

FL

Definition: [fact type](#) that has exactly one [role](#)

Source: [ISO 1087-1 \(English\)](#) (3.2.4) [‘characteristic’]

Definition: abstraction of a property of an object [[thing](#)] or of a set of objects

Synonym: [unary fact type](#)

Reference Scheme: a [role of the fact type](#)

Example: The [fact type](#) ‘[shipment](#) is late’ whose instances are actualities of shipments being late. There is one instance of the fact type for each shipment that is late.

Note: A characteristic always has exactly one role, but it can be defined using fact types having multiple roles.

Example: The characteristic 'driver is of age' with this definition: "the age of the driver is at least the EU-Rent Minimum Driving Age". The semantic formulation of this rule appears in the introduction to Chapter 9 - Logical Formulation of Semantics Vocabulary.

### binary fact type

FL

Definition: fact type that has exactly 2 roles

Example: The fact type 'shipment has drop-off location' whose instances are actualities of shipments having drop-off locations.

Example: The fact type 'number is greater than number' whose instances are actualities of numbers being greater than other numbers, there being one instance for every pair of numbers where one is greater than the other.

Note: A fact type can have two roles that seem to be identical (e.g., 'person is married to person' where each role can be called 'spouse'). But the two roles are distinct within the fact type, each one specializing a more general role (e.g., the role 'spouse' defined as a participant in a marriage).

### individual concept

FL

Source: ISO 1087-1 (English) (3.2.2) ['individual concept']

Definition: concept that corresponds to only one object [thing]

General Concept: noun concept

Concept Type: concept type

Note: An individual concept always has one instance, but not necessarily the same instance in all possible worlds.

Example: The individual concept 'California' whose one instance is an individual state in the United States of America

### 8.1.1.1 About Concepts

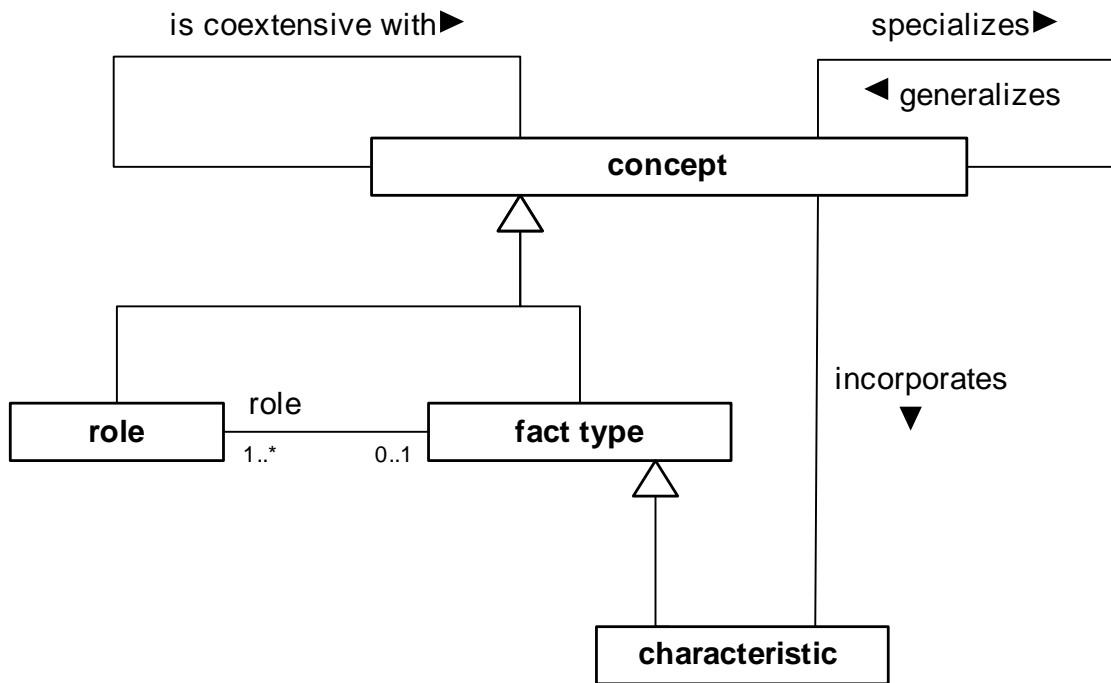


Figure 8.2

---

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

---

#### concept<sub>1</sub> specializes concept<sub>2</sub>

FL

- Definition:            the concept<sub>1</sub> incorporates each characteristic incorporated into the concept<sub>2</sub> plus at least one differentiator
- Synonymous Form:    concept<sub>2</sub> generalizes concept<sub>1</sub>
- Note:                    The extension of a concept that specializes another is always a subset of the extension of the other, but not necessarily a proper subset. The differentiator that makes one concept more specific than the other is conceptual and does not necessarily restrict the extension of the concept.
- Example:                The noun concept ‘whole number’ specializes the noun concept ‘integer’, the differentiator being that whole numbers are nonnegative.
- Example:                The role ‘sum’ specializes the noun concept ‘number’, the differentiator being that each sum is the result of adding up a set of numbers. It turns out that every number is a sum of that number added to zero, so the extensions of the concepts ‘sum’ and ‘number’ are always the same. Nevertheless, the role ‘sum’ incorporates the differentiating characteristic of being the result of addition, so it specializes the noun concept ‘number’.
- Example:                The individual concept ‘Los Angeles’ specializes the concept ‘city’, the differentiator being that Los Angeles is one particular city in California.



### concept<sub>1</sub> is coextensive with concept<sub>2</sub>

FL

- Definition: the extension of the concept<sub>1</sub> always equals the extension of the concept<sub>2</sub>
- Note: Semantic integrations between communities often involve recognizing where different concepts (having different intensions) have the same extensions in all possible worlds. Also, it is possible that concepts employing different methods of conceptualization have the same extension in all cases. For example, a noun concept that specializes the concept ‘actuality’ can be equivalent to a fact type.
- Example: The role ‘sum’ is coextensive with the noun concept ‘number’.

### concept incorporates characteristic

FL

- Definition: the characteristic is an abstraction of a property of each instance of the concept
- Synonymous Form: characteristic is incorporated into concept
- Example: The concept ‘qualified driver’ incorporates the characteristic ‘driver is licensed’ because it is necessary (by the definition of ‘qualified driver’) that each qualified driver is licensed.

### fact type has role

FL

- Definition: the role is an abstraction of a thing playing a part in instances of the fact type

## 8.1.2 Propositions

### proposition

FL

- Definition: meaning that is asserted when a sentence is uttered or inscribed and which is true or false
- Note: The word “proposition” has two common meanings: first, a statement that affirms or denies something, and second, the meaning of such a statement. The concept ‘proposition’ is here defined in the second sense and should not be confused with the statement of a proposition.
- Reference Scheme: a closed logical formulation that means the proposition

### fact

FL

- Definition: proposition that is taken as true
- Note: How one ascertains what is true, whether by assertion, observation, or other means, is outside the scope of this specification. However, taking a proposition as true must be consistent with epistemic commitment. The concept ‘fact’ is here defined to be consistent with the operations of truth-functional logic, which produce results based on true and false.

### modality

FL

- Definition: noun concept that classifies a proposition based on any of a number of operations of modal logics such as deontic and alethic logics
- Concept Type: concept type
- Necessity: Each modality specializes the concept ‘proposition’.

### necessity

FL

- Definition: fact that is necessarily true, always true
- Concept Type: modality
- Synonym: logical necessity

### possibility

FL

Definition: [proposition that](#) is possibly true, there being no necessity that it be false  
Concept Type: [modality](#)  
Synonym: [logical possibility](#)

### obligation

FL

Definition: [proposition that](#) is required to be true, that is obligatory, that is not permitted to be false  
Concept Type: [modality](#)

### permissibility

FL

Definition: [proposition that](#) is permitted to be true, there being no obligation that it be false  
Concept Type: [modality](#)

## 8.1.3 Questions

### question

Definition: [meaning](#) of an interrogatory  
Note: The word “question” has two common meanings: first, a written or spoken expression of inquiry, and second, the meaning of such an inquiry. By the second definition, a single question could be asked in two languages. But by the first definition, using two language results in two expressions, and therefore, two questions. The concept ‘[question](#)’ is here defined in the second sense (meaning) and should not be confused with the expression or representation of a [question](#).  
Reference Scheme: [a closed projection that means the question](#)

## 8.2 Expressions

### expression

Definition: something that expresses or communicates, but independent of its interpretation  
Example: the sequence of characters “car”  
Example: the sequence of speech sounds (t), (r), and (ē)  
Example: a smile  
Example: a diagram  
Example: The entire text of a book

### signifier

Definition: [expression that](#) is a linguistic unit or pattern, such as a succession of speech sounds, written symbols or gestures, used in the [designation](#) of a [concept](#)  
Concept Type: [role](#)  
Example: the sequence of characters “car” used in a [designation](#) of the [concept](#) ‘automobile’ or used in a [designation](#) of the [concept](#) ‘railroad car’  
Example: the sequence of speech sounds (t), (r), and (ē) used in a [designation](#) of the [concept](#) ‘tree’  
Example: The character “€” used in a [designation](#) of the [concept](#) ‘Euro’

## text

- Source: [Unicode 4.0.0 Glossary](#) ['Character Sequence']
- General Concept: [expression](#)
- Note: The [concept](#) 'text' has no explicit [reference scheme](#), but rather, is used as a target for reference schemes.
- Note: A detailed vocabulary concerning text is provided by the Unicode specification. Taking the concept 'text' from the Unicode specification does not mean that a text is a Unicode encoding, but rather, it implies that a text can be represented by a Unicode encoding in electronic communications. Unicode encodings provide the common means of text representation in word processors, mail systems, the Internet, and so on. The encodings tend to be invisible to people writing and reading the text.

## starting character position

- Definition: [positive integer](#) **that** is an ordinal position where a text starts within an encompassing text
- Concept Type: [role](#)

## URI

- Source: [Uniform Resource Identifiers Vocabulary](#) ['URI']
- General Concept: [text](#)
- Concept Type: [role](#)
- Synonym: [uniform resource identifier](#)
- Note: The [concept](#) 'URI' is introduced into this specification in order to provide a universal context for reference schemes.

## 8.3 Representations

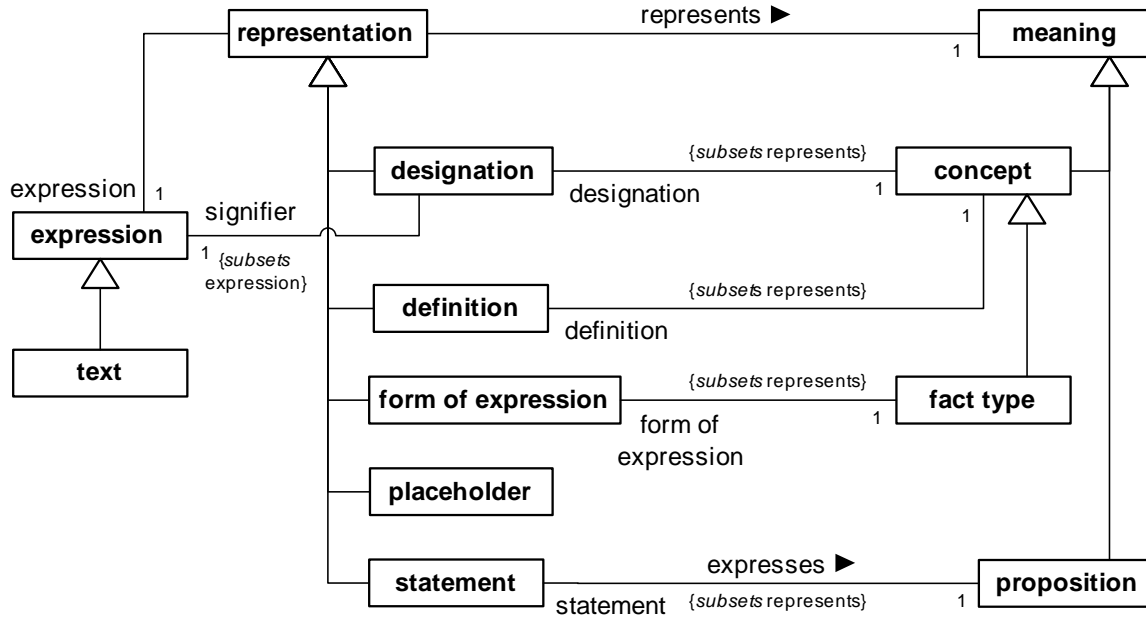


Figure 8.3

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

### representation

- Definition: portrayal of a [meaning](#) by an [expression](#)  
 Necessity: Each [representation](#) has exactly one [expression](#).  
 Necessity: Each [representation](#) represents exactly one [meaning](#).

### representation has expression

### representation represents meaning

Synonymous Form: [meaning](#) is represented by [representation](#)

## 8.3.1 Designations

### designation

- Source: [ISO 1087-1 \(English\)](#) (3.4.1) ['designation']  
 Definition: [representation](#) of a [concept](#) by a sign which [means the concept and] denotes [the extension of] it  
 Necessity: Each [designation](#) represents a [concept](#).  
 Reference Scheme: the [signifier](#) of the [designation](#) and a [namespace](#) that includes the [designation](#)  
 Reference Scheme: the [signifier](#) of the [designation](#) and the [concept](#) that is represented by the [designation](#)

### designation has signifier

Definition: the signifier is the expression of the designation

### concept has designation

Definition: the designation represents the concept

## 8.3.2 Definitions

### definition

Source: ISO 1087-1 (English) (3.3.1) ['definition']

Definition: representation of a concept by a descriptive statement [expression] which serves to differentiate it from related concepts

Necessity: Each definition represents a concept.

Necessity: Each closed projection that formalizes a definition of a concept defines the concept.

Reference Scheme: the expression of the definition and the concept that is represented by the definition

### concept has definition

Definition: the definition represents the concept

## 8.3.3 Statements

### statement

Definition: representation of a proposition by an expression of the proposition

Necessity: Each statement expresses exactly one proposition.

Necessity: Each closed logical formulation that formalizes a statement means the proposition that is expressed by the statement.

Reference Scheme: the expression of the statement and a closed logical formulation that formalizes the statement

### statement expresses proposition

Definition: the statement represents the proposition

Synonymous Form: proposition has statement

Synonymous Form: proposition is expressed by statement

### 8.3.4 Forms of Expression

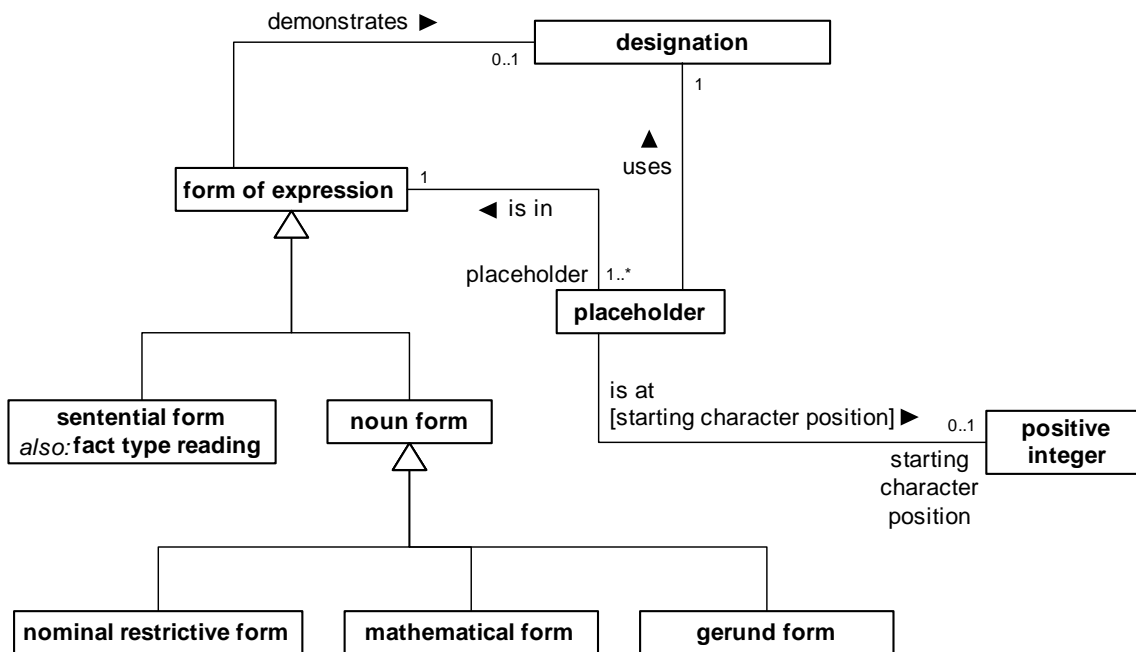


Figure 8.4

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

#### form of expression

- Definition: [representation](#) of a [fact type](#) by a pattern or template of expressions based on the [fact type](#)
- Necessity: [Each form of expression represents exactly one fact type.](#)
- Necessity: [Each form of expression has at least one placeholder.](#)
- Necessity: [At most one role of a fact type that has a form of expression is not represented by a placeholder of the form of expression.](#)
- Necessity: [No form of expression is a designation.](#)
- Necessity: [Each form of expression demonstrates at most one designation.](#)
- Necessity: [If a designation is demonstrated by a form of expression of a fact type then the fact type has the designation.](#)
- Example: The [form of expression](#) ‘customer rents car’ demonstrates the [designation](#) ‘rents’ and has two placeholders. One [placeholder](#) uses the [designation](#) ‘customer’ and is at the [starting character position](#) 1. The other [placeholder](#) uses the [designation](#) ‘car’ and is at the [starting character position](#) 16.
- Example: The [form of expression](#) ‘driver of car’ demonstrates a [designation](#) ‘of’ and has two placeholders, one using the [designation](#) ‘driver’ at the [starting character position](#) 1, and the other using the [designation](#) ‘car’ at the [starting character position](#) 11.
- Example: The [form of expression](#) ‘country charges tax rate on date’ demonstrates the [designation](#) ‘charges on’ that represents the same [fact type](#) as the [form of expression](#).

Note: In some languages, forms of expression occur that involve only a positioning of placeholders with no other designation — no verb or preposition.

Reference Scheme: [the expression of the form of expression](#) and the set of each [placeholder of the form of expression](#) and a [namespace that includes the form of expression](#)

### fact type has form of expression

Definition: [the form of expression](#) provides a pattern or template for expressions denoting [the fact type](#)

### form of expression demonstrates designation

Definition: [the form of expression](#) shows a pattern of using [the designation](#) in an [expression](#)

Synonymous Form: [designation is demonstrated by form of expression](#)

### form of expression has placeholder

Definition: [the placeholder](#) indicates a place for expression of what fills a [role](#) in [the form of expression](#)

Synonymous Form: [placeholder is in form of expression](#)

### sentential form

Definition: [form of expression that](#) is a pattern or template that can be used for stating a proposition based on a fact type

Synonym: [fact type reading](#)

Example: ‘[car](#) is used in [rental agreement](#)’ is a [sentential form](#) of a [binary fact type](#).

Example: ‘[car](#) is unavailable’ is a [sentential form](#) of a [unary fact type](#).

Example: Assuming there is a role ‘renter’ specializing the concept ‘customer’, the following can all be alternative sentential forms of the same fact type:

[car](#) has [renter](#)  
[customer](#) rents [car](#)  
[car](#) is rented by [customer](#)  
[renter](#) rents [car](#)

Necessity: [Each role of the fact type that has a sentential form is represented by a placeholder of the sentential form.](#)

### noun form

Definition: [form of expression that](#) acts as a noun rather than forming a proposition

Example: ‘[car](#) used in [rental agreement](#)’ — see ‘[nominal restrictive form](#)’.

Example: ‘[number](#) + [number](#)’ — see ‘[mathematical form](#)’.

Example: ‘[car](#) being used in [rental agreement](#)’ — see ‘[gerund form](#)’.

### mathematical form

Definition: [noun form that](#) is a form for a mathematical expression whose result corresponds to a role not represented by a placeholder, but rather, by the whole form

Necessity: [if a fact type has a mathematical form then exactly one role of the fact type is represented by no placeholder of the mathematical form](#)

Note: In a mathematical form, there is no placeholder representing the result of an expression based on that form. This is a form common in mathematical expression.

Example: ‘number + number’ is a mathematical form of a fact type having three roles. That same fact type could have sentential forms such as ‘number + number = number’ and nominal restrictive forms such as ‘sum of number and number’.

Example: ‘number’ is a mathematical form of a binary fact type. That same fact type could have a sentential form: ‘number has absolute value’ and a nominal restrictive form: ‘absolute value of number’.

### nominal restrictive form

Definition: noun form whose result corresponds to the role represented by its first placeholder

Necessity: Each role of the fact type that has a nominal restrictive form is represented by a placeholder of the nominal restrictive form.

Example: ‘car used in rental agreement’ is a nominal restrictive form that is a pattern of expression to refer to cars based on their being used in rental agreements. It is of the same fact type that could have a sentential form, ‘car is used in rental agreement’.

Example: ‘of’ is used very commonly in nominal restrictive forms in English for binary fact types having a sentential form using the word ‘has’. For example, the sentential form ‘customer has name’ implies a nominal restrictive form, ‘name of customer’.

### gerund form

Definition: noun form of a fact type used to denote an actuality that is an instance of the fact type

Necessity: Each role of the fact type that has a gerund form is represented by a placeholder of the gerund form.

Example: ‘car being used in rental agreement’ is a gerund form that is a pattern of expression to refer to actualities of cars being used in rental agreements. It is of the same fact type that could have a sentential form, ‘car is used in rental agreement’.

Example: ‘car being rented’ is a gerund form of the characteristic ‘car is rented’. It is used to denote actualities of cars being rented.

### placeholder

Definition: representation of a role within a form of expression marking a place where, in uses of the form of expression, an expression denotes what fills the role

Necessity: Each placeholder is in exactly one form of expression.

Necessity: Each placeholder represents exactly one role.

Necessity: Each placeholder of each form of expression of a fact type represents a role of the fact type.

Necessity: Each placeholder has at most one starting character position.

Necessity: Each placeholder of a form of expression that has a text has a starting character position.

Necessity: Each placeholder uses exactly one designation.

Reference Scheme: the form of expression that has the placeholder and the designation that is used for the placeholder and the starting character position of the placeholder

### placeholder is at starting character position

Synonymous Form: placeholder has starting character position



### placeholder uses designation

Definition: **the placeholder** uses **the designation** of a **concept** for which use of the **form of expression** is understood

Synonymous Form: **designation is used for placeholder**

## 8.3.5 Namespaces

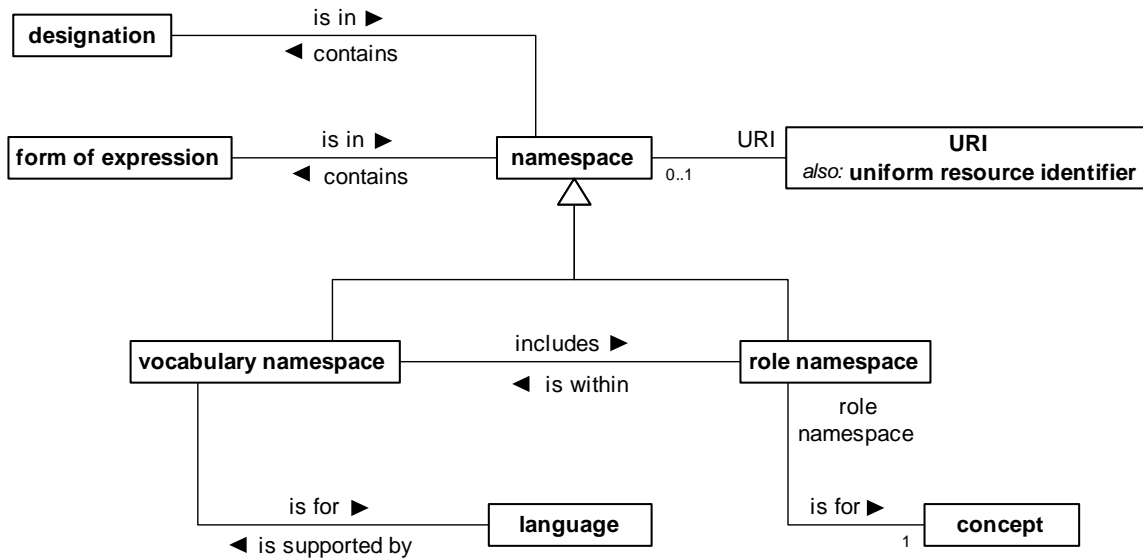


Figure 8.5

---

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

---

### namespace

Definition: container of distinguishable designations and/or forms of expression

Reference Scheme: **a URI of the namespace**

### designation is in namespace

Definition: **the namespace** contains **the designation** such that the signifier of **the designation** is the signifier of no other designation in **the namespace**

Synonymous Form: **namespace contains designation**

### form of expression is in namespace

Definition: **the namespace** contains **the form of expression** such that it is distinguishable from every other form of expression in **the namespace**

Synonymous Form: **namespace contains form of expression**

### namespace has URI

Definition: **the URI** uniquely identifies **the namespace**

Necessity: Each [URI](#) is the [URI](#) of at most one [namespace](#).

### vocabulary namespace

Definition: [namespace](#) for a vocabulary which includes all of the vocabulary's designations and forms of expression that are distinguishable by the uniqueness of signifier or form, and which possibly includes role namespaces containing designations and forms of expression of the vocabulary

### role namespace

Definition: [namespace](#) that contains designations recognizable in the context of being attributed to instances of a particular concept

Necessity: Each [role namespace](#) is for exactly one [concept](#).

Reference Scheme: a [vocabulary namespace](#) that includes the [role namespace](#) and the [concept](#) that has the [role namespace](#)

Note: A [designation](#) in a [role namespace](#) is typically a [role](#) of a [binary fact type](#). In English, such a designation can typically be used with any of several attributive forms, such as "... has ..." or "... of ...". A [designation](#) in a [role namespace](#) can also be for a [characteristic](#).

Example: Given a [role namespace](#) for the [concept](#) 'rental', a [designation](#) 'drop-off date' can be used in any of several attributive forms: "rental has drop-off date", "drop-off date of rental", "rental's drop-off date", "drop-off date is of rental", etc.

Example: Given a [role namespace](#) for the [concept](#) 'rental', the [designation](#) 'assigned' for the [characteristic](#) 'rental is assigned' is recognized where it applies to a rental, as in "assigned rental".

### role namespace is for concept

Definition: the designations in the [role namespace](#) are for concepts attributable to instances of the [concept](#)

Synonymous Form: [concept](#) has [role namespace](#)

### role namespace is within vocabulary namespace

Definition: the [role namespace](#) is a section of the [vocabulary namespace](#) attributable to the [concept](#) that has the [role namespace](#)

Synonymous Form: [vocabulary namespace](#) includes [role namespace](#)

### language

Definition: system of arbitrary signals (such as voice sounds or written symbols) and rules for combining them as used by a nation, people or other distinct community

Source: based on AH

Note: A language can be a natural language or an unnatural one, such as a computer language or a system of mathematical symbols.

Note: A language is often identified by its name. ISO provides names of many languages in [ISO 639-2 \(English\)](#) and provides short (at most 3 letters) language-independent codes in [ISO 639-2 \(Alpha-3 Code\)](#).

Example: English, French, German, Arabic

Example: Moroccan Arabic (a dialect of Arabic)

Example: Unified Modeling Language (a graphical modeling language)

## vocabulary namespace is for language

Definition: each representation in [the vocabulary namespace](#) is for expression in [the language](#)  
Synonymous Form: [language is supported by vocabulary namespace](#)

## 8.4 Reference Schemes

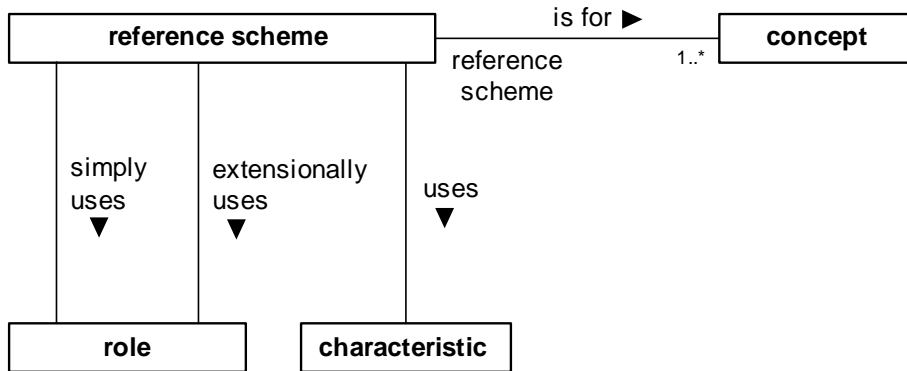


Figure 8.6

---

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

---

### reference scheme

FL

Definition: chosen way of identifying instances of a given [concept](#)

Note: A [reference scheme](#) is a way of referring to instances of a [concept](#) by way of related things that are either lexical or are otherwise identifiable. A reference scheme uses one or more roles of binary fact types in order to identify an instance of a concept from facts about the instance.

Note: A [reference scheme](#) can be partial or complete. It is complete if it can always be used to refer to every instance of the concept. An overall complete reference scheme for a concept can result from there being multiple partial reference schemes for that concept, its more general concepts, and its categories.

Note: Choice of reference schemes must be based on uniqueness (providing an identifier that refers to exactly one thing), but it should consider more than uniqueness. It should also consider permanence – if the actualities considered by the scheme change often, then references can become invalid. A reference scheme should also not lead into an inescapable reference cycle where things only identify each other, but should lead either directly or indirectly to an expression. It should also consider convenience and relevance from a business perspective.

Note: A role is used in a reference scheme in either of two ways. A simple use of a role involves a single instance of the role in each reference based on the scheme. An extensional use of a role involves the entire set of related instances of the role in each reference based on the scheme.

Note: A reference scheme implies that there is uniqueness – that whatever facts are used to reference an individual thing uniquely identify that one thing.

Reference Scheme: [the set of each role that is simply used by the reference scheme](#) and the set of each [role that is extensionally used by the reference scheme](#)

### reference scheme is for concept

FL

- Definition: instances of **the concept** can be identified using **the reference scheme**
- Synonymous Form: concept has reference scheme
- Necessity: **Each reference scheme is for at least one concept.**

### reference scheme extensionally uses role

FL

- Definition: a set of instances of **the role**, which is of a **binary fact type**, serves as identification or partial identification of an **instance** of the **concept** having **the reference scheme** where the set is the set of all instances of **the role** related by way of the **binary fact type** that has the **role**
- Synonymous Form: role is extensionally used by reference scheme
- Necessity: **Each role that is extensionally used by a reference scheme is of a binary fact type.**

### reference scheme simply uses role

FL

- Definition: any given **instance** of **the role**, which is of a **binary fact type**, serves as identification or partial identification of an **instance** of the **concept** having **the reference scheme** where the given **instance** is related by way of the **binary fact type** that has the **role**
- Synonymous Form: role is simply used by reference scheme
- Necessity: **Each role that is simply used by a reference scheme is of a binary fact type.**

### reference scheme uses characteristic

FL

- Definition: having or not having **the characteristic** serves as identification or partial identification of an **instance** of the **concept** having **the reference scheme**
- Synonymous Form: characteristic is used by reference scheme
- Note: Characteristics can contribute to reference schemes. This is most typically useful in compound references schemes that also involve a binary fact type. Using a characteristic in a reference scheme is equivalent to using a binary fact type with a Boolean role whose value is true then the characteristic holds and false otherwise. But business vocabularies don't tend to define binary relationships to Booleans, but rather, they define characteristics.

## 8.5 Conceptual Schemas and Models

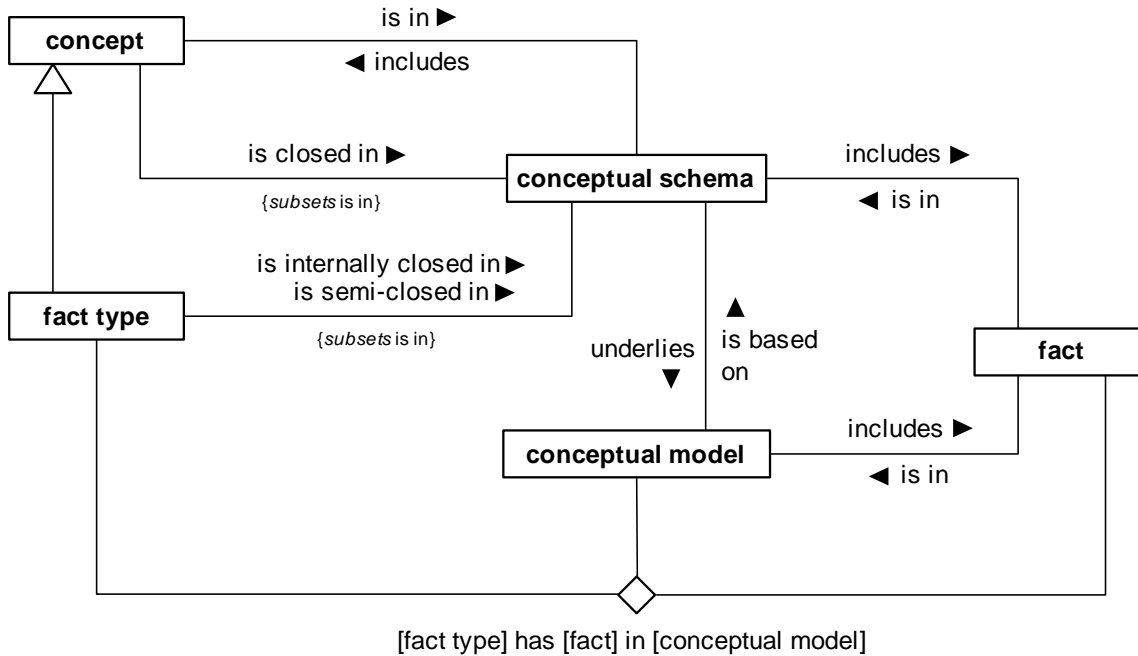


Figure 8.7

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

### conceptual schema

Definition: combination of concepts and facts (with semantic formulations that define them) of what is possible, necessary, permissible, and obligatory in each possible world

FL

### conceptual schema includes concept

Definition: the concept is used in models based on the conceptual schema

Synonymous Form: concept is in conceptual schema

Necessity: Each role of each fact type that is in a conceptual schema is in the conceptual schema.

FL

### conceptual schema includes fact

Definition: the fact determines something possible, necessary, permissible, or obligatory in each possible world that can be modeled based on the conceptual schema

Synonymous Form: fact is in conceptual schema

FL

### fact type is internally closed in conceptual schema

Definition: in each conceptual model based on the conceptual schema, for each instance of the fact type, the conceptual model includes a corresponding fact if, for each thing filling any of the

FL

fact type's roles in the instance, the [conceptual model](#) also includes a fact of the existence of that thing

Synonymous Form: [fact type is semi-closed in conceptual schema](#)

Necessity: [Each fact type that is semi-closed in a conceptual schema is in the conceptual schema.](#)

Note: Open world semantics are assumed by default, but closure may be explicitly asserted for any fact type, on an individual basis, to declare that each conceptual model population agrees with that of the fact type's extension in the actual business domain. Semi-closure is with respect to the domain model population of the noun concepts playing a role in the fact type. In other words, if the things participating in a fact are known within a model, then the fact is also known within that model.

### [concept is closed in conceptual schema](#)

FL

Definition: in each [conceptual model](#) based on [the conceptual schema](#), the entire extension of [the concept](#) is given in the facts included in the [conceptual model](#)

Necessity: [Each concept that is closed in a conceptual schema is in the conceptual schema.](#)

Note: A concept can be closed in one conceptual schema and not in another. For example, consider a corporate customer of EU-Rent that adopts several of EU-Rent's concepts. The corporate customer's conceptual schema might have the concept '[rental](#)' as not closed because the customer is not aware of all rentals, but EU-Rent's conceptual schema has the concept as closed.

### [conceptual model](#)

FL

Definition: combination of a conceptual schema and, for one possible world, a set of facts (defined by semantic formulations using only the concepts of the conceptual schema)

Note: Each necessity of the conceptual schema is satisfied by a conceptual model, but obligations are not necessarily satisfied.

### [conceptual model is based on conceptual schema](#)

FL

Definition: [the conceptual schema](#) provides the concepts and modal facts of [the conceptual model](#)

Synonymous Form: [conceptual schema underlies conceptual model](#)

### [conceptual model includes fact](#)

FL

Definition: [the fact](#) corresponds to an actuality in the possible world modeled by [the conceptual model](#)

Synonymous Form: [fact is in conceptual model](#)

### [fact type has fact in conceptual model](#)

FL

Definition: [the fact is in the conceptual model](#) and [the fact corresponds to an instance of the fact type](#)

## 8.6 Extensions

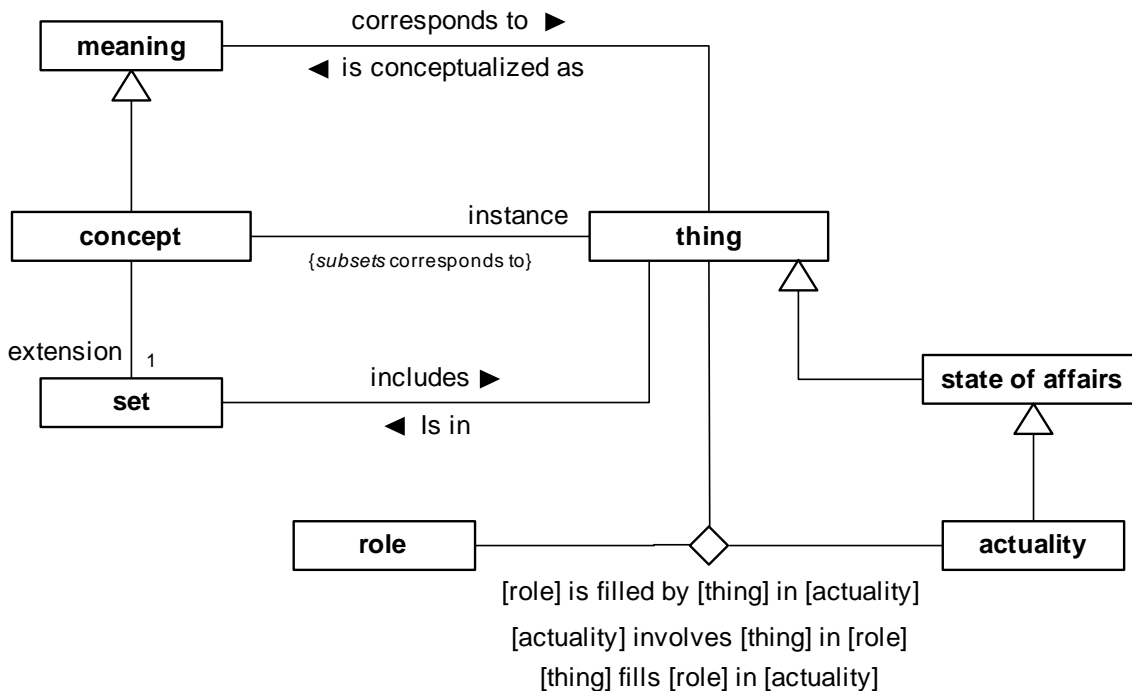


Figure 8.8

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

### state of affairs

FL

- Definition: event, activity, situation or circumstance
- Reference Scheme: a [proposition](#) that *corresponds to* the [state of affairs](#)
- Note: A [state of affairs](#) can be possible or impossible. Some of the possible ones are actualities. A [state of affairs](#) is what is denoted by a [proposition](#). A [state of affairs](#) either occurs or does not occur, whereas a [proposition](#) is either true or false.
- Example: EU-Rent owning 10,000 rental cars is a state of affairs corresponding to the proposition “EU-Rent owns 10,000 rental cars.”
- Example: It being obligatory that each rental have at most three additional drivers is a state of affairs corresponding to the rule, “Each rental must have at most three additional drivers.”

### actuality

FL

- Definition: [state of affairs](#) that occurs in the actual world

### actuality involves thing in role

FL

- Definition: the [actuality](#) is an instance of the [fact type](#) that has the [role](#) and the [thing](#) plays the [role](#) in that [actuality](#)

Synonymous Form: [role is filled by thing in actuality](#)  
 Synonymous Form: [thing fills role in actuality](#)  
 Note: This fact type supports talking generically about involvement of things in the instances of fact types.

### extension

FL

Source: [ISO 1087-1 \(English\)](#) (3.2.8) [*'extension'*]  
 Definition: totality of objects [every [thing](#)] to which a [concept](#) corresponds  
 Concept Type: [role](#)  
 General Concept: [set](#)

### instance

FL

Definition: [thing that is in an extension of a concept](#)  
 Concept Type: [role](#)  
 Example: The actual City of Los Angeles is an [instance](#) of the [concept](#) 'city'. It is also the one [instance](#) of the [individual concept](#) 'Los Angeles'.

## 8.6.1 Relating Meaning to Extension

### meaning corresponds to thing

Definition: [the thing](#) is the actual thing conceptualized by [the meaning](#)  
 Synonymous Form: [thing is conceptualized as meaning](#)  
 Note: A concept corresponds to each instance of the concept. A proposition corresponds to a state of affairs (which might or might not be actual). A fact corresponds to an actuality.

### concept has extension

Definition: [the extension](#) is the set of things to which [the meaning](#) corresponds

### concept has instance

Definition: [the concept corresponds to the instance](#)

## 8.6.2 Necessities Concerning Extension

The following statements of necessity apply to the relationships between a meaning and its extension. Other necessities stated in the context of the [Meaning and Representation Vocabulary](#) concern the contents of conceptual schemas and their representations. But the following necessities concern each conceptual model in relation to the conceptual schema that underlies it.

Necessity: [Each concept has exactly one extension.](#)  
 Necessity: [A thing is an instance of a concept if and only if the thing is in the extension of the concept.](#)  
 Necessity: [Each individual concept has exactly one instance.](#)  
 Necessity: [Each instance of a fact type is an actuality.](#)  
 Necessity: [Each proposition corresponds to at most one state of affairs.](#)  
 Necessity: [Each fact corresponds to exactly one actuality.](#)  
 Necessity: [Each actuality of each fact type involves some thing in each role of the fact type.](#)



- Necessity: Each thing that *fills a role in an actuality* is an instance of the role.
- Necessity: An actuality is an instance of a fact type if the actuality involves a thing in a role of the fact type.
- Necessity: If a concept incorporates a characteristic then each instance of the concept is an instance of the role of the characteristic.
- Necessity: If a concept<sub>1</sub> is coextensive with a concept<sub>2</sub> then the extension of the concept<sub>1</sub> equals the extension of the concept<sub>2</sub>.

## 8.7 Elementary Concepts

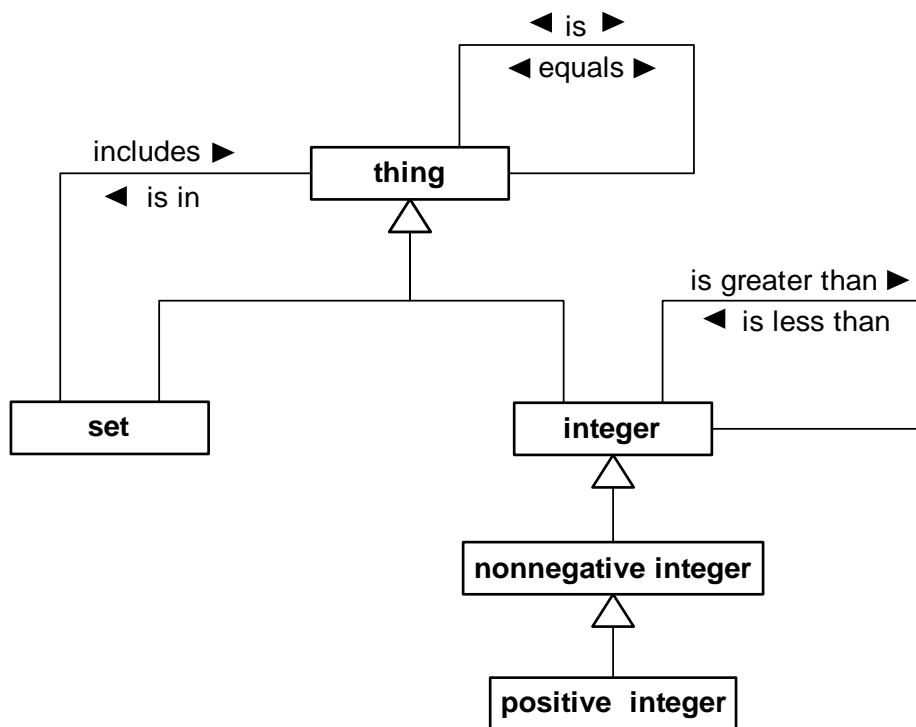


Figure 8.9

---

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

---

### thing

- Source: [ISO 1087-1 \(English\)](#) (3.1.1) ['object']
- Definition: anything perceivable or conceivable
- Note: Every other concept implicitly specializes the concept 'thing'.
- Reference Scheme: an individual concept that *corresponds to the thing*

FL

### **thing<sub>1</sub> is thing<sub>2</sub>**

FL

- Definition: The [thing<sub>1</sub>](#) and the [thing<sub>2</sub>](#) are the same [thing](#)
- Synonymous Form: [thing<sub>1</sub> equals thing<sub>2</sub>](#)
- Synonymous Form: [thing<sub>1</sub> is equal to thing<sub>2</sub>](#)
- Synonymous Form: [thing<sub>1</sub> = thing<sub>2</sub>](#)

### **set**

FL

- Definition: collection of zero or more [things](#) considered together without regard to order

### **thing is in set**

FL

- Definition: [the thing](#) is an element of [the set](#)
- Synonymous Form: [set includes thing](#)

### **integer**

FL

- Definition: number with no fractional part
- Note: The [Integer Namespace](#), in the [Namespace Registration Vocabulary](#), has designations for all of the integers

### **integer<sub>1</sub> is less than integer<sub>2</sub>**

FL

- Definition: The [integer<sub>1</sub>](#) is numerically less than [integer<sub>2</sub>](#)
- Synonymous Form: [integer<sub>1</sub> < integer<sub>2</sub>](#)
- Synonymous Form: [integer<sub>2</sub> is greater than integer<sub>1</sub>](#)
- Synonymous Form: [integer<sub>2</sub> > integer<sub>1</sub>](#)

### **nonnegative integer**

FL

- Definition: [integer](#) that is greater than or equal to zero

### **positive integer**

FL

- Definition: [nonnegative integer](#) that is not zero

## 9 Logical Formulation of Semantics Vocabulary

The vocabulary in this section is not intended for use by business people in general, but rather, it is a vocabulary used to describe the formal semantic structures of business discourse. It is not for discussing business, but for discussing the semantic structures underlying business communications of concepts, facts, and rules. For example, a typical business person does not tend to talk about quantifications, but he expresses quantifications in almost every statement he makes. He doesn't tend to talk about conjuncts, disjuncts, negands, antecedents and consequents, but these are all part of the formulation of his thinking. The vocabulary in this section is for talking about these conceptual devices that people use all the time.

Semantic formulations are not representations or expressions of meaning. Rather, they are structures of meaning – the logical composition of meaning.

Business rules are generally expressed in natural language, although some rules are at times illustrated graphically. SBVR does not provide a logic language for restating business rules in some other language that business people don't use. Rather, SBVR provides a means for describing the structure of the meaning of rules expressed in the natural language that business people use. Semantic formulations are not expressions or statements. They are structures that make up meaning. Using SBVR, the meaning of a definition or statement is communicated as facts about the semantic formulation of the meaning, not as a restatement of the meaning in a formal language.

There are two kinds of semantic formulations. The first kind, logical formulation, structures propositions, both simple and complex. Specializations of that kind are given for various logical operations, quantifications, atomic formulations based on fact types and other formulations for special purposes such as objectification and nominalization.

The second kind of semantic formulation is projection. It structures intension with regard to what comprises sets or multisets. Projections formulate definitions, aggregations, and questions. A projection is over a logical formulation of a condition that determines what is included in a resulting set or multiset.

Semantic formulations are recursive. Several kinds of semantic formulations embed other semantic formulations. Logic variables are introduced by quantifications (a kind of logical formulation) and projections so that embedded formulations can refer to instances of concepts. A logic variable used in a formulation is free within that formulation if it is not introduced within that formulation. A formulation is closed if no variable is free within it. Only a closed semantic formulation can formulate a meaning. If a formulation has a variable that is free within it, then it can be part of a larger formulation of a meaning (one that introduces the variable) but it does not by itself formulate a meaning.

The hierarchical composition of semantic formulations is seen in the following example of a very simple business rule. The rule is stated in different ways but is one rule having one meaning. Many other statements are possible.

- A rental must have at most three additional drivers.
- It is obligatory that each rental has at most three additional drivers.

Below is a representation of a semantic formulation of the rule above as sentences that convey the full structure of the rule as a collection of facts about it. Note that different semantic formulations are possible for the same meaning. Two semantic formulations can be determined to have the same meaning either by logical analysis or by assertion (as a matter of definition). A single formulation is shown below.

The rule is a proposition meant by an obligation claim.  
. That obligation claim embeds a universal quantification.  
. . The universal quantification introduces a first variable.  
. . . The first variable ranges over the concept 'rental'.

- . . The universal quantification scopes over an at-most-n quantification.
- . . . The at-most-n quantification has the maximum cardinality 3.
- . . . The at-most-n quantification introduces a second variable.
- . . . . The second variable ranges over the concept ‘additional driver’.
- . . . The at-most-n quantification scopes over an atomic formulation.
- . . . . The atomic formulation is based on the fact type ‘rental has additional river’.
- . . . . . The atomic formulation has a role binding.
- . . . . . The role binding is of the role ‘rental’ of the fact type.
- . . . . . The role binding binds to the first variable.
- . . . . . The atomic formulation has a second role binding.
- . . . . . The second role binding is of the role ‘additional driver’ of the fact type.
- . . . . . The second role binding binds to the second variable.

Note that designations like ‘rental’ and ‘additional driver’ are used above to refer to concepts. The semantic formulations involve the concepts themselves, so identifying the concept ‘rental’ by another designation (such as from another language) does not change the formulation.

The indentation in the example shows a hierarchical structure in which a semantic formulation at one level operates on, applies a modality to, or quantifies over one or more semantic formulations at the next lower level. Each kind of logical formulation, including modality claims, quantifications, and logical operations, can be embedded in other semantic formulations to any depth and in almost any combination.

Within the one atomic formulation in the example are bindings to two variables. The variables are free within the atomic formulation because they are introduced outside of it (higher in the hierarchical structure). For this reason, the atomic formulation has no meaning. But the obligation claim has a meaning (the rule), and even the logical universal quantification within the obligation claim each has a meaning because each of those two formulations is closed.

Semantic formulations are further exemplified for a simple definition of a characteristic, “driver is of age”.

Definition: the age of the driver is at least the EU-Rent Minimum Driving Age

Below is a representation of a semantic formulation of the definition. Note that different semantic formulations are possible. A single formulation is shown below.

The characteristic is defined by a set projection.

- . The set projection is on a first variable.
- . . The first variable ranges over the concept ‘driver’.
- . . The first variable maps to the one role of the characteristic.
- . The set projection is constrained by a first universal quantification.
- . . The first universal quantification introduces a second variable.
- . . . The second variable ranges over the concept ‘age’.
- . . . The second variable is unitary.
- . . The first universal quantification is restricted by an atomic formulation.
- . . . The atomic formulation is based on the fact type ‘driver has age’.
- . . . The atomic formulation has a role binding.
- . . . . The role binding is of the role ‘driver’ of the fact type.
- . . . . The role binding binds to the first variable.

- ... The atomic formulation has a second role binding.
- ... The second role binding is of the role 'age' of the fact type.
- ... The second role binding binds to the second variable.
- .. The first universal quantification scopes over a second universal quantification.
- ... The second universal quantification introduces a third variable.
- ... The third variable ranges over the concept 'EU-Rent Minimum Driving Age'.
- ... The third variable is unitary.
- ... The second universal quantification scopes over an atomic formulation.
- ... The atomic formulation is based on the fact type 'quantity<sub>1</sub> ≥ quantity<sub>2</sub>'.
- ... The atomic formulation has a role binding.
- ... The role binding is of the role 'quantity<sub>1</sub>' of the fact type.
- ... The role binding binds to the second variable.
- ... The atomic formulation has a second role binding.
- ... The second role binding is of the role 'quantity<sub>2</sub>' of the fact type.
- ... The second role binding binds to the third variable.

The set projection that defines the characteristic is on a single variable. A set projection defining a binary fact type is on two variables, one mapped to each role. Note that the definition of the characteristic above uses two binary fact types, but each of the roles of those fact types are bound to variables introduced by the projection or by formulations within in, so the projection is closed and conveys a meaning.

SBVR does not attempt to provide special semantic formulations for tenses or the variety of ways states and events can relate to each other with respect to time or can be related to times, periods, and durations. However, an objectification is a logical formulation that enables a state or event indicated propositionally to be the subject or object of other propositions. An encompassing formulation can relate a state or event indicated using objectification to points in time, periods, and durations, or to another state or event (possibly also identified using objectification) with respect to time (e.g., occurring after or occurring before). The specific relations of interest can be defined as fact types. SBVR's treatment of time in relation to states and events allows temporal relations to be defined generically and orthogonally to the many fact types whose extensions change over time.

A propositional nominalization is similar to objectification. It is a kind of logical formulation that structures the meaning represented by a mention of a statement or proposition as opposed to a use of it. Other similar types of formulations structure meanings represented by mention of concepts, questions, and answers. Furthermore, rules about change often involve concept formulations, which are special formulations that allow concepts to be a subject or object of a proposition in much the same way that proposition nominalization allows propositions to a subject or object.

---

### Logical Formulation of Semantics Vocabulary

Language: [English](#)  
 Included Vocabulary: [Meaning and Representation Vocabulary](#)

---

## 9.1 Semantic Formulations

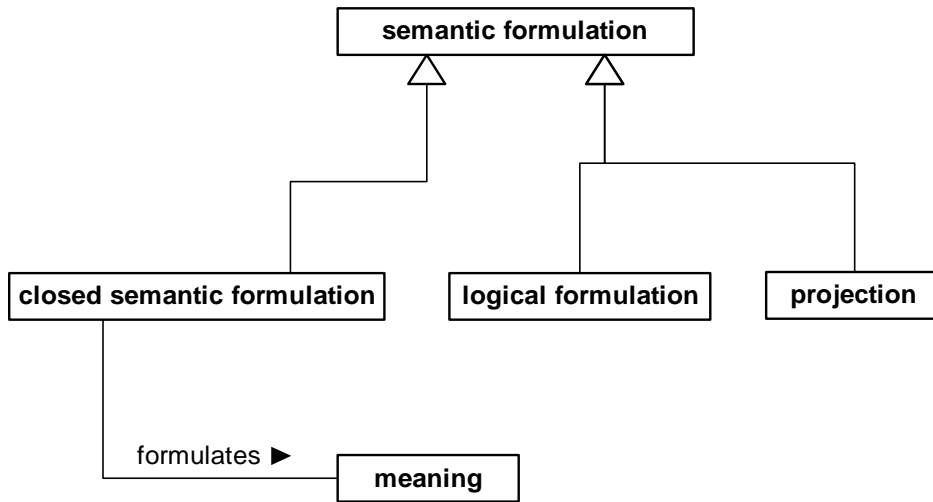


Figure 9.1

---

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

---

### semantic formulation

FL

Definition: conceptual structure of meaning

### closed semantic formulation

FL

Definition: semantic formulation that *includes no variable without binding*

### closed semantic formulation formulates meaning

Definition: the meaning is structured by the closed semantic formulation

Synonymous Form: meaning is formulated by closed semantic formulation

## 9.1.1 Logical Formulations

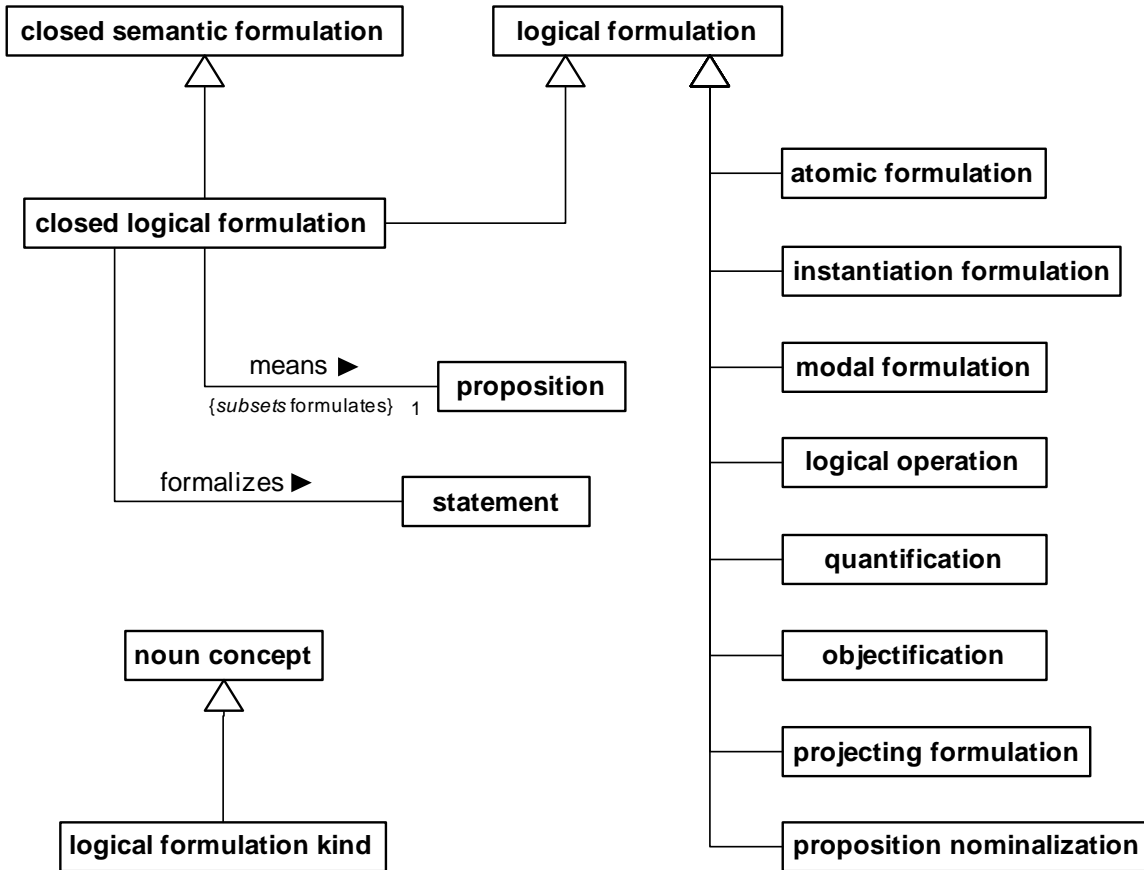


Figure 9.2

---

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

---

### logical formulation

- Definition: [semantic formulation](#) **that** is an abstract interpretation of a well-formed logical formula
- Note: Predicate and modal logics are covered. Mathematical formulations are based on mathematical fact types.
- Necessity: Each [logical formulation](#) **is an instance of exactly one** [logical formulation kind](#).

FL

### logical formulation kind

- Definition: [noun concept](#) **that specializes the concept 'logical formulation' and that** classifies a [logical formulation](#) based on the presence or absence of a main logical operation or quantification
- Note: The absence of a main logical operator occurs for an [atomic formulation](#) or [instantiation formulation](#).
- Example: [logical negation](#), [conjunction](#), [universal quantification](#)

FL

### **closed logical formulation**

FL

- Definition: logical formulation *that is* a closed semantic formulation
- Necessity: *The* meaning formulated by each closed logical formulation *is* a proposition.
- Necessity: *Each* closed logical formulation *means* *exactly one* proposition.

### **closed logical formulation *means* proposition**

FL

- Definition: *the* closed logical formulation *formulates* *the* proposition
- Synonymous Form: proposition *is meant by* closed logical formulation

### **closed logical formulation *formalizes* statement**

FL

- Definition: *the* closed logical formulation *means* *the* proposition *that is expressed by the* statement
- Synonymous Form: statement *is formalized by* closed logical formulation



### 9.1.1.1 Variables and Bindings

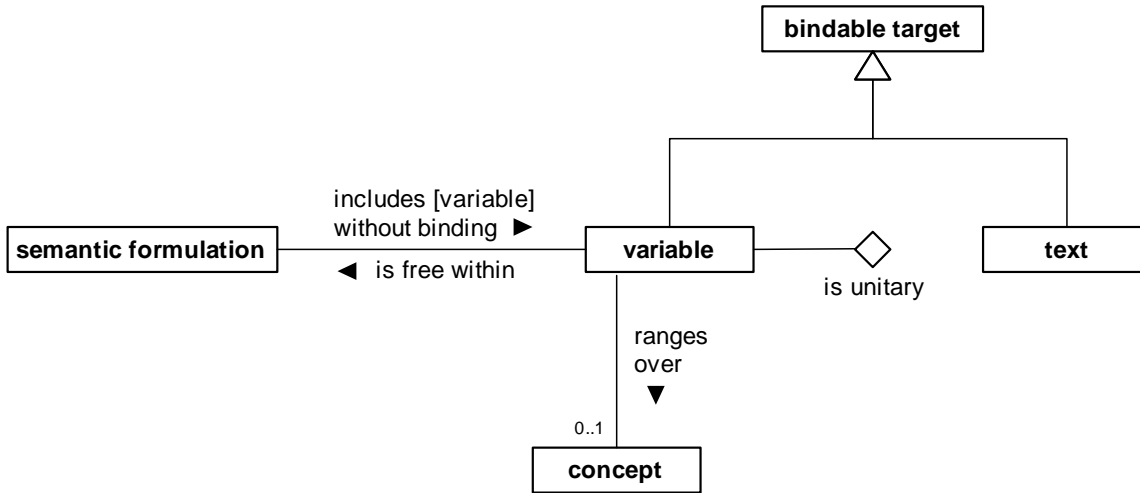


Figure 9.3

---

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

---

#### variable

FL

- Definition: reference to an element of a set, whose referent may vary or is unknown
- Necessity: Each variable *ranges over* at most one concept.
- Reference Scheme: a quantification that *introduces* the variable and the set of each concept that *is ranged over by* the variable and whether the variable *is unitary*
- Reference Scheme: a projection that *is over* the variable and the set of each concept that *is ranged over by* the variable and whether the variable *is unitary*

#### variable ranges over concept

FL

- Definition: each referent of the variable is necessarily an instance of the concept
- Synonymous Form: concept *is ranged over by* variable

#### variable is unitary

FL

- Definition: the variable is meant to have exactly one referent in the context where the variable is introduced
- Note: This characteristic used particularly in the formulation of definite descriptions.  
 If a set projection is on one variable and that variable is unitary, then the projection is meant to have exactly one result. For any other projection on a unitary variable, the projection is meant to have one referent for that variable for each referent of each other variable (including auxiliary variables) in the same projection.  
 If a unitary variable is introduced by a universal quantification, the variable ranges over a concept and is restricted by a logical formulation, then the quantification is satisfied if:

1. the unitary variable has exactly one referent, an instance of the concept, for which the restricting logical formulation is satisfied.
2. the logical formulation that the universal quantification scopes over is also satisfied for that one referent.

An exactly-one quantification introducing a non-unitary variable is satisfied differently:

1. the variable has at least one referent, an instance of the concept, for which the restricting logical formulation is satisfied.
2. the logical formulation that the exactly-one quantification scopes over is satisfied for exactly one referent from 1 above.

Example:

Given the individual concept ‘London-Heathrow Branch’ defined as “the EU-Rent branch located at London-Heathrow Airport”, the definition can be formulated as a projection on a variable that ranges over the concept ‘EU-Rent branch’. The variable is unitary indicating the sense of the definite article “the”. Based on this formulation, the concept is understood to be an individual concept. If the variable is not made unitary, then the formulation captures only the characteristic of being located at London-Heathrow Airport without any indication of the intended meaning that there is exactly one such branch.

Example:

A sensible projection formulating “the renter of a given rental” is on a unitary variable (renter) and has an auxiliary variable (rental). The rental variable being unitary indicates there is exactly one renter for each rental. But a set projection formulating “renter of at least one rental” is not on a unitary variable because the variable for rental is introduced within the logical formulation that constrains the projection and not by the projection itself. The projection result can include multiple renters and does not relate these to particular rentals.

Example:

A possible formulation of the rule, “The pick-up location of each rental must be a EU-Rent branch”, has a variable for ‘pick-up location’ that is unitary with respect to each rental as indicated by the use of the definite article “the”. The possible formulation is an obligation claim that embeds a universal quantification introducing a variable ranging over the concept “rental” and that embeds a second universal quantification introducing a second variable that is restricted by an atomic formulation based on the fact type “rental has pick-up location”. That second variable is unitary indicating that exactly one pick-up location is meant for each rental. The second universal quantification scopes over a formulation of the pick-up location being a EU-Rent branch. The overall formulation applies the obligation claim to the pick-up location being an EU-Rent branch. It does not apply the obligation claim to there being one pick-up branch per rental, which is understood structurally as what is meant in the expression of the rule and not part of the obligation.

Note that if the universal quantifications of the formulation above are reversed such that a quantification introducing the variable for ‘pick-up location’ embeds the quantification introducing the variable for ‘rental’, then the variable for ‘pick-up rental’ is not unitary because it would have multiple referents (one for each distinct pick-up location). Such a formulation would not properly capture the sense of the rule statement.

### variable is free within semantic formulation

FL

Definition: [the semantic formulation](#) employs [the variable](#), but does not introduce it

Synonymous Form: [semantic formulation includes variable without binding](#)

### bindable target

FL

Definition: [variable or text](#)

### 9.1.1.2 Atomic Formulations

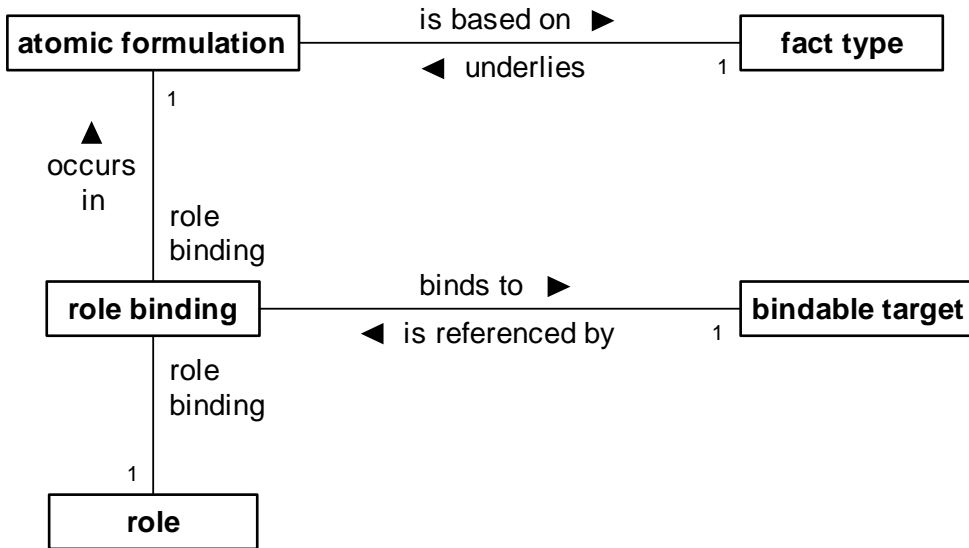


Figure 9.4

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

#### atomic formulation

FL

Definition:	<u>logical formulation</u> that is based directly upon a <u>fact type</u> , thus involving no logical operation
Concept Type:	<u>logical formulation kind</u>
Necessity:	Each <u>atomic formulation</u> is based on exactly one <u>fact type</u> .
Reference Scheme:	the set of each <u>role binding</u> of the <u>atomic formulation</u>
Note:	The meaning invoked by an atomic formulation puts each referent of each role binding in its respective role. Where a role specializes some other concept, that meaning implies (as a separate secondary meaning) that the referent of the role binding for that role is an instance of the other concept.
Example:	Consider an atomic formulation based on a binary fact type that is the meaning of the verb “employs”. There is an “employer” role played by an organization and an “employee” role played by a person. The direct meaning invoked by an atomic formulation puts one referent in the “employer” role and another in the “employee” role. That the employer is an organization and the employee is a person is a secondary meaning implied by the direct meaning.

#### atomic formulation has role binding

FL

Definition:	the <u>atomic formulation</u> includes the <u>role binding</u> for a particular <u>role</u> of the <u>fact type</u> that is the basis of the <u>atomic formulation</u>
Synonymous Form:	<u>role binding occurs in atomic formulation</u>

### atomic formulation is based on fact type

FL

Definition: the meaning invoked by the atomic formulation is that of the fact type  
Synonymous Form: fact type underlies atomic formulation

### role binding

FL

Definition: connection of an atomic formulation to a bindable target  
Necessity: Each role binding occurs in exactly one atomic formulation.  
Necessity: Each role binding is of a role of the fact type that underlies the atomic formulation that has the role binding.  
Necessity: Each role binding binds to exactly one bindable target.  
Necessity: Each role binding is of exactly one role.  
Necessity: Each variable that is referenced by a role binding of an atomic formulation is free within the atomic formulation.  
Reference Scheme: the bindable target that is referenced by the role binding and the role that has the role binding

### role binding binds to bindable target

FL

Definition: the bindable target provides what thing fills the role for propositions meant by the atomic formulation that has the role binding  
Synonymous Form: bindable target is referenced by role binding  
Note: The meaning of a role binding to a variable is that a referent of the variable is the thing involved in the role. The meaning of a role binding to a text is that the thing involved in the role is the text itself.

### role binding is of role

FL

Definition: the role binding is a binding of the role of the fact type that underlies an atomic formulation

## 9.1.1.3 Instantiation Formulations

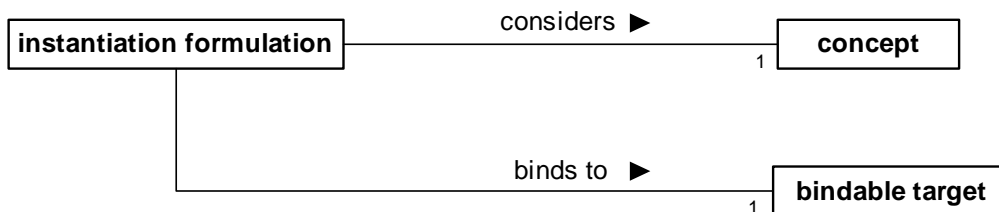


Figure 9.5

---

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

---

### instantiation formulation

FL

Definition: logical formulation indicating that a referent thing is an instance of a particular concept  
Concept Type: logical formulation kind  
Necessity: Each instantiation formulation considers exactly one concept.

Necessity: Each instantiation formulation binds to exactly one bindable target.

Necessity: Each variable that is bound to an instantiation formulation is free within the instantiation formulation.

Reference Scheme: the bindable target that is bound to the instantiation formulation and the concept that is considered by the instantiation formulation

Note: An instantiation formulation is equivalent to an existential quantification that introduces a variable ranging over the concept considered by the instantiation formulation and that scopes over an atomic formulation based on the fact type 'thing is thing' where one role binding is to the variable and the other is to the bindable target bound to the instantiation formulation.

Example: A formulation of the statement, "A customer is a preferred customer if ...", uses an instantiation formulation to conceptualize a customer being an instance of the concept 'preferred customer'.

### instantiation formulation considers concept

FL

Definition: the instantiation formulation classifies things to be an instance of the concept

Synonymous Form: concept is considered by instantiation formulation

### instantiation formulation binds to bindable target

FL

Definition: the bindable target indicates what thing is being classified by the instantiation formulation

Synonymous Form: bindable target is bound to instantiation formulation

## 9.1.1.4 Modal Formulations

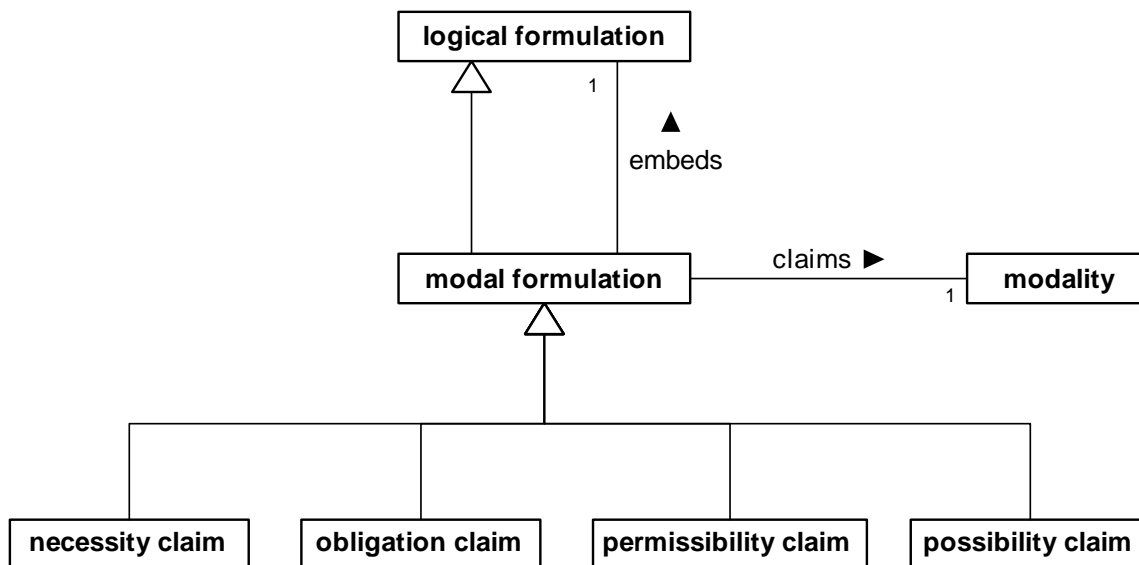


Figure 9.6

---

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

---

## **modal formulation**

FL

- Definition: [logical formulation](#) that applies a [modality](#) to an another [logical formulation](#)
- Necessity: Each [modal formulation](#) *claims* exactly one [modality](#).
- Necessity: Each [modal formulation](#) *embeds* exactly one [logical formulation](#).
- Necessity: Each [variable](#) that *is free within a* [logical formulation](#) that *is embedded in a* [modal formulation](#) *is free within the* [modal formulation](#).
- Note: The [meaning](#) of a [modal formulation](#) is the [proposition](#) that the [proposition](#) meant by the embedded [logical formulation](#) is an instance of the [modality](#) claimed by the [modal formulation](#).

## **modal formulation claims modality**

FL

- Definition: the [modal formulation](#) *relates* to the [modality](#) that is claimed to be applicable to the embedded [logical formulation](#)
- Synonymous Form: [modality](#) *is claimed by* [modal formulation](#)

## **modal formulation embeds logical formulation**

FL

- Definition: the [modality](#) claimed by the [modal formulation](#) is applied to the [logical formulation](#)
- Synonymous Form: [logical formulation](#) *is embedded in* [modal formulation](#)

## **necessity claim**

FL

- Definition: [modal formulation](#) that *claims* the [modality](#) '[necessity](#)'
- Concept Type: [logical formulation kind](#)
- Reference Scheme: the [logical formulation](#) that *is embedded in* the [necessity claim](#)

## **obligation claim**

FL

- Definition: [modal formulation](#) that *claims* the [modality](#) '[obligation](#)'
- Concept Type: [logical formulation kind](#)
- Reference Scheme: the [logical formulation](#) that *is embedded in* the [obligation claim](#)

## **permissibility claim**

FL

- Definition: [modal formulation](#) that *claims* the [modality](#) '[permissibility](#)'
- Concept Type: [logical formulation kind](#)
- Reference Scheme: the [logical formulation](#) that *is embedded in* the [permissibility claim](#)

## **possibility claim**

FL

- Definition: [modal formulation](#) that *claims* the [modality](#) '[possibility](#)'
- Concept Type: [logical formulation kind](#)
- Reference Scheme: the [logical formulation](#) that *is embedded in* the [possibility claim](#)

### **9.1.1.5 Logical Operands**

## **logical operand**

FL

- Definition: [logical formulation](#) upon which a [logical operation](#) operates
- Concept Type: [role](#)

### **logical operand 1**

Definition: [logical operand](#) **that** is the first of at least two operands to a [logical operation](#)  
Concept Type: [role](#)  
Necessity: Each [logical operation](#) *has at most one* [logical operand 1](#).

FL

### **logical operand 2**

Definition: [logical operand](#) **that** is the second of at least two operands to a [logical operation](#)  
Concept Type: [role](#)  
Necessity: Each [logical operation](#) *has at most one* [logical operand 2](#).

FL

### **antecedent**

Definition: [logical operand](#) **that** is the condition considered by a [logical operation](#) such as an [implication](#) (e.g, what is meant by the  $p$  in “if  $p$  then  $q$ ”)  
Concept Type: [role](#)

FL

### **condition 1**

Definition: [logical operand 1](#) **that** is the first of two operands to an [equivalence](#)  
Concept Type: [role](#)

FL

### **condition 2**

Definition: [logical operand 2](#) **that** is the second of two operands to an [equivalence](#)  
Concept Type: [role](#)

FL

### **conjunct 1**

Definition: [logical operand 1](#) **that** is the first of two operands to a [conjunction](#)  
Concept Type: [role](#)

FL

### **conjunct 2**

Definition: [logical operand 2](#) **that** is the second of two operands to a [conjunction](#)  
Concept Type: [role](#)

FL

### **consequent**

Definition: [logical operand](#) **that** is the implied or result operand to a [logical operation](#) such as an [implication](#) (e.g., what is meant by the  $q$  in “if  $p$  then  $q$ ”)  
Concept Type: [role](#)

FL

### **disjunct 1**

Definition: [logical operand 1](#) **that** is the first of two operands to a [disjunction](#)  
Concept Type: [role](#)

FL

### **disjunct 2**

Definition: [logical operand 2](#) **that** is the second of two operands to a [disjunction](#)  
Concept Type: [role](#)

FL

### **exclusive disjunction 1**

Definition: [logical operand 1](#) **that** is the first of two operands to an [exclusive disjunction](#)

FL

Concept Type: [role](#)

### **exclusive disjunction 2**

FL

Definition: [logical operand 2](#) **that** is the second of two operands to an [exclusive disjunction](#)

Concept Type: [role](#)

### **inconsequent**

FL

Definition: [logical operand](#) **that** is an operand irrelevant to the logical result of a [logical operation](#) such as of a [whether-or-not formulation](#)

Concept Type: [role](#)

### **negand**

FL

Definition: [logical operand](#) **that** is the operand of a [logical negation](#)

Concept Type: [role](#)



### 9.1.1.6 Logical Operations

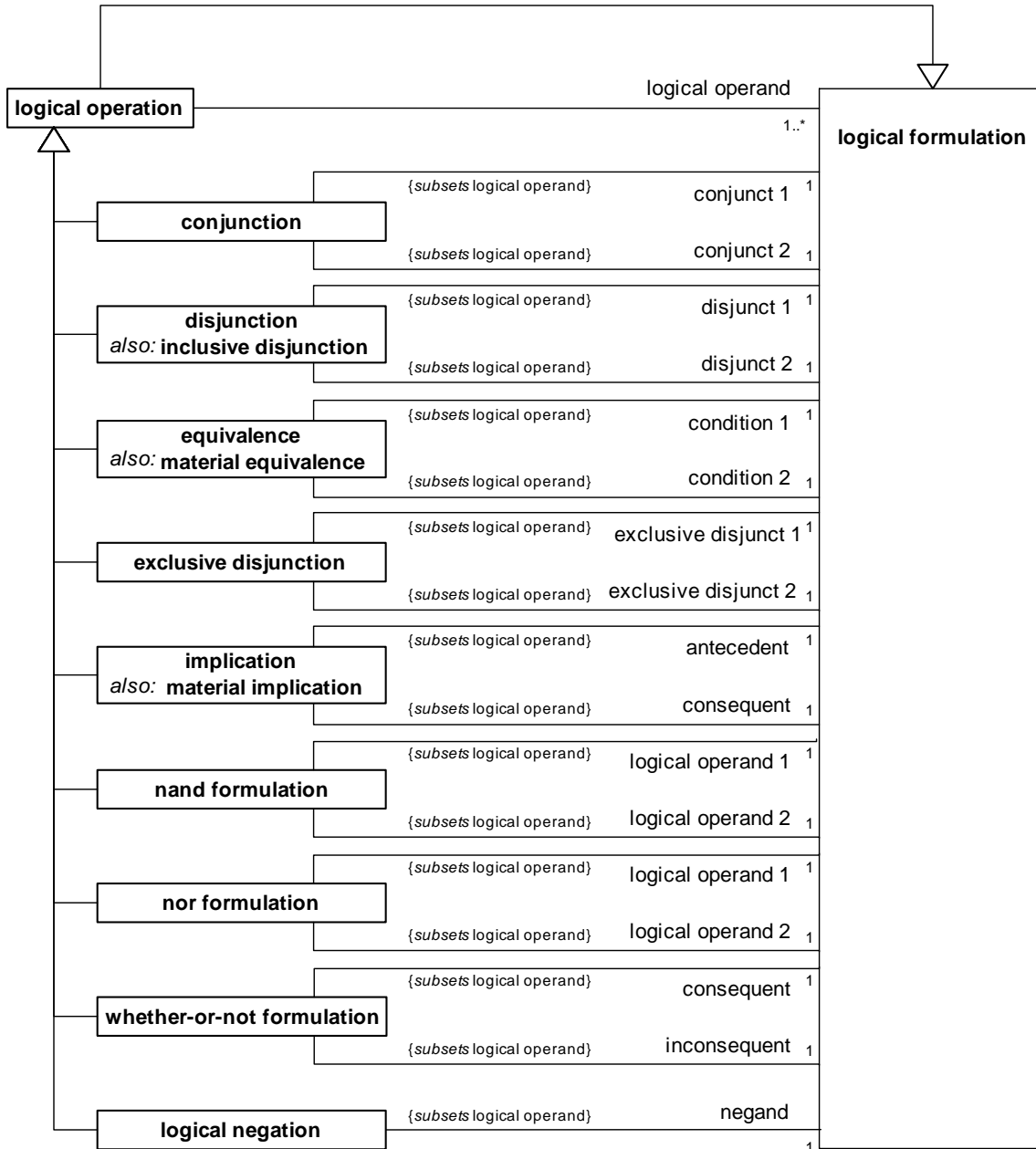


Figure 9.7

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

#### logical operation

Definition: logical formulation that operates on a logical operand

FL

Necessity: Each [logical operation](#) *has* at least one [logical operand](#).  
Necessity: Each [variable](#) that *is free within* a [logical operand](#) of a [logical operation](#) *is free within* the [logical operation](#).

### **logical operation has logical operand**

FL

Definition: the [logical operation](#) operates on the [logical operand](#)

### **conjunction**

FL

Definition: [logical operation](#) that applies the logical 'AND' operation (&) to a [conjunct<sub>1</sub>](#) and a [conjunct<sub>2</sub>](#)

Concept Type: [logical formulation kind](#)

Necessity: Each [conjunction](#) *has* exactly one [conjunct 1](#).

Necessity: Each [conjunction](#) *has* exactly one [conjunct 2](#).

Reference Scheme: the [conjunct 1](#) of the [conjunction](#) and the [conjunct 2](#) of the [conjunction](#)

### **conjunction has conjunct 1**

FL

Definition: the [conjunction](#) operates on the [conjunct 1](#)

Synonymous Form: [conjunction](#) *has* [logical operand 1](#)

### **conjunction has conjunct 2**

FL

Definition: the [conjunction](#) operates on the [conjunct 2](#)

Synonymous Form: [conjunction](#) *has* [logical operand 2](#)

### **disjunction**

FL

Definition: [logical operation](#) that applies the logical 'INCLUSIVE OR' operation ( $\vee$ ) to a [disjunct 1](#) and a [disjunct 2](#)

Concept Type: [logical formulation kind](#)

Synonym: [inclusive disjunction](#)

Necessity: Each [disjunction](#) *has* exactly one [disjunct 1](#).

Necessity: Each [disjunction](#) *has* exactly one [disjunct 2](#).

Reference Scheme: the [disjunct 1](#) of the [disjunction](#) and the [disjunct 2](#) of the [disjunction](#)

### **disjunction has disjunct 1**

FL

Definition: the [disjunction](#) operates on the [disjunct 1](#)

Synonymous Form: [disjunction](#) *has* [logical operand 1](#)

### **disjunction has disjunct 2**

FL

Definition: the [disjunction](#) operates on the [disjunct 2](#)

Synonymous Form: [disjunction](#) *has* [logical operand 2](#)

### **equivalence**

FL

Definition: [logical operation](#) that applies the logical '(MATERIAL) EQUIVALENCE' operation ( $\equiv$ ) to a [condition 1](#) and a [condition 2](#)

Concept Type: [logical formulation kind](#)

Synonym: [material equivalence](#)

Necessity: Each [equivalence](#) *has* exactly one [condition 1](#).  
Necessity: Each [equivalence](#) *has* exactly one [condition 2](#).  
Reference Scheme: [the condition 1 of the equivalence](#) and [the condition 2 of the equivalence](#)

### **equivalence has condition 1**

FL

Definition: [the equivalence](#) operates on [the condition 1](#)  
Synonymous Form: [equivalence has logical operand 1](#)

### **equivalence has condition 2**

FL

Definition: [the equivalence](#) operates on [the condition 2](#)  
Synonymous Form: [equivalence has logical operand 2](#)

### **exclusive disjunction**

FL

Definition: [logical operation](#) that applies the logical ‘EXCLUSIVE OR’ operation ( $\vee$ ) to an [exclusive disjunct 1](#) and an [exclusive disjunct 2](#)  
Concept Type: [logical formulation kind](#)  
Necessity: Each [exclusive disjunction](#) *has* exactly one [exclusive disjunct 1](#).  
Necessity: Each [exclusive disjunction](#) *has* exactly one [exclusive disjunct 2](#).  
Reference Scheme: [the exclusive disjunct 1 of the exclusive disjunction](#) and [the exclusive disjunct 2 of the exclusive disjunction](#)

### **exclusive disjunction has exclusive disjunct 1**

FL

Definition: [the exclusive disjunction](#) operates on [the exclusive disjunct 1](#)  
Synonymous Form: [exclusive disjunct has logical operand 1](#)

### **exclusive disjunction has exclusive disjunct 2**

FL

Definition: [the exclusive disjunction](#) operates on [the exclusive disjunct 2](#)  
Synonymous Form: [exclusive disjunct has logical operand 2](#)

### **implication**

FL

Definition: [logical operation](#) that applies the logical ‘(MATERIALLY) IMPLIES’ operation ( $\Rightarrow$ ) to an [antecedent](#) and a [consequent](#)  
Concept Type: [logical formulation kind](#)  
Synonym: [material implication](#)  
Necessity: Each [implication](#) *has* exactly one [antecedent](#).  
Necessity: Each [implication](#) *has* exactly one [consequent](#).  
Reference Scheme: [the antecedent of the implication](#) and [the consequent of the implication](#)

### **implication has antecedent**

FL

Definition: [the implication](#) operates on [the antecedent](#)  
Synonymous Form: [implication has logical operand 1](#)

### **implication has consequent**

FL

Definition: [the implication](#) operates on [the consequent](#)  
Synonymous Form: [implication has logical operand 2](#)

## logical negation

FL

- Definition: [logical operation](#) that applies the logical ‘NOT’ operation ( $\sim$ ) to a [negand](#)
- Concept Type: [logical formulation kind](#)
- Necessity: Each [logical negation](#) has exactly one [negand](#).
- Reference Scheme: the [negand](#) of the [logical negation](#)

## logical negation has negand

FL

- Definition: the [logical negation](#) operates on the [negand](#)
- Synonymous Form: [logical negation](#) has [logical operand](#)

## nand formulation

FL

- Definition: [logical operation](#) that applies the logical ‘NAND’ operation ( $\downarrow$ ) to a [logical operand 1](#) and a [logical operand 2](#)
- Concept Type: [logical formulation kind](#)
- Necessity: Each [nand formulation](#) has exactly one [logical operand 1](#).
- Necessity: Each [nand formulation](#) has exactly one [logical operand 2](#).
- Reference Scheme: the [logical operand 1](#) of the [nand formulation](#) and the [logical operand 2](#) of the [nand formulation](#)

## nand formulation has logical operand 1

FL

- Definition: the [nand formulation](#) operates on the [logical operand 1](#)

## nand formulation has logical operand 2

FL

- Definition: the [nand formulation](#) operates on the [logical operand 2](#)

## nor formulation

FL

- Definition: [logical operation](#) that applies the logical ‘NOR’ operation ( $\Downarrow$ ) to a [logical operand 1](#) and a [logical operand 2](#)
- Concept Type: [logical formulation kind](#)
- Necessity: Each [nor formulation](#) has exactly one [logical operand 1](#).
- Necessity: Each [nor formulation](#) has exactly one [logical operand 2](#).
- Reference Scheme: the [logical operand<sub>1</sub>](#) of the [nor formulation](#) and the [logical operand<sub>2</sub>](#) of the [nor formulation](#)

## nor formulation has logical operand 1

FL

- Definition: the [nor formulation](#) operates on the [logical operand 1](#)

## nor formulation has logical operand 2

FL

- Definition: the [nor formulation](#) operates on the [logical operand 2](#)

## whether-or-not formulation

FL

- Definition: [logical operation](#) that applies the logical whether-or-not operator ( $\Omega$ ) to a [consequent](#) and an [inconsequent](#)
- Concept Type: [logical formulation kind](#)
- Necessity: Each [whether-or-not formulation](#) has exactly one [consequent](#).

Necessity: Each whether-or-not formulation *has* exactly one inconsequent.  
Reference Scheme: the consequent of the whether-or-not formulation and the inconsequent of the whether-or-not formulation

**whether-or-not formulation *has* consequent**

FL

Definition: the whether-or-not formulation operates on the consequent  
Synonymous Form: whether-or-not formulation *has* logical operand 1

**whether-or-not formulation *has* inconsequent**

FL

Definition: the whether-or-not formulation operates on the inconsequent  
Synonymous Form: whether-or-not formulation *has* logical operand 2

### 9.1.1.7 Quantifications

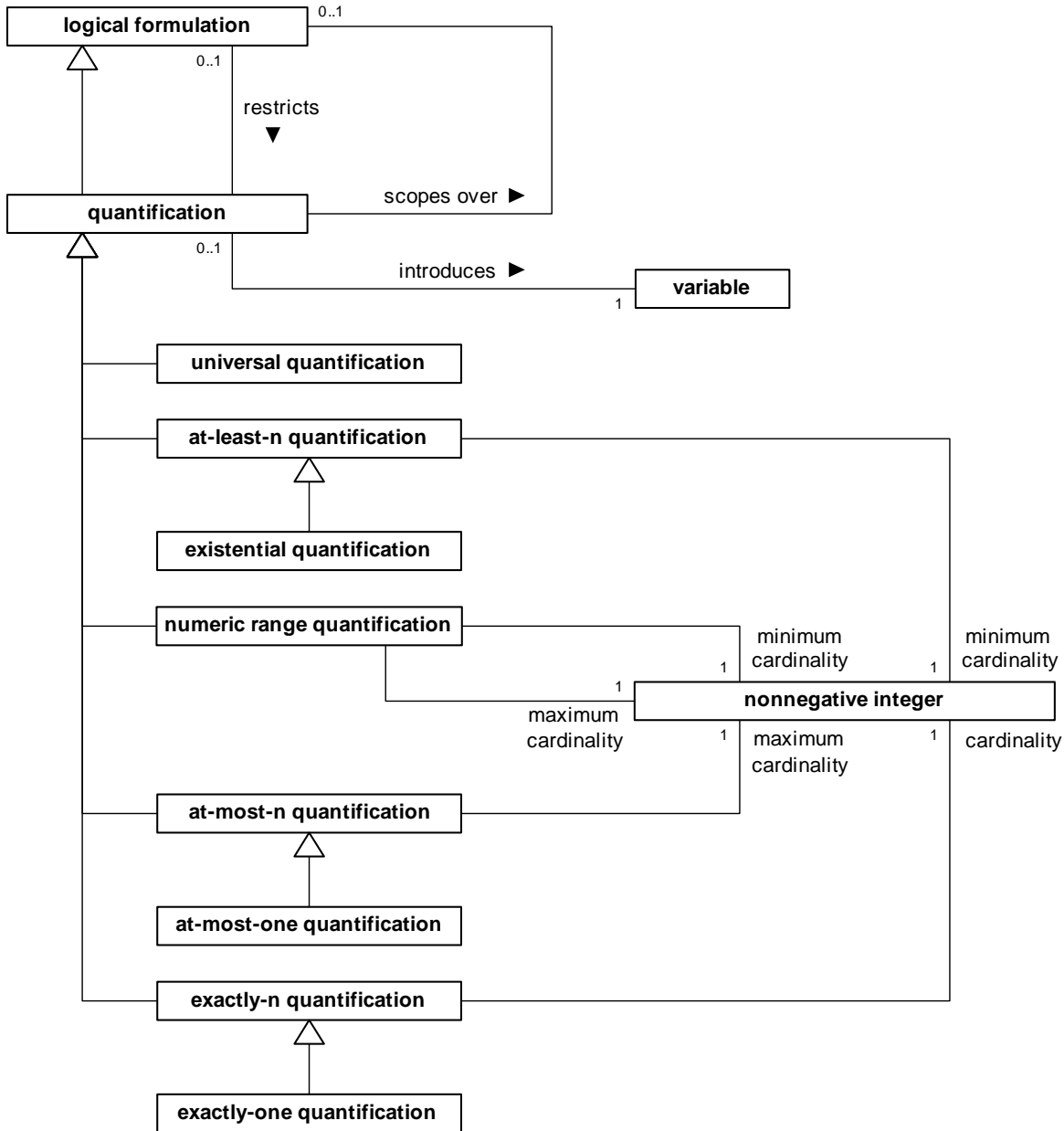


Figure 9.8

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

#### quantification

FL

Definition: [logical formulation](#) that applies a logical quantification operation to a [variable](#)

Necessity: Each [quantification](#) introduces exactly one [variable](#).

- Necessity: Each variable is introduced by at most one quantification.
- Necessity: Each quantification scopes over at most one logical formulation.
- Necessity: Each quantification is restricted by at most one logical formulation.
- Necessity: A variable that is free within a logical formulation that is scoped over by a quantification is free within the quantification if and only if the quantification does not introduce the variable.
- Necessity: A variable that is free within a logical formulation that restricts a quantification is free within the quantification if and only if the quantification does not introduce the variable.

### quantification introduces variable

FL

- Definition: the quantification is over referents of the variable
- Synonymous Form: variable is introduced by quantification

### logical formulation restricts quantification

FL

- Definition: results of the quantification consider only the referents of the variable introduced by the quantification for which the logical formulation is satisfied
- Synonymous Form: quantification is restricted by logical formulation
- Note: A logical formulation restricts a quantification in the same way that a concept that is ranged over by a variable restricts the quantification that introduces the variable. In the formulation of a statement, a restricting logical formulation is used to capture the sense of a restrictive clause in a way that does not confuse it with the primary sense of the statement.
- Example: A possible formulation of the rule, “Each scheduled drop-off location of each rental must be a EU-Rent branch”, includes a universal quantification introducing a variable for ‘rental’ that scopes over another universal quantification introducing a variable for ‘scheduled drop-off location’. The second quantification is restricted by a logical formulation so that for each rental the result considers only scheduled drop-off locations of that rental.

### quantification scopes over logical formulation

FL

- Definition: the overall scope of the quantification is the logical formulation
- Synonymous Form: logical formulation is scoped over by quantification
- Note: A quantification other than a universal quantification does not necessarily scope over a logical formulation (e.g., formulation of “some customer exists” can simply be an existential quantification introducing a variable that ranges over the concept ‘customer’).

### universal quantification

FL

- Definition: quantification that applies the universal quantification operation ( $\forall$ ) scoping over a logical formulation
- Concept Type: logical formulation kind
- Necessity: Each universal quantification scopes over a logical formulation.
- Reference Scheme: the logical formulation that is scoped over by the universal quantification and the variable that is introduced by the universal quantification and the set of each logical formulation that restricts the universal quantification

### existential quantification

FL

- Definition: at-least-n quantification that applies the existential quantification operation ( $\exists$ ), ‘n’ being 1
- Concept Type: logical formulation kind

Reference Scheme: [the set of each logical formulation that is scoped over by the existential quantification and the variable that is introduced by the existential quantification](#) and [the set of each logical formulation that restricts the existential quantification](#)

### cardinality

FL

Definition: [nonnegative integer](#) that is a number of elements in a collection

Concept Type: [role](#)

### maximum cardinality

FL

Definition: [cardinality](#) that is a maximum in a range of cardinalities, such as for an [at-most-n quantification](#)

Concept Type: [role](#)

### minimum cardinality

FL

Definition: [cardinality](#) that is a minimum in a range of cardinalities, such as for an [at-least-n quantification](#)

Concept Type: [role](#)

### at-least-n quantification

FL

Definition: [quantification](#) that applies the ‘at least  $n$ ’ quantification operation ( $\exists^{\geq n}$ ), ‘ $n$ ’ representing a [minimum cardinality](#)

Concept Type: [logical formulation kind](#)

Necessity: Each [at-least-n quantification](#) has exactly one [minimum cardinality](#).

Necessity: The [minimum-cardinality of each at-least-n quantification](#) is a positive integer.

Reference Scheme: [the minimum cardinality of the at-least-n quantification](#) and [the set of each logical formulation that is scoped over by the at-least-n quantification and the variable that is introduced by the at-least-n quantification](#) and [the set of each logical formulation that restricts the at-least-n quantification](#)

### at-least-n quantification has minimum cardinality

FL

Definition: [the at-least-n quantification](#) is satisfied by [the minimum cardinality](#) or greater

### at-most-n quantification

FL

Definition: [quantification](#) that applies the ‘AT MOST  $n$ ’ quantification operation ( $\exists^{\leq n}$ ), ‘ $n$ ’ representing a [maximum cardinality](#)

Concept Type: [logical formulation kind](#)

Necessity: Each [at-most-n quantification](#) has exactly one [maximum cardinality](#).

Necessity: The [maximum-cardinality of each at-most-n quantification](#) is a positive integer.

Reference Scheme: [the maximum cardinality of the at-most-n quantification](#) and [the set of each logical formulation that is scoped over by the at-most-n quantification and the variable that is introduced by the at-most-n quantification](#) and [the set of each logical formulation that restricts the at-most-n quantification](#)

### at-most-n quantification has maximum cardinality

FL

Definition: [the at-most-n quantification](#) is satisfied by [the maximum cardinality](#) or less



### at-most-one quantification

FL

- Definition: at-most-n quantification that applies the ‘AT MOST ONE’ quantification operation ( $\exists^{0..1}$ ), ‘n’ being 1
- Concept Type: logical formulation kind
- Reference Scheme: the set of each logical formulation that is scoped over by the at-most-one quantification and the variable that is introduced by the at-most-one quantification and the set of each logical formulation that restricts the at-most-one quantification
- Note: An at-most-one quantification is logically equivalent to an at-most-n quantification that has a maximum cardinality of 1.

### exactly-n quantification

FL

- Definition: quantification that applies the ‘EXACTLY n’ quantification operation ( $\exists^n$ ), ‘n’ representing a cardinality
- Concept Type: logical formulation kind
- Necessity: Each exactly-n quantification has exactly one cardinality.
- Necessity: The cardinality of each exactly-n quantification is a positive integer.
- Reference Scheme: the cardinality of the exactly-n quantification and the set of each logical formulation that is scoped over by the exactly-n quantification and the variable that is introduced by the exactly-n quantification and the set of each logical formulation that restricts the exactly-n quantification
- Note: An exactly-n quantification is logically equivalent to a conjunction of an at-least-n quantification and an at-most-n quantification using the cardinality as minimum cardinality and maximum cardinality respectively.

### exactly-n quantification has cardinality

FL

- Definition: the exactly-n quantification is satisfied only by the cardinality

### exactly-one quantification

FL

- Definition: exactly-n quantification that applies the ‘EXACTLY 1’ quantification operation ( $\exists^1$ ), ‘n’ being 1
- Concept Type: logical formulation kind
- Reference Scheme: the set of each logical formulation that is scoped over by the exactly-one quantification and the variable that is introduced by the exactly-one quantification and the set of each logical formulation that restricts the exactly-one quantification
- Note: An exactly-one quantification is logically equivalent to an exactly-n quantification that has a cardinality of 1.

### numeric range quantification

FL

- Definition: quantification that applies the ‘NUMERIC RANGE’ quantification operation ( $\exists^{n..m}$ ), ‘n’ representing a minimum cardinality and ‘m’ representing a maximum cardinality
- Concept Type: logical formulation kind
- Necessity: Each numeric range quantification has exactly one maximum cardinality.
- Necessity: Each numeric range quantification has exactly one minimum cardinality.
- Necessity: The minimum cardinality of each numeric range quantification is less than the maximum cardinality of the numeric range quantification.

Reference Scheme: the minimum cardinality of the numeric range quantification and the maximum cardinality of the numeric range quantification and the set of each logical formulation that is scoped over by the numeric range quantification and the variable that is introduced by the numeric range quantification and the set of each logical formulation that restricts the numeric range quantification

Note: A numeric range quantification is logically equivalent to a conjunction of an at-least-n quantification and an at-most-n quantification using the minimum cardinality and maximum cardinality respectively.

### numeric range quantification has maximum cardinality

FL

Definition: the numeric range quantification cannot be satisfied by a number greater than the maximum cardinality

### numeric range quantification has minimum cardinality

FL

Definition: the numeric range quantification cannot be satisfied by a number less than the minimum cardinality

### 9.1.1.8 Objectifications

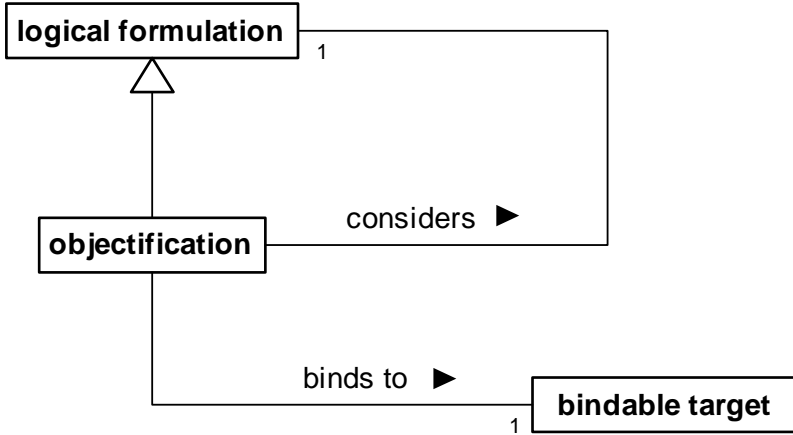


Figure 9.9

---

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

---

#### objectification

FL

- Definition: [logical formulation](#) that a referent state of affairs or event corresponds to the meaning of another [logical formulation](#)
- Concept Type: [logical formulation kind](#)
- Necessity: Each [objectification](#) *considers* exactly one [logical formulation](#).
- Necessity: Each [objectification](#) *binds to* exactly one [bindable target](#).
- Necessity: Each [variable](#) that *is bound to* an [objectification](#) *is free within* the [objectification](#).
- Necessity: Each [variable](#) that *is free within* the [logical formulation](#) of an [objectification](#) *is free within* the [objectification](#).
- Reference Scheme: the [bindable target](#) that *is bound to* the [objectification](#) and the [logical formulation](#) that *is considered by* the [objectification](#)
- Example: In the statement, “A car assignment of a rental car to a rental is an actuality that the car is assigned to the rental”, an [objectification](#) considers a [logical formulation](#) formulating “the car is assigned to the rental” with respect to a referent of a variable that ranges over the concept ‘actuality’. The objectification is satisfied if the referent is the actuality corresponding with the meaning of the logical formulation.

#### objectification considers logical formulation

FL

- Definition: the [objectification](#) is of the state or event that corresponds to the meaning of the [logical formulation](#)
- Synonymous Form: [logical formulation](#) *is considered by* [objectification](#)

#### objectification binds to bindable target

FL

- Definition: the [bindable target](#) indicates the referent state or event identified by the [objectification](#)
- Synonymous Form: [bindable target](#) *is bound to* [objectification](#)

### 9.1.1.9 Projecting Formulations

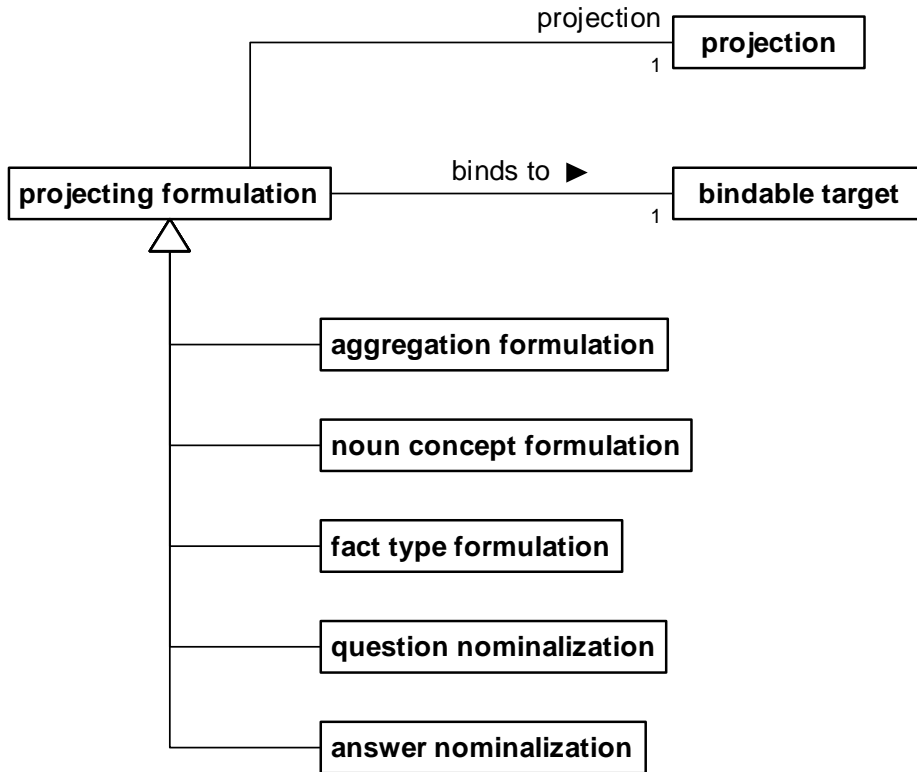


Figure 9.10

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

#### projecting formulation

FL

- Definition: logical formulation of a referent thing considered with respect to a particular projection
- Necessity: Each projecting formulation *has exactly one* projection.
- Necessity: Each projecting formulation *binds to exactly one* bindable target.
- Necessity: Each variable that *is bound to a* projecting formulation *is free within the* projecting formulation.
- Necessity: Each variable that *is free within the* projection of a projecting formulation *is free within the* projecting formulation.
- Example: See 'aggregation formulation', 'question nominalization' and 'answer nominalization'.

#### projecting formulation *has* projection

FL

- Definition: *the* projecting formulation is based on the projection

#### projecting formulation *binds to* bindable target

FL

- Definition: *the* bindable target indicates the referent thing considered by *the* projecting formulation

Synonymous Form: [bindable target is bound to projecting formulation](#)

### aggregation formulation

FL

- Definition: [projecting formulation](#) that a referent set or multiset is the result of a particular [projection](#)
- Concept Type: [logical formulation kind](#)
- Necessity: [The projection of each aggregation formulation is on exactly one variable.](#)
- Note: An aggregation formulation is satisfied for each referent set or multiset that exactly matches the result of the projection, taking the result as the set or multiset of referents of the variable in the projection.
- Reference Scheme: [the bindable target that is bound to the aggregation formulation and the projection of the aggregation formulation](#)
- Example: In the statement, “The average of ages of rental cars owned by each local area must be less than 5 years”, the ages of rental cars are aggregated into a multiset for each local area. A [bag projection](#) structures the meaning of “ages of rental cars”. An [aggregation formulation](#) considers that projection and binds to a variable such that each satisfying referent is the multiset determined by the projection. An atomic formulation can then be used to relate the average to the multiset of ages.

### noun concept formulation

FL

- Definition: [projecting formulation](#) of a referent noun concept whose intension is formulated in a particular [projection](#)
- Concept Type: [logical formulation kind](#)
- Necessity: [The projection of each noun concept formulation is on exactly one variable.](#)
- Note: A noun concept formulation is satisfied for each referent that is a noun concept defined by the projection. For a [closed projection](#), [the projection defines the noun concept](#). Otherwise, a satisfying noun concept is the meaning formulated by the projection plus an understood reference to the referent of each variable that is free within the projection.
- Reference Scheme: [the bindable target that is bound to the noun concept formulation and the projection of the noun concept formulation](#)
- Example: In the condition, “If a corporate renter that is accredited under a corporate rental agreement becomes a corporate customer that contracts for that corporate rental agreement then ...”, the object of “becomes” is not a corporate customer but a noun concept that specializes the concept ‘corporate customer’. The noun concept denotes those corporate customers that contract for the particular agreement.
- Example: In the condition, “If a rental car’s odometer reading increases by more than 1000 kilometers during a rental period then ...”, the subject of “increases” is not a distance, but an individual concept of a distance. An instance of the fact type ‘[individual quantity concept increases by quantity during period](#)’ would, in this case, not involve a particular distance, but an individual concept a distance denoting one particular distance at any single point in time — the one odometer reading of one particular rental car. The condition considers change over time in the extension of that individual concept.
- Example: In the statement, “The local area that owns a rental car must not change”, the subject of “change” is not a local area, but an individual concept of a local area. An instance of the fact type ‘[thing \[individual concept\] changes](#)’ would, in this case, not be identified by a particular local area, but by an individual concept of a local area — the one local area owning a particular rental car. This is an individual concept with respect to each rental car (which is necessarily owned by exactly one local area). The statement requires that the extension of that individual concept not change.

## fact type formulation

FL

- Definition: [projecting formulation](#) of a referent fact type whose intension is formulated in a particular [projection](#)
- Concept Type: [logical formulation kind](#)
- Necessity: [The projection of each fact type formulation is a set projection.](#)
- Note: A fact type formulation is satisfied for each referent that is a fact type defined by the projection. For a [closed projection](#), the [projection defines the fact type](#). Otherwise, a satisfying fact type is the meaning formulated by the projection plus an understood reference to the referent of each variable that is free within the projection.
- Reference Scheme: [the bindable target that is bound to the fact type formulation and the projection of the fact type formulation](#)
- Example: In the statement, “drinking and driving violates a rental agreement”, the subject of “violates” is a characteristic whose meaning is formulated as a projection over a conjunction of atomic formulations based on the characteristics ‘[person drinks](#)’ and ‘[person drives](#)’.

### 9.1.1.10 Nominalizations of Propositions and Questions

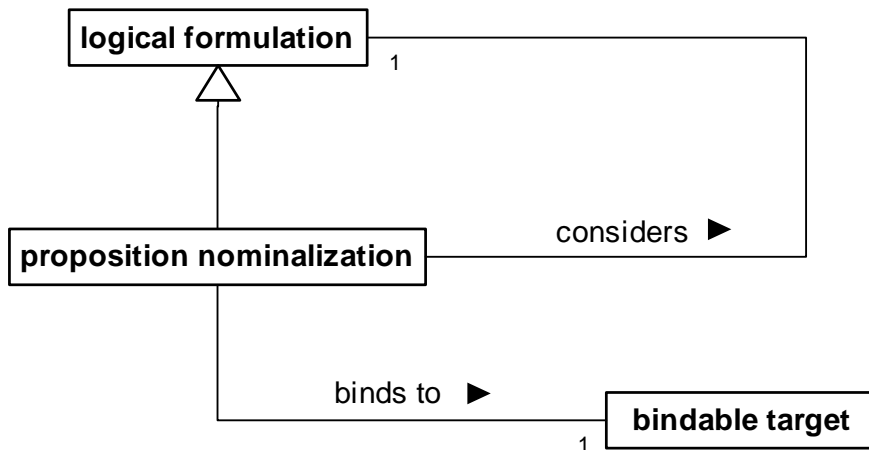


Figure 9.11

---

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

---

## proposition nominalization

FL

- Definition: [logical formulation](#) that a referent proposition is formulated by a particular [logical formulation](#)
- Concept Type: [logical formulation kind](#)
- Necessity: [Each proposition nominalization considers exactly one logical formulation.](#)
- Necessity: [Each proposition nominalization binds to exactly one bindable target.](#)
- Necessity: [Each variable that is bound to a proposition nominalization is free within the proposition nominalization.](#)
- Necessity: [Each variable that is free within the logical formulation of a proposition nominalization is free within the proposition nominalization.](#)

Note:	A <a href="#">proposition nominalization</a> is satisfied for each referent of its bindable target that is a proposition that is formulated by the considered logical formulation. For a <a href="#">closed logical formulation</a> , <a href="#">the closed logical formulation means the proposition</a> . Otherwise, a satisfying proposition has its meaning formulated by the logical formulation plus an understood reference to the referent of each variable that is free within the logical formulation.
Note:	The truth of a nominalized proposition is not relevant to the satisfaction of the <a href="#">proposition nominalization</a> .
Reference Scheme:	<a href="#">the bindable target that is bound to the proposition nominalization and the logical formulation that is considered by the proposition nominalization</a>
Example:	In the statement, “An agent must tell each new customer that EU-Rent accepts no checks”, an <a href="#">atomic formulation</a> based on a fact type ‘ <a href="#">person</a> tells <a href="#">person proposition</a> ’ binds to three variables, one for each role. The variable bound from the ‘ <a href="#">proposition</a> ’ role is also the <a href="#">bindable target</a> of a <a href="#">proposition nominalization</a> that considers a <a href="#">logical formulation</a> formulating “EU-Rent accepts no checks”.
Example:	In the statement, “An agent must tell each new customer that the customer cannot use a check”, an <a href="#">atomic formulation</a> based on a fact type ‘ <a href="#">person</a> tells <a href="#">person proposition</a> ’ binds to three variables, one for each role. The variable bound from the ‘ <a href="#">proposition</a> ’ role is also the <a href="#">bindable target</a> of a <a href="#">proposition nominalization</a> that considers a <a href="#">logical formulation</a> formulating “the customer cannot use a check”. Because the variable for ‘customer’ is free within the logical formulation, each satisfying answer includes a reference to a customer (the one being told).

### [proposition nominalization considers logical formulation](#)

FL

Definition:	<a href="#">the proposition nominalization</a> is based on <a href="#">the logical formulation</a>
Synonymous Form:	<a href="#">logical formulation is considered by proposition nominalization</a>

### [proposition nominalization binds to bindable target](#)

FL

Definition:	<a href="#">the bindable target</a> indicates the referent proposition identified by <a href="#">the proposition nominalization</a>
Synonymous Form:	<a href="#">bindable target is bound to proposition nominalization</a>

### [question nominalization](#)

Definition:	<a href="#">projecting formulation</a> of a referent question being formulated by a particular <a href="#">projection</a>
Concept Type:	<a href="#">logical formulation kind</a>
Necessity:	<a href="#">The projection of each question nominalization is a set projection.</a>
Note:	A <a href="#">question nominalization</a> is satisfied for each referent that is a question that is formulated by the projection. For a <a href="#">closed projection</a> , <a href="#">the projection formulates the question</a> . Otherwise, a satisfying question is the meaning formulated by the projection plus an understood reference to the referent of each variable that is free within the projection.
Reference Scheme:	<a href="#">the bindable target that is bound to the question nominalization and the projection of the question nominalization</a>
Example:	In the statement, “An agent must ask each new customer what kind of car the customer wants”, an <a href="#">atomic formulation</a> based on a fact type ‘ <a href="#">person</a> asks <a href="#">person question</a> ’ binds to three variables, one for each role. The variable bound from the ‘ <a href="#">question</a> ’ role is also the <a href="#">bindable target</a> of a <a href="#">question nominalization</a> that has a <a href="#">projection</a> formulating “what kind of car the customer wants”. Because the variable for ‘customer’ is free within the projection, each

satisfying question includes, in addition to what is formulated by the projection, a reference to a customer (the one being asked).

## answer nominalization

- Definition: [projecting formulation](#) of a referent fact correctly and completely holding the result of a particular [projection](#)
- Concept Type: [logical formulation kind](#)
- Necessity: [The projection of each answer nominalization is a set projection.](#)
- Note: An [answer nominalization](#) is satisfied for each referent that is a fact that correctly and completely holds the result of its projection. A satisfying fact incorporates the meaning formulated by the projection plus an understood reference to the referent of each variable that is free within the projection. It also includes an understood reference to each referent in the projection result. If the result has multiple elements, a satisfying fact holds them all, conjunctively.
- Reference Scheme: [the bindable target that is bound to the answer nominalization and the projection of the answer nominalization](#)
- Example: In the statement, “An agent must tell each new customer what special offer is available to the customer”, an [atomic formulation](#) based on a fact type ‘[person](#) tells [person](#) [proposition](#)’ binds to three variables, one for each role. The variable bound from the ‘[proposition](#)’ role is also the [bindable target](#) of an [answer nominalization](#) that has a [projection](#) formulating “what special offer is available to the customer”. Because the variable for ‘customer’ is free within the projection, each satisfying answer includes a reference to a customer (the one being told). Also, a satisfying answer must include a reference to each special offer in the projection result. Possible satisfying propositions are the meanings of the following statements (assuming the new customers Arthur, Bob and Charlie): “A free upgrade is the only special offer available to Arthur”, “A free upgrade and a 10% discount are the only special offers available to Bob” and “No special offer is available to Charlie”. In each answer, the entire result is incorporated, using conjunction as necessary (as for Bob). If the projection result is the empty set, then the answer must say so (as for Charlie).



## 9.1.2 Projections

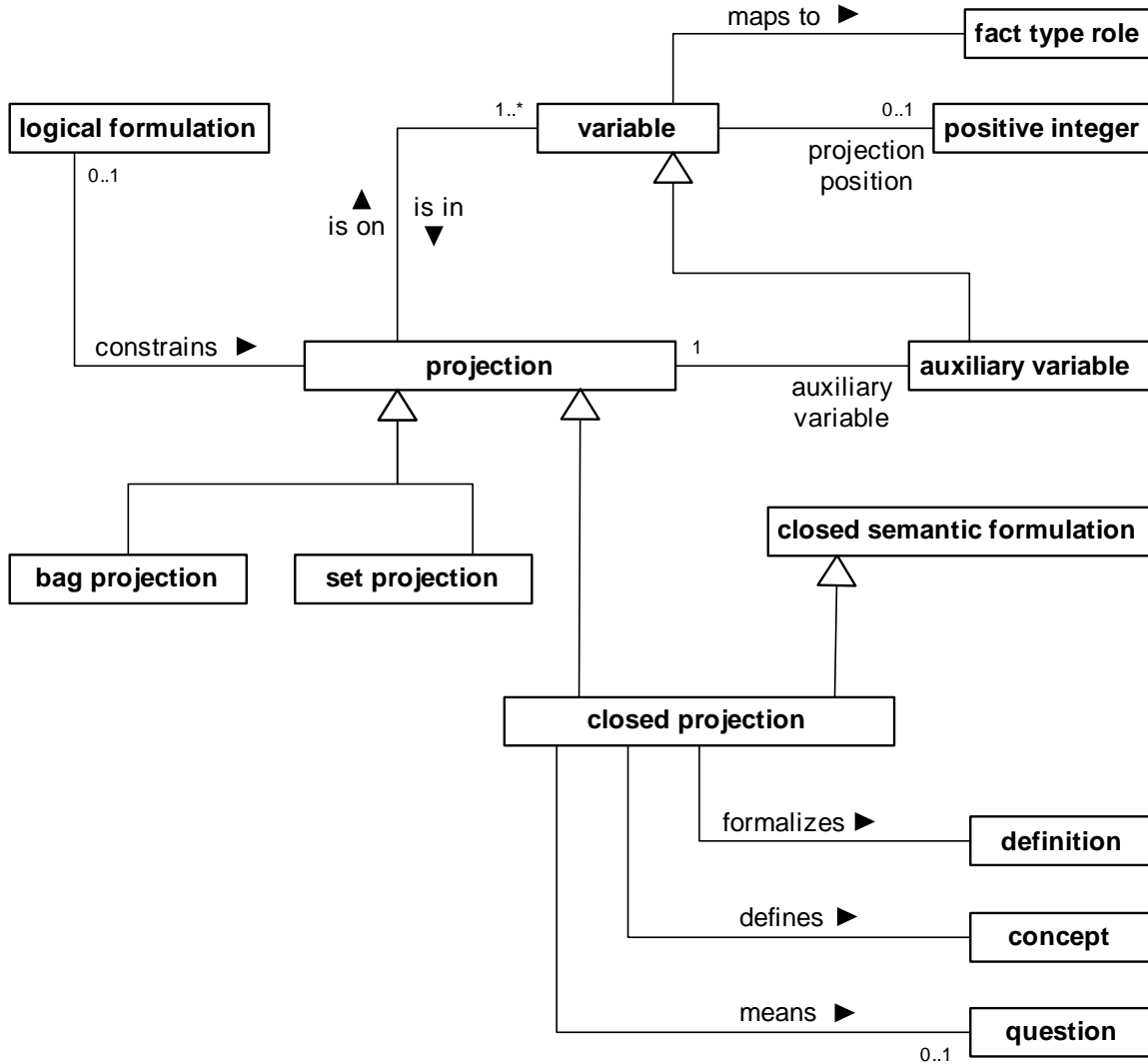


Figure 9.12

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

### projection

- Definition: [semantic formulation](#) that operates over one or more variables and results in a set or multiset
- Necessity: Each [projection](#) is on at least one [variable](#).
- Necessity: Each [projection](#) is constrained by at most one [logical formulation](#).
- Necessity: A [variable](#) that is free within a [logical formulation](#) that constrains a [projection](#) is free within the [projection](#) if and only if the [projection](#) is not on the [variable](#) and the [variable](#) is not an [auxiliary variable](#) of the [projection](#).

FL

Necessity: **No projection is a logical formulation.**

Reference Scheme: **the set of each variable that is in the projection and set of each auxiliary variable of the projection and the set of each logical formulation that *constrains* the projection**

Note: A projection's result can be taken in multiple ways. Which way depends on how the projection is used. When used in an aggregating formulation or as defining a concept other than a fact type, the result elements are simply the referents of the variable in the projection. When used to define a fact type, each result element is taken as an actuality that involves the referents of the variables in the projection.

### **projection is on variable**

FL

Definition: **the projection introduces the variable such that satisfying referents of the variable are in the result of the projection**

Synonymous Form: **variable is in projection**

### **projection has auxiliary variable**

FL

Definition: **the auxiliary variable is introduced by the projection, but is left out of the result of the projection thereby giving the possibility of duplicates in a result**

### **logical formulation constrains projection**

FL

Definition: **the logical formulation determines which referents of the variables introduced by the projection are in the result of the projection**

Synonymous Form: **projection is constrained by logical formulation**

Note: A logical formulation that constrains a projection restricts the results of the projection. If there is no constraining logical formulation, then there is no restriction other than what is on variables in the projection.

### **auxiliary variable**

FL

Definition: **variable that is introduced by a projection, but which is left out of the result of the projection thereby giving the possibility of duplicate results**

Necessity: **Each auxiliary variable is of exactly one projection.**

Reference Scheme: **the projection that has the auxiliary variable and the projection position of the variable**

### **projection position**

FL

Definition: **positive integer that distinguishes a variable introduced by a projection from others introduced by the same projection**

Concept Type: **role**

### **variable has projection position**

FL

Definition: **the variable is introduced by a projection and has the unique projection position among the set of variables introduced by that projection**

Necessity: **Each variable has at most one projection position.**

Necessity: **Each variable that is in a projection has exactly one projection position.**

Necessity: **Each auxiliary variable has exactly one projection position.**

### set projection

FL

- Definition: [projection that has no auxiliary variable](#)
- Example: A [projection](#) formalizing the expression, “customers that are preferred”, is on a single [variable](#) (customer). There is no [auxiliary variable](#), so the result is necessarily a set.

### bag projection

FL

- Definition: [projection that has an auxiliary variable](#)
- Example: A [projection](#) formalizing the expression, “account balances of customers that are preferred”, is on a [variable](#) (account balance) and has an [auxiliary variable](#) (customer). Only balances are in the result, but there can be duplicates where multiple customers have the same balance.

### closed projection

FL

- Definition: [projection that is a closed semantic formulation](#)
- Necessity: [Each closed projection that defines a fact type is a set projection.](#)
- Necessity: [Each variable that is in a closed projection maps to exactly one role of each fact type that is defined by the closed projection.](#)
- Necessity: [If a closed projection defines a fact type then each role of the fact type is mapped from exactly one variable that is in the closed projection.](#)
- Necessity: [A variable maps to a role of a fact type only if the variable is of a closed projection that defines the fact type.](#)
- Necessity: [A closed projection that defines a noun concept is on at most one variable.](#)
- Necessity: [If a closed projection that defines a noun concept is a bag projection then the noun concept is a role.](#)
- Necessity: [If a closed projection formalizes a definition of a concept then the closed projection defines the concept.](#)
- Necessity: [If a closed projection that defines a noun concept is a set projection that is on a variable that is unitary then the noun concept is an individual concept.](#)
- Necessity: [Each closed projection that means at most one question.](#)
- Example: A [projection](#) formalizing the expression, “customers that are preferred”, is closed – there is no variable that is not introduced. But within a formulation of the expression, “Each branch must report the number of car models offered by the branch”, the [projection](#) of “car models offered by the branch” is open because it binds to a [variable](#) (branch) that is introduced outside of the [projection](#).

### closed projection formalizes definition

- Definition: [the definition](#) conveys the meaning formulated by [the closed projection](#)
- Synonymous Form: [definition is formalized by closed projection](#)

### closed projection defines concept

FL

- Definition: [the closed projection](#) formulates a definition of [the concept](#) such that the result of the [projection](#) is the [extension](#) of [the concept](#)
- Synonymous Form: [concept is defined by closed projection](#)
- Example: For a noun concept ‘barred driver’ having the definition, “person that must not rent or drive a car from EU-Rent”, a closed projection structures the meaning of the definition. The projection is on one variable, which ranges over the concept ‘person’. The satisfying referents of that variable make up the extension of the concept ‘barred driver’.

Example: For a fact type 'branch is sold out' having the definition, "the branch has no rental car available", a closed projection structures the meaning of the definition. The projection is on one variable which maps to the role 'branch'.

### closed projection means question

Definition: the closed projection formulates the question such that the result of the projection answers the question

Synonymous Form: question is meant by closed projection

### variable maps to role

FL

Definition: the variable is in a closed projection that defines the fact type that has the role such that for each element in the projection result the referent of the variable is involved in the role in a corresponding actuality in the extension of the fact type

Synonymous Form: role is mapped from variable

# 10 Providing Semantic and Logical Foundations for Business Vocabulary and Rules

This section lists and explains foundational concepts taken from respected works on formal logics and mathematics. A mapping is then shown from the concepts of the SBVR [Logical Formulation of Semantics Vocabulary](#) to these foundational concepts.

A conceptual model includes both a conceptual schema and a population of facts that conform to the schema. A conceptual model may cover any desired time span, and contain facts concerning the past, present or future. This notion is distinct from changes made to a conceptual model. Any change to a conceptual model, including any change to any fact in the fact population, creates a different conceptual model. Each conceptual model is distinct and independent, although there may be relationships between conceptual models that share the same conceptual schema.

'Facts' are one of the primary building blocks of SBVR. A 'Fact' is of a particular 'Fact Type'. The lowest level logical unit in SBVR – an 'Atomic Formulation' – is a logical formulation based directly upon a fact type, involving no logical operation. An atomic formulation may be considered as an invocation of a predicate.

SBVR makes no distinction about how facts are known: for example, whether they are asserted as 'ground facts' or obtained by inference. Inferences can only be performed at one time within a particular conceptual model. SBVR does not define any kind of inference that can be made between conceptual models.

Control over the order in which inferences can be made is a common feature in the automation of inference, as found, for example, in rules engines. SBVR deals with declarative rules expressed from a business perspective. Transitions between conceptual models and the mechanization of those rules in an automated system are outside the scope of SBVR.

Closed-world assumptions are often used in automated systems, such as the well-known 'negation by failure' in the Prolog language. The business orientation of SBVR makes it natural to assume open-world semantics by default. For example, if we assume that 'Customers' have some unary fact such as 'Credit OK' then we can not assume anything like 'Credit not OK' in the absence of this fact. SBVR permits fact types to be explicitly identified as closed where this makes business sense. For example, it may be appropriate to infer 'Credit not OK' for a subset of customers identified as 'Credit-Checked Customers' in the absence of a 'Credit OK' fact.

The detailed definition of SBVR uses the vocabulary defined in SBVR – in other words, SBVR is defined in terms of itself. This inevitably makes the SBVR definition higher order, but this does not force any modeler to produce exclusively higher-order models. Models based on SBVR can be first order if that is what is desired by the modeler.

## 10.1 Logical Foundations for SBVR

### 10.1.1 SBVR Formal Grounding Model Interpretation

#### 10.1.1.1 Introduction

The SBVR (Semantics of Business Vocabulary and Business Rules) initiative is intended to capture business facts and business rules that may be expressed either informally or formally. Business rule expressions are classified as formal only if they are expressed purely in terms of fact types in the pre-declared schema for the business domain, as well as certain logical/ mathematical operators, quantifiers, etc. Formal rules may be transformed into a logical formulation that is used for exchange with other rules-based software tools. Informal rules may be exchanged as un-interpreted comments. The following discussion of business rule semantics is confined to formal business rules. (A closer definition of terms is given as needed later throughout this section.)

The rest of this section is structured as follows. 10.1.1.2 provides some basic background and terminology, explaining our usage of terms such as “schema”, “model” and “fact”. 10.1.1.3 reviews the approach to choosing open or closed world semantics. 10.1.1.4 provides an overview of the use of quantifiers as well as alethic or deontic modal operators in specifying business rules. 10.1.1.5 and 10.1.1.6 respectively discuss the formal semantics for static, alethic constraints and static, deontic constraints. 10.1.1.7 considers derivation rules. 10.1.1.8 examines dynamic constraints. 10.1.1.9 reviews the option for using higher-order logic.

### 10.1.1.2 Facts, Schemas and Models

For any given business, the “universe of discourse” indicates those aspects of the business that are of interest. The term “business domain” is commonly used in the modeling community, with equivalent meaning. A “model”, in the sense used here, is a structure intended to describe a business domain, and is composed of a conceptual *schema* (fact structure) and a *population* of ground facts (see later). A *fact* is a proposition taken to be true by the business. Population facts are restricted to elementary and existential facts (see later).

Facts refer to individuals (such as “Employee 123” or “the sales department”). These individuals are considered as being of a particular type (such as “Employee” or “Department”) where *type* denotes “set of possible individuals”.

The schema declares the *fact types* (kinds of facts, such as “Employee works for Department”) and *logical rules* (typically constraints or derivation rules) relevant to the business domain.

Logical rules, in the sense used here, are regulations or principles governing conduct, procedure, etc. Logical rules are effectively higher-level facts (i.e., facts about facts), and in a loose sense are also sometimes considered under the generic term ‘fact’. For clarity, the term “ground fact” is used here to explicitly exclude such (meta) facts.

*Constraints* are used to define bounds, borders or limits on fact populations, and may be static or dynamic. A *static constraint* imposes a restriction on what fact populations are possible or permitted, for each fact population taken individually.

Static constraint

**Each Employee was born on at most one Date**

A *dynamic constraint* imposes a restriction on transitions between fact populations.

Dynamic constraint

A person’s marital status may change from single to married, but not from divorced to single

*Derivation rules* indicate how a fact type may be derived from one or more other fact types or how a type of individual may be defined in terms of other types of individuals and fact types.

Derivation rules

Person<sub>1</sub> is an uncle of Person<sub>2</sub> **if** Person<sub>1</sub> is a brother of **some** Person<sub>3</sub> **who** is a parent of Person<sub>2</sub>,

**Each FemaleAustralian is a Person who** was born in Country ‘Australia’ **and** has Gender ‘Female’

A model of the kind considered here is a *fact model*, not a process model. The term *knowledge base* is sometimes used to reflect this focus (on what is known, as opposed to what must be done). At least two kinds of fact model may be specified: reality models; and in-practice models. Although both these models use the same set of fact types, they may differ in the constraints imposed on those fact types. A *reality model* of a business domain is intended to reflect the constraints that actually apply to the business

domain in the real world. An *in-practice model* of a business domain reflects the constraints that the business chooses in practice to impose on its knowledge of the business domain.

Suppose the following two fact types are of interest: Employee was born on Date; Employee has PhoneNumber. In the real world, each employee is born, and may have more than one phone number. Hence the reality model includes the constraint “**Each** Employee was born on **at least one** Date” and allows that “**It is possible that the same** Employee has **more than one** PhoneNumber”. Now suppose that the business decides to make it optional whether it knows an employee’s birth date. Suppose also that the business is interested in knowing at most one phone number for any given employee. In this case, the in-practice model excludes the reality constraint “**Each** Employee was born on **at least one** Date”, but it includes the following constraint that doesn’t apply in the reality model: **Each** Employee has **at most one** PhoneNumber.

Constraint differences between reality and in-practice models have some restrictions (for instance, in-practice uniqueness constraints need to be at least as strong as the corresponding real world uniqueness constraints, and if a fact type role is optional in the real world it is optional in the in-practice world, but the converse need not apply).

Reality schemas are sometimes constructed first to help determine in-practice schemas. Although a population may be added to any schema to form a model, it is common to add populations only to in-practice schemas. So in-practice models are more common than reality models. The possibility of incomplete knowledge arises for both reality and in-practice models but is more prevalent with in-practice models since these tend to include more optional aspects. Adoption of open or closed world assumptions is discussed in 10.1.1.3.

#### Example of incomplete knowledge

The business might know just some of a given employee’s phone numbers

We use the term “fact model” or “knowledge base” in a broad sense. Conceptually, the fact model is represented by a set of sentences, each of which connotes either a logical rule or a ground fact. The fact model may be fully automated (as in, say, a database system), manual (as in, say, a paper record system), or semi-automated. The knowledge may even be stored in human memory (belonging to the business domain experts who may be collectively regarded as the authoritative source of those business facts that are of interest). However, the knowledge must ultimately be expressible by sentences communicated between humans.

A fact model is specified as a conceptual model of the business domain, using a suitable high level vocabulary and language that is readily understood by the business domain experts. Typically this language will be a formal subset of a natural language. In particular, the language is not a machine-oriented technical language (such as C#, Java, or SQL) that might be used to implement a system to enforce at least some of the business rules included in the model. Business domain models are meant to capture the relevant business rules, not to implement them. Whether a given business rule is implemented at all, or how it might be implemented (automated, semi-automated or manual) are not issues here. Typically however, it is expected that many business rules specified in a business domain model will likely be enforced in an automated way; and in such cases, the logical rules need to be formally expressed.

Any fact model passes through a sequence of *states*, each of which includes a set of *ground facts*, which are either elementary or existential. Roughly speaking, an *elementary fact* is a declaration that an individual has a property, or that one or more individuals participate in a relationship, where the fact cannot be split into simpler facts with the same individuals (without information loss).

Examples of elementary facts

The Country named 'Australia' is large

The Prime Minister named 'John Howard' was born in the Country named 'Australia'

An elementary fact may be treated as an instantiation of a typed, irreducible predicate of interest to the business, except that multiple fact type readings using different predicates, possibly based on different orderings of the individuals, are considered to express the same fact if they mean the same. Individuals are typically denoted by definite descriptions.

The sentences (1) and (2) below express the same fact:

(1) The President named 'Mary McAleese' governs the Country that has the Country Name 'Ireland'.

(2) The Country that has the Country Name 'Ireland' is governed by the President named 'Mary McAleese'.

"The President named 'Mary McAleese'" is treated here as shorthand for "The President who has the President Name 'Mary McAleese'".

Instead of definite descriptions, proper names may be used if they function as individual constants in the business domain. Lexical individuals denote themselves. Individual constants may also be introduced as abbreviations of definite descriptions.

Example of a self-denoting lexical individual

The country code 'US'

We use the term "fact" in the sense of "proposition taken to be true by the business" (i.e., the business members are prepared to act as if they believed the proposition is true; their attitude toward the proposition is one of epistemic commitment). This sense of epistemic commitment does not require any special interpretation of logical operators, or use of epistemic or doxastic logic. The logical connectives (and, or, not, if-then, etc.) may be interpreted just like truth functional operators (conjunction, disjunction, negation, material implication, etc.) in 2-valued classical logic. An *existential fact* is used to simply assert the existence of an individual,

Example of an existential fact

There is a Country that has the Country Code 'US'



A *fact type* may be identified by one or more fact type readings that declare typed predicates.

Examples of fact type readings

The President named 'Mary McAleese' governs the Country that has the Country Name 'Ireland'  
is an instance of the fact type  
President governs Country

The Country that has the Country Name 'Ireland' is governed by the President named 'Mary McAleese'  
is an instance of the fact type  
Country is governed by President

Section 10.1.1 uses initial capitals to denote types of individuals (other styles may be used for this purpose), and in general allows predicates in mixfix notation.

Example of mixfix notation

President visited Country on Date

More conventional but less readable syntaxes may also be used.

Example of more conventional notation

President governs Country  
may be expressed as  
governs(x:President; y:Country)

Each predicate has a fixed arity, so variadic predicates are not supported.

For example, the unary "smokes" predicate in 'Person smokes' is considered to be different from the binary "smokes" predicate in 'Person smokes Cigar Brand.'

Note that we do not identify untyped predicates simply by their name and arity.

For example, the "has" in 'Person has Disease' is considered to be a different predicate from the "has" in 'Disease has Cure.'

The fact model includes both the conceptual schema and the ground fact population (set of fact instances that instantiate the fact types in the schema). The conceptual schema includes a generic component and a domain-specific component. The generic component is common to all conceptual schemas: this includes relevant axioms from logic and mathematics<sup>1</sup>. The domain-specific component includes declarations of the ground fact types and logical rules relevant to the specific business domain.

1. For a detailed discussion of one way to formalize this, see [Halp1989]. A fact model is specified as a set of sentences in a language based on predicate logic with identity. An interpretation is defined in the usual way (e.g., each predicate symbol maps onto a relation over the domain of individuals) and a model (not the same as fact model) is an interpretation where all the sentences are true.

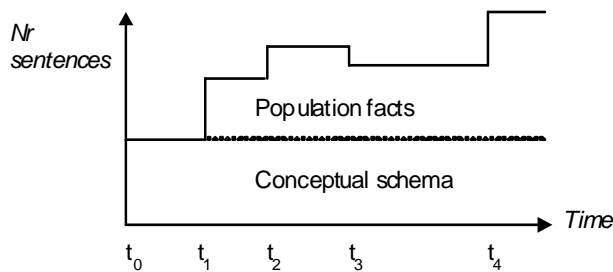
Trivially, each fact model includes existential facts to declare the existence of generic constants such as numbers, but we ignore these in our discussion, confining the use of “population” to the domain-specific population of interest. With that understanding, the fact model at any point in time may be declared as a set of sentences that collectively express the conceptual schema and the fact population of the domain-specific fact types in the conceptual schema.

Although in practice the conceptual schema may evolve over time (if the business domain changes its structure or scope of interest) we ignore schema evolution here, treating the conceptual schema as fixed. Schema evolution may be handled as a metametalevel concern. Model exchange must be enabled between a system supporting SBVR and other systems identified as desirable targets for interoperability. Any exchange of a fact model takes place at a given point in time, and at that time the conceptual schema is fixed (later exchanges may be used to update the fact model as required). Also, when a necessity is originally stated, the intent is that by default the logical rule should stay in force.

In contrast to the conceptual schema, the (domain-specific) fact population is typically highly variable.

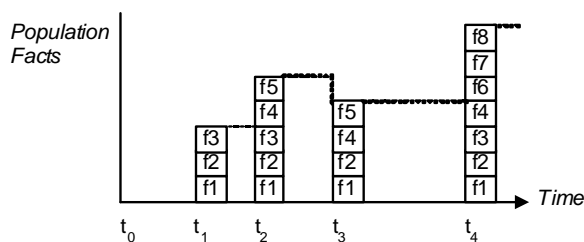
For example, the fact type "Employee works on Project" may initially have no instances, but over time thousands of employees may be added or removed from various project teams.

Figure 10-1 provides a simplified picture of this situation, indicating that the fact model of sentences expressing population facts (instances of domain-specific fact types) is a varset (variable-set) whose population at any given time is a set of facts.



**Figure 10.1 - Evolution of the fact model (schema plus ground fact instances)**

The fact model may be initially empty or pre-populated with some facts. The fact model may expand or shrink over time as facts are added or removed from it. At any point in time, the fact model includes a set of facts. Figure 10-2 depicts this situation in more detail, using a labeled box to denote a fact instance (f1 = fact 1, etc.).



**Figure 10.2- Evolution of the ground fact population**

In treating a fact model as a varset of facts that typically changes over time, we allow facts to be added or deleted (see Figure 10-22). We might delete a fact because we revise our decision on whether it is (taken to be) true (for instance, we might discover a

mistake), or because we decide that fact is no longer of interest. Now consider the following description by [Anto2001] of non-monotonic logic.

The term "non-monotonic logic" covers a family of formal frameworks devised to capture and represent *defeasible inference*, i.e., that kind of inference of everyday life in which reasoners draw conclusions tentatively, reserving the right to retract them in the light of further information. Such inferences are called "non-monotonic" because the set of conclusions warranted on the basis of a given knowledge base does not increase (in fact, it can shrink) with the size of the knowledge base itself. This is in contrast to classical (first-order) logic, whose inferences, being deductively valid, can never be "undone" by new information.

On the surface, it would appear that we are committing to a non-monotonic logic, given that we allow facts to be deleted in going from one state to another. However it seems reasonable to formalize those logical rules that are static constraints in terms of classical, non-monotonic logic.

For example, we might formalize the static constraint that each person was born on some date as an SBVR logical formulation of the formula "x:Person \$y:Date x was born on y."

In classifying the logical rule as a static constraint, we assert that it is true for each state of the fact model, taken individually. This seems to be enough, from the point of view of exchanging fact models, which always involves just one state at that time. Note also that the characterization of fact models as variable sets of sentences does not claim that propositions change their truth value over time. We regard propositions to be atemporal: they are timelessly true or false, so never change their truth value.

At least superficially, it is possible that a sentence in one fact model state expresses a different proposition from that expressed by the same sentence in another fact model state. For example, the meaning of time-deictic sentence occurrences depends on the time they were uttered or inscribed.

For instance, given the static constraint that each person lives in at most one country, we might assert for the fact model state 1 that Terry lives in Australia, for fact model state 2 we delete "Terry lives in Australia" and add that Terry lives in Utah, and for fact model state 3 we delete "Terry lives in Utah" and add that Terry lives in Australia. This does not involve any change in proposition truth values, because different propositions were being asserted in the different states. Here the verb phrase "lives in" means "currently lives in", where 'currently' may be unpacked into a time-indexed expression that includes the time of that fact model state.

### 10.1.1.3 Open/Closed World Semantics

Adopting *closed world* semantics basically means that all relevant facts are known (either as primitives – not defined in terms of other things – or derivable). So if a proposition cannot be proved true, it is assumed to be false. This *closed world assumption* entails *negation by failure*, since failure to find a fact implies its negation. *Open world semantics* allows that some knowledge may be incomplete; so if a proposition and its negation are both absent, it is unknown whether the proposition is true.

In modeling any given business domain, attention can be restricted to propositions *of interest* to that domain. If a proposition is not relevant to that domain, it is not included as a fact there, but we do not assume it is false; rather we simply dismiss it from consideration. For any business domain, we have a *finite set of types of individuals and fact types* (typed predicates), and any type of individual or fact type outside this set is simply disregarded.

It is a practical issue whether one's knowledge pertaining to the population of a given fact type is complete or not, since this may impact how the business derives other facts (e.g., negations) or how it reacts to query results (e.g., whether to treat "not" as "not the case" or merely "not known to be the case"). So we regard the issue of open/closed world semantics to be relevant to the fact model itself, not just automated implementations of the fact model.

Many implementations treat “not” in the closed-world sense of either “not known” (as a primitive or derivable fact), i.e., negation as failure, or “not known as a primitive fact”, i.e., semi-positive negation. For instance, Prolog-based rule engines rely on negation by failure, and the “not” in SQL means “not recorded in a base table or derivable in a view”.

SQL example,

Figure 10-3 depicts the relational schema and a sample population for a database fragment used to store the employee number and name of each employee, as well as the cars they drive (if any).

Employee		Drives	
<i>empNr</i>	<i>empName</i>	<i>empNr</i>	<i>carRegNr</i>
1	John Smith	1	ABC123
2	Ann Jones	2	AAA246
3	John Smith	2	DEF001


*E m p l o y e e* ( *e m p N r*, *e m p N a m e* )  
  
*D r i v e s* ( *e m p N r*, *c a r R e g N r* )

Figure 10.3- A sample database storing some facts about employees

Suppose we want to know the employee number and name of each employee. In SQL we might formulate this query as **select \* from Employee**, which returns the three rows of data shown in the Employee table. This result returns the employee number and name of those employees referenced in the database. Whether this includes all the employees in the business domain depends on whether the database is complete with respect to the population of the elementary fact type Employee has EmployeeName. If it is complete, the fact type is closed, and we may treat the SQL query as equivalent to our intended query about the business domain. If it is not complete, then the fact type is open, and we may need to take into account that there may be more employees than listed in the result.

Knowledge about completeness is typically not stored in databases, although in principle it could be. Users typically adopt the closed world assumption when interpreting data in relational databases. If independently of the database system they know how complete the data is, they may take that into account in deciding how completely the query results from the database system relate to the real world of their business domain.

Suppose we want to know the employee number of each employee who does not drive a car for the database shown in Figure 10-3. In SQL we might formulate this query as **select empNr from Employee where empNr not in (select empNr from Drives)**. This returns just one employee number (viz. 3). Whether this covers all the non-driver employees in the business domain depends on whether the population of the two fact types (Employee has EmployeeName and Employee drives Car) is complete or not. Again, this knowledge about completeness could be stored in the database, but typically isn't, in which case users need to rely on their own knowledge about completeness to decide whether the data returned is complete or not.

The approach adopted here is fact-based (as opposed to attribute-based), where each fact type is modeled as a type of relationship, never as an attribute. Annex J provides extended examples of fact types expressed in this way using a popular fact-based modeling approach.

#### Example fact-based representation of a database schema

The information structure implied by the database schema shown in Figure 10-3 can be expressed as a set of fact types and constraints as follows, using the capitalized mixfix notational style described earlier:

##### Types of individuals

Employee

Car

Employee Number

Employee Name

Car Registration Number

(Note that here Employee and Car represent the kind of real world individuals that typically change state. Employee Number, Employee Name and Car Registration Number represent simple self-identifying lexical constants.)

##### Fact types

Employee has Employee Number

Employee has Employee Name

Car has Car Registration Number

Employee drives Car

##### Constraints

Each Employee has exactly one Employee Number.

For each Employee Number, at most one Employee has that Employee Number.

Each Employee has exactly one Employee Name.

Each Car has exactly one Car Registration Number.

For each Car Registration Number, at most one Car has that Car Registration Number.

It is possible that the same Employee drives more than one Car and that more than one Employee drives the same Car.

Completeness claims about a schema can be clarified by referring to whether fact type roles are mandatory and whether instances of fact type roles are unique. A fact type role is mandatory if, for each state of the fact model, each instance in the population of the associated type of individual must play that fact type role. A fact type role (or combination of fact type roles) is unique if, for each state of the fact model, each individual that instantiates the fact type role (or each sequence of individuals that instantiates the fact type role sequence) does so once only.

In the schema given above:

each Employee has exactly one Employee Name (mandatory fact type role) but it is optional whether an Employee drives a car.

each Employee has exactly one Employee Name: the Employee fact type role is unique in this fact type but the Employee Name fact type role is not (an Employee has only one Employee Name, but the same Employee Name could refer to more than one Employee).

To consider completeness claims, we can express additional requirements in terms of the fact model populations of types of individuals and the sequences of fact type roles they play in the population of fact types. A schema, as described earlier, is useful for clarifying the conditions under which completeness claims may be made.

Referring again to the Employee-Car schema, for any state of the fact model, let  $pop(I)$  denote the fact model population of the type of individual  $I$  in that state, and let  $pop(F)$  denote the fact model population of the fact type role sequence for the fact type  $F$  in that state. If the fact model is complete with regard to capturing the real world business domain, then for each state of the fact model the following three additional conditions are satisfied:

(1)  $pop(\text{Employee})$  = set of employees in the (real world) business domain (at that time)

(2)  $pop(\text{Car})$  = set of cars in the business domain

(3)  $pop(\text{Employee drives Car})$  = set of (employee, car) pairs from  $pop(\text{Employee}) \times pop(\text{Car})$  where that employee drives that car in the business domain.

Requirements (1) and (2) declare that the fact model population of the Employee and Car types of individuals always matches that of the business domain being modeled. We may regard this as asserting the closed world assumption for those types of individuals. Requirement (3) asserts that for those employees and cars that are included in the fact model, if they drive a car then this fact is known. In combination, requirements (1) – (3) entail the closed world assumption for the drives fact type (if an employee drives a car in the business domain, this is known in the fact model).

Given the schema, and requirement (1), the closed world assumption is implied for the employee name fact type. This follows because of the mandatory and uniqueness constraints on the first fact type role (employee is closed, so we have all the employees; having a name is mandatory, so we have at least one name for each employee; the uniqueness constraint means that each employee has at most one name; so for all employees we now have all their names). Note that open world semantics still applies to the employee name fact type; in the presence of (1) and the constraints, this is equivalent to closed world semantics for that fact type.

For any given schema, the business might have complete knowledge about some parts and incomplete knowledge about other parts. So in practice, a mixture of open and closed world assumptions may apply. We use the term “*local closure*” (or “*relative closure*”) for the application of the closed world assumption to just some parts of the overall schema. One might assume open world semantics by default, and then apply local closure to specific parts as desired; or alternatively, assume closed world semantics by default and then apply “*local openness*”. We adopt the former approach as it seems more realistic when modeling real business domains.

Closure (i.e., local closure) may be explicitly asserted for any type of individual, on a one-by-one basis, to declare that for each state the fact model population agrees with that of the population of that type of individual in the actual business domain. The relevant meta-fact type is: “type of individual is closed”. It may be reasonable to assume closure for types of individual by default, but it seems unrealistic to assume closure for predicates.

Closure may also be asserted for fact types. *Semi-closure* is with respect to the fact model population of the types of individual playing a fact type role in the predicate. If closure has also been declared for these types, then (full) closure also holds for the fact type (i.e., closure with respect to the domain population of the types of individuals). The relevant meta-fact types are: “fact type is semi-closed” and “fact type is closed”. The meta-fact type “concept is closed” applies to both types of individuals and fact types, since both are concepts.

As seen earlier, closure for a fact type is sometimes implied. A *functional fact type role* is the complete argument of a uniqueness constraint. For schemas whose functional fact type roles are also functional in the business domain, the following implications hold. If a predicate includes a mandatory, functional fact type role, then that predicate is semi-closed by implication (as in the employee name example earlier). This result may be generalized to the case of a mandatory fact type role that has a frequency constraint of exactly  $n$  (although some attribute-based approaches do not deal reliably with various  $n$ -ary cases). If a type of individual has a set of functional fact type roles that are disjunctively mandatory and mutually exclusive (in other words, they are spanned by an exclusive-or constraint), then the predicates that include those fact type roles are semi-closed by implication. If the type of individual has also been declared complete in such cases, then (full) closure applies.

For many fact types in a business domain, especially those without functional fact type roles, it is impractical to include all the negative instances as primitive facts.

For example, for the fact type “Employee drives Car,” there might be many thousands of cars, so one would normally not explicitly include negated facts such as “Employee 1 does <b>not</b> drive Car ‘AAA246’.”
--

In some cases however, especially with functional fact type roles or when the population is small, it is practical to include negated facts as base facts.

#### Example

To provide a concrete example of the alternative, we can consider the unary fact type 'Person smokes', and three instances of Person: Fred, Sue and Tom (for simplicity we will ignore reference schemes and assume that a person may be identified by their first name).

Assume that we know that Fred smokes. If we use open-world semantics, then it is unknown whether Sue or Tom smoke. If we apply closed world semantics, then the absence of facts that Sue or Tom smoke entails that they don't smoke.

If, for each Person, it is known whether that person smokes or not, then we could adopt one of two approaches to model our business domain.

(a) Use two unary fact types, such as 'Person smokes' and 'Person is a nonsmoker', with an exclusive-or constraint between the fact types. In other words, a Person must play one fact type role or the other, but cannot play both.

(b) Use a binary fact type such as 'Person has Smoker Status' where Smoker Status is indicated by some suitable code such as 'S' or 'NS' (for smoker or nonsmoker respectively), together with the constraint that a Person has exactly one Smoker Status.

In each of these cases, negated facts are explicitly treated as primitive facts and the predicates are given open world semantics. Semi-closure is implied because of the constraints.

Now consider a business domain where we know that Fred smokes, and that Sue doesn't smoke, but are unsure whether Tom smokes. In this case we have three alternative approaches that we could consider.

(a) Use two unary fact types, such as 'Person smokes' and 'Person is a nonsmoker', with an exclusion constraint between the fact types. In other words, a Person may play one fact type role or the other (but not both) or may play neither fact type role. For the given scenario, we would have the facts 'Fred smokes', 'Sue is a nonsmoker' and no information for Tom.

(b) Use a binary fact type such as 'Person has Smoker Status' where Smoker Status is indicated by some suitable code such as 'S' or 'NS' (for smoker or nonsmoker respectively), together with the constraint that a Person has zero or one Smoker Status value. For the given scenario we would have the facts 'Fred has Smoker Status 'S'', 'Sue has Smoker Status 'NS'' and no information for Tom.

(c) Use a binary fact type such as 'Person has Smoker Status' where Smoker Status is indicated by some suitable code such as 'S', 'NS' or '?' (for smoker, nonsmoker, or unknown, respectively), together with the constraint that a Person has exactly one Smoker Status. In this case we treat the 'unknown' value ('?') like any other value using 2-valued logic, rather than adopt a generic null based on 3-valued logic, as in SQL. For the given scenario we would have the facts "Fred has Smoker Status 'S'," "Sue has Smoker Status 'NS'," and "Tom has Smoker Status '?'."

The above discussion indicates some ways of declaring and inferring various kinds of closure in the underlying fact model, based on a default, open world semantics. Here, all business rules that are parsed as formal are given a logical formulation based on the fact types in the underlying model. When people formulate queries on the model population, they may either adopt whatever closure guarantees are formally captured in the model, or instead informally rely on their own knowledge about closure to decide whether the data returned is complete or not. Such informal knowledge is outside the fact model, and does not impact the formal semantics of the logical formulation used in exchanging fact models.

In addition to specifying fact models at a conceptual level, languages may be defined for querying these models directly at a conceptual level. These may include features such as the ability to specify projections in the scope of negation, as well as projections in the scope of the "whether-or-not" operator which is used to perform conceptual left outer joins [Bloe1996, Bloe1997]. Further details are outside the scope of this section.



### 10.1.1.4 Quantifiers and Modalities

Static constraints apply to each state of the fact model, taken individually. These may typically be expressed as logical formulations that are equivalent to formulae in 2-valued, first-order predicate calculus with identity. The 2-valued restriction applies because the fact types on which the logical rules are based are elementary (irreducible), so their instances never involve nulls. For convenience, we can use mixfix notation for predicates, and predefine some numeric quantifiers in addition to  $\forall$  and  $\exists$ . Table 10-1 summarizes the pre-defined quantifiers.

**Table 10.1- Quantifiers**

<i>Symbol</i>	<i>Example</i>	<i>Name</i>	<i>Meaning</i>
$\forall$	$\forall x$	Universal Quantifier	For each and every $x$ , taken one at a time
$\exists$	$\exists x$	Existential Quantifier	At least one $x$
$\exists^1$	$\exists^1 x$	Exactly-one quantifier	There is exactly one (at least one and at most one) $x$
$\exists^{0..1}$	$\exists^{0..1} x$	At-most-one quantifier	There is at most one $x$
$\exists^{0..n}$ ( $n \geq 1$ )	$\exists^{0..2} x$	At-most- $n$ quantifier	There is at most $n$ $x$ <i>Note: <math>n</math> is always instantiated by a number <math>\geq 1</math>. So this is really a set of quantifiers (<math>n = 1</math>, etc.)</i>
$\exists^{n..}$ ( $n \geq 1$ )	$\exists^{2..} x$	At-least- $n$ quantifier	There is at least $n$ $x$ <i>Note: <math>n</math> is always instantiated by a number <math>\geq 1</math>. So this is really a set of quantifiers (<math>n = 1</math>, etc.)</i>
$\exists^n$ ( $n \geq 1$ )	$\exists^2 x$	Exactly- $n$ quantifier	There is at exactly (at least and at most) $n$ $x$ <i>Note: <math>n</math> is always instantiated by a number <math>\geq 1</math>. So this is really a set of quantifiers (<math>n = 1</math>, etc.)</i>
$\exists^{n..m}$ ( $n \geq 1, m \geq 2$ )	$\exists^{2..5} x$	Numeric range quantifier	There is at least $n$ and at most $m$ $x$

The additional existential quantifiers are easily defined in terms of the standard quantifiers.

For example, the exactly-two quantifier  $\exists^2$  may be defined as follows. Let  $x, x_1, x_2$  be individual variables and  $\Phi x$  be a well formed formula with no free occurrences of  $x_1, x_2$ . Then:

$$\exists^2 x \Phi x =_{df} \exists x_1 \exists x_2 [\Phi x_1 \ \& \ \Phi x_2 \ \& \ x_1 \neq x_2 \ \& \ \forall y (\Phi y \supset (y = x_1 \vee y = x_2))]$$

Definition schemas for the other quantifiers may be found on page 4-11 of [Halp1989].

The logical rule formulations covered here may use any of the basic alethic or deontic modal operators shown in Table 2. These modal operators are treated as proposition-forming operators on propositions (rather than actions). Other equivalent readings may be used in whatever concrete syntax is used to originally declare the logical rule (e.g., “necessary” might be replaced by “required”, and “obligatory” might be replaced by “ought to be the case”). Derived modal operators may also be used in the surface syntax, but are translated into the basic modal operators plus negation ( $\sim$ ).

For example, “It is impossible that  $p$ ” is defined as “It is not possible that  $p$ ” ( $\sim\ddagger p$ ), and “It is forbidden that  $p$ ” is defined as “It is not permitted that  $p$ ” ( $Fp =_{df} \sim Pp$ ).

**Table 10.2 - Alethic and deontic modal operators**

Alethic		Deontic	
Reading	Symbol	Reading	Symbol
It is necessary that	$\leq$	It is obligatory that	<b>O</b>
It is possible that	$\diamond$	It is permitted that	<b>P</b>

The following *modal negation rules* apply: it is not necessary that  $\equiv$  it is possible that not ( $\sim\leq p \equiv \diamond\sim p$ ); it is not possible that  $\equiv$  it is necessary that not ( $\sim\diamond p \equiv \leq\sim p$ ); it is not obligatory that  $\equiv$  it is permitted that it is not the case that ( $\sim Op \equiv P\sim p$ ); it is not permitted that  $\equiv$  it is obligatory that it is not the case that ( $\sim Pp \equiv O\sim p$ ). In principle, these logical rules could be used with double negation to get by with just one alethic modal operator (e.g.,  $\diamond p$  could be defined as  $\sim\leq\sim p$ , and  $Pp$  could be defined as  $\sim O\sim p$ ).

Every constraint has an associated *modality*, determined by the logical modal operator that functions explicitly or implicitly as its main operator. We can distinguish between positive, negative, and default verbalizations of constraints. In positive verbalizations, an alethic modality of necessity is often assumed (if no modality is explicitly specified), but may be explicitly prepended.

For example, the following static constraint

C1     **Each** Person was born in **at most one** Country.

may be explicitly verbalized with an alethic modality thus:

C1'    **It is necessary that each** Person was born in **at most one** Country.

We interpret this in terms of *possible world semantics*, as introduced by Saul Kripke and other logicians in the 1950s. A proposition is necessarily true if and only if it is true in all possible worlds. With respect to a *static constraint* declared for a given business domain, a possible world corresponds to a *state of the fact model* that might exist at some point in time.

The constraint C1 in the example above means that for each state of the fact model, each instance in the population of Person is born in at most one country.

A proposition is possible if and only if it is true in at least one possible world. A proposition is impossible if and only if it is true in no possible world (i.e., it is false in all possible worlds).

In the example above, constraint C1 may be reformulated as the following negative verbalization:

C1” **It is impossible that the same** Person was born in **more than one** Country.

In practice, both positive and negative verbalizations are useful for validating constraints with domain experts, especially when illustrated with sample populations that provide satisfying examples or counter-examples respectively. The approach described here does not stipulate a high level language for logical rule verbalization, so many alternative verbalizations may be used.

Many business constraints are deontic rather than alethic in nature. To avoid confusion, we recommend that, when declaring a deontic constraint, the deontic modality always be explicitly included.

Consider the following static, deontic constraint.

C2 **It is obligatory that each** Person is a husband of **at most one** Person.

If this logical rule were instead expressed simply as “**each** Person is a husband of **at most one** Person”, it would not be obvious that a deontic interpretation was intended. The deontic version indicates a condition that *ought* to be satisfied, while recognizing that the condition *might* not be satisfied. Including the obligation operator makes the logical rule much weaker than a necessity claim, since it allows that there could be some states of the fact model where a person is a husband of more than one wife (excluding same-sex unions from instances of the husband relationship). For such cases of polygamy, it is important to know the facts indicating that the person has multiple wives. Rather than reject this possibility, we allow it and then typically perform an action that is designed to minimize the chance of such a situation arising again (e.g., send a message to inform legal authorities about the situation).

Constraint C2 may be reformulated as either of the following negative verbalizations:

C2’ **It is forbidden that the same** Person is a husband of **more than one** Person.

C2” **It is not permitted that the same** Person is a husband of **more than one** Person.

In practice, most logical rules include only one modal operator, and this operator is the main operator of the whole logical rule expression. For these cases, we simply tag the constraint as being of the modality corresponding to its main operator, without committing to any particular modal logic. Apart from this modality tag, there are some basic modal properties that may be used in transforming the original high level expression of the logical rule into a standard logical formulation. At a minimum, these include the modal negation rules.

We also make use of equivalences that allow one to move the modal operator to the front of the formula.

For example, suppose the user formulates logical rule C1 instead as:

**For each Person, it is necessary that that Person was born in at most one Country.**

The modal operator is now embedded in the scope of a universal quantifier. To transform this logical rule to a standard logical formulation that classifies the logical rule as an alethic necessity, we move the modal operator before the universal quantifier, to give:

**It is necessary that each Person was born in at most one Country.**

For such tasks, we assume that the Barcan formulae and their converses apply, so that  $\leq$  and  $\forall$  are commutative, as are  $\diamond$  and  $\exists$ . In other words:

$$\forall x \leq Fx \equiv \leq \forall x Fx$$

$$\exists x \diamond Fx \equiv \diamond \exists x Fx$$

While these commutativity results are valid for all normal, alethic modal logics, some philosophical concerns have been raised about these equivalences (e.g., see sections 4.6-4.8 of [Gir12000]).

As a deontic example, suppose the user formulates logical rule C2 instead as:

**For each Person, it is obligatory that that Person is a husband of at most one Person.**

Using a deontic variant of the Barcan equivalences, we commute the  $\forall$  and **O** operators, thus transforming the logical rule to the deontic obligation:

**It is obligatory that each Person is a husband of at most one Person.**

So far, our logical rule examples have included just one modal operator, which (perhaps after transformation) also turns out to be the main operator. Ignoring dynamic aspects, we may handle such cases without needing to commit to the formal semantics of any specific modal logic. The only impact of tagging a logical rule as a necessity or obligation is on the logical rule enforcement policy. Enforcement of a necessity rule should never allow the necessity rule to be violated. Enforcement of an obligation rule should allow states that do not satisfy the obligation rule condition, and take some other remedial action: the precise action to be taken is not specified in SBVR, as it is out of scope for the proposal (logical rule enforcement and logical rule management are to be addressed in separate proposals). At any rate, a business person ought to be able to specify a deontic rule first at a high level, without committing at that time to the precise action to be taken if the condition is not satisfied; of course, the action still needs to be specified later in refining the logical rule to make it fully operational.

### 10.1.1.5 Static, Alethic Constraints

Logic rule formulations may make use of two alethic modal operators:  $\leq$  = it is necessary that;  $\diamond$  = it is possible that. Static constraints are treated as alethic necessities by default, where each state of the fact model corresponds to a possible world.

Given the fact type Person was born in Country, the constraint “**Each Person was born in at most one Country**” may be captured by an SBVR logical formulation that may be automatically translated to the formula  $\forall x:\text{Person} \exists^{0..1} y:\text{Country } x \text{ was born in } y$ . This formula is understood to be true for each state of the knowledgebase. Pragmatically, the logical rule is understood to apply to all future states of the fact model, until the logical rule is revoked or changed. This understanding could be made explicit by prepending the formula with  $\leq$  to yield the modal formula  $\leq \forall x:\text{Person} \exists^{0..1} y:\text{Country } x \text{ was born in } y$ .

For compliance with Common Logic, formulae such as those in the preceding example could then be treated as irregular expressions, with the modal necessity operator treated as an uninterpreted symbol (e.g., using “[N]” for  $\leq$ ). However we leave this understanding as implicit, and do *not* commit to any particular modal logic.

For the model theory, we omit the necessity operator from the formula. Instead, we merely tag the logical rule as a necessity. The implementation impact of the alethic necessity tag is that any attempted change that would cause the model of the business domain to violate the constraint must be dealt with in a way that ensures the constraint is still satisfied (e.g., reject the change, or take some compensatory action).

Typically, the only modal operator in an explicit logical rule formulation is  $\leq$ , and this is at the front of the logical rule. This common case was covered earlier. If an alethic modal operator is placed elsewhere in the logical rule, we first try to “normalize” it by moving the modal operator to the front, using transformation rules such as the modal negation rules ( $\sim\leq p \equiv \diamond\sim p$ ;  $\sim\diamond p \equiv \leq\sim p$ ) and/or the Barcan formulae and their converses ( $\forall x\leq\Phi x \equiv \leq\forall x\Phi x$  and  $\exists x\diamond\Phi x \equiv \diamond\exists x\Phi x$ , i.e.  $\leq$  and  $\forall$  are commutative, as are  $\diamond$  and  $\exists$ ).

For example, the embedded formulation “ $\forall x:\text{Person } \leq \exists^{0..1}y:\text{Country } x \text{ was born in } y$ ” (**For each Person, it is necessary that that Person was born in at most one Country.**) may be transformed into “ $\leq \forall x:\text{Person } \exists^{0..1}y:\text{Country } x \text{ was born in } y$ ” (**It is necessary that each Person was born in at most one Country.**)

We also allow use of the following equivalences:  $\leq\leq p \equiv \leq p$ ;  $\diamond\diamond p \equiv \diamond p$ ;  $\leq\diamond\leq p \equiv \leq\diamond p$ ;  $\diamond\leq\diamond p \equiv \diamond\leq p$ . These hold in S4, but not in some modal logics, e.g., K or T [Girl2000, p. 35].

To make life interesting, SBVR also allows a single logical rule to include multiple occurrences of modal operators, including the nesting of a modal operator within the scope of another modal operator. While this expressibility may be needed to capture some real business rules, it complicates attempts to provide a formal semantics.

In extremely rare cases, a formula for a static logical rule might contain an embedded alethic modality that cannot be eliminated by transformation. For such cases, we could retain the modal operator in the logical rule formulation and adopt the formal semantics of a particular modal logic. There are many normal modal logics to choose from (e.g., K, K4, KB, K5, DT, DB, D4, D5, T, Br, S4, S5) as well as many non-normal modal logics (e.g., C2, ED2, E2, S0.5, S2, S3). For a discussion of these logics, and their inter-relationships, see [Girl2000] (esp. pp. 48, 82). For SBVR, if we decide to retain the embedded alethic operator for such cases, we choose S4 for the formal semantics. The possibility of schema evolution along with changes to necessity constraints may seem to violate S4, where the accessibility relationship between possible worlds is transitive, but we resolve this by treating such evolution as a metalevel concern. Alternatively, we may handle such very rare cases by moving the embedded alethic operators down to domain-level predicates (e.g., is necessary) in a similar fashion to the way we deal with embedded deontics (see later).

### 10.1.1.6 Static, Deontic Constraints

Constraint formulations may make use of the standard deontic modal operators ( $O$  = it is obligatory that;  $P$  = it is permitted that) as well as  $F$  = it is forbidden that (defined as  $\sim P$ , i.e., “It is not permitted that”).

If the logical rule includes exactly one deontic operator,  $O$ , and this is at the front, then the logical rule may be formalized as  $Op$ , where  $p$  is a first-order formula that is tagged as obligatory (rather than necessary). For the purposes of this section, this tag is assigned only the following informal semantics: it ought to be the case that  $p$  (for all future states of the fact model, until the constraint is revoked or changed). The implementation impact is that it is possible to have a state in which the logical rule’s condition is violated (i.e., not satisfied), in which case some appropriate action (currently unspecified) ought to be taken to help reduce the chance of future violations. A later submission to the OMG is intended to address logical rule enforcement, including the specification of appropriate actions in response to deontic rule “violations”.

From a model-theoretic perspective, a model is an interpretation where each *non-deontic* formula evaluates to true, and the model is classified as a *permitted model* if the  $p$  in each deontic formula (of the form  $Op$ ) evaluates to true, otherwise the model is a *forbidden model* (though it is still a model). Note that this approach removes any need to assign a truth value to expressions of the form  $Op$ .

For example, suppose the fact type Person is a husband of Person is declared to be many to many, but that each role of this fact type has a deontic uniqueness constraint to indicate that the fact type *ought* to be 1:1. The deontic constraint on the husband fact type role verbalizes as: **It is obligatory that each Person is a husband of at most one Person**. This formalizes as  $O"x:Person \text{ \$}^{0..1}y:Person$   $x$  is a husband of  $y$ , which may be captured by entering the logical rule body as " $x:Person \text{ \$}^{0..1}y:Person$   $x$  is a husband of  $y$  and tagging the logical rule as deontic. The other deontic constraint (each wife should have at most one husband) may be handled in a similar way. A more detailed treatment of this example is included in Annex J.

Note that some formulae allowed by SBVR are illegal in some deontic logics (e.g., iterating modal operators such as  $OPp$  is forbidden in von Wright's deontic logic), and deontic logic itself is "rife with disagreements about what should be the case" [Girl2000, p. 173].

If a deontic modal operator is embedded later in the logical rule formulation, we first try to "normalize" the formula by moving the modal operator to the front, using transformation logical rules such as  $p \supset Oq \equiv O(p \supset q)$  or deontic counterparts to the Barcan formulae.

In some cases, a formula for a static logical rule might contain an embedded deontic modality that cannot be eliminated by transformation. In this case, we still allow the business user to express the logical rule at a high level using such embedded deontic operators, but *where possible* we transform the formula to a first-order formula without modalities by *replacing the modal operators by predicates at the business domain level*. These predicates (e.g., is forbidden) are treated like any other predicate in the domain, except that their names are reserved, and they are given some basic additional formal semantics to capture the deontic modal negation rules: it is not obligatory that  $\equiv$  it is permitted that it is not the case that ( $\sim Op \equiv P\sim p$ ); it is not permitted that  $\equiv$  it is obligatory that it is not the case that ( $\sim Pp \equiv O\sim p$ ). For example, these logical rules entail an exclusion constraint between the predicates is forbidden and is permitted.

This latter approach may also be used as an alternative to tagging a logical rule as deontic, thereby (where possible) moving deontic aspects out of the metamodel and into the business domain model.

For example, consider the following logical rule:

Car rentals ought not be issued to people who are barred drivers at the time the rental was issued.

This deontic constraint may be captured by the following textual constraint on the domain fact type CarRental is forbidden:

CarRental is forbidden **if**

CarRental was issued at Time **and**

CarRental was issued to Person **and**

Person is a barred driver at Time.

The fact type Person is a barred driver at Time is derived from other fact types (Person was barred at Time, Person was unbarred at Time) using the derivation rule:

Person is a barred driver at Time<sub>1</sub> **iff**

Person was barred at a Time<sub>2</sub> <= Time<sub>1</sub> **and**

Person was **not** unbarred at a Time<sub>3</sub> **between** Time<sub>2</sub> **and** Time<sub>1</sub>.

The deontic constraint may be formalized by the first-order formula:  $\forall x:\text{CarRental} \forall y:\text{Person} \forall t:\text{Time} [(x \text{ was issued at } t \ \& \ x \text{ was issued to } y \ \& \ y \text{ is a barred driver at } t) \supset x \text{ is forbidden}]$ . This schema allows for the possible existence of forbidden car rentals; if desired, some fact types could be added to describe actions (e.g., sending messages) to be taken in reaction to such an event.

As a second example, consider the following deontic rule:

It is forbidden that more than three people are on the EU-Rent Board.

Suppose the underlying schema includes the fact type: Person is on Board. This may be used to define the derived fact type Board has NrMembers using the derivation rule: nrMembers **of** Board = **count each** Person **who** is on Board. Objectify this derived fact type as BoardHavingSize, and then add the fact type BoardHavingSize is forbidden. The deontic constraint may now be captured by the following textual constraint on the derived fact type:

BoardHavingSize is forbidden **if**

BoardHavingSize is of a Board

**that** has BoardName 'EU-Rent Board'

**and** has NrMembers > 3.

As a third example, our earlier schema for current marriage may be recast by objectifying the fact type Person is a husband of Person as CurrentMarriage, and recognizing the link fact types Person is a husband in CurrentMarriage and Person is a wife in CurrentMarriage. The deontic constraints may now be formulated as textual constraints on the fact type CurrentMarriage is forbidden as follows:

CurrentMarriage is forbidden **if**

**a** Person<sub>1</sub> **who** is a husband in CurrentMarriage

is a husband of **more than one** Person<sub>2</sub>.

CurrentMarriage is forbidden **if**

**a** Person<sub>1</sub> **who** is a wife in CurrentMarriage

is a wife of **more than one** Person<sub>2</sub>.

Extended treatments of the examples above are provided in Annex J.

The approach to objectification described here works for those cases where a fact (proposition taken to be true) is being objectified (which covers the usual cases of nominalization, including the EU-Rent Board and current marriage examples discussed earlier), but it does not handle cases where no factual claim is being made of the proposition.

SBVR is intended to cater for logical rules that embed possibly non-factual propositions. However, there does not appear to be any simple solution to providing explicit, formal semantics for such logical rules.



As a nasty example, consider the following logical rule:

It is not permitted that some department adopts a logical rule that says it is obligatory that each employee of that department is male.

This example includes the mention (rather than use) of an open proposition in the scope of an embedded deontic operator. One possible, though weak, solution is to rely on reserved domain predicates to carry much of the semantics implicitly. For example, suppose the schema includes the following fact types: Person is male; Person works for Department; Department adopts Logic Rule. Objectify Department adopts Logical rule as LogicRuleAdoption, and add the following fact types: LogicRuleAdoption is forbidden; LogicRule obligates the actualization of PossibleAllMaleState; PossibleAllMaleState is actual. This uses the special predicates “obligates the actualization of” and “is actual”, as well as a type of individual “PossibleAllMaleState” which includes all conceivable all-male-states of departments, whether actual or not. The derived fact type PossibleAllMaleState is actual may be defined using the derivation rule:

PossibleAllMaleState is actual **iff**  
PossibleAllMaleState is of **a** Department **and**  
**each** Person **who** works for **that** Department is male.

i.e., "x:PossibleAllMaleState [x is actual  $\frac{1}{2}$   $\exists$ y:Department (x is of y & "z:Person (z works for y ... z is male))]. The deontic constraint may now be captured by the following textual constraint on the fact type LogicRuleAdoption is forbidden:

LogicRuleAdoption is forbidden **if**  
LogicRuleAdoption is by **a** Department  
**and** is of **a** LogicRule  
**that** obligates the actualization of **a** PossibleAllMaleState  
**that** is of **the same** Department.

i.e., "x:LogicRuleAdoption "y:Department "z:LogicRule "w:PossibleAllMaleState [(x is by y & x is of z & z obligates the actualization of w & w is of y) ... x is forbidden]

The formalization of the deontic constraint works, because the relevant instance of PossibleAllMaleState exists, regardless of whether or not the relevant depart actually is all male. The “obligates the actualization of” and “is actual” predicates embed a lot of semantics, which is left implicit. While the connection between these predicates is left informal, the derivation rule for PossibleAllMaleState is actual provides enough semantics to enable human readers to understand the intent. An extended treatment of this example is provided in Annex J.

Alternatively, we could capture the structure of the logical rule using the current semantic formulation machinery, and then adopt one of two extremes: (1) treat the logical rule overall as an uninterpreted sentence, or informal comment, for which humans are to provide the semantics; (2) translate the semantic formulation directly into higher-order logic, which permits logical formulations (which connote propositions) to be predicated over. The complexity and implementation overhead of option (2) would seem to be very substantial.

We could try to push such cases down to first-order logic by providing the equivalent of the semantic formulation machinery as a predefined package that may be imported into a domain model, and then identifying propositions by means of a structured logical formulation. But that seems a fudge, because in order to assign formal semantics to such expressions, we must effectively adopt the higher-order logic proposal mentioned in the previous paragraph.

Pat Hayes has indicated his intent to add support for reification as an extension to Common Logic at some future date. This support is intended to cater for objectification of propositions that are already being asserted as facts (i.e., propositions being used), as well as propositions for which no factual claim is made (i.e., propositions being mentioned). When available, his treatment for the latter case may offer a better solution for the problem under consideration. His intent is to allow quantification and predication over propositions (or expressions that declare propositions), regardless of whether truth claims are being asserted of those propositions, while still retaining a first-order approach. We might be able to adopt whatever he proposes in this regard to provide a formal semantics for such problematic logical rules.

### 10.1.1.7 Derivation Rules

The SBVR approach supports logical rules for deriving types of individuals (subtype definitions) or fact types using either 'if-and-only-if' (equivalence) formulations for full derivation, or 'if'-rules for partial derivation. A subtype may be fully derived (defined in terms of fact type roles played by its supertype), asserted (without a derivation rule), or partly derived.

Here is one simple example of each kind of derivation rule, stated first using a high-level textual language, as described earlier, and then recast as a predicate logic formula. The transformation from a semantic formulation structure in a high level language into predicate logic is straightforward.

Derivation rule for fully derived subtype:

**Each** Australian **is a** Person **who** was born in Country 'AU'.

$\forall x [\text{Australian } x \equiv (\text{Person } x \ \& \ \exists y:\text{Country} \ \exists z:\text{CountryCode} (x \text{ was born in } y \ \& \ y \text{ has } z \ \& \ z = \text{'AU'}))]$

Derivation rule for partly derived subtype:

Person<sub>1</sub> **is a** Grandparent **if** Person<sub>1</sub> is a parent of **some** Person<sub>2</sub> who is a parent of **some** Person<sub>3</sub>.

$\forall x:\text{Person} [\text{Grandparent } x \subset \exists y:\text{Person} \ \exists z:\text{Person} (x \text{ is a parent of } y \ \& \ y \text{ is a parent of } z)]$

Derivation rule for fully derived fact type:

Person<sub>1</sub> is an uncle of Person<sub>2</sub> **iff** Person<sub>1</sub> is a brother of **some** Person<sub>3</sub> **who** is a parent of Person<sub>2</sub>.

$\forall x:\text{Person} \ \forall y:\text{Person} [x \text{ is an uncle of } y \equiv \exists z:\text{Person} (x \text{ is a brother of } z \ \& \ z \text{ is a parent of } y)]$

Derivation rule for partly derived fact type:

**If a** Patient **smokes then that** Patient is cancer-prone.

$\forall x:\text{Patient} (\text{smokes } x \supset \text{cancer-prone } x)$

### 10.1.1.8 Dynamic Constraints

Dynamic constraints apply restrictions on possible transitions between business states. The constraint may simply compare one state to the next.

Salaries should never decrease.

Alternatively, the constraint may compare states separated by a given period.

Invoices ought to be paid within 30 days of being issued.

The invoice logical rule might be formally expressed in a high level rules language thus, assuming the fact types Invoice was issued on Date and Invoice is paid on Date are included in the conceptual schema:

**For each Invoice, if that Invoice was issued on Date<sub>1</sub>  
then it is obligatory that  
that Invoice is paid on Date<sub>2</sub> where Date<sub>2</sub> <= Date<sub>1</sub> + 30 days.**

This might now be normalized to the following formulation, moving the deontic operator to the front:

**It is obligatory that each Invoice that was issued on Date<sub>1</sub> is paid on Date<sub>2</sub>  
where Date<sub>2</sub> <= Date<sub>1</sub> + 30 days.**

There are two issues here. First, what logical rules did we rely on to license the transformation of the logical rule? It would seem that we require an equivalence rule such as  $p \supset Oq \equiv O(p \supset q)$ . While this formula is actually illegal in some deontic logics, it does seem intuitively acceptable. At any rate, the preliminary transformation work in normalizing a logical rule formulation might involve more than just the Barcan equivalences or their deontic counterparts. In principle, this issue might be ignored for interoperability purposes, so long as the business domain expert is able to confirm that the final, normalized formulation (perhaps produced manually by the business rules modeler) agrees with their intended semantics; it is only the final, normalized formulation that is used for exchange with other software tools.

The second issue concerns the dynamic nature of the logical rule. While it is obvious how one may actually implement this logical rule in a database system, capturing the formal semantics in an appropriate logic (e.g., a temporal or dynamic logic) is a harder task. One possibility is to provide a temporal package that may be imported into a domain model, in order to provide a first-order logic solution. Another possibility is to adopt a temporal modal logic (e.g., treat a possible world as a sequence of accessible states of the fact model). It may well be reasonable to defer decisions on formal semantics for dynamic rules to a later version of the SBVR standard.

### 10.1.1.9 Higher-order Logic

Currently, SBVR allows users to either stay with first-order logic, or adopt higher-order logic restricted to Henkin semantics (e.g., for dealing with categorization types). In general, standard higher-order logic allows quantification over uncountably many possible predicates (or functions). If  $D$  = the domain of individuals, then the range of any unary predicate variable  $R$  is the entire power set  $P(D)$  (i.e., the set of all subsets of  $D$ ), the range of any binary predicate variable is the Cartesian product  $P(D) \times P(D)$ , and so on for higher arity predicates. If  $D$  includes a denumerable (countable infinite, i.e.,  $|D| = \aleph_0$ ) set, such as the natural numbers, then  $P(D)$  is uncountably infinite. In contrast, Henkin semantics restricts quantifiers to range over only individuals and those predicates (or functions) that are specified in the universe of discourse (a.k.a. business domain), where the  $n$ -ary predicates/functions ( $n > 0$ ) range over a fixed set of  $n$ -ary relations/operations. By restricting the ranges of predicate and function variables, the Henkin interpretation retains certain desirable first-order properties (e.g., completeness, compactness, and the Skolem-Löwenheim theorems) that are lost in the standard interpretation of higher-order logic.

Common Logic adopts the Henkin restriction on quantifier ranges, but does not adopt the Axiom of Comprehension, which states that for each property there exists a set of elements having that property, i.e., for any formula  $\delta(x)$  where  $x$  (possibly a vector) is free in  $\delta$ ,  $\exists A \forall x [x \in A \equiv \delta(x)]$ . The intent of the Comprehension axiom (to ensure that every formula specifies a set) may also be achieved by using lambda abstraction to name the set, e.g.,  $\lambda x. \delta(x)$ , which is equivalent to the set comprehension  $\{x / \delta(x)\}$ . The Axiom of Comprehension leads to Russell's paradox (substituting  $x \notin x$  for  $\delta(x)$  generates a contradiction since  $\{x / x \notin x\}$  is simultaneously a member of itself and not a member of itself). The paradox may be avoided either by rejecting the comprehension

axiom (e.g., replacing it by the weaker axiom of separation, as in Zermelo-Fraenkel set theory) or by restricting the language so that formulae such as  $x \notin x$  are illegal (as in Russell's type theory, where a set may belong only to a set of higher order).

Here we use set comprehensions (in a restricted sense) to define projections on schema path expressions, as a way to specify result sets.

For example, given the fact type Employee(EmpNr) works for Company(Name), the query "Who works for Microsoft?" corresponds to the following set comprehension:

$$\{x:\text{Employee} \mid \exists y:\text{Company}; z:\text{CompanyName} (x \text{ works for } y \ \& \ y \text{ has } z \ \& \ z = \text{'Microsoft'})\}$$

The formal semantics of such conceptual queries is based on that of the Conquer language, which provides a sugared version of sorted finitary first-order logic with set comprehension [Anto2001].

The use here of set comprehension is quite restricted. Any expression we use to define a set must ultimately be expressible only in terms of some basic logical operators (e.g.,  $\&$ ) as well as predefined ground fact types which must be either elementary or existential. Hence we adopt a limited version of the axiom of comprehension. Common Logic is open to extensions that adopt restricted versions of the comprehension axiom. To avoid Russell's paradox, we treat formulae such as  $x \notin x$  as illegal. The "is an instance of" predicate caters for set membership, but is constrained to be irreflexive, and the formation rules do not permit expressions of the form  $x \in x$  – in other words, we cannot make statements involving self-membership. We do not adopt a type theory such as Russell's type theory, where each set may belong only to a set of a higher type.

The decision on whether to use higher-order types mainly impacts the following three aspects of fact modeling: categorization schemes, un-normalized structures, and crossing levels/metalevels within the same model. In [Halp2004], some ways are suggested to avoid higher-order types, by treating types as intensional individuals whose instances may sometimes be in 1:1 correspondence (but not identical) to subtypes, by requiring subtype definitions to be informative, by remodeling (including demotion of metadata to data), and by treating types as individuals in separate models. For further discussion, see [Halp2004].

*Acknowledgement:* We gratefully acknowledge the assistance of Pat Hayes ([www.ihmc.us/users/phayes](http://www.ihmc.us/users/phayes)) in addressing some of the logical semantics topics in this document.

## 10.1.2 Formal Logic & Mathematics in General

---

### Formal Logic & Mathematics Vocabulary

Language: [English](#)

---

#### alethic modality

Source: CDP

Definition: Historically, any of the four central ways or modes in which a given proposition might be true or false: [necessity](#), contingency [see note 2], [possibility](#), and impossibility.

Note: (1) Although these "modes" have historically been thought of as ways in which a proposition might be true, we think of them as ways in which one might think of the truth of a proposition: e.g., that a proposition be qualified with the alethic modality "necessity" does not imply it is a fact, but only signifies that the semantic community is considering it (takes it to be) necessarily true. For some issues arising from the former approach, cf. CDP, s.v. *intensional logic*. For a thorough critique of

it, see PEIL. The four “modal negation equivalences” (MLP, p. 3), such as  $\Box p \equiv \sim\Diamond\sim p$ , still hold under the latter approach (cf. LEVS, p. 135), which is the more useful one in the fields of linguistic semantics and linguistic pragmatics.

Note: (2) The four alethic modalities which we consider most basic, and to which the four “modal negation equivalences” (MLP, p. 3) apply, are [necessity](#), [possibility](#), and their respective negations (non-necessity and impossibility). ‘Contingency’, the idea “neither impossible nor necessary” (CDP), is not relevant in a business rules context.

Note: (3) Alethic modal logic differs from deontic modal logic in that the former deals with people’s estimate(s) of the possible truth of some proposition, whereas deontic modal logic deals with people’s estimate(s) of the social desirability of some particular party’s making some proposition true.

### antecedent

Source: adapted from GFOL

Definition: The [wff](#) in [or more specifically, the proposition-[wff](#) in or else the proposition denoted by] the if-clause of an [implication](#).

Note: Interpolation ours. Otherwise the definition is from GFOL.

### argument

Source: GFOL

Definition: a [logical-] subject-term for a [predicate](#).

Note: Interpolation in square brackets ours. By “logical subject” we mean an object playing a role (i.e., an object filling an object hole) in a logical predicate. Thus there may be one or more logical-subject-terms in a logical predicate.

### arity

Source: IMRD (pp. 10, 64)

Definition: A logical predicate’s number of roles (i.e., of object holes).

Note: A function may be thought of as a relation; accordingly, we treat a function as a logical predicate. MATH defines arity of a function thus: “The number of arguments taken by something, usually applied to functions: an  $n$ -ary function is one with an arity of  $n$ , i.e., it takes  $n$  arguments. Unary is a synonym for 1-ary, and binary is a synonym for 2-ary.”

### atomic formula

Source: GFOL [“atom”]

Definition: In predicate logic, a [wff](#) without [quantifiers](#) or connectives.

Note: (1) This definition is from the cited source s.v. atom, which we deem a synonym.

Note: (2) LSO says of atomic formula: “The simplest sort of [wff](#) of a formal language; an atomic formula of the language of predicate logic is a predicate letter followed by zero or more name letters”. Yet it can also be a propositional variable or a propositional constant, depending on context.

### consequent

Source: GFOL

Definition: The [wff](#) in [or more specifically, the proposition-[wff](#) in or else the proposition denoted by] the then-clause of an [implication](#).

Note: Interpolation ours.

## deontic modality

- Source: CDP ["deontic operator"]; LEVS (pp. 276-77); LSO (p. 302); MLP (pp. 170-76)
- Definition: Any of the four central ways or modes in which one might think of the social desirability of a certain other person(s)'s making true some proposition, that is, the social desirability that the act(s) be performed, by a certain other person(s), that would make the proposition true; viz., [permission](#), [obligation](#), and their respective negations: nonpermission (forbidden/prohibition) and non-obligation.
- Note: (1) The definition given is not quoted directly from any source, since we have not found the term defined as such anywhere. Rather, we have based our definition on passages mainly in the above-cited sources.
- Note: (2) Alethic modal logic differs from deontic modal logic in that the former deals with people's estimate(s) of the possible truth of some proposition, whereas deontic modal logic deals with people's estimate(s) of the social desirability of some particular party's making some proposition true.

## domain

- Source: GFOL
- Definition: Of an interpretation of a formal language of predicate logic, the set of objects that may serve as the assigned referents of the constants of the language, the [arguments](#) of functions, and the [arguments](#) of [predicates](#).

## domain grammar

- Source: META (p. 4); HALT89 (sec. 3.2); IMRD (pp. 27-30)
- Definition: The formation rules determining what is a [wff](#) in a given domain-specific formal language.
- Note: Another term for that which is called in ORM "conceptual schema". The definition given above is not quoted directly from any source, since we have not found the term defined as such anywhere. Rather, we have based our definition on passages mainly in the above-cited sources.

## first-order instance

- Source: GFOL
- Definition: The objects or elements taken as the [logical] subjects of the [predicates](#) of first-order predicate logic.
- Definition: [CLARIFIED DEFINITION] object or element taken as a logical subject of a predicate of first order logic.
- Note: And the distinguishing characteristic of "first-order" predicate logic, in turn, is the additional restriction, re the formation of [wffs](#), that subjects of [predicates](#) cannot themselves be [types](#) or [predicates](#), but rather only individuals (or individual-constants, individual-variables, or function-expressions). See [first-order type](#).

## first-order type

- Source: LSO (pp. 280-84) [and "type system"]; META (p. 140); TTGG (p. 5)
- Definition: A [type](#) whose extension includes no types or predicates, only first order [instances](#), in accordance with the grammatical restrictions in first-order predicate logic.
- Note: The definition given is not quoted directly from any source, since we have not found the term defined as such anywhere. Rather, we have based our definition on passages mainly in the above-cited sources.

### formal model

Source: based on GFOL ["model"]; META (pp. 5,6, 148-49)

Definition: An *interpretation* supplies semantics (referents) for a given formal language, in relation to some domain or universe. It specifies referents for the nonlogical symbols occurring in the formal language. A *formal model* of a given wff or set of wffs in a formal language is an interpretation of the language for which the wffs are considered true.

### implication

Source: GFOL

Definition: expression of the form, "if A, then B," when A and B stand for wffs or propositions. The wff in the if-clause is called the antecedent (also the implicans and protasis). The wff in the then-clause is called the consequent (also the implicate and apodosis). Also called a conditional, or a conditional statement.

Note: In SBVR we treat "implication" as if it is "material implication", i.e., ' $p \rightarrow q$ ' is equivalent to ' $\sim p \vee q$ '.

### integer

Source: GFOL ["integers"]

The natural numbers supplemented by their negative counterparts. The set {...-3, -2, -1, 0, 1, 2, 3...}.

### logical variable

Source: GFOL

Definition: A symbol whose referent varies or is unknown. A place-holder, as opposed to an abbreviation or name (a constant).

Note: This definition is from the cited source s.v. variable, which we deem a synonym.

### member

Source: DEAN (p. 6); GFOL ["membership"]

Definition: An element belonging to a set.

Note: The definition given is not quoted directly from any source, since we have not found the term defined as such anywhere. Rather, we have based our definition on passages mainly in the above-cited sources.

### modal logic

Source: SEP

Definition: Narrowly construed, modal logic studies reasoning that involves the use of the expressions 'necessarily' and 'possibly'. However, the term 'modal logic' is used more broadly to cover a family of logics with similar rules and a variety of different symbols.

### necessity

Source: CDP

Definition: a modal property that qualifies an assertion of a whole proposition just when it is not considered possible that the proposition is false.

Note: The definition given is not quoted directly from any source. Rather, we have based our definition on passages mainly in the above-cited source. See also alethic modality.

### obligation

- Source: CDP ["deontic logic"]; MLP (pp. 170-76)
- Definition: One of the four main [deontic modalities](#), which qualifies as socially obligatory the making true a certain proposition (i.e., the doing a certain act) by a certain party or parties.
- Note: The definition given is not quoted directly from any source, since we have not found the term defined as such anywhere. Rather, we have based our definition on passages mainly in the above-cited sources.

### permission

- Source: CDP ["deontic logic"]; MLP (pp. 170-76)
- Definition: One of the four main [deontic modalities](#), which qualifies as socially permissible the making true a certain proposition (i.e., the doing a certain act) by a certain party or parties.
- Note: The definition given is not quoted directly from any source, since we have not found the term defined as such anywhere. Rather, we have based our definition on passages mainly in the above-cited sources.

### population

- Source: IMRD (p. 164)
- Definition: The extension of a [type](#) (whether object type, fact type, or role) for a given state of the business domain.

### possibility

- Source: CDP
- Definition: a modal property that qualifies an assertion of a whole proposition just when it is considered possible that the proposition is true.
- Note: The definition given is not quoted directly from any source. Rather, we have based our definition on passages mainly in the above-cited source. See also [alethic modality](#).

### predicate

- Source: GFOL
- Definition: Intuitively, whatever is said of the subject[s] of a sentence. A function from individuals (or a sequence of individuals) to truth-values.
- Note: Interpolation in square brackets ours. A predicate is distinguished from others by sentence structure, not by proposition/meaning (see IMRD, pp. 63-66). Propositions or meanings distinguish fact types, each of which may have 1 or more predicates.

### prohibition

- Source: CDP ["deontic logic"]; MLP (pp. 170-76)
- Definition: One of the four main [deontic modalities](#), nonpermissibility, which qualifies as socially not permissible the making true a certain proposition (i.e., the doing a certain act) by a certain party or parties.
- Note: See also [permission](#). The definition given is not quoted directly from any source, since we have not found the term defined as such anywhere. Rather, we have based our definition on passages mainly in the above-cited sources.



### proposition

- Source: DL (p. 4)
- Definition: That which is asserted when a sentence is uttered or inscribed.
- Note: Generally understood as “the meaning of” a declarative sentence. GFOL defines it thus: “In logic generally (for some), the meaning of a sentence that is invariant through all the paraphrases and translations of the sentence.”

### propositional operator

- Source: PLTS
- Definition: An operator (or connective) joins ... statements [i.e., propositions or proposition-wffs] into compounds.... Connectives include conjunction, disjunction, implication and equivalence. Negation is the only operator that is not a connective; it affects single statements [i.e., propositions or proposition-wffs] only, and does not join statements [i.e., propositions or proposition-wffs] into compounds.
- Note: By “proposition-wff” we mean a proposition-constant or proposition-variable, or a predicate supplied with arguments so as to yield a proposition.

### quantifier

- Source: GFOL
- Definition: In predicate logic, a symbol telling us ... how many objects (in the domain) [instantiate] the predicate.... The quantifier applies to, or binds, variables which stand as the arguments of predicates. In first-order logic these variables must range over individuals; in higher-order logics they may range over predicates.
- Note: Interpolation in square brackets ours.

### restricted higher-order instance

- Source: HALT2004 (pp. 2-4, 7); MEN97 (pp. 378-80)
- Definition: instance of a restricted higher-order type.
- Note: The definition given is not quoted directly from any source, since we have not found the term defined as such anywhere. Rather, we have based our definition on passages mainly in the above-cited sources.

### restricted higher-order type

- Source: HALT2004 (pp. 2-4, 7, 8); MEN97 (pp. 378-80)
- Definition: A *higher-order type* includes an instance that is itself a type. For SBVR, we *restrict higher-order types* to Henkin semantics, limiting the range of predicates/functions over which we may quantify to a fixed set, rather than allowing full range over power-sets. This restriction retains useful properties of first-order logic (e.g., completeness).
- Note: The definition given is not quoted directly from any source, since we have not found the term defined as such anywhere. Rather, we have based our definition on passages mainly in the above-cited sources.

### set

- Source: GFOL
- Definition: Intuitively, a collection of elements (called members). In a set, the order of members is irrelevant, and repetition of members is [also irrelevant]. The intuitive notion of a set leads to

paradoxes, and there is considerable mathematical and philosophical disagreement on how best to refine the intuitive notion.

Note: Interpolation in square brackets ours.

### state of affairs

Source: CDP

Definition: A possibility, actuality or impossibility of the kind expressed by a nominalization of a declarative sentence. E.g., “This die comes up six” may be nominalized by “that this die comes up six” or “this die’s coming up six”. The resulting nominalizations might be interpreted as naming corresponding propositions or states of affairs.

### subset

Source: GFOL

Definition: set all of whose members belong to a second set (a superset of the subset).

### type

Source: adapted from HALT2004 (p. 8); cf. TTGG (p. 84)

Definition: named set of possible instances, where for any given state of the business domain, exactly one subset of the type is the population of the type in that state.

Note: At any given time, the population of a type is the set of instances of that type that exist in the business domain (i.e., that are referenced within facts that are known and are of interest to the business) at that time. It follows that if two types are equal, then for each state of the business domain they must have the same population.

Note: “Possible instances” here means “instances which are considered part of the type’s population, for some state of the business domain”.

Note: Because it is a formal object that behaves quite differently in first-order predicate logic than in second-order predicate logic (and differently still in third order, and so on), the definition of “type” proves to be anaphoric, having a different denotation depending on whether, in the situation where used, the intended formalization is first-order, second-order, or other-order. In our definitions of first-order type and restricted higher order type, at least some of this indefiniteness is removed (by the specifying of either first-order logic or restricted higher-order logic).

### unbound variable

Source: GFOL

Definition: free variable [which, in GFOL, is defined thus:] In predicate logic, an individual variable at least one of whose occurrences in a wff does not lie within the scope of a quantifier on the same letter.

### wff

Source: GFOL

Definition: Acronym of “well-formed formula”. A string of symbols, each from the alphabet of a formal language, that conforms to the grammar of the formal language. In predicate logic, a closed wff is a wff with no free occurrences of any variable; either it has constants in place of variables, or its variables are bound, or both. Also called a sentence. An open wff is a wff with at least one free occurrence of a variable.

## world

Source:	CSILL
Definition:	A universe, whether real, imaginary, or hypothetical.
Note:	From CSILL: The truth-conditional approach to meaning allows model theory to be extended to the study of natural languages. Sentences and their parts are mapped on to elements of a model, which represents the truth-conditions for the sentences. In possible world semantics, models are not restricted to domains of real entities but include possible objects; that is, model theory can provide truth-conditions in terms of possible worlds, thus allowing meaningful expressions without requiring ontological commitment.

## 10.2 Formal Logic Interpretation Placed on SBVR Terms

This document defines how the SBVR concepts as defined for the [Meaning and Representation Vocabulary](#) and for the [Logical Formulation of Semantics Vocabulary](#) may be mapped to terms of formal logic.

NOTE: In this section, the following textual styles have semantic importance:

- [Logical Formulation of Semantics Vocabulary](#) designations and [Meaning and Representation Vocabulary](#) designations are in *italics*
- Formal logic terms are underlined

## 10.3 Mapping of SBVR Business Terms to Formal Logic

The following tables show the suggested mappings to formal logics terms for the selected subset of (important) SBVR business terms. Any formatted term that is used in the “Mapping to Formal Logic” is either a formal logic term (underlined) or an SBVR business term (*italics*) that is mapped somewhere else in this table.

### 10.3.1 Concepts with Different Mappings for 'First-Order' and 'Restricted Higher-order'

SBVR Term	'First-Order' Mapping to Formal Logic	'Restricted Higher-Order' Mapping to Formal Logic	Comment
<a href="#">object type</a>	<a href="#">first-order type</a>	<a href="#">restricted higher-order type</a>	
<a href="#">instance</a>	<a href="#">first-order instance</a>	<a href="#">restricted higher-order instance</a>	

### 10.3.2 Concepts with a Single Mapping for 'First-Order' and 'Restricted Higher-order'

#### 10.3.2.1 Mapping to Formal Logic

SBVR Term	Mapping to Formal Logic	Comment
<a href="#">state of affairs</a>	<a href="#">state of affairs</a>	
<a href="#">antecedent</a>	<a href="#">antecedent</a>	
<a href="#">atomic formulation</a>	<a href="#">atomic formula</a>	
<a href="#">cardinality</a>	<a href="#">nonnegative integer</a> that is the number of <a href="#">members</a> in a <a href="#">set</a>	
<a href="#">concept</a>	<a href="#">type</a>	
<a href="#">conceptual model</a>	<a href="#">formal model</a>	
<a href="#">conceptual schema</a>	<a href="#">domain grammar</a>	
<a href="#">consequent</a>	<a href="#">consequent</a>	
<a href="#">equivalence / material equivalence</a>	bidirectional <a href="#">implication</a> .	
<a href="#">extension</a>	<a href="#">population</a>	

<a href="#">fact</a>	<a href="#">logical formulation</a> that is taken as true, in the sense that it means a <a href="#">proposition</a> that is taken as true. Each actuality instantiates an unordered set of one or more <a href="#">roles</a> , but for each ordering of <a href="#">roles</a> <a href="#">predicate readings</a> may be given, e.g., ‘Terry likes Norma’ and ‘Norma is liked by Terry’ express the same fact. There is no formal way of determining whether two primitive (non-derived) <a href="#">predicates</a> have the same <a href="#">meaning</a> , but equivalences between <a href="#">predicates</a> may be explicitly asserted if known.	How one ascertains what is true, whether by assertion, observation or other means, is outside the scope of the SBVR specification. However, taking a formulation as true must be consistent with epistemic commitment. The term ‘ <a href="#">fact</a> ’ is here defined to be consistent with the operations of truth-functional logic, which produce results based on true and false.
<a href="#">fact type</a>	<a href="#">type of actuality</a> , e.g., Person likes Person.	
<a href="#">implication / material implication</a>	<a href="#">implication</a>	
<a href="#">inconsequent</a>	<a href="#">logical formulation</a> that is a <a href="#">logical operand</a> irrelevant to the logical result of a <a href="#">logical formulation</a> such as of a <a href="#">whether-or-not formulation</a> .	A term invented for SBVR.
<a href="#">integer</a>	integer	
<a href="#">logical formulation</a>	an abstract expression of a well-formed logical formula ( <a href="#">wff</a> )	
<a href="#">logical negation</a>	<a href="#">logical formulation</a> that applies the logical “NOT” operation ( $\sim$ ) to a <a href="#">negand</a>	
<a href="#">maximum cardinality</a>	<a href="#">cardinality</a> that is a maximum in a range of <a href="#">cardinalities</a> , such as for an <a href="#">at-most-n quantification</a>	
<a href="#">minimum cardinality</a>	<a href="#">cardinality</a> that is a minimum in a range of <a href="#">cardinalities</a> , such as for an <a href="#">at-least-n quantification</a>	
<a href="#">modal formulation</a>	<a href="#">logical formulation</a> ( <a href="#">wff</a> ) that applies a <a href="#">modality</a> to an embedded <a href="#">logical formulation</a> ( <a href="#">wff</a> ).	The embedded wff must either be or else contain (embed directly or indirectly) a closed <a href="#">wff</a> .
<a href="#">necessity</a>	<a href="#">logical formulation</a> whose main (top level) operator is the alethic (it is necessary that). The SBVR term actually means the <a href="#">proposition</a> expressed by such a <a href="#">logical formulation</a> .	

<a href="#">obligation</a>	<a href="#">logical formulation</a> whose main (top level) operator is the deontic O (it is obligatory that). The SBVR term actually means the <a href="#">proposition</a> expressed by such a <a href="#">logical formulation</a> .	
<a href="#">permissibility</a>	<a href="#">logical formulation</a> whose main (top level) operator is the deontic P (it is permitted that). The SBVR term actually means the <a href="#">proposition</a> expressed by such a <a href="#">logical formulation</a> .	
<a href="#">possibility</a>	<a href="#">logical formulation</a> whose main (top level) operator is the alethic $\diamond$ (it is possible that). The SBVR term actually means the <a href="#">proposition</a> expressed by such a <a href="#">logical formulation</a> .	
<a href="#">proposition</a>	<a href="#">proposition</a>	
<a href="#">quantification</a>	<a href="#">logical formulation</a> (wff) that applies a logical quantification operation (i.e., a <a href="#">quantifier</a> ) to a <a href="#">logical variable</a> .	
<a href="#">reference scheme</a>	chosen way of identifying <a href="#">instances</a> of a given <a href="#">concept</a>	
<a href="#">role</a>	an object hole in a <a href="#">predicate</a>	
<a href="#">role binding</a>	connection of an <a href="#">atomic formulation</a> to a literal value or <a href="#">variable</a> for a particular <a href="#">role</a> of the <a href="#">fact type</a> that is the basis of the <a href="#">atomic formulation</a>	
<a href="#">set</a>	<a href="#">set</a>	
<a href="#">statement</a>	The communication act of uttering or inscribing a sentence to declare a <a href="#">proposition</a>	
<a href="#">variable</a>	<a href="#">logical variable</a>	

### 10.3.3 Concepts that Inherit their Formal Logic Mapping from their Generalization

#### 10.3.3.1 Generalization Providing Formal Logic Mapping

SBVR Term	Generalization Providing Formal Logic Mapping	Comment
<a href="#">actuality</a>	<a href="#">state of affairs</a>	
<a href="#">aggregation formulation</a>	<a href="#">logical formulation</a>	
<a href="#">at-least-n quantification</a>	<a href="#">quantification</a>	
<a href="#">at-most-n quantification</a>	<a href="#">quantification</a>	
<a href="#">at-most-one quantification</a>	<a href="#">quantification</a>	
<a href="#">auxiliary variable</a>	<a href="#">variable</a>	
<a href="#">bag projection</a>	<a href="#">projection</a>	
<a href="#">binary fact type</a>	<a href="#">fact type</a>	
<a href="#">characteristic</a>	<a href="#">fact type</a>	
<a href="#">closed logical formulation</a>	<a href="#">logical formulation</a>	
<a href="#">closed projection</a>	<a href="#">projection</a>	
<a href="#">closed semantic formulation</a>	<a href="#">semantic formulation</a>	
<a href="#">concept type</a>	<a href="#">object type</a>	
<a href="#">condition 1</a>	<a href="#">logical operand</a>	
<a href="#">condition 2</a>	<a href="#">logical operand</a>	
<a href="#">conjunct 1</a>	<a href="#">logical operand</a>	
<a href="#">conjunct 2</a>	<a href="#">logical operand</a>	
<a href="#">conjunction</a>	<a href="#">logical formulation</a>	
<a href="#">disjunct 1</a>	<a href="#">logical operand</a>	
<a href="#">disjunct 2</a>	<a href="#">logical operand</a>	
<a href="#">disjunction / exclusive disjunction</a>	<a href="#">logical formulation</a>	
<a href="#">exactly-n quantification</a>	<a href="#">quantification</a>	
<a href="#">exactly-one quantification</a>	<a href="#">quantification</a>	

<a href="#">exclusive disjunct 1</a>	<a href="#">logical operand</a>	
<a href="#">exclusive disjunct 2</a>	<a href="#">logical operand</a>	
<a href="#">exclusive disjunction</a>	<a href="#">logical formulation</a>	
<a href="#">existential quantification</a>	<a href="#">quantification</a>	
<a href="#">fact type formulation</a>	<a href="#">logical formulation</a>	
<a href="#">individual concept</a>	<a href="#">object type</a>	
<a href="#">instantiation formulation</a>	<a href="#">logical formulation</a>	
<a href="#">logical formulation kind</a>	<a href="#">object type</a>	
<a href="#">logical operand</a>	<a href="#">logical formulation</a>	
<a href="#">logical operand 1</a>	<a href="#">logical operand</a>	
<a href="#">logical operand 2</a>	<a href="#">logical operand</a>	
<a href="#">logical operation</a>	<a href="#">logical formulation</a>	
<a href="#">nand formulation</a>	<a href="#">logical formulation</a>	
<a href="#">necessity claim</a>	<a href="#">modal formulation</a>	
<a href="#">negand</a>	<a href="#">logical operand</a>	
<a href="#">nonnegative integer</a>	<a href="#">integer</a>	
<a href="#">nor formulation</a>	<a href="#">logical formulation</a>	
<a href="#">noun concept formulation</a>	<a href="#">logical formulation</a>	
<a href="#">numeric range quantification</a>	<a href="#">quantification</a>	
<a href="#">objectification</a>	<a href="#">logical formulation</a>	
<a href="#">obligation claim</a>	<a href="#">modal formulation</a>	
<a href="#">permissibility claim</a>	<a href="#">modal formulation</a>	
<a href="#">positive integer</a>	<a href="#">integer</a>	
<a href="#">possibility claim</a>	<a href="#">modal formulation</a>	
<a href="#">projecting formulation</a>	<a href="#">logical formulation</a>	
<a href="#">projection</a>	<a href="#">semantic formulation</a>	
<a href="#">projection position</a>	<a href="#">positive integer</a>	



<a href="#">proposition nominalization</a>	<a href="#">logical formulation</a>	
<a href="#">set projection</a>	<a href="#">projection</a>	
<a href="#">universal quantification</a>	<a href="#">quantification</a>	
<a href="#">whether-or-not formulation</a>	<a href="#">logical formulation</a>	

### 10.3.4 Concepts whose Formal Logic Mapping is specified in their Specializations

SBVR Term	Specialization(s) Providing Formal Logic Mapping	Comment
<a href="#">bindable target</a>	<a href="#">variable</a> , <a href="#">text</a>	
<a href="#">modality</a>	<a href="#">alethic modality</a> , <a href="#">deontic modality</a>	
<a href="#">semantic formulation</a>	<a href="#">logical formulation</a> , <a href="#">projection</a>	
<a href="#">thing</a>	(Every other concept is a specialization of <i>thing</i> .)	The current definition of <i>thing</i> is: “anything perceivable or conceivable”.



# 11 Business Vocabulary

The following vocabulary provides words for describing business vocabularies along with the symbols and forms of expression they contain. A full description of a business vocabulary involves its relationship to semantic communities and speech communities, its relationship to other vocabularies, the concepts represented, their definitions and other information about them.

---

## Vocabulary for Describing Business Vocabularies

Language: [English](#)

Included Vocabulary: [Meaning and Representation Vocabulary](#)

---

# 11.1 Business Meaning

## 11.1.1 Communities, Meanings & Vocabularies

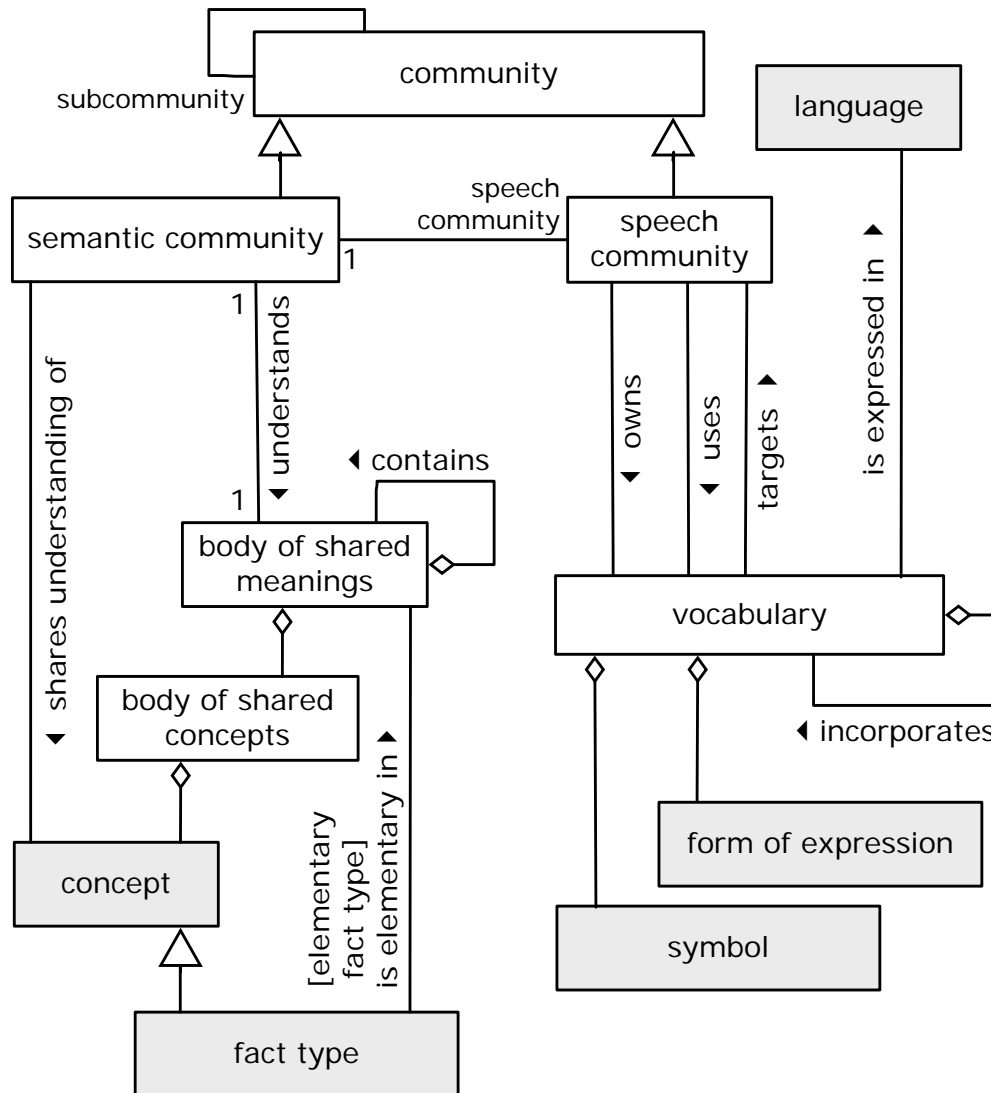


Figure 11.1

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

### 11.1.1.1 Communities

#### community

Definition: group of people having a particular unifying characteristic in common

Dictionary Basis: group of people having a religion, race, profession, or other particular characteristic in common [NODE 'community']

Example: The [Car Rental Community](#) -- people who work in the car rental business

Example: The [EU-Rent Community](#) -- all EU-Rent employees

Example: The [EU-Rent German Community](#) -- employees of EU-Rent's German division

### **semantic community**

Definition: [community](#) whose unifying characteristic is a shared understanding (perception) of the things that they have to deal with

Example: The [EU-Rent Community](#) -- those who share the body of concepts about general and specific things of importance to the EU-Rent business.

### **speech community**

Definition: [community](#) whose unifying characteristic is the [vocabulary](#) that it uses

Dictionary Basis: group of people sharing a characteristic vocabulary, and grammatical and pronunciation patterns for use in their normal intercommunication [W3ID 'speech community']

Example: The [EU-Rent German Community](#) shares the German-based vocabulary of symbols used in EU-Rent's business. The symbols include German words for EU-Rent's concepts plus symbols adopted from other languages.

### **speech community is of semantic community**

Synonymous Form: [semantic community](#) *has* [speech community](#)

Necessity: *Each* [speech community](#) *is of exactly one* [semantic community](#).

### **subcommunity**

Concept Type: [role](#)

Definition: [community](#) *that is* a distinct grouping within *another* [community](#)

Dictionary Basis: distinct grouping within a community [NODE 'sub-community']

### **subcommunity is of community**

Definition: *the* [subcommunity](#) *is a distinct grouping within* *the* [community](#)

Synonymous Form: [community](#) *has* [subcommunity](#)

## **11.1.1.2 Bodies of Shared Meanings**

### **body of shared meanings**

Definition: *set of* [concepts](#) and [guidance](#) for which there is a shared understanding in *a given* [semantic community](#)

Dictionary Basis: set containing all objects or elements and of which all other sets are subsets [NODE 'universal set']

Example: The [EU-Rent Car Rental Business](#) has a [body of shared meanings](#) which contains the set of concepts of general and specific things of importance to the EU-Rent car rental business

### **semantic community understands body of shared meanings**

Synonymous Form: [body of shared meanings](#) *is understood by* [semantic community](#)

Necessity: *Each* [semantic community](#) *understands exactly one* [body of shared meaning](#).

Necessity: Each body of shared meaning *is understood by exactly one* semantic community.

### body of shared meanings *includes* body of shared concepts

#### body of shared concepts

Definition: all of the concepts within a body of shared meanings

### body of shared concepts *includes* concept

Concept Type: partitive fact type

Synonymous Form: concept *is included in* body of shared concepts

### semantic community *shares understanding of* concept

Synonymous Form: concept *has shared understanding by* semantic community

### body of shared meanings<sub>1</sub> *contains* body of shared meanings<sub>2</sub>

Concept Type: partitive fact type

Definition: *the* body of shared meanings *includes* everything in *the other* body of shared meanings

#### elementary fact type

Definition: fact type whose facts cannot be split into smaller units of information that collectively provide the same information as the original

Concept Type: role

Example: branch *has* storage capacity

Example: service depot *is included in* local area

Example: rental car *has* fuel level *at* date/time

Example: Counter-example (this would *not* be considered an elementary fact type): car manufacturer *delivers* consignment *to* branch. This is not elementary because a consignment is always from at most one car manufacturer and is always to at most one branch. So the counter-example is equivalent to the combination of two binary fact types: car manufacturer *delivers* consignment and consignment *is delivered to* branch.

### fact type *is elementary in* body of shared meanings

Definition: within *the* body of shared meanings, *the* fact type cannot be decomposed into a set of two or more fact types that collectively have the same meaning as *the* fact type

Synonymous Form: body of shared meanings *has* elementary fact type

Necessity: Each elementary fact type *of a* body of shared meanings *is in the* body of shared meanings.

Necessity: A fact of an elementary fact type of a body of shared meanings is not equivalent to the conjunction of two or more Facts of other fact types in the body of shared meanings.

## 11.1.1.3 Vocabularies

### vocabulary

Definition: set of symbols and forms of expression primarily drawn from a single language to express concepts within a body of shared meanings

Dictionary Basis: sum or stock of words employed by a language, group, individual, or work, or in a field of knowledge [MWCD 'vocabulary']

Reference Scheme: a [URI of the vocabulary](#)

Example: The sets of symbols represented in EU-Rent's internal glossaries, in the natural languages in which the company does business, together with the vocabularies it has adopted, including those defined in:

- \* Industry standard glossaries for car rental business,
- \* Standard (e.g., ISO) glossaries of business terms,
- \* Authoritative dictionaries for the relevant natural languages.

### **speech community owns vocabulary**

Definition: the [speech community](#) creates and evolves the [vocabulary](#)

Synonymous Form: [vocabulary is owned by speech community](#)

Note: The speech community that owns a vocabulary has the authority to change the content of the vocabulary.

### **speech community uses vocabulary**

Synonymous Form: [vocabulary is used by speech community](#)

Note: A speech community may use a vocabulary that is owned by a different speech community.

### **vocabulary targets speech community**

Synonymous Form: [speech community is targeted by vocabulary](#)

### **vocabulary is expressed in language**

Definition: the [symbols](#) of the [vocabulary](#) are primarily within the [language](#)

Synonymous Form: [language expresses vocabulary](#)

Synonymous Form: [vocabulary uses language](#)

Note: Typically, the language would be a natural language, but not necessarily. See '[language](#)'.

### **vocabulary includes symbol**

Concept Type: [partitive fact type](#)

Synonymous Form: [symbol is included in vocabulary](#)

### **vocabulary includes form of expression**

Concept Type: [partitive fact type](#)

Synonymous Form: [form of expression is included in vocabulary](#)

### **vocabulary<sub>1</sub> incorporates vocabulary<sub>2</sub>**

Concept Type: [partitive fact type](#)

Definition: the [vocabulary<sub>1</sub>](#) includes each [symbol](#) that is included in the [vocabulary<sub>2</sub>](#)

Note: When more than one vocabulary is included, a hierarchy of inclusion can provide priority for selection of definitions.

Synonymous Form: [vocabulary<sub>2</sub> is incorporated into vocabulary<sub>1</sub>](#)

## 11.1.2 Concepts & Characteristics

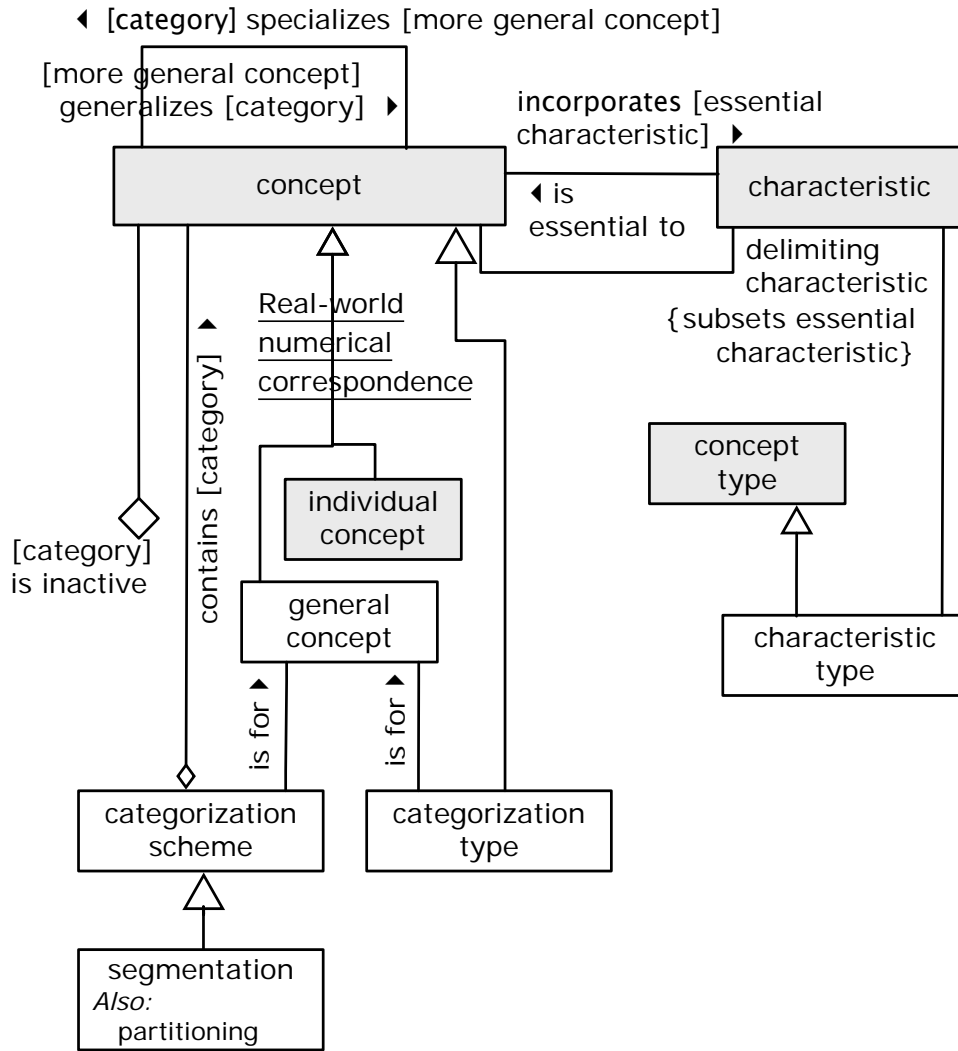


Figure 11.2

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

### 11.1.2.1 Kinds of Concept

#### Real-world Numerical Correspondence

Definition: the categorization scheme of the concept 'concept' that classifies a concept based on whether or not the concept always corresponds to one specific real-world individual

Necessity: The concept 'individual concept' is included in Real-world Numerical Correspondence.



### general concept

Source:	based on <a href="#">ISO 1087-1 (English)</a> (3.2.3) ['general concept']
Definition:	<a href="#">concept</a> that conceivably corresponds to a number of <a href="#">things</a> by reason of their having something in common
Note:	A definition is given along with the source in order to state the definition in terms of SBVR.
Note:	<a href="#">ISO 1087-1 (English)</a> partitions ' <a href="#">concept</a> ' into ' <a href="#">individual concept</a> ' (necessarily corresponding to one thing at a time) and ' <a href="#">general concept</a> ' (possibly corresponding to two or more things at a time).
Necessity:	No <a href="#">general concept</a> is an <a href="#">individual concept</a> .
Necessity:	The <a href="#">concept</a> ' <a href="#">general concept</a> ' is included in <a href="#">Real-world Numerical Correspondence</a> .
Example:	The concept 'rental car' corresponding to cars that are rented
Example:	The concept 'rental customer' corresponding to EU-Rent customers that rent cars

### 11.1.2.2 Kinds of Characteristic

#### essential characteristic

Source:	<a href="#">ISO 1087-1 (English)</a> (3.2.6) ['essential characteristic']
Definition:	<a href="#">characteristic</a> which is indispensable to understanding a <a href="#">concept</a>
Concept Type:	<a href="#">role</a>

#### characteristic is essential to concept

Definition:	the <a href="#">concept</a> <i>incorporates</i> the <a href="#">characteristic</a> and the <a href="#">characteristic</a> is essential to understanding the <a href="#">concept</a>
Synonymous Form:	<a href="#">concept</a> <i>incorporates</i> <a href="#">essential characteristic</a>
Concept Type:	<a href="#">is-property-of fact type</a>

#### delimiting characteristic

Source:	<a href="#">ISO 1087-1 (English)</a> (3.2.7) ['delimiting characteristic']
Definition:	<a href="#">essential characteristic</a> used for distinguishing a <a href="#">concept</a> from related <a href="#">concepts</a>
Concept Type:	<a href="#">role</a>
Note:	Delimiting characteristics of a concept are inherited as essential characteristics by all categories of that concept.

#### concept has delimiting characteristic

Definition:	the <a href="#">characteristic</a> <i>is essential to</i> the <a href="#">concept</a> and the <a href="#">characteristic</a> serves to distinguish the <a href="#">concept</a> from others
Concept Type:	<a href="#">is-property-of fact type</a>

#### characteristic type

Source:	<a href="#">ISO 1087-1 (English)</a> (3.2.5) ['type of characteristics']
Definition:	category of [the concept] ' <a href="#">characteristic</a> ' which serves as a criterion of subdivision when establishing concept systems
General Concept:	<a href="#">concept type</a>
Necessity:	Each <a href="#">instance</a> of each <a href="#">characteristic type</a> is a <a href="#">characteristic</a> .

Example: The extension of the [characteristic type](#) 'color' includes the characteristics '[thing is blue](#)', '[thing is red](#)', '[thing is green](#)', etc.

### 11.1.2.3 Categorization Schemes

#### [category](#)

Source: [ISO 1087-1 \(English\)](#) (3.2.16) ['specific concept']  
Definition: [concept](#) in a generic relation having the broader intension  
Concept Type: [role](#)  
Dictionary Basis: secondary or subordinate category [NODE 'subcategory']

#### [more general concept](#)

Source: [ISO 1087-1 \(English\)](#) (3.2.15) ['generic concept']  
Definition: [concept](#) in a generic relation having the narrower intension  
Concept Type: [role](#)

#### [categorization scheme](#)

Definition: scheme for partitioning [things](#) in [the extension of a given general concept](#) into [the extensions of categories of that general concept](#)  
Example: The [general concept](#) 'person' categorized by age range and gender into categories '[boy](#)', '[girl](#)', '[man](#)', '[woman](#)'.  
Dictionary Basis: an orderly combination of related parts [AH (3) 'scheme']

#### [categorization scheme is for general concept](#)

Definition: [the general concept](#) is divided into [category\(s\)](#) by [the categorization scheme](#)  
Necessity: [Each categorization scheme is for at least one general concept.](#)  
Synonymous Form: [general concept has categorization scheme](#)

#### [categorization scheme contains category](#)

Definition: [the category](#) is included in [the categorization scheme](#) as one of the categories divided into by the scheme  
Synonymous Form: [category is included in categorization scheme](#)  
Concept Type: [partitive fact type](#)  
Necessity: [Each category that is included in a categorization scheme of a general concept is a category of that general concept.](#)

#### [segmentation](#)

Definition: [categorization scheme](#) whose contained [categories](#) are complete (total) and disjoint with respect to the [general concept](#) that has the [categorization scheme](#)  
Synonym: [partitioning](#)

#### [partitioning](#)

See: [segmentation](#)

### **categorization type**

- Definition: [concept](#) whose [instances](#) are, or are in one-to-one correspondence with, meaningful-to-the-business [categories](#) of [another concept](#)
- Note: A [categorization type](#) is either partial or complete. It is complete if it necessarily categorizes everything of the general concept that it is for.
- Example: EU-Rent's categorization type for EU-Rent's concept of 'branch' whose instances are categories of branch: 'airport branch', 'agency' and 'city branch'.

### **categorization type is for general concept**

- Synonymous Form: [general concept has categorization type](#)

### **category is inactive**

- Necessity: A [category is inactive](#) if and only if the [category is a concept](#) that plays [no role](#) in a [fact type](#).

### 11.1.3 Kinds of Definition

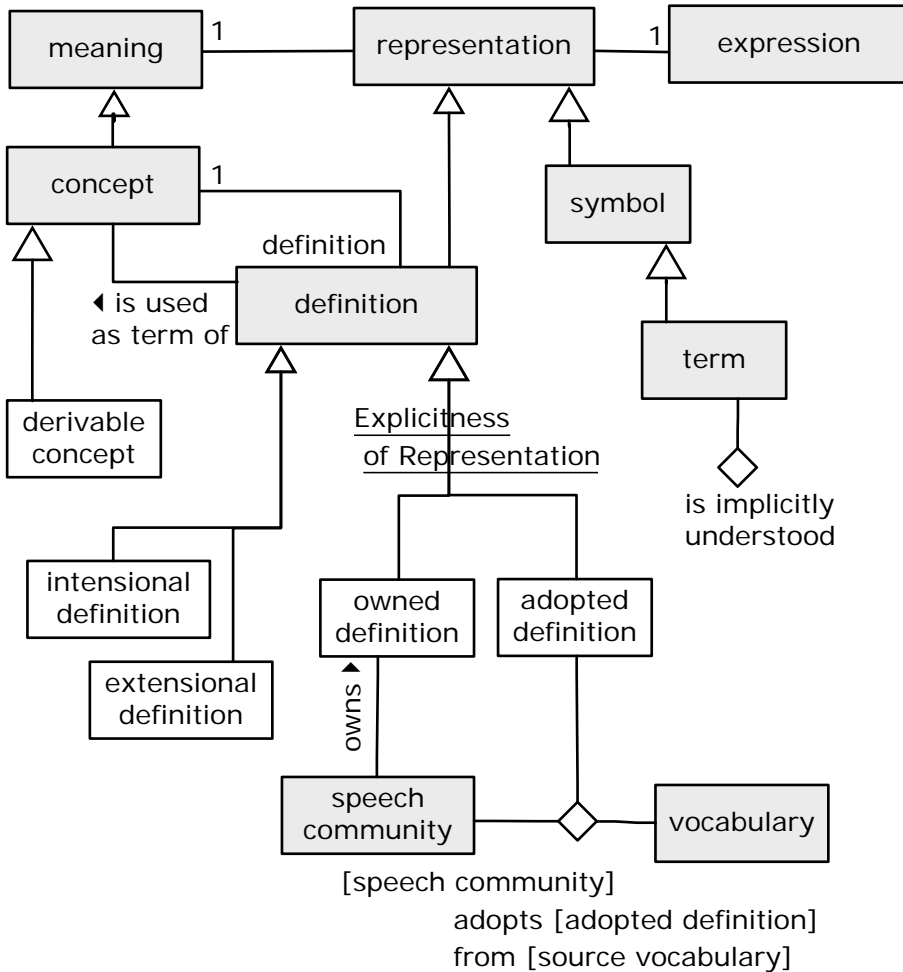


Figure 11.3

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

#### intensional definition

- Source: [ISO 1087-1 \(English\)](#) (3.3.2) ['intensional definition']
- Definition: [definition](#) which describes the intension of a concept by stating the superordinate concept and the delimiting characteristics
- General Concept: [definition](#)
- Necessity: No [intensional definition](#) is an [extensional definition](#).

### extensional definition

Source:	<a href="#">ISO 1087-1 (English)</a> (3.3.3) [ <a href="#">‘extensional definition’</a> ]
Definition:	description of a concept by enumerating all of its subordinate concepts under one criterion of subdivision
General Concept:	<a href="#">definition</a>
Necessity:	No <a href="#">extensional definition</a> is an <a href="#">intensional definition</a> .

### Explicitness of Representation

Definition:	<a href="#">the categorization scheme of the concept ‘definition’ that classifies a definition</a> based on whether it is owned by its <a href="#">speech community</a> or adopted by its <a href="#">speech community</a>
-------------	--

### owned definition

Definition:	<a href="#">definition that a speech community ‘owns’ and</a> is responsible for creating and maintaining
Necessity:	The <a href="#">concept ‘owned definition’ is included in</a> <a href="#">Explicitness of Representation</a> .
Example:	EU-Rent ‘owns’ its definition of the concept of ‘barred driver.’

### speech community owns owned definition

### adopted definition

Definition:	<a href="#">definition that a speech community adopts from a source vocabulary that the speech community does not own</a>
Necessity:	The <a href="#">concept ‘adopted definition’ is included in</a> <a href="#">Explicitness of Representation</a> .
Necessity:	Each <a href="#">adopted definition</a> must be for a <a href="#">concept</a> in <a href="#">the body of shared meanings of the semantic community of the speech community</a> .
Necessity:	The <a href="#">speech community that adopts an adopted definition</a> does not own the <a href="#">source vocabulary of the adopted definition</a> .
Reference Scheme:	<a href="#">source vocabulary</a>
Example:	EU-Rent adopts definition 2b of ‘law’ from Merriam-Webster Unabridged, using the terms ‘law’ (primary) and ‘statute’ for the concept.
Note:	The primary term used for the concept does not have to be the same as the primary term in the source. For example, EU-Rent might have taken the definition of ‘law’ from MWU, but used ‘statute’ as the primary term for the concept.

### source vocabulary

Concept Type:	<a href="#">role</a>
Definition:	<a href="#">vocabulary that is the source of an adopted definition</a>
Note:	The speech community that owns a source vocabulary and any speech community that adopts definitions from the source vocabulary are usually in different semantic communities.

### speech community adopts adopted definition from source vocabulary

### definition is used as term of concept

Definition:	<a href="#">the definition</a> of <a href="#">the concept</a> that has no term defined is used as <a href="#">the term</a> for <a href="#">that concept</a>
Note:	In the case of a concept for which no term is given, the concept is represented by its definition

**term is implicitly understood**

Definition: **the term** is generally understood within by its owning community without an explicit definition

**derivable concept**

Definition: concept whose extension can be determined from its definition or from rules

### 11.1.4 Conceptualization Decisions

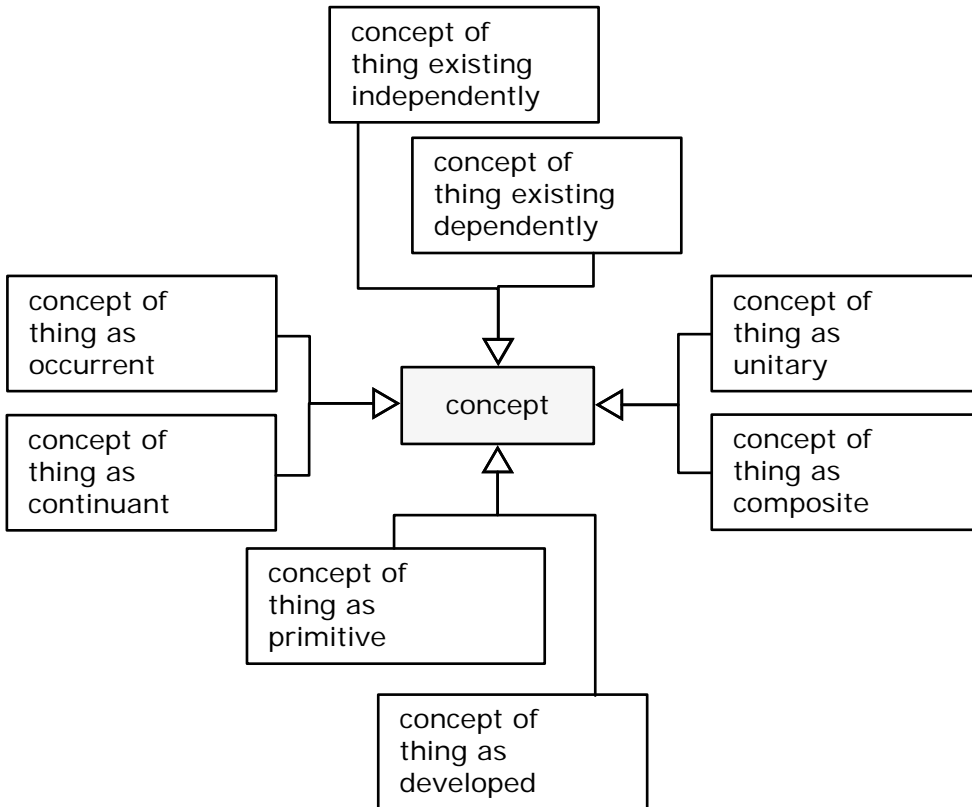


Figure 11.4

---

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

---

#### concept of thing as unitary

- Definition: **concept that** conceptualizes its **instances** as **not** being made up of discrete parts or elements
- Note: A thing is conceptualized as unitary if a semantic community doesn't think of it as having components, even though some other community may be aware of and concerned about its decomposition.
- Example: EU-Rent finance department treats a car as unitary, while its maintenance staff treat it as composite.

#### concept of thing as composite

- Definition: **concept that** conceptualizes its **instances** as being made of discrete parts or elements that have corresponding **concepts** in their own right
- Necessity: **No concept of thing as unitary is a concept of thing as composite.**

### concept of thing as primitive

Definition: concept that conceptualizes its instances as **not** being developed or derived from anything else

Dictionary Basis: not developed or derived from anything else [NODE 'primitive']

### concept of thing as developed

Definition: concept that conceptualizes its instances as being developed or derived from something else

Necessity: **No** concept of thing as primitive *is a* concept of thing as developed.

### concept of thing as occurrent

Definition: concept that conceptualizes its instances as existing only at a point in time

Dictionary Basis: the fact of something existing or being found in a place or under a particular set of conditions [NODE 'occurrence' 2] + the fact or frequency of something happening [NODE 'occurrence' 1]

### concept of thing as continuant

Definition: concept that conceptualizes its instances as existing over a period of time

Dictionary Basis: a thing that retains its identity even though its states and relations may change. [NODE 'continuant' 2]

Necessity: **No** concept of thing as occurrent *is a* concept of thing as continuant.

### concept of thing existing independently

Definition: concept that conceptualizes **each** instance to exist independently of other things such that existence cannot be ended by the ending of the existence of any other thing

### concept of thing existing dependently

Definition: concept that conceptualizes **each** instance as existing only as long as one or more other things continue to exist

Necessity: **No** concept of thing existing independently *is a* concept of thing existing dependently.



## 11.1.5 Fact Type Templating

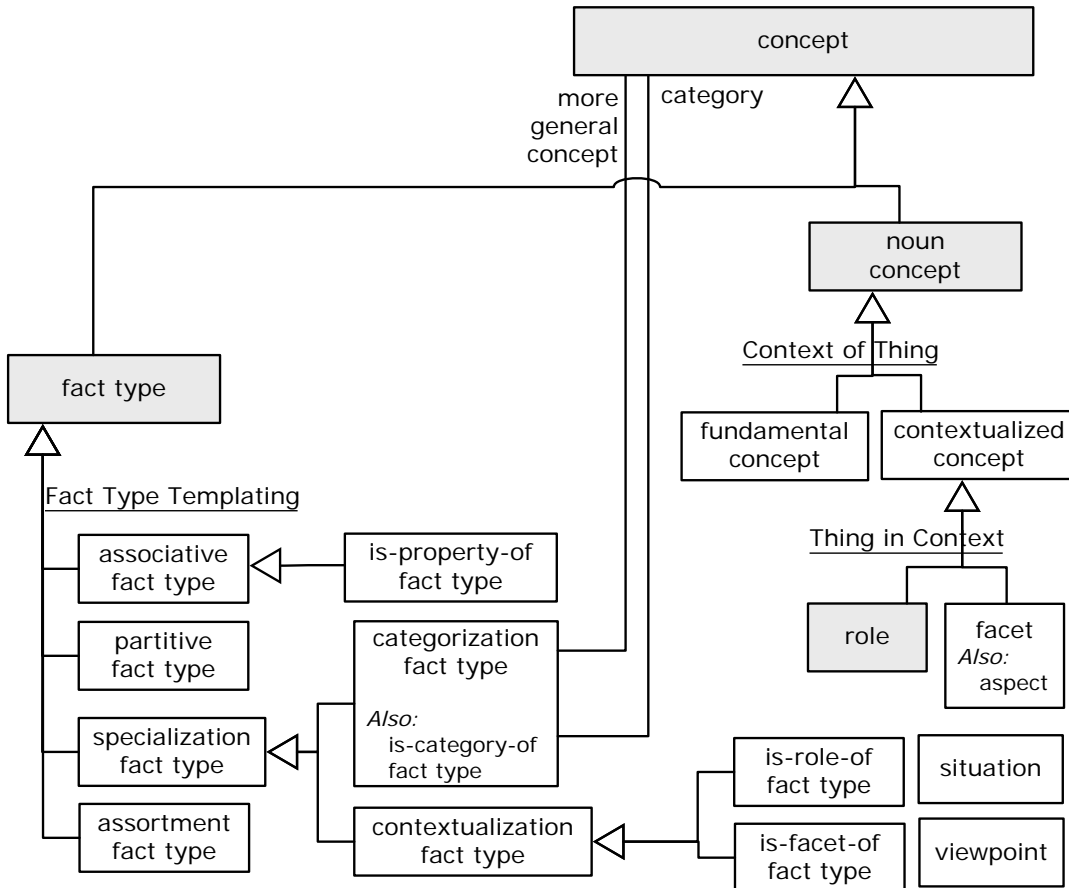


Figure 11.5

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

### 11.1.5.1 Kinds of Fact Type

#### Fact Type Templating

Definition: the categorization scheme of the concept 'fact type' that classifies a fact type based on the semantic nature of the fact type

#### associative fact type

Definition: fact type that has more than one role and that has a nonhierarchical subject-oriented connection drawn from experience, based on practical rather than theoretical considerations

Source: based on [ISO 1087-1 \(English\)](#) (3.2.23) ['associative relation', 'pragmatic relation']

Necessity: The concept 'associative fact type' is included in [Fact Type Templating](#).

Example: The fact type 'additional driver is authorized in rental'.

Example: The fact type 'car manufacturer supplies car model'.  
Example: The fact type 'car manufacturer delivers consignment to branch'.

### is-property-of fact type

Definition: associative fact type that is defined with respect to a first given concept and a second given concept such that each instance of the fact type is an actuality that an instance of the first concept constitutes an essential quality of an instance of the second concept

Dictionary Basis: an essential quality [SOED 'property']

Dictionary Basis: so important as to be indispensable. Something so important to a thing's very nature that without it that thing would not be the same thing. Something that cannot be removed without destroying the thing itself or its distinguishing character [MWDS & NODE 'essential']

Dictionary Basis: an intelligible mark or indication by means of which a thing may be identified or its constitution understood [MWDS 'quality']

Necessity: Each instance of an is-property-of fact type is an actuality that a thing has a particular property.

Necessity: The concept 'is-property-of fact type' is included in Fact Type Templating.

Example: The fact type 'engine size is property of car model'.

### partitive fact type

Definition: fact type that has two roles and where each instance is an actuality that a given part is in the composition of a given whole

Source: based on ISO 1087-1 (English) (3.2.22) ['partitive relation']

Necessity: Each instance of a partitive fact type is an actuality that a part is in the composition of a whole.

Necessity: The concept 'partitive fact type' is included in Fact Type Templating.

Example: The fact type 'country is included in region'. An example of an instance of that fact type is that Sweden is included in Scandinavia.

Example: The fact type 'branch is included in local area'.

Example: The fact type 'car model is included in car group'.

### specialization fact type

Definition: categorization fact type or contextualization fact type

General Concept: fact type

Note: The essential property is that, for these kinds of fact types, an instance of a more specific concept is one and the same thing as an instance of a more general concept.

Necessity: The concept 'specialization fact type' is included in Fact Type Templating.

### assortment fact type

Definition: fact type that is defined with respect to a given general concept and a given individual concept such that each instance of the fact type is an actuality that the one instance of the individual concept is an instance of the general concept

Dictionary Basis: to place in the same group with others : associate in a class [MWU (3) "assort"]

Necessity: Each instance of an assortment fact type is an actuality that the instance of a given individual concept is an instance of a given general concept.

Necessity: The concept 'assortment fact type' is included in Fact Type Templating.

- Example: A fact type with the form of expression ‘Euro is a currency’. The one instance of the fact type would be Euro being a currency.
- Example: A fact type with the form of expression ‘Ford Motor Company is a car manufacturer’. The one instance of the fact type would be the Ford Motor Company being a car manufacturer.
- Example: A fact type with the form of expression ‘Switzerland is a country’. The one instance of the fact type would be Switzerland being a country.

### 11.1.5.2 Contextualization

#### aspect

See: [facet](#)

#### facet

- Definition: [concept](#) that incorporates only those [characteristics of another concept](#) being contextualized which are relevant to [a given viewpoint](#)
- General Concept: [contextualized concept](#)
- Dictionary Basis: a particular way in which some thing may be considered; its particular nature, appearance, or quality; the particular part or feature of it [NODE ‘aspect’]
- Necessity: [The concept ‘facet’ is included in Thing in Context.](#)

#### situation

- Definition: set of circumstances that provides the context from which [roles](#) played by [instances of a concept](#) may be understood or assessed
- Dictionary Basis: a set of circumstances in which one finds oneself; a state of affairs [NODE ‘situation’]
- Dictionary Basis: the circumstances that form the setting for an event, statement, or idea, and in terms of which it can be fully understood or assessed [NODE ‘context’]
- Note: A situation typically pertains for some period of time, during which changes may occur.
- Example: The situation "breakdown during rental" is the set of circumstances that starts with the breakdown of a car while on rental and continues until the broken-down car, having been replaced by another car, has been returned to a EU-Rent location.

#### viewpoint

Definition: perspective from which something is considered

#### categorization fact type

- Definition: [fact type](#) that is defined with respect to [a given concept](#) and another [concept that is a category of that concept](#) such that [each instance of the fact type is an actuality that a particular instance of the concept is also an instance of the category](#)
- Synonym: [is-category-of fact type](#)
- General Concept: [specialization fact type](#)
- Necessity: [No categorization fact type is a contextualization fact type.](#)
- Example: A fact type with the form of expression “‘customer is of the category ‘high-end customer’”. An instance of the fact type would be a particular customer being a high-end customer.
- Example: A fact type with the form of expression “‘customer is of the category ‘risky customer’”. An instance of the fact type would be a particular customer being a risky customer.

## contextualization fact type

- Definition: [is-role-of fact type](#) or [is-facet-of fact type](#)  
General Concept: [specialization fact type](#)  
Necessity: The [concept 'contextualization fact type'](#) is included in [Fact Type Templating](#).

## is-role-of fact type

- Definition: [fact type](#) that is defined with respect to a given [concept](#), a given [role](#) and a given [category of the concept 'situation'](#) such that [each instance of the fact type is an actuality that a particular instance of the concept plays the role](#) in a particular [instance of the category of 'situation'](#)
- Necessity: Each [instance of an is-role-of fact type](#) is an [actuality that a particular thing plays a role](#) in a particular [situation](#).
- Necessity: Each [is-role-of fact type](#) is defined with respect to a [concept](#), a [role](#) and a [category of the concept 'situation'](#).
- Necessity: The [concept 'is-role-of fact type'](#) is included in [Fact Type Templating](#).
- Example: A fact type with the form of expression “[rental car](#) plays the role '[replacement car](#)' in the fact type '[breakdown during rental](#) has [replacement car](#)’”. An instance of the fact type would be a particular breakdown during a particular rental having a particular replacement car. Note that a separate fact type relates a breakdown during rental to a rental.
- Example: A fact type with the form of expression “[branch](#) plays the role '[pick-up branch](#)' in the fact type '[rental has pick-up branch](#)’”. An instance of the fact type would be a particular rental having a particular pick-up branch.
- Note: A fact type is understood to be an is-role-of fact type based on a pattern of meaning, not on a form of expression. There is no requirement of any particular wording of the form of expression of an is-role-of fact type.

## is-facet-of fact type

- Definition: [fact type](#) that is defined with respect to a given [concept](#), a given [facet](#) and a given [category of the concept 'viewpoint'](#) such that [each instance of the fact type is an actuality that a particular instance of the category of the concept 'viewpoint' gives consideration to the facet of a particular instance of the concept](#)
- General Concept: [contextualization fact type](#)
- Necessity: Each [instance of an is-facet-of fact type](#) is an [actuality that a particular viewpoint gives consideration to a facet of particular thing](#).
- Necessity: Each [is-facet-of fact type](#) is defined with respect to a [concept](#), a [facet](#) and a [category of the concept 'viewpoint'](#).
- Necessity: The [concept 'is-facet-of fact type'](#) is included in [Fact Type Templating](#).
- Example: A fact type with the form of expression “[financial accounting](#) considers [rental car](#) as asset’. An instance of the fact type would be a particular financial accounting considering a particular rental car to be an asset.
- Example: A fact type with the form of expression '[rental](#) considers [person](#) as driver’. An instance of the fact type would be a particular rental considering a particular person to be a driver.
- Note: A fact type is understood to be an is-facet-of fact type based on a pattern of meaning, not on a form of expression.
- Note: A given community may choose to include only one facet.

### 11.1.5.3 Contextualized Concepts

#### Context of Thing

Definition: the segmentation of the concept 'noun concept' that *classifies a noun concept based on* whether the noun concept's real-world individuals are perceived by the semantic community as in their uninvolved essence or as to their involvement in a situation or from a viewpoint

#### fundamental concept

Definition: noun concept whose real-world individuals are perceived by a given semantic community as being in their essence, apart from any situation in which they are involved or viewpoint from which they are considered

Dictionary Basis: a property or group of properties of something without which it would not exist or be what it is [NODE 'essence']

Necessity: No fundamental concept is a contextualized concept.

Necessity: The concept 'fundamental concept' is included in Context of Thing.

Example: car (as contrasted with 'rental car')

Example: person (as contrasted with 'customer')

Note: Each semantic community decides what is within its body of shared meanings. A concept that is considered as fundamental by one community may, to another community, be a role or facet or category of a more broadly-defined concept.

#### contextualized concept

Definition: role or facet

General Concept: noun concept

Necessity: The concept 'contextualized concept' is included in Context of Thing.

#### Thing in Context

Definition: the segmentation of the concept 'contextualized concept' that *classifies a contextualized concept based on* whether the contextualization reflects a situation (i.e., a role) or a viewpoint (i.e., a facet)

Necessity: The concept 'role' is included in Thing in Context.

## 11.2 Business Representation

### 11.2.1 Symbolization

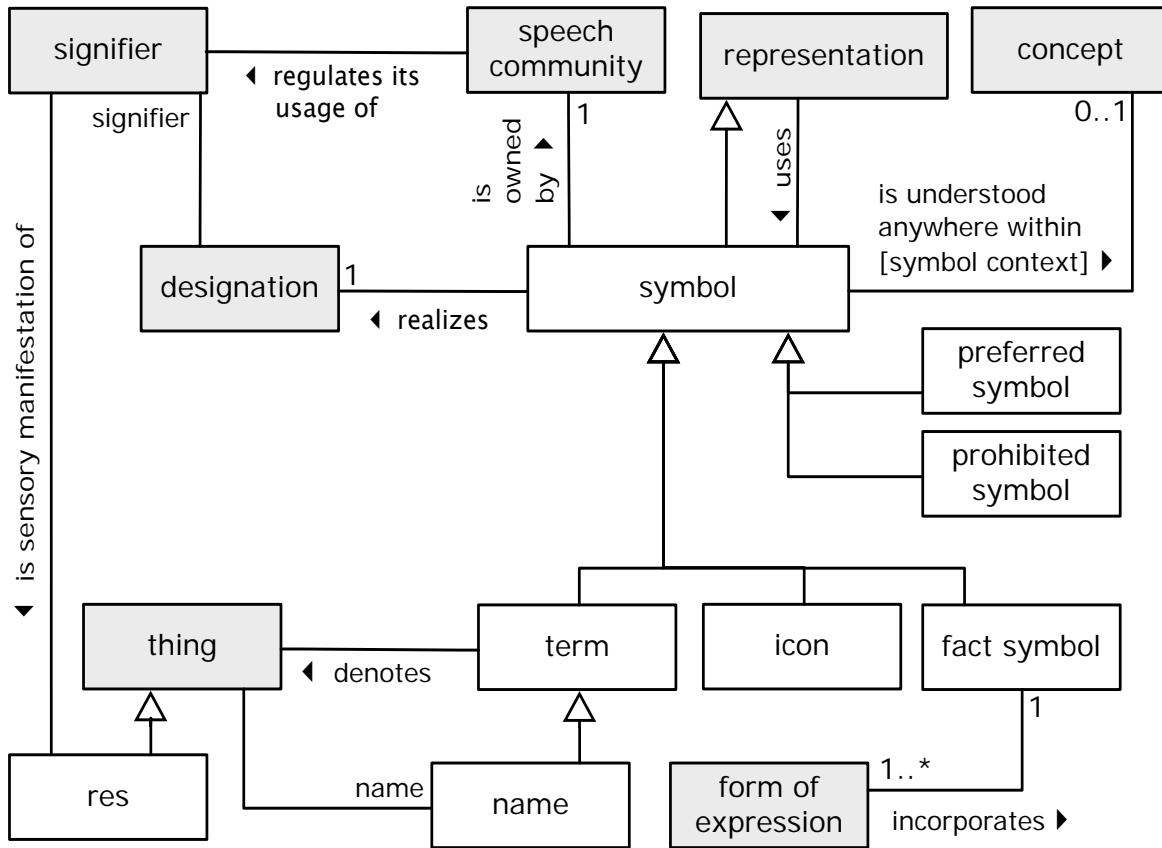


Figure 11.6

#### 11.2.1.1 Symbols

##### symbol

Definition: [representation of a concept](#) by [a signifier](#) as [owned by a speech community](#) and used within [a symbol context](#) which means [the concept](#) and denotes its [extension](#)

Dictionary Basis: thing conventionally regarded as representing or recalling something else (by possessing analogous qualities or by association in fact or thought, especially a material object representing something abstract) [SOED 'symbol']

Necessity: Each [symbol](#) [realizes exactly one designation](#).

Necessity: Each [symbol](#) [is owned by exactly one speech community](#).

Necessity: Each [symbol](#) [is understood anywhere within at most one symbol context](#).

Note:	The ' <u>symbol context</u> ' component of a <u>symbol</u> is only required for a <u>symbol</u> when needed for disambiguation of <u>signifier</u> -- i.e., in the case where a <u>speech community</u> uses one <u>signifier</u> to represent two (or more) distinct <u>concepts</u> .
Necessity:	Given a <u>speech community</u> , <u>signifier</u> and <u>symbol context</u> , at most one <u>symbol</u> is owned by the <u>speech community</u> and is understood anywhere within <u>symbol context</u> and realizes a <u>designation</u> that has the <u>signifier</u> .
Reference Scheme:	the <u>speech community</u> that owns the <u>symbol</u> and the <u>designation</u> that realizes the <u>symbol</u> and a <u>symbol context</u> that qualifies understanding of the <u>symbol</u>
Reference Scheme:	the <u>speech community</u> that owns the <u>symbol</u> and the <u>designation</u> that realizes the <u>symbol</u> and the <u>set of each symbol context</u> that qualifies understanding of the <u>symbol</u>

### symbol realizes designation

#### symbol is owned by speech community

Synonymous Form: speech community owns symbol

#### symbol is understood anywhere within symbol context

Synonymous Form: symbol context is of symbol

Synonymous Form: symbol context qualifies understanding of symbol

Note: The sense of 'anywhere' means that the symbol is understood inside of, but not outside of, the symbol context. In other words, this specifies the one symbol context that is the largest (broadest) within which the symbol is understood.

### symbol context

Concept Type: role

Definition: concept that represents the scope within which a signifier has a unique meaning for a given semantic community

Necessity: The symbol for a symbol context provides a disambiguating context where that symbol, via the concept that it symbolizes, defines a unique context within which the signifier of a second symbol is uniquely connected to its concept.

Example: When EU-Rent uses the term 'site':

- \* within the context of the concept termed 'vehicle rental' (another EU-Rent term), it denotes EU-Rent's shared understanding of 'a place from which EU-Rent vehicles are picked up and returned.'
- \* within the context of the concept termed 'vehicle maintenance' (another EU-Rent term), it denotes EU-Rent's shared understanding of 'a place where EU-Rent's vehicle fleet is serviced and repaired.'

Example: When EU-Rent uses the term 'customer':

- \* within the context of the concept termed 'vehicle rental' (another EU-Rent term), it denotes EU-Rent's shared understanding of 'rental-customer-ness' (Definition: 'an individual who currently has a EU-Rent car on rental, or has a reservation for a future car rental, or has rented a car from EU-Rent in the past 5 years').
- \* within the context of the concept termed 'vehicle sales' (another EU-Rent term), it denotes EU-Rent's shared understanding of 'car-purchaser-ness' (Definition: 'an individual who has purchased at least one car from EU-Rent that is still within its warranty period').

## representation uses symbol

Synonymous Form: [symbol is used in representation](#)

Example: EU-Rent: late rental return of a car damaged by an additional driver -- i.e., a driver, named on the rental contract, who is not the renter. Used symbols include 'late rental return', 'car', 'damaged' and 'additional driver'.

### 11.2.1.2 Kinds of Symbol

#### term

Definition: [symbol that is for a concept and](#) that is a word or phrase

Source: based on [ISO 1087-1 \(English\)](#) (3.4.3 & 3.4.2) ['term' or 'appellation']

Note: A [term](#) is typically a common noun or noun phrase, unless it is a [name](#).

Definition: [symbol that](#) has a precisely-limited meaning in some uses, or is peculiar to a science, art, profession, trade, or special subject

Dictionary Basis: word or expression that has a precise meaning in some uses or is peculiar to a science, art, profession, or subject [[MWCD 'term'](#)]

Example: EU-Rent agrees the word 'car' denotes its shared understanding of 'rental-car-ness' within <rental context>.

Example: EU-Rent agrees the word 'vehicle' denotes its shared understanding of 'car-ness' within <rental context>.

Example: EU-Rent agrees the word 'customer' denotes its shared understanding of 'rental-customer-ness' within <rental context>.

Example: EU-Rent agrees the word 'customer' denotes its shared understanding of 'car-purchaser-ness' within <car-sales context> -- i.e., when EU-Rent disposes of cars after they reach their mileage or age threshold.

Example: EU-Rent agrees the word 'renter' denotes its shared understanding of 'rental-customer-ness.' (within any context).

#### name

Definition: [term that is for an individual concept](#)

Source: based on [ISO 1087-1 \(English\)](#) (3.4.2) ['appellation']

Note: A [name](#) is often a proper noun.

#### icon

Definition: [symbol](#) whose signifier is a picture

Dictionary Basis: a usu. pictorial representation [[MWCD 'icon'](#)]

Example:  as a symbol for a 'u-turn'

Necessity: No [icon](#) is a [term](#).

Necessity: No [icon](#) is a [fact symbol](#).



### **fact symbol**

- Definition: symbol that is for a fact type and that is understood in an ordered context indicated by a form of expression
- Necessity: No fact symbol is a term.
- Reference Scheme: a form of expression that incorporates the fact symbol
- Example: In the expression, “Each customer rents a car”, ‘rents’ is a fact symbol denoting a fact type.
- Example: In the expression, “A driver of a car returns the car to a branch office”, ‘of’ is a fact symbol for one fact type (relating a driver to a car) and ‘returns to’ is another fact symbol denoting a fact type (relating a driver to a car and a branch office).

### **form of expression incorporates fact symbol**

- Definition: the form of expression lays out a pattern for using the fact symbol in an expression
- Synonymous Form: fact symbol is incorporated into form of expression
- Necessity: Each form of expression incorporates at most one fact symbol.
- Necessity: Each fact symbol is incorporated into at least one form of expression.

## **11.2.1.3 Symbols and Things in the Real-world**

### **term denotes thing**

- Definition: the thing is an instance of the concept that is represented by the term
- Synonymous Form: thing is denoted by term

### **thing has name**

- Definition: the thing is the instance of the individual concept that is represented by the name
- Synonymous Form: name references thing

### **res**

- Definition: thing that is literally anything that exists but is not the result of conceptualization

### **res is sensory manifestation of signifier**

## **11.2.1.4 Symbol Preference and Prohibition**

### **preferred symbol**

- Definition: symbol that is selected by its owning speech community for a given concept from among alternative symbols for that concept as being most desirable or productive
- Example: EU-Rent’s preferred terms for indicating the USA Dollar, Canadian Dollar and Mexican Peso are, respectively, “USD”, “CAD” and “MXN” (ISO 4217 currency codes).

### **prohibited symbol**

- Definition: symbol that is declared unacceptable by its owning speech community creating a vocabulary that excludes the symbol
- Example: In EU-Rent, use of the dollar sign (\$) by itself is prohibited, to avoid confusion between the USA Dollar, Canadian Dollar and Mexican Peso.

Note: What is prohibited is the use of a given signifier to represent a given meaning. The same signifier may be permitted, even preferred, to represent another meaning.

Necessity: **No preferred symbol is a prohibited symbol.**

**speech community regulates its usage of signifier**

**11.2.2 Forms of Business Representation**

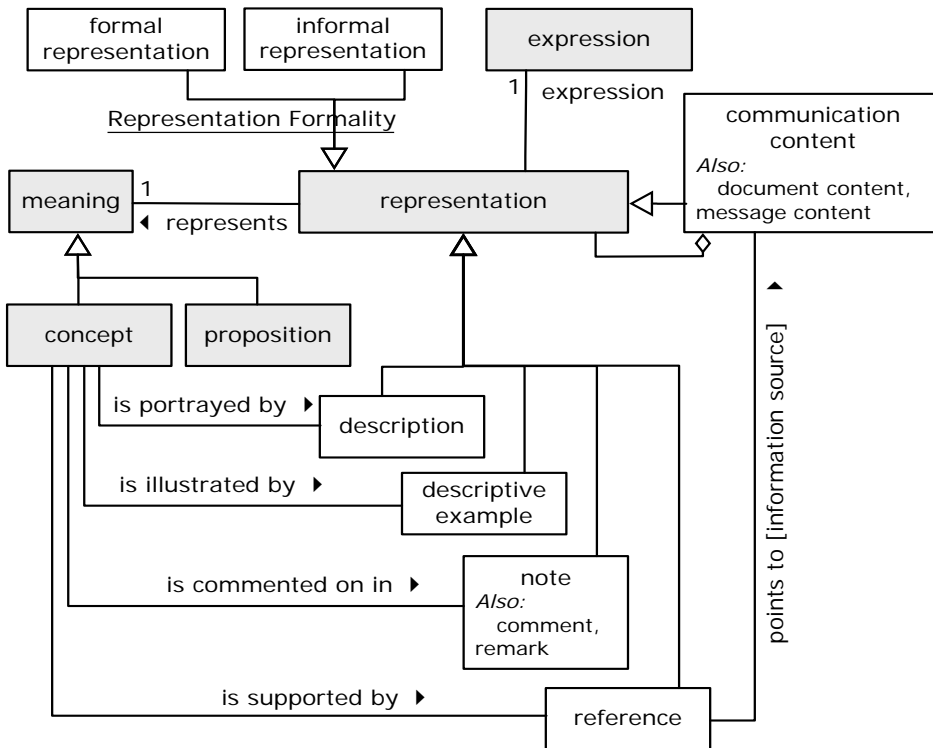


Figure 11.7

**11.2.2.1 Representation Formality**

Representation Formality

Definition: the segmentation of the concept 'representation' that classifies a representation based on whether or not it is 'formal'

informal representation

Definition: representation in which not every word is annotated ('tagged') in accordance with a notation that can be mapped to SBVR

Necessity: **No informal representation is a formal representation.**

Necessity: **The concept 'informal representation' is included in Representation Formality.**

Note: Some of the words of an informal representation may be annotated -- i.e., defined, or 'tagged', terms, names, verbs, or keywords.

### formal representation

- Definition: [representation](#) in which every word is annotated ('tagged') in accordance with a notation that can be mapped to SBVR
- Necessity: [No formal representation is an informal representation.](#)
- Necessity: [The concept 'formal representation' is included in Representation Formality.](#)

### 11.2.2.2 Concept Expression

#### concept is portrayed by description

- Synonymous Form: [description portrays concept](#)
- Note: The meaning of a description that portrays a concept is most likely not that concept. A description can be a statement, in which case, its meaning is a proposition.

#### description

- Definition: [representation that](#) provides a detailed account of something, a verbal portrait
- Dictionary Basis: a spoken or written representation or account of a person, object, or event [NODE 'description']
- Necessity: [No description that portrays a concept is a descriptive example that illustrates that concept.](#)
- Necessity: [No description that portrays a concept is a note that comments on that concept.](#)
- Necessity: [No description that portrays a concept is a reference that supports that concept.](#)

#### concept is illustrated by descriptive example

- Synonymous Form: [descriptive example illustrates concept](#)
- Note: The meaning of a descriptive example is typically a proposition.

#### descriptive example

- Definition: [representation that](#) provides descriptive material that is a sample of the thing defined
- Source: [based on MWCD and NODE](#)
- Dictionary Basis: one (as an item or incident) that is representative of all of a group or type [MWCD 'example']
- Dictionary Basis: a thing characteristic of its kind or illustrating a general rule [NODE 'example']
- Necessity: [No descriptive example that illustrates a concept is a definition of that concept.](#)
- Necessity: [No descriptive example that illustrates a concept is a description that portrays that concept.](#)
- Necessity: [No descriptive example that illustrates a concept is a note that comments on that concept.](#)
- Necessity: [No descriptive example that illustrates a concept is a reference that supports that concept.](#)
- Example: Chris Cushing is an example of EU-Rent's concept of 'rental customer'
- Example: The vehicle with VIN#88744332 is an example of EU-Rent's concept of 'rental car'

#### concept is commented on in note

- Synonymous Form: [note comments on concept](#)
- Note: The meaning of a note that comments on a concept is most likely not that concept. A note is typically a statement whose meaning is a proposition.

## note

- Definition: [representation](#) that annotates or explains
- Necessity: No [note](#) that *comments on a [concept](#)* is a [definition of that \[concept\]\(#\)](#).
- Necessity: No [note](#) that *comments on a [concept](#)* is a [description that \*portrays that \[concept\]\(#\)\*](#).
- Necessity: No [note](#) that *comments on a [concept](#)* is a [descriptive example that \*illustrates that \[concept\]\(#\)\*](#).
- Necessity: No [note](#) that *comments on a [concept](#)* is a [reference that \*supports that \[concept\]\(#\)\*](#).
- Synonym: [remark](#)
- Synonym: [comment](#)

## comment

- See: [note](#)

## remark

- See: [note](#)

### 11.2.2.3 Business Content of a Communication

#### communication content

- Definition: [representation](#) that is a subdivision of a written composition that consists of one or more statements and deals with one point or gives the words of one speaker
- Source: MWCD (1a)
- Synonym: [message content](#)
- Synonym: [document content](#)

#### document content

- See: [communication content](#)

#### message content

- See: [communication content](#)

#### communication content is composed of representation

- Concept Type: [partitive fact type](#)

#### concept is supported by reference

- Synonymous Form: [reference supports \[concept\]\(#\)](#)

## reference

- Definition: [representation](#) that is the mention or citation of a source of information used to direct a reader elsewhere for additional information about [a given \[concept\]\(#\)](#)
- Dictionary Basis: a mention or citation of a source of information in a book or article [NODE 'reference']
- Necessity: No [reference](#) that *supports a [concept](#)* is a [definition of that \[concept\]\(#\)](#).
- Necessity: No [reference](#) that *supports a [concept](#)* is a [description that \*portrays that \[concept\]\(#\)\*](#).
- Necessity: No [reference](#) that *supports a [concept](#)* is a [descriptive example that \*illustrates that \[concept\]\(#\)\*](#).
- Necessity: No [reference](#) that *supports a [concept](#)* is a [note that \*comments on that \[concept\]\(#\)\*](#).

Example: 'The Highway Code' published by HMSO, 2005.

Example: The descriptions of car models' capacity, fuel economy, and performance taken from the manufacturers' specifications.

### reference points to information source

Definition: the communication content plays the role of an information source for the reference

### information source

Concept Type: role

Definition: communication content that is used as a resource to supply information or evidence



# 12 Business Rules

## Vocabulary for Describing Business Rules

Language: [English](#)

Included Vocabulary: [Vocabulary for Describing Business Vocabularies](#)

### 12.1 Categories of Guidance

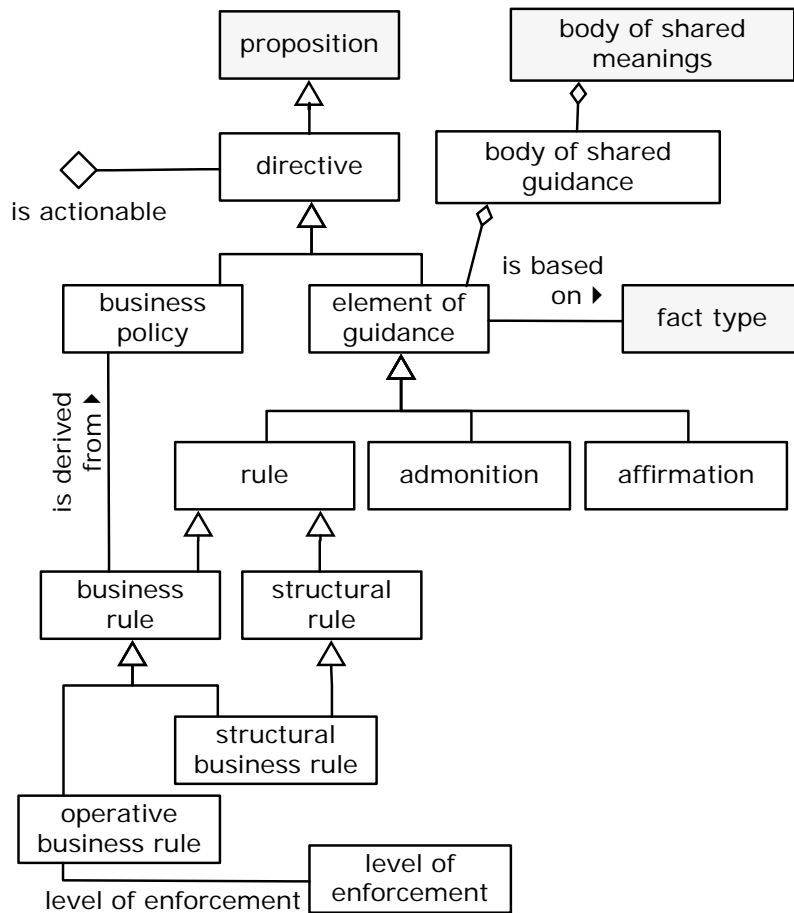


Figure 12.1

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

## 12.1.1 Guidance

### directive

Definition:	means that defines or constrains some aspect of an enterprise
General Concept:	<a href="#">proposition</a>
Note:	This sense of ‘means’ (as in ‘ends and means’, rather than ‘is meant as’) arises from the Business Motivation Model [BMM].
Note:	Intended to assert business structure or to control or influence the behavior of the enterprise.
Note:	Its formulation is under the enterprise’s control by a party authorized to manage, control or regulate an enterprise, by selection from alternatives in response to a combination of assessments.
Dictionary Basis:	an official or authoritative instruction [NODE]

### directive is actionable

Concept Type:	<a href="#">characteristic</a>
Note:	‘Actionable’ means that a person who knows about the directive could observe a relevant situation (including his or her own behavior) and decide directly whether or not the business was complying with the directive.
Dictionary Basis:	subject to or affording ground for an action or suit at law [MWUD ‘actionable’]
Dictionary Basis:	a thing done : DEED [MWUD (5a) ‘action’]
Note:	The sense intended is: “It’s actually something you can put to use or apply.”

### business policy

Definition:	<a href="#">directive that is not actionable</a> whose purpose is to guide an enterprise
Note:	Compared to a business rule, a business policy tends to be: <ul style="list-style-type: none"><li>- less structured</li><li>- less discrete or not atomic</li><li>- less carefully expressed in terms of standard vocabulary</li><li>- not directly actionable.</li></ul>
Dictionary Basis:	definite course or method of action selected (as by a government, institution, group, or individual) from among alternatives and in the light of given conditions to guide and usually determine present and future decisions [MWUD “Policy” 5a]

### element of guidance

Definition:	<a href="#">directive that is actionable</a> , whose purpose is to advise or inform with a goal of resolving a problem or difficulty, especially as given by someone in authority
Necessity:	No <a href="#">business policy is an element of guidance</a> .

### element of guidance is based on fact type

Definition:	<a href="#">the element of guidance</a> is formulated using <a href="#">the fact type</a>
Example:	The EU-Rent element of guidance (business rule) that is expressed as “It is obligatory that each rental specifies a car group.” (or, in RuleSpeak, “A rental must have a car group.”) is based on the EU-Rent fact type ‘ <a href="#">rental</a> specifies <a href="#">car group</a> ’.

### body of shared guidance

Definition:	all of <a href="#">the elements of guidance</a> within a <a href="#">body of shared meanings</a>
-------------	--



body of shared meanings includes body of shared guidance

body of shared guidance includes element of guidance

## 12.1.2 Rules

### rule

- Definition: element of guidance that introduces an obligation or a necessity
- Dictionary Basis: standard by which something is judged or valued; criterion [MWUD (2B) 'rule']
- Dictionary Basis: principle or standard by which something may be judged or decided [determined] [NODE 'criterion']
- Dictionary Basis: standard on which a judgment or decision may be based [MWCD (2) 'criterion']

### business rule

- Definition: rule that is under business jurisdiction

### business rule is derived from business policy

- Synonymous Form: business policy is basis for business rule

### structural rule

- Definition: rule that is intended as a definitional criterion
- Necessity: Each structural rule is a proposition that another proposition is a necessity.

### structural business rule

- Definition: structural rule that is a business rule

### operative business rule

- Definition: business rule that is intended to produce an appropriate or designed effect
- Definition: business rule that covers conduct, action, practice, or procedure within a particular activity or sphere
- Definition: business rule that there is an obligation concerning conduct, action, practice or procedure
- Dictionary Basis: a prescribed, suggested, or self-imposed guide for conduct or action : a regulation or principle <his parents laid down the rule that he must do his homework before going out to play> <a very sound rule for any hiker is to mind his own business [...] F.D.Smith & Barbara Wilcox> <made it a rule never to lose his temper> [...] [MWU (1a) 'rule']
- Dictionary Basis: a prescribed guide for conduct or action [MWCD 'rule']
- Necessity: Each operative business rule is a proposition that another proposition is an obligation.
- Necessity: No operative business rule is a structural business rule.

## 12.1.3 Enforcement

### level of enforcement

- Definition: something that represents a position in a graded or ordered scale of values that specifies the severity of action imposed in order to put or keep an operative business rule in force
- Dictionary Basis: a position on a real or imaginary scale of amount, quantity, extent, or quality [NODE 'level']

Dictionary Basis:	compel observance of or compliance with [NODE 'enforcement']
Example:	An example set of levels of enforcement, based on [BMM]
Enforcement Level:	<u>strict</u>
Definition:	strictly enforced (If you violate the rule, you cannot escape the penalty.)
Enforcement Level:	<u>deferred</u>
Definition:	deferred enforcement (Strictly enforced, but enforcement may be delayed — e.g., waiting for resource with required skills.)
Enforcement Level:	<u>pre-authorized</u>
Definition:	pre-authorized override (Enforced, but exceptions allowed, with prior approval for actors with before-the-fact override authorization.)
Enforcement Level:	<u>post-justified</u>
Definition:	post-justified override (If not approved after the fact, you may be subject to sanction or other consequences.)
Enforcement Level:	<u>override</u>
Definition:	override with explanation (Comment must be provided when the violation occurs.)
Enforcement Level:	<u>guideline</u>
Definition:	guideline (suggested, but not enforced.)

## operative business rule has level of enforcement

### 12.1.4 Admonitions and Affirmations

#### admonition

Definition:	<u>element of guidance</u> that there is <b>not an obligation or necessity</b> where, by custom or practice, one might be assumed
Note:	The purpose of an <u>admonition</u> is to preempt application of “rules” that might be assumed by some members of a semantic community, but are not actually rules admitted by the community. Often, the reason for this assumption in a business is that other, similar, businesses have such rules. Typically, the reason for an explicit <u>admonition</u> is that people in the business have mistakenly applied the non-existent “rule” in the past.
Example:	(In a bank) There is no rule that a person must be over some given age in order open a savings account: “There is no minimum age for opening a savings account”.
Example:	(In EU-Rent) There is no rule that a rented car can be dropped off only at the return branch specified in the rental agreement: “At the end of a rental, a rental car can be dropped off at any EU-Rent branch”. There is a related rule that if the drop-off branch is not the specified return branch, the rental will incur a penalty charge, but the importance of this <u>admonition</u> is that EU-Rent wants its cars back, even if they are in the wrong places. It does not want a branch to refuse to accept a car on the grounds that it is not the specified return branch.

Necessity: Each admonition *is* a proposition that another proposition *is* not an obligation or a necessity.

Necessity: No rule *is* an admonition.

### affirmation

Definition: element of guidance that there is a permissibility or possibility where, by custom or practice, one might not be assumed

Example: (In a bank) “It is possible that an account balance is negative”.

Example: (In EU-Rent) “A rental car may be dropped off at any EU-Rent branch, even one that is not its scheduled drop-off branch”.

Necessity: Each affirmation *is* a proposition that another proposition *is* a possibility or a permissibility.

Necessity: No rule *is* an affirmation.

Necessity: No admonition *is* an affirmation.

## 12.2 Statements of Guidance

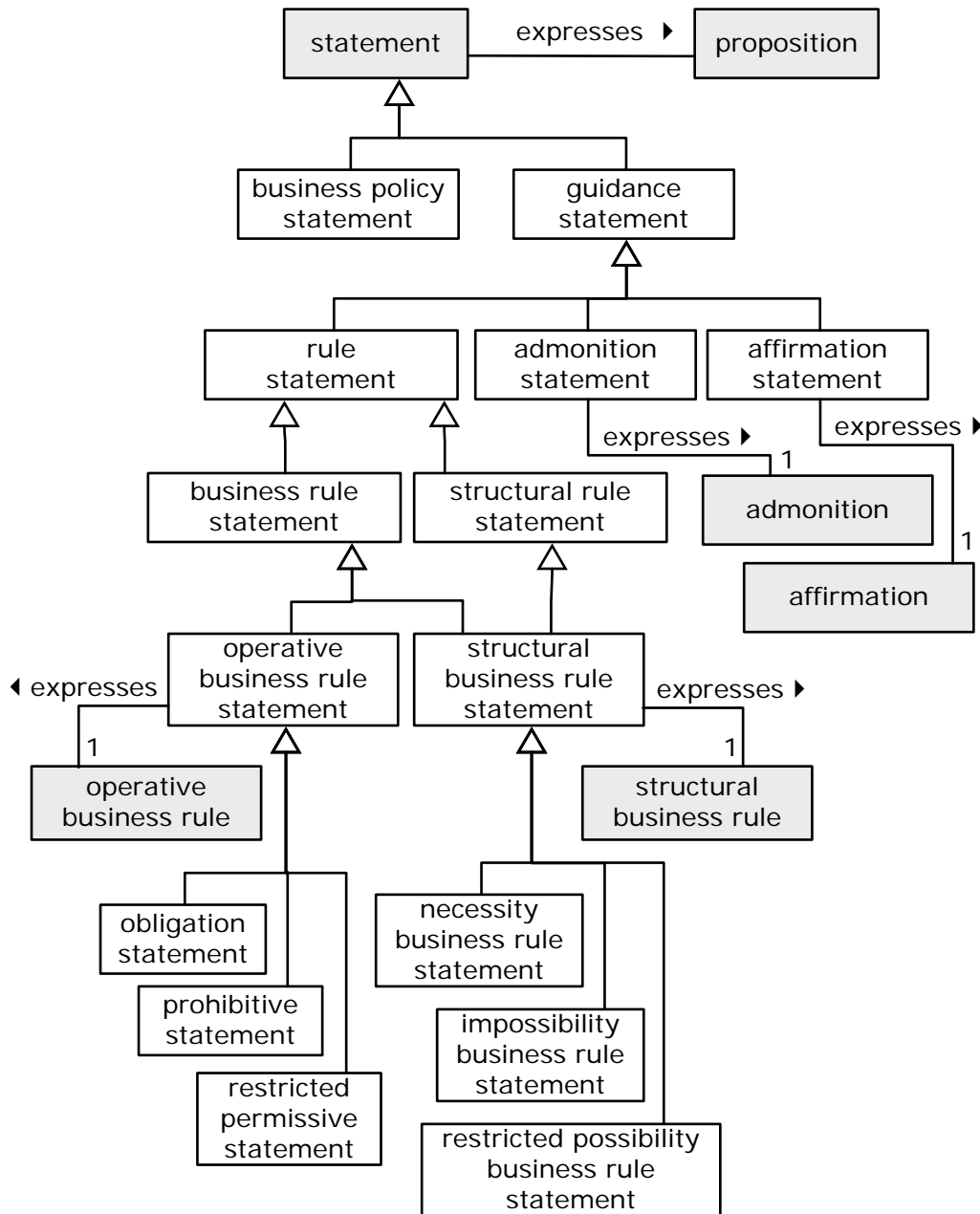


Figure 12.2

---

This diagram is not normative abstract syntax for SBVR, but is for illustration only.

---

## 12.2.1 Categories of Business Statement

### business policy statement

Definition: statement that expresses a business policy

### guidance statement

Definition: statement that expresses an element of guidance

Definition: statement that provides advice or information aimed at resolving a problem or difficulty, especially as given by someone in authority

Dictionary Basis: a statement that provides advice or information aimed at resolving a problem or difficulty, especially as given by someone in authority [NODE 'guidance']

Necessity: No guidance statement is a business policy statement.

### rule statement

Definition: guidance statement that expresses something to be a necessity or obligation

### business rule statement

General Concept: rule statement

Definition: operative business rule statement or structural business rule statement

### structural rule statement

Definition: rule statement of necessary characteristics

Necessity: Each structural rule statement expresses exactly one structural rule.

### structural business rule statement

Definition: structural rule statement that is a business rule statement

Necessity: Each structural business rule statement expresses exactly one structural business rule.

### operative business rule statement

Definition: business rule statement that expresses an operative business rule

Definition: business rule statement that is a definite or clear expression of an operative business rule in speech or writing

Dictionary Basis: a definite or clear expression of something in speech or writing [NODE 'statement']

Necessity: Each operative business rule statement expresses exactly one operative business rule.

Necessity: No operative business rule statement is a structural business rule statement.

### admonition statement

Definition: guidance statement that expresses an admonition

Necessity: Each admonition statement expresses exactly one admonition.

Example: "It is permitted that the drop-off branch of a rental is not the return branch of the rental."

Example: "The drop-off branch of a rental need not be the return branch of the rental."

### affirmation statement

Definition: guidance statement that expresses an affirmation

- Necessity: Each [affirmation statement](#) expresses exactly one [affirmation](#).
- Necessity: No [admonition statement](#) is an [affirmation statement](#).
- Example: “It is possible that the notification date/time of a bad experience that occurs during a rental is after the actual return date/time of the rental.”
- Example: “The notification date/time of a bad experience that occurs during a rental is sometimes after the actual return date/time of the rental.”

## 12.2.2 Business Statements

### obligation statement

- Definition: [operative business rule statement](#) that is expressed in a syntactical form for expressing an [operative business rule](#) in terms of [obligation](#)
- Definition: [operative business rule statement](#) that is expressed in an obligatory form, regardless of language, such as ‘It is obligatory that ...’ or using the word ‘must’ (but not ‘must not’), or listed as a ‘requirement’
- Necessity: No [obligation statement](#) is a [prohibitive statement](#).
- Necessity: No [obligation statement](#) is a [restricted permissive statement](#).
- Note: The same rule could also be expressed in another form, such as a prohibitive form, by introducing or removing negation.
- Example: “If the drop-off location of a rental is not the EU-Rent site of the return branch of the rental then it is obligatory that the rental incurs a location penalty charge.”
- Example: “A rental must incur a location penalty charge if the drop-off location of the rental is not the EU-Rent site of the return branch of the rental.”

### prohibitive statement

- Definition: [operative business rule statement](#) that is expressed in a syntactical form for expressing an [operative business rule](#) in terms of [prohibition](#)
- Definition: [operative business rule statement](#) that is expressed in a prohibitive form, regardless of language, such as ‘It is prohibited that ...’ or using the words ‘must not’
- Necessity: No [prohibitive statement](#) is a [restricted permissive statement](#).
- Note: The same rule could also be expressed in another form, such as an obligatory form, by introducing or removing negation.
- Example: “It is prohibited that a rental is open if a driver of the rental is a barred driver.”
- Example: “A rental must not be open if a driver of the rental is a barred driver.”

### restricted permissive statement

- Definition: [operative business rule statement](#) that is expressed in a syntactical form for expressing an [operative business rule](#) in terms of [permission](#), providing the condition is met
- Definition: [operative business rule statement](#) that is expressed in a permissive form, but with a condition that must be satisfied, regardless of language, such as “It is permitted that ... only if ...” or using the word “may” but subjected to “only if” or “only when”
- Note: The same rule could also be expressed in another form, such as an obligatory form, by introducing or removing negation.
- Example: “It is permitted that a rental is open only if an estimated rental charge is provisionally charged to the credit card of the renter of the rental.”

Example: “A rental may be open only if an estimated rental charge is provisionally charged to the credit card of the renter of the rental.”

### necessity business rule statement

Definition: [structural business rule statement](#) **that** is expressed in a syntactical form for expressing a [business rule statement](#) in terms of [necessity](#)

Definition: [business rule statement](#) **that** is expressed in a form indicating a necessity, regardless of language, such as ‘It is necessary that ...’

Necessity: **No** [necessity business rule statement](#) **is an** [impossibility business rule statement](#).

Necessity: **No** [necessity business rule statement](#) **is a** [restricted possibility business rule statement](#).

Example: “It is necessary that each rental has exactly one requested car group.”

Example: “Each rental always has exactly one requested car group.”

### impossibility business rule statement

Definition: [structural business rule statement](#) **that** is expressed in a syntactical form for expressing a [business rule statement](#) in terms of impossibility

Definition: [business rule statement](#) **that** is expressed in a form indicating impossibility, regardless of language, such as ‘It is impossible that ...’

Necessity: **No** [impossibility business rule statement](#) **is a** [restricted possibility business rule statement](#).

Example: “It is impossible that the pick-up branch of a one-way rental is the return branch of that rental.”

Example: “The pick-up branch of a one-way rental is never the return branch of that rental.”

### restricted possibility business rule statement

Definition: [structural business rule statement](#) **that** is expressed in a syntactical form for expressing a [business rule statement](#) in terms of [possibility](#), providing the condition is met

Definition: [structural business rule statement](#) **that** is expressed in a form indicating [possibility](#), but with a condition that must be satisfied, regardless of language, such as ‘It is possible that ... only if ...’

Example: “It is possible that a rental is an open rental only if the rental car of the rental has been picked up.”

Example: “A rental can be an open rental only if the rental car of the rental has been picked up.”





## 13 Vocabulary-Driven Interchange Using MOF and XMI

This section presents the [Vocabulary-to-MOF/XMI Vocabulary](#) and the [Vocabulary-to-MOF/XMI Mapping Rule Set](#). The vocabulary is used by the rule set, which contains rules for transforming any vocabulary defined in terms of the Logical Formulation of Semantics Vocabulary into a MOF model and XMI-based XML schema that supports repository services and data interchange of facts that can be formulated as atomic or instantiation formulations.

The UML diagrams shown in the vocabulary sections of this document might at first appear to make up a MOF model of SBVR. However, these diagrams represent vocabulary, not MOF classes. The diagrams show orthogonal dimensions of specialization in which a single thing can be an instance of multiple concepts at the same time without there being a single most specific concept. This sort of specialization is normal in language but is not supported by MOF which requires an object to be an instance of exactly one most-specific class. The diagrams show characteristics as if they were unary associations, but unary associations are not supported by MOF, nor are ternary associations which also occur in the diagrams. The MOF model of SBVR is created by a transformation from the SBVR vocabulary following the rules in this section.

### 13.1 Vocabulary-to-MOF/XMI Vocabulary

This vocabulary includes the Logical Formulation of Semantics Vocabulary as its starting point. It further incorporates many concepts from UML 2, MOF 2 and XMI 2.1 specifications.

UML 2 concepts are limited to the subset of UML 2 included in EMOF as described by the final adopted specification of the *Meta Object Facility (MOF) 2.0 Core*.<sup>1</sup>

XMI concepts are used to tailor XML schema production. See “Tailoring Schema Production” in OMG’s *Meta Object Facility (MOF) 2.0 XMI Mapping* specification for further descriptions of how MOF tags are used to represent XMI directives.<sup>2</sup>

Reference schemes are given for several of the types. These reference schemes satisfy the needs of the SBVR-to-MOF/XMI mapping and are consistent with constraints in the UML 2 specification.

---

#### Vocabulary to MOF/XMI Vocabulary

Language:	<a href="#">English</a>
Included Vocabulary:	<a href="#">Logical Formulation of Semantics Vocabulary</a>
Included Vocabulary:	<a href="#">Essential SBVR Vocabulary</a>

---

- 
1. MOF 2 is in finalization at the time of writing, so the limit to EMOF must be reverified with the final specification.
  2. This specification is in finalization at the time of writing. The [XMI for MOF 2 Relevant Vocabulary](#) will be updated as needed to match the final specification.

## 13.1.1 Mapping

### representation maps to UML element

- Definition: [the representation](#) is mapped based on [the Vocabulary-to-MOF/XML Mapping Rule Set](#) to the [UML element](#)
- Synonymous Form: [UML element comes from representation](#)

### vocabulary namespace maps to package

- Definition: [the vocabulary namespace](#) is mapped based on [the Vocabulary-to-MOF/XML Mapping Rule Set](#) to [the UML package](#) that contains UML classes providing a MOF/XML implementation of the vocabulary namespace
- Synonymous Form: [package comes from vocabulary namespace](#)

### text is for placeholder

- Definition: [the text](#) represents [the placeholder](#) of [a form of expression](#) based on [the designation](#) corresponding to [the placeholder](#) with the conditional addition of a numeral in the case where [the same designation](#) also occurs for another [placeholder](#) within [the form of expression](#)
- Example: The placeholder 'Note that a text is normalized to have no leading or trailing spaces and no other white space than single spaces.

### XMI name is derived from text

- Definition: [the XMI name](#) is derived from [the text](#) as described below:  
An XMI name derived from a text takes each character directly from the sequence except for the following:
1. A space character is replaced by a hyphen ('-').
  2. A hyphen is replaced by two consecutive hyphens ('--').
  3. An underscore character is replaced by two consecutive underscores ('\_\_').
  4. A character specified to be invalid by the *Extensible Markup Language (XML) 1.0* specification (Second Edition, see [www.w3.org/TR/REC-xml](http://www.w3.org/TR/REC-xml)), either in general or as a first character if in the first character position of the text, is replaced with an underscore followed by the hexadecimal encoding (using lower case) of the character with leading zeros suppressed followed by another underscore (e.g., '\_d7ff\_').
- Note: Note that a text is normalized to have no leading or trailing spaces and no other white space than single spaces.

### instantiation prefix

- Definition: [text that](#) is used with a designation to form a class name for objects representing that a thing is an instance of the designated concept
- Concept Type: [role](#)

### instantiation prefix is used for language

- Definition: [the instantiation prefix](#) is used when generating class names for designations of the [language](#)

## 13.1.2 Relevant Concepts from UML 2

### UML element

Source: [UML 2 Infrastructure](#) [Core::Abstractions::Elements::Element]

### name

Source: [UML 2 Infrastructure](#) [Core::Basic::NamedElement::name]

General Concept: [string](#)

Concept Type: [role](#)

### named element

Source: [UML 2 Infrastructure](#) [Core::Basic::NamedElement]

General Concept: [UML element](#)

### named element has name

Source: based on [UML 2 Infrastructure](#) [Core::Basic::NamedElement::name]

### type

Source: [UML 2 Infrastructure](#) [Core::Basic::Type]

General Concept: [named element](#)

Reference Scheme: [the name of the type and a package that owns the type](#)

### typed element

Source: [UML 2 Infrastructure](#) [Core::Basic::TypedElement]

General Concept: [named element](#)

### typed element has type

Source: based on [UML 2 Infrastructure](#) [Core::Basic::TypedElement::type]

### class

Source: [UML 2 Infrastructure](#) [Core::Basic::Class]

General Concept: [type](#)

### property

Source: [UML 2 Infrastructure](#) [Core::Basic::Property]

General Concept: [typed element](#)

### lower bound

Source: [UML 2 Infrastructure](#) [Core::Abstractions::Multiplicities:: MultiplicityElement::lower]

General Concept: [integer](#)

Concept Type: [role](#)

### upper bound

Source: [UML 2 Infrastructure](#) [Core::Abstractions::Multiplicities:: MultiplicityElement::upper]

General Concept: [unlimited natural number](#)

Concept Type: [role](#)

### unlimited natural number

Source: [UML 2 Infrastructure](#) [Core::PrimitiveTypes::UnlimitedNaturalNumber]

### Infinity

Source: based on [UML 2 Infrastructure](#) [‘\*’ designating an instance of Core::PrimitiveTypes::UnlimitedNaturalNumber]

General Concept: [unlimited natural number](#)

### owned attribute

Source: [UML 2 Infrastructure](#) [Core::Basic::Class::ownedAttribute]

General Concept: [property](#)

Concept Type: [role](#)

Reference Scheme: the [name](#) of the [owned attribute](#) and a [class](#) that [has](#) the [owned attribute](#)

### class has owned attribute

Source: based on [UML 2 Infrastructure](#) [Core::Basic::Class::ownedAttribute]

### superclass

Source: [UML 2 Infrastructure](#) [Core::Basic::Class::superClass]

General Concept: [class](#)

Concept Type: [role](#)

### class has superclass

Source: based on [UML 2 Infrastructure](#) [Core::Basic::Class::superClass]

### class is abstract

Source: based on [UML 2 Infrastructure](#) [Core::Basic::isAbstract]

### data type

Source: [UML 2 Infrastructure](#) [Core::Basic::DataType]

General Concept: [type](#)

### primitive type

Source: [UML 2 Infrastructure](#) [Core::DataTypes::PrimitiveType]

General Concept: [data type](#)

### Integer Type

Source: [UML 2 Infrastructure](#) [‘Core::PrimitiveTypes::Integer’]

General Concept: [primitive type](#)

### string

Source: [UML 2 Infrastructure](#) [Core::PrimitiveTypes::String]

General Concept: [text](#)

### String Type

Source: [UML 2 Infrastructure](#) [Core::PrimitiveTypes::String]  
General Concept: [data type](#)

### package

Source: [UML 2 Infrastructure](#) [Core::Basic::Package]  
General Concept: [named element](#), [namespace](#)

### owned type

Source: [UML 2 Infrastructure](#) [Core::Basic::Package::ownedType]  
General Concept: [type](#)  
Concept Type: [role](#)

### package has owned type

Source: based on [UML 2 Infrastructure](#) [Core::Basic::Package::ownedType]

## 13.1.3 XMI for MOF 2

### named element has XMI name

Definition: the [named element](#) is tagged with the [XMI name](#)

### package has XMI namespace prefix

Definition: the [package](#) is tagged with the [XMI namespace prefix](#)

### package has XMI namespace URI

Definition: the [package](#) is tagged with the [XMI namespace URI](#)

### XMI name

Source: [XMI 2.1 Tags](#) [XMIname]  
Definition: [string](#) that is the value of an 'org.omg.xmi.XMIname' tag  
Concept Type: [role](#)

### XMI namespace prefix

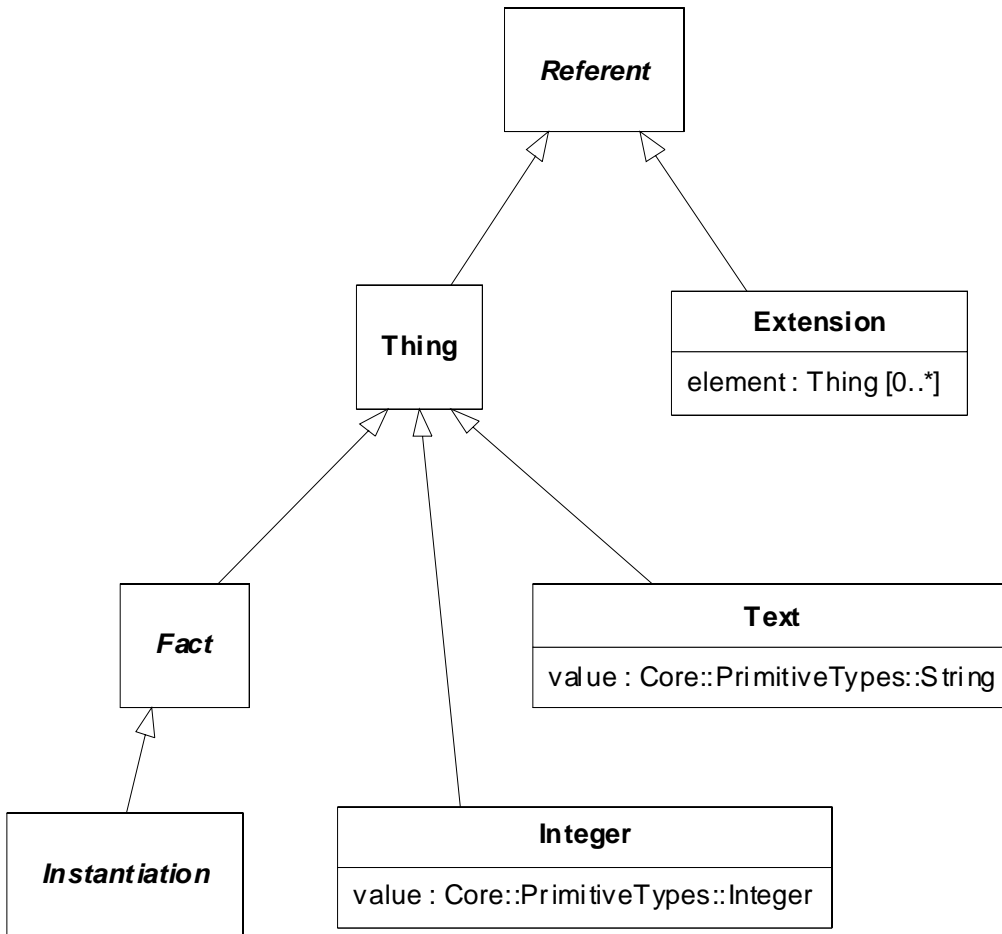
Source: [XMI 2.1 Tags](#) [nsPrefix]  
Definition: [string](#) that is the value of an 'org.omg.xmi.nsPrefix' tag  
Concept Type: [role](#)

### XMI namespace URI

Source: [XMI 2.1 Tags](#) [nsURI]  
Definition: [string](#) that is the value of an 'org.omg.xmi.nsURI' tag  
Concept Type: [role](#)

## 13.2 Essential SBVR

The Essential SBVR Package is a UML/MOF package containing classes used by target packages of the [Vocabulary-to-MOF/XMI Mapping Rule Set](#). Its contents are all defined in the [Essential SBVR Vocabulary](#) below.



**Figure 13.1** The Essential SBVR Package

The [Referent Class](#) is used to represent what a fact can refer to. It has two subclasses. One, the [Thing Class](#), is used to represent individual things. The other, the [Extension Class](#), is used to represent a set of things, the entire set for which a fact is true. Each instance of the [Extension Class](#) has zero or more members which are instances of the [Thing Class](#).

The [Thing Class](#) is used to represent things in general that are the subjects or objects of facts. The expressed type or types of a thing are known from facts about it. The [Thing Class](#) has subclasses for different kinds of representations of things: the [Fact Class](#), the [Text Class](#) and the [Integer Class](#).

A subclass of the [Fact Class](#) is created according to the [Vocabulary-to-MOF/XMI Mapping Rule Set](#) for each sentential form in a vocabulary. Each instance of one of these subclasses represents a fact of the fact type that has the sentential form.

A subclass of the [Instantiation Class](#) is created according to the [Vocabulary-to-MOF/XMLI Mapping Rule Set](#) for each designation in a vocabulary. Each instance of one of these subclasses represents a fact that a thing is an instance of the concept denoted by the designation.

The [Text Class](#) and the [Integer Class](#) provide convenient ways to use text and integers in representing facts.

The [Essential SBVR Vocabulary](#) is used to refer to model elements of the [Essential SBVR Package](#).

---

## Essential SBVR Vocabulary

Language: [English](#)

---

## Essential SBVR Package

Definition: the [package](#) that contains classes used in general by MOF/XMLI implementations of vocabularies generated according to the [Vocabulary-to-MOF/XMLI Mapping Rule Set](#)

Necessity: 'Essential SBVR' is the [name of the Essential SBVR Package](#).  
'ESBVR' is the [XML namespace prefix of the Essential SBVR Package](#).

## Referent Class

Definition: the [class](#) that is owned by the [Essential SBVR Package](#) and that has the [name](#) 'Referent'

Necessity: The [Referent Class](#) is abstract.

## Thing Class

Definition: the [class](#) that is owned by the [Essential SBVR Package](#) and that has the [name](#) 'Thing'

Necessity: The [Thing Class](#) is not abstract.  
The [Referent Class](#) is a superclass of the [Thing Class](#).

## Fact Class

Definition: the [class](#) that is owned by the [Essential SBVR Package](#) and that has the [name](#) 'Fact'

Necessity: The [Fact Class](#) is abstract.  
The [Thing Class](#) is a superclass of the [Fact Class](#).  
'fact' is the [XML name of the Fact Class](#).

## Instantiation Class

Definition: the [class](#) that is owned by the [Essential SBVR Package](#) and that has the [name](#) 'Instantiation'

Necessity: The [Instantiation Class](#) is abstract.  
The [Fact Class](#) is a superclass of the [Instantiation Class](#).  
'instantiation' is the [XML name of the Instantiation Class](#).

## Integer Class

Definition: the [class](#) that is owned by the [Essential SBVR Package](#) and that has the [name](#) 'Integer'

Necessity: The [Thing Class](#) is a superclass of the [Integer Class](#).  
The [Integer Class](#) is not abstract.

'value' is the name of an owned attribute of the Integer Class and the Integer Type is the type of that owned attribute.

'integer' is the XMI name of the Integer Class.

### Text Class

Definition: the class that is owned by the Essential SBVR Package and that has the name 'Text'

Necessity: The Thing Class is a superclass of the Text Class.

The Text Class is not abstract.

'value' is the name of an owned attribute of the Text Class and the String Type is the type of that owned attribute.

'text' is the XMI name of the Text Class.

### Extension Class

Definition: the class that is owned by the Essential SBVR Package and that has the name 'Extension'

Necessity: The Referent Class is a superclass of the Extension Class.

The Extension Class is not abstract.

'element' is the name of an owned attribute of the Extension Class, the Thing Class is the type of the owned attribute, 0 is the lower bound of the owned attribute and Infinity is the upper bound of the owned attribute.

'extension' is the XMI name of the Extension Class.

## 13.3 Vocabulary-to-MOF/XMI Mapping Rule Set

The Vocabulary-to-MOF/XMI Mapping Rule Set guides the transformation of a vocabulary namespace defined in terms of the Logical Formulation of Semantics Vocabulary into a MOF-compliant model with tags for XML schema production based on the XMI 2.1 specification.

In general, the rules should be strictly enforced, but deviation is permissible for rules that map symbols to UML and XMI names as long as all parties using the resulting MOF model or XML schema are aware of the deviations.

The rules aim at producing a UML package. The source is a vocabulary namespace. The rules are expressed using the Vocabulary-to-MOF/XMI Vocabulary and the Essential SBVR Vocabulary.

---

### Vocabulary-to-MOF/XMI Mapping Rule Set

Language: English

Vocabulary: Vocabulary-to-MOF/XMI Vocabulary

---

#### 13.3.1 Namespace Mapping Rules

1. A vocabulary namespace must map to a package.

Possibility: It is possible that a vocabulary namespace maps to more than one package.



### 13.3.2 Designation Mapping Rules

1. Each designation that *is in* a vocabulary namespace that *maps to* a package must *map to* exactly one owned type of the package if the signifier of the designation is a text.
2. Each owned type that *comes from* a designation must *be* a class.
3. The Instantiation Class must *be* a superclass of each class that *comes from* a designation.
4. Each class that *comes from* a designation must *not be* abstract.
5. The name of each class that *comes from* a designation that *is in* a vocabulary namespace that *is for* a language must *be* the text that *combines* the instantiation prefix that *is used for* the language with the signifier of the designation.
6. The instantiation prefix that *is used for* English must *be* "is".
7. The XMI name of each class that *comes from* a designation must *be derived from* the name of the class.
8. Each class that *comes from* a designation *has* exactly one owned attribute.
9. The name of the owned attribute of each class that *comes from* a designation must *be* the signifier of the designation.
10. The type of the owned attribute of each class that *is from* a designation must *be* the Referent Class.
11. The lower bound and the upper bound of the owned attribute of each class that *comes from* a designation must *be* 1.

### 13.3.3 Sentential Form Mapping Rules

1. Each sentential form that *is in* a vocabulary namespace that *maps to* a package must *map to* exactly one owned type of the package if the expression of the sentential form is a text.
2. Each owned type that *comes from* a sentential form must *be* a class.
3. The Fact Class must *be* a superclass of each class that *comes from* a sentential form.
4. Each class that *comes from* a sentential form must *not be* abstract.
5. The name of each class that *comes from* a sentential form must *be* the text that *represents* the sentential form.
6. The XMI name of each class that *comes from* a sentential form must *be derived from* the text that *represents* the sentential form.

### 13.3.4 Placeholder Mapping Rules

1. Each placeholder of a sentential form that *is mapped to* a class must *be mapped to* exactly one owned attribute of the class.
2. The Referent Class must *be* the type of each property that *comes from* a placeholder.
3. The name of each property that *comes from* a placeholder must *be* the text that *is for* the placeholder.

4. The XMI name of a property that comes from a placeholder must be derived from the text that is for the placeholder.
5. The lower bound and the upper bound of each property that comes from a placeholder must be 1.

### 13.3.5 Notes and Limitations

1. If there are specifications of XMI names in addition to or overriding what is provided by the mapping rules, then a writer and reader of XML documents must both share those specifications.
2. If different reference schemes are used, then the XML document contents should satisfy the reference schemes of both (or at least that of the reader). But this is a matter of selecting content and is independent of the XML format of the content.
3. Only designations and forms of expression expressed in Unicode are used to derive MOF and XMI names according to mapping rules. Other kinds of expression require additional specification beyond what is covered by the rules.

## 14 Index of Vocabulary Entries

### A

actuality .....	33
actuality involves thing in role .....	33
admonition .....	140
admonition statement .....	143
adopted definition .....	119
affirmation .....	141
affirmation statement .....	143
aggregation formulation .....	63
alethic modality .....	94
answer nominalization .....	66
antecedent .....	49
antecedent .....	95
argument .....	95
arity .....	95
aspect .....	125
associative fact type .....	123
assortment fact type .....	124
at-least-n quantification .....	58
at-least-n quantification has minimum cardinality .....	58
at-most-n quantification .....	58
at-most-n quantification has maximum cardinality .....	58
at-most-one quantification .....	59
atomic formula .....	95
atomic formulation .....	45
atomic formulation has role binding .....	45
atomic formulation is based on fact type .....	46
auxiliary variable .....	68

### B

bag projection .....	69
binary fact type .....	17
bindable target .....	44
bindable target is bound to instantiation formulation .....	47
bindable target is referenced by role binding .....	46
body of shared concepts .....	112
body of shared concepts includes concept .....	112
body of shared guidance .....	138
body of shared guidance includes element of guidance .....	139
body of shared meanings .....	111
body of shared meanings has elementary fact type .....	112
body of shared meanings includes body of shared concepts .....	112
body of shared meanings includes body of shared guidance .....	139
body of shared meanings is understood by semantic community .....	111
body of shared meanings1 contains body of shared meanings2 .....	112
business policy .....	138
business policy is basis for business rule .....	139
business policy statement .....	143
business rule .....	139
business rule is derived from business policy .....	139
business rule statement .....	143

## C

cardinality .....	58
categorization fact type .....	125
categorization scheme .....	116
categorization scheme contains category .....	116
categorization scheme is for general concept .....	116
categorization type .....	117
categorization type is for general concept .....	117
category .....	116
category is inactive .....	117
category is included in categorization scheme .....	116
characteristic .....	16
characteristic is essential to concept .....	115
characteristic is incorporated into concept .....	19
characteristic is used by reference scheme .....	30
characteristic type .....	115
class .....	149
class has owned attribute .....	150
class has superclass .....	150
class is abstract .....	150
closed logical formulation .....	42
closed logical formulation formalizes statement .....	42
closed logical formulation means proposition .....	42
closed projection .....	69
closed projection defines concept .....	69
closed projection formalizes definition .....	69
closed projection means question .....	70
closed semantic formulation .....	40
closed semantic formulation formulates meaning .....	40
comment .....	134
communication content .....	134
communication content is composed of representation .....	134
community .....	110
community has subcommunity .....	111
concept .....	15
concept has definition .....	23
concept has delimiting characteristic .....	115
concept has designation .....	23
concept has extension .....	34
concept has instance .....	34
concept has reference scheme .....	30
concept has role namespace .....	28
concept has shared understanding by semantic community .....	112
concept incorporates characteristic .....	19
concept incorporates essential characteristic .....	115
concept is closed in conceptual schema .....	32
concept is commented on in note .....	133
concept is considered by instantiation formulation .....	47
concept is illustrated by descriptive example .....	133
concept is in conceptual schema .....	31

concept is included in body of shared concepts .....	112
concept is portrayed by description .....	133
concept is supported by reference.....	134
concept of thing as composite .....	121
concept of thing as continuant .....	122
concept of thing as developed .....	122
concept of thing as occurrent .....	122
concept of thing as primitive .....	122
concept of thing as unitary .....	121
concept of thing existing dependently .....	122
concept of thing existing independently .....	122
concept type .....	15
concept1 is coextensive with concept2 .....	19
concept1 specializes concept2 .....	18
concept2 generalizes concept1 .....	18
conceptual model .....	32
conceptual model includes fact .....	32
conceptual model is based on conceptual schema .....	32
conceptual schema .....	31
conceptual schema includes concept .....	31
conceptual schema includes fact .....	31
conceptual schema underlies conceptual model .....	32
condition 1 .....	49
condition 2 .....	49
conjunct 1 .....	49
conjunct 2 .....	49
conjunction .....	52
conjunction has conjunct 1 .....	52
conjunction has conjunct 2 .....	52
conjunction has logical operand 1 .....	52
conjunction has logical operand 2 .....	52
consequent .....	49
consequent .....	95
contextualization fact type .....	126
contextualized concept .....	127

## D

definition .....	23
definition is used as term of concept .....	119
delimiting characteristic .....	115
deontic modality.....	96
derivable concept .....	120
description .....	133
description portrays concept.....	133
descriptive example .....	133
descriptive example illustrates concept .....	133
designation .....	22
designation has signifier .....	23
designation is demonstrated by form of expression .....	25
designation is in namespace .....	27
designation is used for placeholder .....	27
directive .....	138
directive is actionable .....	138
disjunct 1.....	49

disjunct 2 .....	49
disjunction .....	52
disjunction has disjunct 1 .....	52
disjunction has disjunct 2 .....	52
disjunction has logical operand 1 .....	52
disjunction has logical operand 2 .....	52
document content .....	134
domain .....	96
domain grammar .....	96

## E

element of guidance .....	138
element of guidance is based on fact type .....	138
elementary fact type .....	112
equivalence .....	52
equivalence has condition 1 .....	53
equivalence has condition 2 .....	53
equivalence has logical operand 1 .....	53
equivalence has logical operand 2 .....	53
essential characteristic .....	115
Essential SBVR Package .....	153
Essential SBVR Vocabulary .....	12
Essential SBVR Vocabulary .....	153
exactly-n quantification .....	59
exactly-n quantification has cardinality .....	59
exactly-one quantification .....	59
exclusive disjunct has logical operand 1 .....	53
exclusive disjunct has logical operand 2 .....	53
exclusive disjunction 1 .....	49
exclusive disjunction 2 .....	50
exclusive disjunction .....	53
exclusive disjunction has exclusive disjunct 1 .....	53
exclusive disjunction has exclusive disjunct 2 .....	53
existential quantification .....	57
Explicitness of Representation .....	119
expression .....	20
extension .....	34
Extension Class .....	154
extensional definition .....	119

## F

facet .....	125
fact .....	19
Fact Class .....	153
fact is in conceptual model .....	32
fact is in conceptual schema .....	31
fact symbol .....	131
fact symbol is incorporated into form of expression .....	131
fact type .....	16

fact type formulation .....	64
fact type has fact in conceptual model.....	32
fact type has form of expression .....	25
fact type has role .....	19
fact type is elementary in body of shared meanings .....	112
fact type is internally closed in conceptual schema.....	31
fact type is semi-closed in conceptual schema.....	32
fact type reading .....	25
Fact Type Templating .....	123
fact type underlies atomic formulation .....	46
first-order instance .....	96
first-order type .....	96
form of expression .....	24
form of expression demonstrates designation .....	25
form of expression has placeholder .....	25
form of expression incorporates fact symbol .....	131
form of expression is in namespace .....	27
form of expression is included in vocabulary .....	113
Formal Logic & Mathematics Vocabulary .....	94
Formal Logic & Mathematics Vocabulary .....	11
formal model .....	97
formal representation .....	133
fundamental concept .....	127

## G

general concept .....	115
general concept has categorization scheme .....	116
general concept has categorization type .....	117
gerund form .....	26
guidance statement .....	143

## I

icon .....	130
implication .....	97
implication has antecedent .....	53
implication has consequent .....	53
implication has logical operand 1 .....	53
implication has logical operand 2 .....	53
impossibility business rule statement .....	145
inconsequent .....	50
individual concept .....	17
Infinity .....	150
informal representation .....	132
information source .....	135
instance .....	34
Instantiation Class .....	153
instantiation formulation .....	46
instantiation formulation binds to bindable target .....	47
instantiation formulation considers concept .....	47
instantiation prefix .....	148
instantiation prefix is used for language .....	148
integer .....	36
integer .....	97

Integer Class .....	153
Integer Namespace .....	12
Integer Type .....	150
integer1 < integer2 .....	36
integer1 is less than integer2 .....	36
integer2 > integer1 .....	36
integer2 is greater than integer1 .....	36
intensional definition .....	118
is-category-of fact type .....	125
is-facet-of fact type .....	126
ISO 1087-1 (English) .....	12
ISO 639-2 (Alpha-3 Code) .....	12
ISO 639-2 (English) .....	12
is-property-of fact type .....	124
is-role-of fact type .....	126

## L

language expresses vocabulary .....	113
language is supported by vocabulary namespace .....	29
level of enforcement .....	139
logical formulation.....	41
logical formulation constrains projection .....	68
logical formulation is embedded in modal formulation.....	48
logical formulation kind .....	41
Logical Formulation of Semantics Vocabulary .....	11
Logical Formulation of Semantics Vocabulary .....	39
logical formulation restricts quantification .....	57
logical necessity .....	19
logical negation .....	54
logical negation has logical operand .....	54
logical negation has negand .....	54
logical operand 1 .....	49
logical operand 2 .....	49
logical operand .....	48
logical operation .....	51
logical operation has logical operand .....	52
logical possibility .....	20
logical variable .....	97
lower bound .....	149

## M

material equivalence .....	52
material implication .....	53
mathematical form .....	25
maximum cardinality .....	58
meaning .....	15
Meaning and Representation Vocabulary .....	11
Meaning and Representation Vocabulary .....	14
meaning corresponds to thing .....	34



meaning is formulated by closed semantic formulation .....	40
meaning is represented by representation .....	22
member.....	97
message content .....	134
message content .....	134
minimum cardinality .....	58
modal formulation claims modality .....	48
modal formulation embeds logical formulation .....	48
modal logic.....	97
modality .....	19
modality is claimed by modal formulation .....	48
more general concept .....	116

## N

name .....	130
name .....	149
name references thing .....	131
named element .....	149
named element has name .....	149
named element has XML name .....	151
namespace .....	27
namespace contains designation .....	27
namespace contains form of expression .....	27
namespace has URI .....	27
nand formulation .....	54
nand formulation has logical operand 1 .....	54
nand formulation has logical operand 2 .....	54
necessity.....	19
necessity .....	97
necessity business rule statement .....	145
necessity claim .....	48
negand.....	50
nominal restrictive form .....	26
nonnegative integer .....	36
nor formulation.....	54
nor formulation has logical operand 1 .....	54
nor formulation has logical operand 2 .....	54
note .....	134
note comments on concept .....	133
noun concept .....	15
noun concept formulation .....	63
noun form .....	25
numeric range quantification .....	59
numeric range quantification has maximum cardinality .....	60
numeric range quantification has minimum cardinality.....	60

## O

object type .....	15
objectification .....	61
objectification binds to bindable target.....	61
objectification considers logical formulation.....	61
obligation .....	20
obligation .....	98

obligation claim .....	48
obligation statement .....	144
operative business rule .....	139
operative business rule has level of enforcement .....	140
operative business rule statement .....	143
owned attribute .....	150
owned definition .....	119
owned type .....	151

## P

package .....	151
package comes from vocabulary namespace .....	148
package has owned type .....	151
package has XML namespace prefix .....	151
package has XML namespace URI .....	151
partitioning .....	116
partitive fact type .....	124
permissibility .....	20
permissibility claim.....	48
permission .....	98
placeholder .....	26
placeholder has starting character position .....	26
placeholder is at starting character position .....	26
placeholder is in form of expression .....	25
placeholder uses designation .....	27
population .....	98
positive integer .....	36
possibility .....	20
possibility .....	98
possibility claim.....	48
predicate.....	98
preferred symbol .....	131
primitive type .....	150
prohibited symbol .....	131
prohibition .....	98
prohibitive statement .....	144
projecting formulation .....	62
projecting formulation binds to bindable target .....	62
projecting formulation has projection .....	62
projection.....	67
projection has auxiliary variable .....	68
projection is on variable .....	68
projection position .....	68
property .....	149
proposition .....	19
proposition .....	99
proposition has statement .....	23
proposition is expressed by statement .....	23
proposition is meant by closed logical formulation .....	42
proposition nominalization .....	64

proposition nominalization binds to bindable target .....	65
proposition nominalization considers logical formulation .....	65
propositional operator .....	99

## Q

quantification .....	56
quantification introduces variable .....	57
quantification scopes over logical formulation .....	57
quantifier .....	99
question .....	20
question nominalization .....	65

## R

Real-world Numerical Correspondence .....	114
reference .....	134
reference points to information source .....	135
reference scheme .....	29
reference scheme extensionally uses role .....	30
reference scheme is for concept .....	30
reference scheme simply uses role .....	30
reference scheme uses characteristic .....	30
reference supports concept .....	134
Referent Class .....	153
remark .....	134
representation .....	22
representation has expression.....	22
representation maps to UML element .....	148
representation represents meaning .....	22
representation uses symbol .....	130
res .....	131
restricted higher-order instance .....	99
restricted higher-order type .....	99
restricted permissive statement .....	144
restricted possibility business rule statement .....	145
role .....	16
role binding .....	46
role binding binds to bindable target .....	46
role binding is of role .....	46
role binding occurs in atomic formulation .....	45
role is extensionally used by reference scheme .....	30
role is filled by thing in actuality .....	34
role is simply used by reference scheme .....	30
role namespace is for concept.....	28
role namespace is within vocabulary namespace .....	28
rule .....	139
rule statement .....	143

## S

segmentation .....	116
semantic community .....	111
semantic community shares understanding of concept.....	112
semantic community understands body of shared meanings .....	111

semantic formulation .....	40
semantic formulation includes variable without binding.....	44
set.....	36
set .....	99
set includes thing .....	36
set projection .....	69
signifier .....	20
situation .....	125
source vocabulary .....	119
specialization fact type .....	124
speech community .....	111
speech community adopts adopted definition from source vocabulary .....	119
speech community is of semantic community .....	111
speech community is targeted by vocabulary .....	113
speech community owns owned definition .....	119
speech community owns symbol .....	129
speech community owns vocabulary.....	113
speech community regulates its usage of signifier .....	132
speech community uses vocabulary .....	113
starting character position .....	21
state of affairs .....	100
state of affairs .....	33
statement .....	23
statement expresses proposition .....	23
statement is formalized by closed logical formulation .....	42
string .....	150
String Type .....	151
structural business rule .....	139
structural business rule statement .....	143
structural rule .....	139
structural rule statement .....	143
subcommunity .....	111
subcommunity is of community .....	111
subset .....	100
superclass .....	150
symbol .....	128
symbol context .....	129
symbol context is of symbol .....	129
symbol context qualifies understanding of symbol .....	129
symbol is included in vocabulary .....	113
symbol is owned by speech community .....	129
symbol is understood anywhere within symbol context .....	129
symbol is used in representation .....	130
symbol realizes designation .....	129

## T

term .....	130
term denotes thing .....	131
term is implicitly understood .....	120
text .....	21

Text Class .....	154
text is for placeholder .....	148
thing .....	35
Thing Class .....	153
thing fills role in actuality .....	34
thing has name .....	131
thing is conceptualized as meaning .....	34
thing is denoted by term .....	131
thing is in set .....	36
thing1 = thing2 .....	36
thing1 equals thing2 .....	36
thing1 is equal to thing2 .....	36
thing1 is thing2 .....	36
type .....	100
type .....	149
typed element .....	149
typed element has type .....	149

## U

UML 2 Infrastructure .....	12
UML element .....	149
UML element comes from representation .....	148
unary fact type .....	16
unbound variable .....	100
Unicode Glossary .....	12
uniform resource identifier .....	21
Uniform Resource Identifiers Vocabulary .....	12
universal quantification .....	57
unlimited natural number .....	150
upper bound.....	149
URI .....	21

## V

variable .....	43
variable has projection position .....	68
variable is free within semantic formulation .....	44
variable is unitary .....	43
variable maps to role .....	70
variable ranges over concept.....	43
verb concept .....	16
viewpoint .....	125
vocabulary .....	112
Vocabulary for Describing Business Rules .....	11
Vocabulary for Describing Business Rules .....	137
Vocabulary for Describing Business Vocabularies .....	109
Vocabulary for Describing Business Vocabularies .....	11
vocabulary includes form of expression .....	113
vocabulary includes symbol .....	113
vocabulary is expressed in language .....	113
vocabulary is owned by speech community .....	113
vocabulary is used by speech community .....	113
vocabulary namespace .....	28
vocabulary namespace includes role namespace .....	28

vocabulary namespace is for language .....	29
vocabulary namespace maps to package .....	148
Vocabulary Registration Vocabulary .....	11
vocabulary targets speech community .....	113
Vocabulary to MOF/XMI Vocabulary .....	147
vocabulary uses language .....	113
vocabulary1 incorporates vocabulary2 .....	113
vocabulary2 is incorporated into vocabulary1 .....	113
Vocabulary-to-MOF/XMI Mapping Rule Set .....	12
Vocabulary-to-MOF/XMI Mapping Rule Set .....	154
Vocabulary-to-MOF/XMI Vocabulary .....	11

## W

wff .....	100
whether-or-not formulation .....	54
whether-or-not formulation has consequent .....	55
whether-or-not formulation has inconsequent .....	55
whether-or-not formulation has logical operand 1 .....	55
whether-or-not formulation has logical operand 2 .....	55
world .....	101

## X

XMI 2.1 Tags .....	12
XMI name .....	151
XMI name is derived from text .....	148
XMI namespace prefix .....	151
XMI namespace URI .....	151

# 15 Supporting Documents

The following documents accompany this specification. Some indicated documents are not immediately available but will be provided later or generated during finalization.

## 15.1 SBVR Metamodel

### 15.1.1 SBVR Metamodel Document

The SBVR Metamodel is created from the [Logical Formulation of Semantics Vocabulary](#) (which includes the [Meaning and Representation Vocabulary](#)) combined with the [Vocabulary for Describing Business Rules](#) (which includes the [Vocabulary for Describing Business Vocabulary](#)). It is a MOF model created following the [Vocabulary-to-MOF/XMI Mapping Rule Set](#). The metamodel is provided in the form of an XML Document based on the UML 2 Infrastructure XML Schema.

To be provided at a later time.

### 15.1.2 SBVR XML Schema

An XMI 2.1 based XML Schema derived from the SBVR Metamodel

document bei/2005-08-02: SBVR.xsd

Contact: Don Baisley

### 15.1.3 Logical Formulation of Semantics

#### 15.1.3.1 Meaning and Representation XML Document

The formal content of section is provided as an XML document based on the SBVR XML Schema.

To be provided at a later time.

#### 15.1.3.2 Logical Formulation of Semantics XML Document

The formal content of section is provided as an XML document based on the SBVR XML Schema.

To be provided at a later time.

#### 15.1.3.3 Logical Formulation of Semantics MOF

A MOF model is created from the [Logical Formulation of Semantics Vocabulary](#) (which includes the [Meaning and Representation Vocabulary](#)) following the [Vocabulary-to-MOF/XMI Mapping Rule Set](#). The metamodel is provided in the form of an XML Document based on the UML 2 Infrastructure XML Schema.

To be provided at a later time.

#### 15.1.3.4 Logical Formulation of Semantics XML Schema

An XMI 2.1 based XML Schema derived from the Logical Formulation of Semantics MOF model

document bei/2005-08-02: LFSV.xsd

Contact: Don Baisley

## 15.2 Business Vocabulary

### 15.2.1 Business Vocabulary XML Document

The formal content of section 2.5 Business Vocabulary, which describes the [Vocabulary for Describing Business Vocabulary](#), is provided as an XML document based on the SBVR XML Schema.

To be provided at a later time.

### 15.2.2 Describing Business Vocabulary MOF

A MOF model is created from the [Vocabulary for Describing Business Vocabulary](#) (which includes the [Meaning and Representation Vocabulary](#)) following the [Vocabulary-to-MOF/XMI Mapping Rule Set](#). The metamodel is provided in the form of an XML Document based on the UML 2 Infrastructure XML Schema.

To be provided at a later time.

### 15.2.3 Describing Business Vocabulary XML Schema

An XMI 2.1 based XML Schema derived from the Business Vocabulary MOF model

document bei/2005-08-02: DBV.xsd

Contact: Don Baisley

## 15.3 Business Vocabulary and Rules

### 15.3.1 Business Rules XML Document

The formal content of section 2.6 Business Rules, which describes the [Vocabulary for Describing Business Rules](#), is provided as an XML document based on the SBVR XML Schema.

To be provided at a later time.

#### 15.3.1.1 Describing Business Vocabulary and Rules MOF

A MOF model is created from the [Vocabulary for Describing Business Rules](#) (which includes the [Vocabulary for Describing Business Vocabulary](#)) following the [Vocabulary-to-MOF/XMI Mapping Rule Set](#). The metamodel is provided in the form of an XML Document based on the UML 2 Infrastructure XML Schema.

To be provided at a later time.

#### 15.3.1.2 Describing Business Vocabulary and Rules XML Schema

An XMI 2.1 based XML Schema derived from the Business Vocabulary and Rules MOF model



document bei/2005-08-02: DBVR.xsd

Contact: Don Baisley

## **15.4 Vocabulary-to-MOF/XMI**

### **15.4.1 Vocabulary-to-MOF/XMI Vocabulary and Mapping Rule Set XML Document**

The [Vocabulary-to-MOF/XMI Vocabulary](#) and the [Vocabulary-to-MOF/XMI Mapping Rule Set](#) as an XML document based on the SBVR Logical Formulation of Semantics XML Schema.

To be provided at a later time.

### **15.4.2 Vocabulary-to-MOF/XMI MOF**

A MOF model created from the [Vocabulary-to-MOF/XMI Vocabulary](#) in the form of an XML Document based on the UML 2 Infrastructure XML Schema

To be provided at a later time.

### **15.4.3 Vocabulary-to-MOF/XMI XML Schema**

An XMI 2.1 based XML Schema derived from the Vocabulary-to-MOF/XMI MOF document bei/2005-08-02: VocabularyToMOFXMI.xsd

Contact: Don Baisley

### **15.4.4 Essential SBVR MOF**

The [Essential SBVR Package](#) in the form of an XML Document based on the UML 2 Infrastructure XML Schema

To be provided at a later time.

### **15.4.5 Essential SBVR XML Schema**

An XMI 2.1 based XML Schema derived from the Essential SBVR MOF model document bei/2005-08-02: EssentialSBVR.xsd

Contact: Don Baisley

## **15.5 XMI XML Schema**

The base XML Schema for XMI 2.1

document bei/2004-07-05: XMI.xsd

Contact: Don Baisley



## Part III - Annexes

This part contains the annexes, including:

A - Overview of the Approach

B - The Business Rules Approach

C - SBVR Structured English

D - SBVR Structured English Patterns

E - EU-Rent Example

F - The RuleSpeak<sup>®</sup> Business Rule Notation

G - Concept Diagram Graphic Notation

H - Use of UML Notation in a Business Context to Represent SBVR-style Vocabularies

I - The ORM Notation for Verbalizing Facts and Business Rules

J - ORM Examples Related to the Logical Foundations for SBVR

K - Design Rationale Details for the Use of MOF and XMI

L - Examples of SBVR's Use of MOF

M - Mappings and Relationships to Other Initiatives

N - Additional References

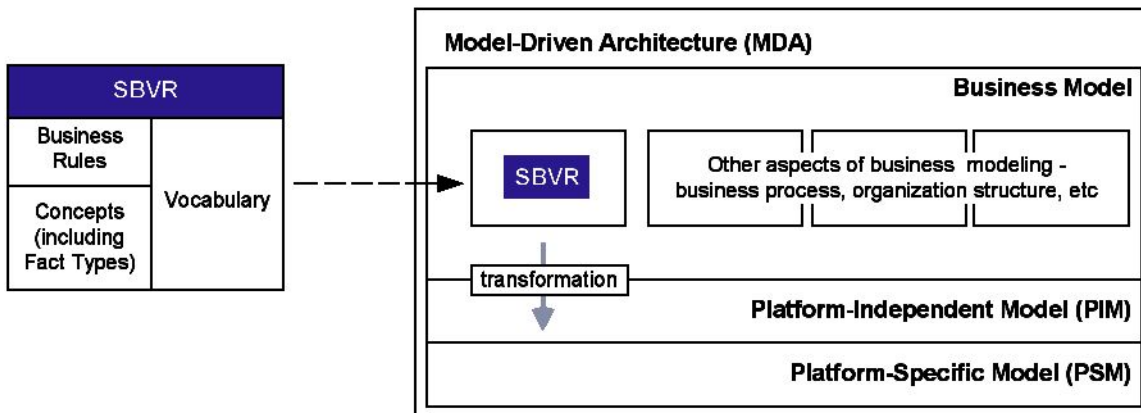


## Annex A (informative)

### Overview of the Approach

#### A.1 Positioning of SBVR in Model-Driven Architecture

SBVR is positioned to be entirely within the business model layer of the OMG's Model Driven Architecture (MDA)<sup>1</sup>.



This positioning has two implications.

- SBVR is targeted at business rules and business vocabularies, including those relevant for usage in conjunction with those rules. Other aspects of business models also have to be developed, including business process and organization structure, but these are to be addressed by the OMG in other initiatives.
- Business models, including the models that SBVR supports, describe businesses and not the IT systems that support them.

In MDA, IT systems are specified using Platform Independent Models (PIMs) and Platform-Specific Models (PSMs). Guidance will be needed for transformation of business models to PIMs. Such guidance is outside the scope of SBVR. It is anticipated that the OMG will ensure that the metamodels for different aspects of business modeling form a coherent whole, and will call for development of guidance on the transformation from business model to PIM as appropriate.

---

1. SBVR enables the specific capture of terminology and meaning for any level of the MDA, so SBVR could be used for PIM and PSM vocabulary and rules. However, this specification is focused on SBVR as a vehicle for describing businesses rather than their information systems. In the kinds of SBVR model assumed here, the concept called "customer" would be a role of a real-world person or organization. In a PIM, it would be a UML class whose objects represent real-world customers; the business rule "a rental car must not be handed over to a customer who appears to be intoxicated" would probably not appear in a PIM.

## A.2 The Key Notions of the SBVR Approach

### A.2.1 What is Semantics?

‘Semantics’ is “the meaning or relationship of meanings of a sign or set of signs” [MWCD]. In SBVR the signs can be of any form: words, phrases, codes, numbers, icons, sounds, etc. SBVR includes two specialized vocabularies:

- the SBVR “Vocabulary for Describing Business Vocabularies”, which deals with all kinds of terms and meanings (other than meanings of Business Rules);
- the SBVR “Vocabulary for Describing Business Rules”, which deals with the specification of the meaning of business rules, and builds on the “Vocabulary for Describing Business Vocabularies”.

The two have been separated so that the “Vocabulary for Describing Business Vocabularies” could be used independently - for example, as a basis for vocabularies for business processes or organizational roles.

The next two sections deal with the semantics of business vocabularies and the semantics of business rules.

### A.2.2 What is a Business Vocabulary?

A business vocabulary contains all the specialized terms and definitions of concepts that a given organization or community uses in their talking and writing in the course of doing business.

The SBVR “Vocabulary for Describing Business Vocabularies” is based on the ISO terminology standards:

- ISO 1087-1 (2000) “Terminology work — Vocabulary — Theory and application” [ISO1087-1]
- ISO 704 (2000) “Terminology work — Principles and methods” [ISO704]
- ISO 860 (1996) “Terminology work – Harmonization of concepts and terms” [ISO860]

These standards have been used for many decades for multilingual vocabularies in support of language translation work. SBVR is the result of the integration of these ISO standards, formal logics, linguistics and practical experience from foremost practitioners in the field of business vocabulary for business rules. They have over ten years experience in the development and application of the applied techniques included in the SBVR approach.

There are additional ISO standards for representing basic concepts such as country names and codes (ISO/IEC 3166), dates and times (ISO/IEC 8601), currency codes (ISO/IEC 4217), addresses (ISO/IEC 11180), which are likely to be adopted into vocabularies using SBVR as a matter of practice, but have not been included in this specification.

An SBVR-based business vocabulary strengthens the semantics of ordinary business glossaries of terms and their definitions in several ways. It provides:

1. A powerful multi-dimensional, hierarchical categorization capability to organize concepts from general to specific such as those used by library/information scientists to index documents. This is often referred to as taxonomies or categorization schemes. The ability to define categories is also included.
2. The capabilities associated with Thesauri including synonyms, abbreviations, ‘see also’, multiple vocabularies for one set of meanings for different languages, etc. The function of the ISO 2788:1986 Monolingual and ISO 5964:1985 Multi-Lingual Thesaurus standards is included in SBVR-based business vocabularies.
3. The ability to specify definitions (both intensional and extensional) formally and unambiguously in terms of other definitions in the business vocabulary as a result of its formal logics and linguistic underpinning.

4. The ability to define connections between concepts that are of interest to the organization. These connections provide the business-level semantic structure required to find information about such relationships in text documents and relational databases, as well as providing the ability to specify business rules formally and unambiguously. The function in the ISO/IEC 13250:2000 “Topic Maps” standard is included in SBVR-based business vocabularies.
5. A semantically rich set of templates to facilitate capturing the full semantics of each concept and connection between concepts of interest to the business community owning the business vocabulary.
6. A basis for identification and/or definition of individual entities, events and states, the relationships among them, and their relationship to time for text document and data mining.
7. The basis for tools that can support powerful visualization and ‘navigation’ of business vocabulary based on business meaning.
8. Business community ownership and management of their independent business vocabularies and business rules.
9. The basis to integrate separately created business vocabularies, using the ‘characteristic analysis’ capability from ISO 1087-1 and ISO 860. When separate business vocabularies are integrated and the business rules based on them are modified to reflect the vocabulary integration, the business rules will also be integrated.
10. The ability to minimize the number of definitions an organization needs to create by providing powerful, pragmatic features for vocabulary adoption on a well-managed basis. The SBVR approach encourages (a) incorporation of ready-made ‘outside’ vocabularies and (b) communication between people in different communities.
11. A comprehensively integrated capability to support the specification of the meaning of all kinds of business rules.

### A.2.3 What is a Business Rule?

The SBVR follows a common-sense definition of ‘business rule’:

Business Rule: *rule that is under business jurisdiction*

‘Under business jurisdiction’ is taken to mean that the business can enact, revise and discontinue business rules as it sees fit. If a rule is not under business jurisdiction in that sense, then it is not a business rule. For example, the ‘law’ of gravity is obviously not a business rule. Neither are the ‘rules’ of mathematics.

The more fundamental question in defining ‘business rule’ is the meaning of ‘rule’. Careful consideration was given to a variety of real-world interpretations of ‘rule’, including numerous authoritative dictionaries and previously-published works on business rules. Foremost consideration was given to how people think naturally about ‘rule’ in everyday life, not only within business activities, but also outside of them. For example, several rule books for professional sports were reviewed.

Clearly, ‘rule’ carries the sense of ‘guide for conduct or action’ both in everyday life and in business. In one way or another, this sense of ‘rule’ can be found in most, if not all, authoritative dictionaries.

Examining the question more closely, it is obvious that if rules are to serve as guides for conduct or action, they must also provide the actual criteria for judging and guiding that conduct or action. In other words, for the context of business rules (and probably in most other contexts), rules serve as *criteria* for making decisions. The SBVR’s interpretation of ‘rule’ therefore encompasses the sense of ‘criteria’ as given by authoritative dictionaries.

This point is fundamentally important for professionals creating business models. In business process engineering, for example, the most prevalent understanding of ‘business rule’ is as criteria for decision points (‘branch points’) in business process models. Often such decision points are relatively simple -- for example, “do we treat a customer as gold level, silver level or bronze level?” In other cases, such decision points may be highly complex -- for example, “should an insurance claim be paid, denied or considered as possibly fraudulent?”. For these more complex cases in particular, special inferencing techniques are quite likely to be helpful -- for example, tools supporting ‘production *rules*’.

### A.2.3.1 Rules and Formal Logic

An additional and no less important driver in the SBVR's treatment of 'rule' is consistency with formal logics. Notable experts in this area recommended that the best treatment for the SBVR's interpretation of rules would involve *obligation* and *necessity* claims.

Consequently, in SBVR, a Rule is "an element of guidance that introduces an obligation or a necessity". The two fundamental categories of Rule are:

- **Structural Rule** (necessities): These are rules about how the business chooses to organize (i.e., 'structure') the things it deals with. Structural Rules supplement definitions. For example (from EU-Rent):

Necessity: A Customer has at least one of the following:

- a Rental Reservation.
  - an in-progress Rental.
  - a Rental completed in the past 5 years.
- **Operative Rules** (obligations): These are rules that govern the conduct of business activity. In contrast to Structural Rules, Operative Rules are ones that can be *directly* violated by people involved in the affairs of the business. For example (from EU-Rent):

Obligation: A Customer who appears intoxicated or drugged must not be given possession of a Rental Car.

### A.2.3.2 Rules, Fact Types and Concepts expressed by Terms

Informally, a fact type is an association<sup>2</sup> between two or more concepts; for example "Rental Car is located at Branch."

In SBVR, rules are always constructed by applying necessity or obligation to fact types. For example, the rule "A Rental must not have more than three Additional Drivers" is based on the fact type "Rental has Additional Driver."

By this means, SBVR realizes a core principle of the Business Rules Approach at the business level, which is that "Business rules build on fact types, and fact types build on concepts as expressed by terms." This notion is well-documented in published material by foremost industry experts over the past 10 years.

The Business Rules Approach is summarized in Annex B.

One important consequence of the SBVR's approach in this regard is that concepts (including fact types) are *distinct* from rules, which are in a separate Compliance Point. This design permits SBVR's support for concepts (including fact types) to be optionally used on its own for building business vocabularies.

### A.2.3.3 Additional Comments about Business Rules

All business rules need to be *actionable*. This means that a person who knows about a business rule could observe a relevant situation (including his or her own behavior) and decide directly whether or not the business was complying with the rule. This assumes, of course, that the business vocabulary on which the rule is based has been adequately developed, and has been made available in some appropriate manner. This points toward the essential role of business vocabulary in supporting business rules; indeed, the bulk of SBVR is devoted to that area.

Just because business rules are actionable, this does *not* imply they are always automatable. Many business rules, especially operative business rules, are *not* automatable in IT systems. For instance, consider the obligation example given above.

---

2. "Association" is used here in its everyday, business sense - not the narrower, technical sense that would apply to a UML class model.



This distinction is not important within SBVR, which focuses on rules only from the business perspective, regardless of whether the rules could be automated. However, it is obviously important in defining a transformation from business model to PIM. In particular, non-automatable business rules need to be implemented as user activity, supported by procedure manuals or rulebooks.

#### **A.2.4 What is Semantic Interchange?**

The SBVR Metamodel is intended to provide for standardized data interfaces and data interchange among tools that collect, organize, analyze and use business vocabularies and rules, as well as tools that bind business vocabularies and rules to other models and implementations. The SBVR Metamodel will eventually facilitate many tools from various vendors for validation, analysis, alignment, merging and composition of business rules (including tools that can support explanations regarding why certain rules were deemed to conflict or overlap with one another) and for exchange of business vocabularies and rules along with their semantics.

An important feature of the SBVR Metamodel is how it is created. It starts with the SBVR Vocabularies. SBVR's Vocabulary-to-MOF/XMI Rule Set governs how a business vocabulary is mapped to a MOF 2 model. An XML Schema is then generated based on XMI 2.1.

The resulting SBVR Metamodel is intended, not for business people, but for software engineers that build tools for business people. The SBVR metamodel is includable and extendable in models that address various business domains. That the SBVR Metamodel is generated without manual intervention guarantees that it accurately represents the concepts of the SBVR Vocabularies.

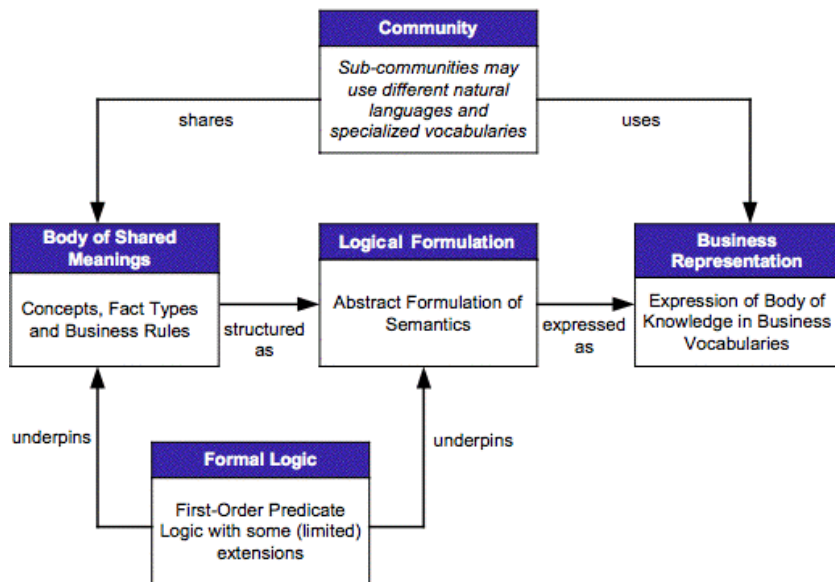
The rules that govern generation of the SBVR Metamodel apply a fact-oriented approach, which provides important advantages for business-level interchange:

1. Fine control over exactly what is communicated to the level of individual facts.
2. Communication of facts about facts.
3. Support for multidimensional categorization.
4. Support for things changing over time, such as a thing with one identity being reclassified over time.
5. Communication for many purposes that cannot be predicted.
6. Extensibility and reuse in other business vocabularies.

The BRT is deeply interested in interoperability of modeling tools and in integration of many kinds of models. These models range from business mission and vision to business vocabulary, rules and processes; to IT models of components and databases; and to models of system deployment and administration. The SBVR Metamodel supports the broad requirements for integration and traceability, and is consistent with the goals of the knowledge representation community.

## A.3 Informal Overview of SBVR

SBVR can be viewed as having five major aspects, as illustrated below:



### A.3.1 Community

The basis for business vocabulary is community. At the business level, communities of primary importance are enterprises for which business rules are being established and expressed. However, other communities - the industry in which an enterprise operates, partner enterprises, standards groups, regulatory authorities, etc. - also need to be recognized.

An important aspect of community is that sub-communities within an enterprise may need its body of shared meanings (starting with fundamental concepts) to be expressed in different vocabularies, ranging from specialized jargon to different natural languages. In SBVR, such sub-communities are called “speech communities”.

### A.3.2 Body of Shared Meanings

A community has a body of shared meanings, comprising concepts (which include fact types) and business rules. What is shared is the meaning, not the form of expression. Clearly, for shared meanings to be exchanged, discussed and validated, they must be expressed. But SBVR separates the business meaning from any particular form of expression. The structure of the body of shared meanings (i.e., which concepts play which roles in facts, which facts form the basis of which rules etc.) is defined by associating abstract concepts, fact types and business rules, not by associating statements in any given language.

### A.3.3 Logical Formulation

Logical formulation provides a formal, abstract, language-independent syntax for capturing the semantics of a body of shared meanings. It supports multiple forms of representation, such as: noun and verb forms of expression, reading of associations in both directions.

Logical formulation supports two essential features of SBVR. First is the mapping of a body of shared meanings to vocabularies used by communities. Second is the mapping to XMI that enables interchange of concepts, facts and business rules between tools that support SBVR.

### **A.3.4 Business Representation**

The concepts and business rules in a body of shared meanings need to be represented in vocabularies acceptable to, and usable by, speech communities that share their meaning. These vocabularies may be in different natural languages, in artificial languages such as the UML, or in specialized subsets of natural languages, as used by, for example, engineers or lawyers.

SBVR supports mapping of business meaning to concrete language by associating elements of the body of shared meanings with signifiers, e.g., terms such as “customer”, “car”, “branch” for concepts, and fact symbols (often verb phrases) such as “rents”, “is located at” for fact types. Logical formulations provide the structure, and signifiers are placed in logical formulations to provide the expression.

SBVR supports adoption from external sources, such as standards bodies and industry groups. For example, SBVR itself adopts some of its basic definitions from ISO standards for terminology and vocabulary (ISO 1087-1 and ISO 704).

### **A.3.5 Formal Logic**

SBVR has a sound theoretical foundation of formal logic, underpinning both logical formulation and the structures of bodies of shared meanings. The base is first-order predicate logic (with some restricted extensions into higher-order logics), with some limited extensions into modal logic – notably some deontic forms, for expressing obligation and prohibition, and alethic forms for expressing necessities.

## **A.4 SBVR Beneficiaries**

A different perspective of SBVR is provided by considering the different groups of people who will benefit from it.

### **A.4.1 Business Analysts and Modelers**

Business analysts and modelers work in enterprises such as EU-Rent. Their business view is the enterprise business view, or perhaps a view of part of the business.

Their view of Community is generally the enterprise in which they work, and its Speech Communities. Within this, they are most concerned with building on the enterprise’s Body of Shared Meanings and Vocabulary in which to express it. They have to negotiate with the Integrators/Administrators (see next subsection) for inclusion of new concepts and business rules and new signifiers in the Vocabularies.

Business analysts and modelers need to specify business policies and rules precisely, but to do so they do not need any in-depth knowledge of SBVR’s Logical Formulation or Formal Logic. They will see the effects of these parts of SBVR in facilities provided by tools that support their enterprise’s business vocabularies and rules, e.g., templates, options, constraints, consistency checks.

### **A.4.2 Business Vocabulary+Rules Integrators/Administrators**

Business Vocabulary+Rules integrators/administrators generally work within enterprises. Their business view is maintaining a consistent enterprise-wide Body of Shared Meanings, plus Vocabularies for Speech Communities within the enterprise.

They are responsible for integrating and quality-assuring content provided by business analysts and modelers. An important part of this is deciding what to adopt from external vocabularies. They will also be responsible for maintaining the Business Vocabulary+Rules over time. This is outside the scope of SBVR; Business Rule Management is a separate issue to be addressed by the OMG as appropriate.

Integrators/administrators will generally be more aware than business analysts and modelers of Logical Formulation. However they do not need to understand it formally: they will see its effects in administration tools.

### A.4.3 Tool Builders

Two kinds of tool will be needed to support SBVR:

- For interchange of business vocabulary and rules between different platforms.
- For developing and maintaining business vocabulary and rules for a community.

Interchange standards (and tools that use them) are of great importance to the OMG. Compliance with MOF and XMI was mandated by the OMG, and its achievement is a major part of SBVR. Developers of interchange tools will have four major concerns:

- The types of construct in a Body of Shared Meanings – Concepts, Fact Types, Facts and Business Rules - and the types of relationship between them.
- The association of elements of the Body of Shared Meanings with elements of Vocabulary – terms, fact symbols, definitions, references to external sources.
- Logical Formulation.
- Mapping to MOF/XMI.

The developers will not be concerned with the content of Business Vocabulary+Rules for enterprises. And although tool architects and designers will need to understand the Formal Logic theory underpinning of SBVR, the developers will not (although it should be reassuring that it is there). For further discussion see Annex L.

Business analysts and modelers and integrators/administrators will need tools for developing and maintaining enterprise Business Vocabulary+Rules.

Development of such tools is not the direct concern of the OMG; they will be developed by vendors to meet market demand. However, it is important that they are developed – it would be futile to have good interchange standards and tools if nobody was developing worthwhile content for interchange.

Ensuring that the SBVR model will provide a sound basis for development and maintenance tools has been a judgment call by the BRT. Tools will need to support Body of Shared Meaning, Business Expression and Logical Formulation, plus multiple Communities and vocabulary adoption between them. Tool developers will also have to work with methodologists to ensure support of processes for development and integration of Business Vocabulary+Rules.

### A.4.4 Logicians, Semanticists and Linguists

Logicians, semanticists, and linguists provide the logical, mathematical, and linguistic capabilities that make it possible to transform business vocabularies and rules from the business perspective to PIM and PSM information systems designs, to structure a variety of natural language statements into SBVR constructs, and to verbalize SBVR entries into any number of natural language statements.

They design the algorithms to ensure integrity in Business Vocabulary+Rules interchange documents, and in the translation between interchange documents and internal tool designs. They also help ensure the formal logic, mathematic and linguistic integrity of the internal designs of Business Vocabulary+Rules tools.

## A.4.5 Summary of Audiences (Business Beneficiaries) by Activity and Business Context

Business Context (*excluding recordkeeping & information system activities*)

- Creating Business Content in a ‘Business Vocabulary+Rules’ (e.g., EU-Rent)  
*Audience:* Business People in General
- Integrating & Quality Assuring Business Content in a ‘Business Vocabulary+Rules’ (e.g., EU-Rent)  
*Audience:* ‘Business Vocabulary+Rules’ Integrator/Administrators

‘Business Vocabulary+Rules’ Technology and Tool Context

- Providing the Semantic and Logical Foundation for all ‘Business Vocabulary+Rules’  
*Audience:* Linguists, Semanticists and Logicians
- Designing a ‘Business Vocabulary+Rules’ Tool for Business People to Document Business Content (e.g., EU-Rent)  
*Audience:* Designers of vocabulary and rules software tools for business people
- Designing Tool capability to interchange Business Content in a ‘Business Vocabulary+Rules’ (e.g., EU-Rent) among Business Communities within and between Organizations  
*Audience:* Infrastructure Designers for Business Vocabulary and Rules Tools

Information System (Recordkeeping) Context (*Out of Scope for SBVR*)

- Designing Information Systems that Talk and Work according to the Business Content in a ‘Business Vocabulary+Rules’ (e.g., EU-Rent)  
*Audience:* Designers of information systems that support business vocabulary or automate business rules

## A.5 Technical Overview of the Approach

SBVR is designed to support interchange of business vocabularies and rules among organizations. SBVR is conceptualized optimally for business people and designed to be used for business purposes independent of information systems designs.

It is also intended to provide the business vocabulary and rules underpinned by First Order Predicate Logic for transformations by IT staff into information system designs. Note that, in most cases, such transformations will not be fully automated; there will be many options for information system design, with decisions required from system architects and PIM modelers.

### A.5.1 How SBVR is Underpinned by Formal Logics

The formal semantics of SBVR is based on the following formal approaches: typed predicate logic; arithmetic; set and bag comprehension (grounded in ur-elements), with some additional basic results from modal logic. The logic is essentially classical logic, so mapping to various logic-based tools should be straightforward. Typed logic is used for convenience but is easily translatable into untyped logic.

SBVR is neutral as to whether types may be instances of other types in the same model. We provide a basic formalization in first-order logic for those who wish to exclude higher-order types. We also provide an extended formalization for those who wish to allow higher-order types. The extended formalization uses a restricted version of higher-order logic that is closely related to Henkin semantics in restricting the range of types over which quantification is permitted. In first-order logic, quantification is permitted only over individuals (objects: lexical or non-lexical). The SBVR’s restricted higher-order formalization also allows quantification over at least one (one may choose either or both) of the following: object types that

are instances of a declared categorization type (whether or not these instances have been explicitly declared); object types (primitive or derived) that are explicitly declared in the schema.

It is well known that any function may be rewritten as an equivalent relation, and vice versa. For simplicity, SBVR treats all functions (including mathematical operations) as relations. Relations may be of any arity (1, 2, 3, etc.).

SBVR has no dependency on artificial identifiers (such as MOF ids, surrogate keys), so that all individuals are identified by definite descriptions that are ultimately grounded in lexical constants (note that this does not prevent businesses from using artificial identifiers within their specific SBVR models). Individual constants may be introduced by definition as a shorthand for definite descriptions. Unnamed structures are permitted. For example, sets may be identified by their extensions, and formulae may be identified by their structural composition. The avoidance of artificial identifiers ensures that business statements may be easily understood and communicated between businesses. This is not to discourage the use of names, which is highly recommended, but only to cater for cases where they are not supplied. This also does not prohibit the use of artificial identifiers by supporting tools, provided that such identifiers are hidden from business users of such tools.

Modal operators used include the alethic operators ‘It is necessary that’ and ‘It is possible that’, and the deontic operators ‘It is obligatory that’ and ‘It is permissible that’. Other modal operators are allowed at the surface level but are translated into these more basic operators with the help of negation (e.g., ‘It is forbidden that’ is captured internally as ‘It is obligatory that it is not the case that’). Apart from standard modal operator transformations involving negation, no other use is made of modal logic theorems, so there is no requirement to choose one out of many specific modal logics for a given modality.

The term ‘fact’ is used in the sense of epistemic commitment, but the underlying logic used for logical connectives is isomorphic to standard truth-functional logic rather than epistemic logic. Ultimately all ground facts are existential or elementary. The truth functional logic is two-valued, with negated existential formulae being used to avoid the use of null values.

## A.5.2 SBVR Inherent Extensibility

1. The SBVR Vocabularies given by this specification are themselves vocabulary that can be included in other business vocabularies. An extended SBVR vocabulary can be created by including an SBVR vocabulary into another business vocabulary that has other symbols. An extended SBVR vocabulary can, for example, provide for expression of additional information about symbols and rules that is not covered by this specification. An extended SBVR vocabulary can add new symbols (terms, names and sentential forms) for existing concepts as well as add new concepts along with symbols that represent them.
2. The SBVR Vocabularies given by this specification are based on the English language, but can be used to define vocabularies in any language. Alternative SBVR vocabularies based on a different language can be defined by providing symbols from the different language for the concepts represented in the SBVR Vocabularies.
3. The Vocabulary-to-MOF/XMI Mapping Rule Set provided with this specification can be applied to any extended SBVR vocabulary in order to produce a repository model and an XML schema that can extend the repository models and XML schemas, respectively, that are provided with this specification (see Annex L). XML documents formed according to that schema can accommodate facts expressible in the extended business vocabulary.
4. The SBVR Vocabularies are used to express rules in this specification concerning the definition of business vocabulary and formation of business rules. The SBVR Vocabularies can be further used to express other rules or to form expressions for other purposes. Such other rules can stipulate additional requirements concerning, among other things, what constitutes valid business vocabulary and what is allowable and required in the expression of rules. This specification describes how such rules, like other rules, are formally modeled and communicated and makes no requirement concerning enforcement of such additional rules.

Use of an SBVR vocabulary outside this specification (as in 1 through 4 above) does not change the SBVR vocabulary itself, but only uses it by way of reference.

### **A.5.3 MOF/XMI Models for SBVR**

A business vocabulary provides a means of recording and communicating facts. Following OMG's Model Driven Architecture, a business vocabulary developed as an information system independent model of business communication is used to drive the creation of a platform independent MOF model. The MOF model is, in turn, used to drive generation of Java interfaces (based on JMI) and an XML schema (based on XMI).

SBVR is mapped to MOF in two ways. First, the SBVR Vocabularies are mapped to a MOF model of repositories that can hold representations of facts that can be meant by any atomic formulation expressible using the business vocabulary. This first mapping does not capture the full SBVR with all of its semantics. It only maps the business vocabulary, using MOF as a mode of representation. The creation of this model is guided by mapping rules given in Part II.

Second, the full SBVR is captured in terms of the MOF model created from the SBVR Vocabularies (the first mapping). This includes the definitions of concepts, terms, business rules and other facts of the SBVR Metamodel that are expressed in terms of the SBVR Vocabularies.

The rules that guide the first mapping are general enough to be used for any vocabulary defined in terms of SBVR. A vocabulary for any business domain can be mapped to a MOF repository model by these same rules.

The MOF repository model mapped from the SBVR Vocabularies, which is the model used to capture the SBVR itself, is also used to capture business vocabularies and business rules in general. For additional detail on the design rationale, see Annex K.

## **A.6 Special Features of SBVR**

### **A.6.1 Coherent Business Example: EU-Rent**

It is valuable to have a common, consistent base for a large body of examples to illustrate the SBVR approach and use of the SBVR Metamodel. SBVR uses EU-Rent, a (fictitious) car rental company that has been used in several other R&D projects and publications, including papers published by the Business Rules Group. EU-Rent was also used as the basis for the *Business Rules Product Derby*, held at the Business Rules Forum in (New Orleans, 2002, Nashville, 2003, and Las Vegas, 2004), and as the common case study for vendors at the European Business Rules Conference (Zurich, 2003, and Amsterdam, 2004).

EU-Rent includes a broad range of concepts, facts and rules. Most readers of this specification should find the business requirements easy to understand. They should be able to move into the detail of the examples without having to spend much time on the general business scenario.

An important feature of EU-Rent is that it is an international business, which has requirements for expression in different natural languages, and for adaptation of some policies and rules to local regulation, custom and practice.

### **A.6.2 Internationalization**

Internationalization is handled from two directions. First, the meanings of concepts (including fact types) and rules within a body of shared meanings are modeled separately from how they are expressed. The same meaning can be expressed in different languages, both natural and artificial (such as UML and XML).

Second, communities who define concepts and set rules can be grouped and associated. An international company could, for example, define core concepts. Each of its regional divisions would adopt the core into its local body of shared meanings, which also addressed adaptation to local regulation, custom, and practice.

The resulting content could then be mapped into different languages. For example, global policies and rules could be expressed globally in a common language such as English, but operational detail mapped to as many languages as are needed. Communities can also adopt business vocabularies, so that the Swiss division could adopt business vocabularies developed

and maintained by the French, German and Italian divisions. SBVR uses “ISO 639-2 Codes for the Representation of Names of Languages” [ISO639-2] to specify the language used to express a given vocabulary (see Part II entry for ‘language’).

One issue still to be addressed in internationalization concerns adoption of business vocabularies from outside the business. Adoption of such business vocabularies, e.g., from trade associations or special interest groups, has two major advantages: it reduces the work needed to maintain the adopting company’s own vocabulary, and it eases communication with other organizations in the same business area. If such business vocabularies are adopted in different natural languages for the same meaning there is some risk of inconsistency in the mappings. The issue that needs further discussion is the trade-off between:

- Adopting an externally-defined vocabulary and supplementing it as needed
- Modifying an externally-defined vocabulary to create a new one and taking on the overhead of maintaining the modifications

The outcome is likely to be heuristics to be applied case by case, rather than a general recommendation one way or the other.

### A.6.3 Independence

**Rule Independence.** SBVR bases the expression of all business rules on structured business vocabularies. By doing so, business rules can be specified independently of all processes and events.

**Enforcement.** SBVR carefully segregates business rule specification from any aspect of enforcement.

**Methodology and Notation.** Although proven compatible with both existing notations and new innovative visualization techniques, SBVR is completely neutral with respect to methodology or notation, permitting the widest possible adoption.

### A.6.4 Notations for Business Vocabulary+Rules

#### A.6.4.1 Special Note on Notations

‘Notation’ is used in SBVR (as instructed by OMG) to mean any language used to represent semantics, or more precisely, abstract syntax. Notations can be verbal, graphical or any combination thereof. Other words for ‘notation’ are ‘grammar,’ ‘syntax,’ and ‘concrete surface syntax.’

It is specifically *not* the intention of SBVR to mandate any particular notation(s) that must or should be used with the SBVR Metamodel. Indeed, this would be neither productive nor desirable. Instead, SBVR encourages wide innovation, experimentation and value-adding software development in the area of compliant notations.

#### A.6.4.2 SBVR Structured English

It should be remembered that SBVR Structured English (presented in Annex C) is just one of possibly many notations that can be used to express the SBVR Metamodel, and, as a notation, is nonnormative in the SBVR standard. Indeed, additional compliant notations are welcomed and encouraged.

Compliant enrichments of various parts of SBVR Structured English itself are also welcomed and encouraged.

Two styles of SBVR Structured English are documented in this specification:

1. Prefixed Rule Keyword Style
2. Embedded (mixfix) Rule Keyword Style



The Prefix Style introduces rules by prefixing a statement with keywords that convey a modality. Examples of some of the prefixes are shown in the table below.

<b>Operative</b>	<b>Structural</b>
It is obligatory that	It is necessary that
It is prohibited that	It is impossible that
It is permitted that	It is possible that

This style, which is explained in Annex C, is included in this specification for two primary reasons:

- It is supported by the commercial reference implementation of Unisys Corporation, an implementation that satisfies the OMG submission’s compliance requirements.
- Its rule keywords correspond to the modal operators in the logical formulation portion of SBVR, so it illustrates the translation of notation to metamodel in the most direct and easy-to-understand fashion.

The Embedded Style features the use of rule keywords embedded (usually in front of verbs) within rules statements of appropriate kinds. Examples of some of the embedded keywords are shown in the table below.

<b>Operative</b>	<b>Structural</b>
... must ...	... always ...
... must not ...	... never ...
... may ...	... sometimes ...

This style of notation, which is introduced in Annex F and examined more closely in Annex I, is included in this specification for two primary reasons:

- It is an existing, documented notation<sup>3</sup> (RuleSpeak<sup>®</sup>, by Business Rule Solutions, LLC) that has been used with business people in actual practice for a number of years.
- It clearly demonstrates that alternative notations for business rules, which some business people find more natural and/or friendly, are easily accommodated under SBVR Structured English.

### **A.6.5 State**

‘State’ is an important notion for business vocabularies and business rules. As far as business people are concerned, ‘state’ is a concept they can refer to and use in creating definitions, facts, and rules. For example, in EU-Rent a car’s states would include: ‘available’, ‘allocated to rental’, ‘on rental,’ ‘damaged,’ and so on. The company uses these state names in defining business rules, e.g., “The car assigned to a walk-in rental must be the available car with the lowest odometer reading in the requested car group.” One way to express states is using unary predicates, e.g., “car is available”.

Businesses name only those states that are useful to them, and these may be only a small subset of the real-world states that real-world cars may have. For example, a car will, early in its EU-Rent life, have a state ‘just delivered and checked out, ready for its first rental’. But EU-Rent can decide that this has no practical difference from ‘returned from rental, cleaned and refueled’ and combine the two (with others, like ‘transferred in from another branch’) into a named state called ‘available.’

---

3. [Ross2003], Chapters 8-12

The SBVR approach to Business Vocabulary+Rules regards state as largely definitional ('available' is the concept we use for a car that is ...), unlike in a system design or implementation, where state handling is often about applying rules to data ("when a car is returned from a rental, its state must be set to 'available'"). And selection of the states that are useful to name and define is a business decision.

States are associated with other kinds of concept, including concepts that represent:

- things in the business (like cars and rentals).
- things happening in the business (like rental reservation, late return from rental).
- other states ("when a car is in state 'due for service' it cannot become 'available' again until it has been serviced -- i.e., been through the pattern of events that describe servicing").

'State' may need some further development; for example, *dynamic* models of events, cycles, schedules, etc. were considered to be outside the scope of SBVR. As SBVR is, states can be represented using concepts and fact types.

## Annex B (informative)

### The Business Rules Approach

SBVR provides a formal foundation for business rules. It also defines what they are. Much of the thinking in this area arose from the work of the Business Rules Group, which has been working exclusively in the area since the late 1980s.

Key notions of the business rules approach are presented succinctly by the BRG's *Business Rules Manifesto*. An extract from the Manifesto is presented below, to assist readers in positioning some of the central notions of SBVR. This brief extract is followed by a figure providing an overview of SBVR support.

A brief word on the BRG follows, along with citations to its work products. The full text of the Business Rules Manifesto<sup>1</sup> can be found in numerous languages at: <http://www.businessrulesgroup.org/brmanifesto.htm>.

#### B.1 Extract from the Business Rules Manifesto

**Primary Requirements, Not Secondary.** Rules are essential for, and a discrete part of, business models and technology models.

**Separate From Processes, Not Contained In Them.** Rules apply across processes and procedures. There should be one cohesive body of rules, enforced consistently across all relevant areas of business activity.

**Deliberate Knowledge, Not A By-Product.** Rules build on facts, and facts build on concepts as expressed by terms. Terms express business concepts; facts make assertions about these concepts; rules constrain and support these facts. Rules are basic to what the business knows about itself — that is, to basic business knowledge. Rules need to be nurtured, protected, and managed.

**Declarative, Not Procedural.** Rules should be expressed declaratively in natural-language sentences for the business audience. A rule is distinct from any enforcement defined for it. A rule and its enforcement are separate concerns.

**Well-Formed Expression, Not Ad Hoc.** Business rules should be expressed in such a way that they can be validated for correctness by business people. Business rules should be expressed in such a way that they can be verified against each other for consistency.

**For the Sake of the Business, Not Technology.** Rules are about business practice and guidance; therefore, rules are motivated by business goals and objectives and are shaped by various influences. The cost of rule enforcement must be balanced against business risks, and against business opportunities that might otherwise be lost.

**Of, By and For Business People, Not IT People.** Rules should arise from knowledgeable business people.

**Managing Business Logic, Not Hardware/Software Platforms.** Rules, and the ability to change them effectively, are fundamental to improving business adaptability.

---

1. [BRM].

## B.2 An Overview of SBVR Support for Key Business Rule Ideas

A core idea of business rules formally supported by SBVR is the following from the Manifesto: “Rules build on facts, and facts build on concepts as expressed by terms. Terms express business concepts; facts make assertions about these concepts; rules constrain and support these facts.”

This core idea, originating in the BRG’s seminal 1995 white paper [BRG2002], has been called the business rules “mantra”. It is often abbreviated for convenience to simply: “*Rules are based on facts, and facts are based on terms.*”

Figure B-1 provides an overview of how SBVR supports the “mantra.” It requires separation of viewpoints as follows.

**Business Rule “Mantra.”** An approximation that simplifies explanation for business people and others new to the approach.

**Representation (in SBVR terminology).** The SBVR notions that classify the words that people use to express their vocabulary+rules.

**Meaning (in SBVR terminology).** The SBVR notions that classify the underlying meaning of the words that people use in expressing their vocabulary+rules.

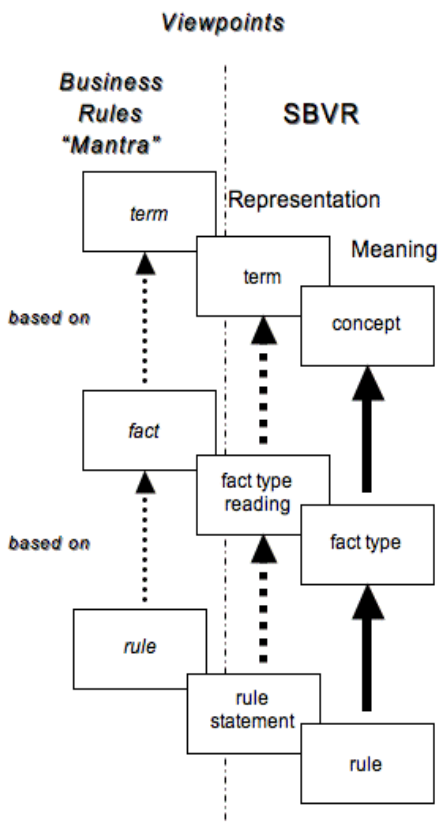


Figure B.1 - How SBVR Supports the Business Rules “Mantra”

## B.3 About the Business Rules Group (BRG<sup>2</sup>)

**Background.** Information Systems analysts have long been able to describe an enterprise in terms of the structure of the data the enterprise uses and the organization of the functions it performs. Unfortunately, there is often neglect of the rules (constraints and conditions) under which the enterprise operates.

Frequently these rules are not articulated until it is time to convert them into program code. While rules that are represented by the structure and functions of an enterprise have been documented to a degree, others have not been articulated well, if at all. The Business Rules Group was organized to carry out that articulation.

**The BRG Charter.** Originally a project within GUIDE International, the Business Rules Group has been an independent organization since the 1990s. Its membership comprises experienced practitioners in the field of systems and business analysis methodology who work in both the public and the private sectors.

The charter of the BRG is to formulate statements and supporting standards about the nature and structure of business rules, the relationship of business rules with the way an enterprise is organized, and the relationship of business rules with systems' architectures.

---

2. [BRJ2005]



## Annex C (informative)

### SBVR Structured English

The most common means of expressing definitions and business rules is through statements, not diagrams. While diagrams are helpful for seeing how concepts are related, they are impractical as a primary means of defining vocabulary and expressing business rules.

This specification defines an English vocabulary for describing vocabularies and stating rules. There are many different ways that this vocabulary and other English vocabularies described using SBVR can be combined with common English words and structures to express definitions and statements. However expressed, the semantics of definitions and rules can be formally represented in terms of the SBVR vocabulary and, particularly, in terms of logical formulations (the SBVR conceptualization of formal logic).

This annex describes one such way of using English that maps mechanically to SBVR concepts. It is not meant to offer all of the variety of common English, but rather, it uses a small number of English structures and common words to provide a simple and straightforward mapping.

All formal definitions and rules in this document that are part of ‘SBVR in terms of itself’ (or of the other vocabularies and rules, such as those for mapping to MOF and XMI) are stated using the SBVR Structured English. These statements can then be interpreted automatically in order to create MOF and/or XMI representations.

The description of the SBVR Structured English is divided into sections.

- Expressions in SBVR Structured English
- Describing a Vocabulary
- Vocabulary Entries
- Specifying a Rule Set
- Guidance Entries

#### C.1 Expressions in SBVR Structured English

This document contains numerous statements and definitions that represent corresponding logical formulations. These statements are recognized by being fully expressed using the fonts listed below. Note that these fonts are also used for individual designations in the context of ordinary, unformalized statements in order to note that defined concepts are being used.

There are four font styles with formal meaning:

term The ‘term’ font is used for a designation for a noun concept (other than an individual concept), one that is part of a vocabulary being used or defined (e.g., modality, modal formulation, fact type). This style is applied to the designation where it is defined and wherever it is used.

Terms are usually defined using lower case letters unless they include a proper noun. Terms are defined in singular form. Plural forms are implicitly available for use.

## Name

The ‘name’ font is used for a designation of an individual concept — a name. Names tend to be proper nouns (e.g., California). This style is applied to a name where it is defined and wherever it is used. Note that names of numerical values in formal statements are also shown in this style (e.g., 25). See the definition of ‘name’ for more details.

Names appear using appropriate capitalization, which is usually the first letter of each word, but not necessarily.

## verb

The ‘verb’ font is used for designations for fact types — usually a verb, preposition or combination thereof. Such a designation is defined in the context of a form of expression. This font is used both in the context of showing a form of expression (e.g.,

‘modal formulation claims modality’

and

‘modality is claimed by modal formulation’)

and in the context of using it in a statement (e.g.,

“Each modal formulation claims exactly one modality.”).

See the definition of ‘form of expression’ in Part II for more details.

Forms of expressions are defined using singular, active forms of verbs with the exception that gerund forms are sometimes defined for characteristics. Infinitive, plural and gerund forms of verbs are implicitly available for us.

## keyword

The ‘keyword’ font is used for linguistic symbols used to construct statements – the words that can be combined with other designations to form statements and definitions (e.g., ‘each’ and ‘it is obligatory that’). Key words and phrases are listed below.

Quotation marks are also in the ‘keyword’ font. The text within quotes is in ordinary font if the meaning of the quotation is uninterpreted text. The text within quotes is in styled text if the meaning of the quotation is formally represented. Single quotation marks are used to quote a designation or form of expression that is being mentioned. If a designation is mentioned (where the designation is itself the subject of a statement) it appears within single quote marks (e.g., ‘modality’ and ‘California’ used to talk about those designations). Single quotes are also used around a form of expression that is being mentioned (e.g., ‘modal formulation claims modality’ used to talk about that form of expression). Double quotation marks are used in other cases, such as to quote a statement.

Single quotation marks are also used to mention a concept – to refer to the concept itself rather than to the things it denotes. In this case, a quoted designation or form of expression is preceded by the word ‘concept’ or by a term for a kind of concept. For example, the statement,

“The concept ‘quantification’ is a category of the concept ‘logical formulation’”,

refers to the named concepts, not to quantifications and logical formulations. A role can be named with respect to a fact type in this same way (e.g.,

The role ‘modality’ of the fact type ‘modal formulation claims modality’).

Periods also appear in the ‘keyword’ font. A period is used to terminate a statement, but not a definition. Other punctuation symbols (e.g., parentheses, comma) also apply the ‘keyword’ font when part of a formal expression.

The SBVR Structured English uses designations and forms of expressions exactly as they are defined in a vocabulary. Plural forms are not used. For example, a formal statement would say “each concept” rather than “all concepts.” Both the active form and the passive form of a verb need to be defined in a vocabulary if both are used.



## C.1.1 Key words and phrases for logical formulations

Key words and phrases are shown below for expressing each kind of logical formulation. The letters ' $n$ ' and ' $m$ ' represent use of a literal whole number. The letters ' $p$ ' and ' $q$ ' represent expressions of propositions.

### C.1.1.1 Quantification

each	<a href="#">universal quantification</a>
some	<a href="#">existential quantification</a>
at least one	<a href="#">existential quantification</a>
at least $n$	<a href="#">at-least-<math>n</math> quantification</a>
at most one	<a href="#">at-most-one quantification</a>
at most $n$	<a href="#">at-most-<math>n</math> quantification</a>
exactly one	<a href="#">exactly-one quantification</a>
exactly $n$	<a href="#">exactly-<math>n</math> quantification</a>
at least $n$ and at most $m$	<a href="#">numeric range quantification</a>
more than one	<a href="#">at-least-<math>n</math> quantification</a> with $n = 2$

### C.1.1.2 Logical Operations

it is not the case that $p$	<a href="#">logical negation</a>
$p$ and $q$	<a href="#">conjunction</a>
$p$ or $q$	<a href="#">disjunction</a>
$p$ or $q$ but not both	<a href="#">exclusive disjunction</a>
if $p$ then $q$	<a href="#">implication</a>
$q$ if $p$	<a href="#">implication</a>
$p$ if and only if $q$	<a href="#">equivalence</a>
not both $p$ and $q$	<a href="#">nand formulation</a>
neither $p$ nor $q$	<a href="#">nor formulation</a>
$p$ whether or not $q$	<a href="#">whether-or-not formulation</a>

Where a subject is repeated when using 'and' or 'or', the repeated subject can be elided. For example, the statement, "An implication has an antecedent and the implication is embedded in a modal formulation," can be abbreviated to this: "An implication has an antecedent and is embedded in a modal formulation." Similarly, a repeated subject and verb can be elided. For example, the statement, "An implication has an antecedent and the implication has a consequent," can be abbreviated to this: "An implication has an antecedent and a consequent."

The keyword 'not' is used within an expression before the verb "is" as a way of introducing a [logical negation](#). Also, the key words "does not" are used before other verbs (modified to be infinitive) to introduce a [logical negation](#).

### C.1.1.3 Modal Operations

it is obligatory that $p$	<a href="#">obligation claim</a>
it is prohibited that $p$	<a href="#">obligation claim</a> embedding a <a href="#">logical negation</a>
it is necessary that $p$	<a href="#">necessity claim</a>
it is impossible that $p$	<a href="#">necessity claim</a> embedding a <a href="#">logical negation</a>
it is possible that $p$	<a href="#">possibility claim</a>
it is permitted that $p$	<a href="#">permissibility claim</a>

The following key words are used within expressions having a verb (often modified to be infinitive) to form verb complexes that add a modal operation.

... <b>must</b> ...	<a href="#">obligation claim</a>
... <b>must not</b> ...	<a href="#">obligation claim</a> embedding a <a href="#">logical negation</a>
... <b>always</b> ...	<a href="#">necessity claim</a>
... <b>never</b> ...	<a href="#">necessity claim</a> embedding a <a href="#">logical negation</a>
... <b>may</b> ...	<a href="#">permissibility claim</a>

The key word phrase “**only if**” is used in combination with some of the key words and phrases shown above to invert a modality.

... <b>may</b> ... <b>only if</b> $p$	<a href="#">obligation claim</a> over an <a href="#">implication</a>
<b>it is permitted that</b> $q$ <b>only if</b> $p$	<a href="#">obligation claim</a> over an <a href="#">implication</a>
<b>it is possible that</b> $q$ <b>only if</b> $p$	<a href="#">necessity claim</a> over an <a href="#">implication</a>

The key word “**only**” can also be used with “**may**” in an expression before a preposition to invert a modality.

... <b>may</b> ... <b>only</b> ...	<a href="#">obligation claim</a> over an <a href="#">implication</a>
------------------------------------	--

### C.1.2 Other Keywords

<b>the</b>	<ol style="list-style-type: none"><li>used with a designation to make a pronominal reference to a previous use of the same designation. This is formally a binding to a variable of a quantification.</li><li>introduction of a name of an individual thing or of a definite description</li></ol>
<b>a, an</b>	universal or existential quantification, depending on context based on English rules
<b>another</b>	(used with a term that has been previously used in the same statement) existential quantification plus a condition that the referent thing is not the same thing as the referent of the previous use of the term
<b>a given</b>	universal quantification pushed outside of a demonstrative expression where ‘ <b>a given</b> ’ is used such that it represents one thing at a time – this is used to avoid ambiguity where the ‘ <b>a</b> ’ by itself could otherwise be interpreted as an existential quantification.
<b>that</b>	<ol style="list-style-type: none"><li>when preceding a designation for a noun concept, this is a binding to a variable (as with ‘<b>the</b>’)</li></ol>

2. when after a designation for a noun concept and before a designation for a fact type, this is used to introduce a restriction on things denoted by the previous designation based on facts about them
3. when followed by a propositional statement, this used to introduce nominalization of the proposition or objectification, depending on whether the expected result is a proposition or an actuality. See C.1.5 below.

**who** the same as the second use of ‘that’ but used for a person

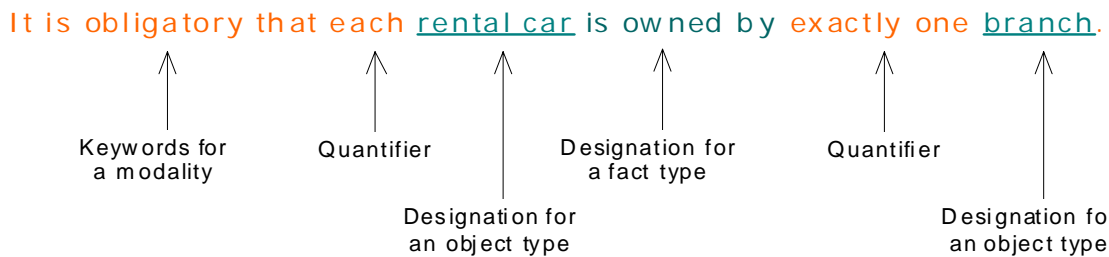
**is of** The common preposition “*of*” is used as a shorthand for “*that is of*”. For any sentential form that takes the general form of ‘<placeholder 1> *has* <placeholder 2>’ there is an implicit reversed form of ‘<placeholder 2> *is of* <placeholder 1>’ that has the same meaning.

**what** used to introduce a variable in a projection as well as indicate that a projection is being formulated to be considered by a question or answer nominalization. See C.1.5 below.

### C.1.3 Examples

It is obligatory that each rental car is owned by exactly one branch.

The example above includes three key words or phrases, two designations for noun concepts and one for a fact type (from a form of expression), as illustrated below.



Below are two statements of a single rule:

1. A rental must have at most three additional drivers.
2. It is obligatory that each rental has at most three additional drivers.

Using the font styles of SBVR Structured English, these rule statements are:

1. A rental must *have* at most three additional drivers.
2. It is obligatory that each rental *has* at most three additional drivers.

A semantic formulation of the rule can be seen in the introduction to “Logical Formulation of Semantics Vocabulary” on page 37.

The characteristic ‘driver is of age’ has the following definition: “the age of the driver is at least the EU-Rent Minimum Driving Age”. Below is the definition using the SBVR Structured English styles.

Definition: the age of the driver is at least the EU-Rent Minimum Driving Age

A semantic formulation of the definition can be seen in the introduction to “Logical Formulation of Semantics Vocabulary” on page 37.

### C.1.4 Qualifying Signifiers by Vocabulary and/or Symbol Context

Some signifiers are used to mean different things in different vocabularies or in different contexts. In SBVR structured English a signifier can be followed by parentheses enclosing the name of a vocabulary and/or a term for a symbol context. If both are present, they are separated by a comma. Qualifications are shown in the example rules below.

Necessity:                    Each customer (car rental responsibility) is a corporate renter or is an individual customer.

The signifier “customer” is used in two ways in the EU-Rent English Vocabulary. So the first rule above uses “customer” for its meaning in the symbol context ‘car rental responsibility’.

If the same rule is stated in a place where the EU-Rent English Vocabulary is not understood to be in use, the rule would be stated as follows in order to fully qualify its terms:

Necessity:                    Each customer (EU-Rent English Vocabulary, car rental responsibility) is a corporate renter (EU-Rent English Vocabulary) or is an individual customer (EU-Rent English Vocabulary).

### C.1.5 Objectification and Nominalization

The keyword ‘**that**’ can introduce a proposition being objectified or nominalized. The following examples use the fact types ‘car is assigned to rental, ‘car assignment involves car,’ ‘car assignment is to rental,’ ‘rental has pick-up date,’ and ‘rental is guaranteed by credit card’.

The first example is objectification. It states that a car assignment is an actuality denoted by the proposition that a given car is assigned to a given rental. Note that only the third use of ‘**that**’ in the example below introduces an objectification. The others introduce restrictions

Necessity:                    A car assignment that involves a car and that is to a rental is an actuality that the car is assigned to the rental.

An objectification uses a propositional expression to identify a state of affairs or event. States and events can then be related to times and durations or be involved in any number of fact types that concern states or events. Consider the following examples of fact types.

state of affairs occurs before point in time

state of affairs<sub>1</sub> occurs before state of affairs<sub>2</sub> occurs

The following rule uses the first fact type above:

A car assignment that is to a rental must occur before the pick-up date of the rental.

SBVR Structured English supports objectification using a convenient mechanism that is based on the word “**occurs**” being in the designation of a fact type after a placeholder. An implicit form of a fact type can be used that objectifies a propositional expression in the position of the placeholder and leaves out the word “**occurs**”. In other words, the rule above can be stated like this:

A car must be assigned to a rental before the pick-up date of the rental.

Using these implicit forms allows objectification to occur implicitly without defining corresponding noun concepts for each fact type whose instances might be objectified. For example, using the second fact type listed above I can form the following rule even though no noun concept is defined for the fact type ‘rental is guaranteed by credit card’.

A [rental](#) must be guaranteed by a [credit card](#) before a [car](#) is assigned to the [rental](#).

The next example is a proposition nominalization. It uses the additional fact types '[report specifies fact](#)' and '[rental has rental report](#)'. The keyword 'that' nominalizes a fact to be specified.

Necessity:                    If a [car](#) is assigned to a [rental](#) then the [rental report of the rental](#) must specify that the [car](#) is assigned to the [rental](#).

The next example is an answer nominalization. The keyword 'what' is used to put variables in a projection.

Necessity:                    The [rental report of each rental](#) must specify what [car](#) is assigned to the [rental](#).

An expression of a statement can include the keyword 'what' multiple times, putting more variables in the projection (for example, "[what car is assigned to what rental](#)"). A question nominalization is formed in the same way as an answer nominalization, but nominalizes the question itself rather than an answer to it.

## C.1.6 Intensional Roles

Some fact types about time and change have what can be called intensional roles. In English, most verbs are about their expressed subjects and objects, but in some cases, a verb involves the meaning of the expression of the subject or object. The verb takes its argument by name rather than by value. Fact types for such verbs are often about time and change.

The SBVR Structured English uses a special syntactic clue to identify placeholders for intensional roles in forms of expression. Normally, a placeholder is shown using a designation for a concept that generalizes its role, but for an intensional role that concept is a concept type and is shown in square brackets after designation for a noun concept that corresponds with syntactic usage of the verb. Some examples of such fact types are listed below.

### [thing](#) [[individual concept](#)] *is changed*

Definition:                    the extension of [the individual concept](#) is different at one point in time from what it is at a subsequent point in time

Example:                      "If the scheduled pick-up time of a rental is changed ...."

### [thing<sub>1</sub>](#) *becomes* [thing<sub>2</sub>](#) [[noun concept](#)]

Definition:                    [the thing](#) is an instance of [the noun concept](#), but having just previously not been an instance of [the noun concept](#)

Example:                      "If a driver of a rental becomes a barred driver before the actual drop-off date of the rental ...."

### [quantity<sub>1</sub>](#) [[individual concept](#)] *increases by* [quantity<sub>2</sub>](#)

Definition:                    [the individual concept](#) refers to a quantity at some point in time and to a different quantity at a later point in time which is greater than the first quantity by [the quantity<sub>2</sub>](#)

Example:                      "If the odometer reading of a rental car increases by 10,000 miles during a rental ...."

Use of such fact types often involves the special semantic formulations [noun concept formulation](#) and [fact type formulation](#), explained and exemplified in Part II. Also, see examples in Annex E.

## C.2 Describing a Vocabulary

A vocabulary is described in a document section having glossary-like entries for concepts having representations in the vocabulary. Those entries are explained in the next section. The introduction to a vocabulary description includes the vocabulary's name and can further include any of the several kinds of details shown in the skeleton below.

### <Vocabulary Name>

Description:

Source:

Speech Community:

Language:

Included Vocabulary:

Note:

## **C.2.1 The Vocabulary Name**

The vocabulary name appears in the ‘Name’ Font.

## **C.2.2 Description**

The ‘Description’ caption is used to introduce the scope and purpose of the vocabulary.

## **C.2.3 Source**

The ‘Source’ caption is used if the vocabulary being described is based on a formally-defined work. For example, if the vocabulary being described is based on a glossary or other document developed independently of the formalisms of SBVR, then that glossary or other document is shown as the source.

## **C.2.4 Speech Community**

The ‘Speech Community’ caption is used to name the speech community that controls and is responsible for the vocabulary.

## **C.2.5 Language**

The ‘Language’ caption is used to name the language that is the basis of the vocabulary. Language names are from [ISO 639-2 \(English\)](#). By default, [English](#) is assumed. Note that the SBVR Structured English is based only on English, so descriptions, definitions and other details are in English but representations being defined can be in another language.

### EU-Rent Vocabulaire Française

Language: [French](#)

## **C.2.6 Included Vocabulary**

The ‘Included Vocabulary’ caption is used to indicate that another vocabulary is fully incorporated into the vocabulary being described. All designations and forms of expressions of an included vocabulary are part of the vocabulary being described.

## **C.2.7 Note**

The ‘Note’ caption labels explanatory notes that do not go under the other captions.

## C.3 Vocabulary Entries

Each entry is for a single concept, called the entry concept. It starts with a primary representation which is either a designation or a form of expression for the concept.

Any of several kinds of captioned details can be listed under the primary representation. A skeleton of a vocabulary entry is shown below followed by an explanation of the use of each caption.

### <primary representation>

Definition:  
Source:  
Dictionary Basis:  
General Concept:  
Concept Type:  
Symbol Type:  
Necessity:  
Possibility:  
Reference Scheme:  
Note:  
Example:  
Synonym:  
Synonymous Form:  
See:  
Qualifier:

### C.3.1 Designation or Form of Expression

The primary representation (designation or form of expression) for an entry can be for any concept type. It is shown in its appropriate font style.

The primary representation for a fact type is a form of expression. In the unusual case where the same designation is used for more than one placeholder, a subscript on each placeholder can be given so that references to the roles from a definition of other text within the entry is unambiguous.

It is recommended that quantifiers (including articles) and logical operators not be embedded within designations and forms of expression.

### C.3.2 Definition

A definition is shown as an expression that can be logically substituted for the primary representation. It is not a sentence, so it does not end in a period.

A definition can be fully formal, partly formal or informal. It is fully formal if all of it is styled as described above. A partially-formal definition starts with a styled designation for a more general concept but other details depend on external concepts.

Styles of definition are explained separately for different types of concepts.

### C.3.2.1 Definition of a Noun Concept

A common pattern of definition begins with a designation for a more general concept followed by the keyword 'that' (used in the second sense defined for 'that' in the Other Keywords section above) and then an expression of necessary and sufficient characteristics that distinguish a thing of the defined concept from other things of the more general concept. Another less used pattern also leads with a designation for a more general concept but then uses the word 'of' with another expression as explained in the Other Keywords section above.

Two kinds of information are formally expressed by a fully formal definition.

1. A fact that the concept being defined is a category of a particular more general concept
2. A closed projection that defines the concept.

Only the first kind of information is formally expressed by a partially formal definition. A partially formal definition leads with a styled designation that is for a more general concept. That designation is generally followed by the keyword 'that' and then an informal expression of necessary and sufficient characteristics.

The following example shows a partially formal definition. It formally expresses the fact that the concept 'icon' is a category of the concept 'symbol,' but it uses words that are external to the formally available vocabulary.

#### icon

Definition: symbol that is a pictorial representation

The next example is fully formal. Its formal interpretation includes that the concept 'obligation claim' specializes the concept 'modal formulation' and also includes a closed projection conveying semantics of the definition.

#### obligation claim

Definition: modal formulation that claims the modality 'obligation'

The next example is not formal at all. It defines the most general concept used by SBVR.

#### thing

Definition: anything perceivable or conceivable

A definition of a noun concept can generally be read as a statement using the following pattern (where "a" represents either "a" or "an"):

A <designation> is a <definition>.

For example: An icon is a symbol that is a pictorial representation.

Another style of formal definition is extensional. It uses disjunction to combine a number of concepts. For example, a semantic formulation is anything that is a logical formulation or a projection.

#### semantic formulation

Definition: logical formulation or projection

### C.3.2.2 Definition of an Individual Concept

A definition of an individual concept is just like a definition of a noun concept described above except that it must be a definite description of one single thing.



A definition of an individual concept can generally be read as a statement using the following pattern. The leading “The” is optionally used depending on the designation.

[The] <designation> is the <definition>.

It is often that case that an individual concept has no definition because it is widely understood. In such a case the ‘General Concept’ caption can be used to state the type of the named thing. Here is an example.

### Switzerland

General Concept:            country

### C.3.2.3 Definition of a Fact Type

A definition given for a fact type is an expression that can be substituted for a simple statement expressed using a form of expression of the fact type.

The definition must refer to the placeholders in the form of expression. This is done in order to relate the definition to the things that play a role in instances of the fact type. Whether or not the definition is formal, each reference to a placeholder appears in the ‘term’ font and is preceded by the definite article, “the”.

Here is an informal example followed by a fully-formal one.

#### statement expresses proposition

Definition:                    the proposition is what is meant by the statement

#### sequence is of general concept

Definition:                    each thing that is included in the sequence is an instance of the general concept

The second definition above is formal such that it translates to a closed projection.

A definition of a fact type can generally be read using the pattern below, which is shown for a binary fact type but works for fact types of any arity (“a” represents either “a” or “an”).

A fact that a given <placeholder 1> <fact type designation> a given <placeholder 2> is a fact that <definition>.

For example: A fact that a given statement expresses a given proposition is a fact that the proposition is what is meant by the statement.

Similarly, the equivalence understood from a definition of a fact type can generally be read using the following pattern:

A <placeholder 1> <fact type designation> a <placeholder 2> if and only if <definition>.

For example: A statement expresses a proposition if and only if the proposition is what is meant by the statement.

### C.3.3 Source

The ‘Source’ caption is used to indicate a source vocabulary or document for a concept.

The source’s designation for the concept is given in square brackets and quoted after the name of the source. It might or might not match the entry’s primary representation. If the source has a name for the concept itself, the name is given in square brackets unquoted. The designation from the source is quoted if it is a term for the concept.

## thing

Source: [ISO 1087-1 \(English\)](#) (3.1.1) ['object']

## individual concept

Source: [ISO 1087-1 \(English\)](#) (3.2.2) ['individual concept']

The keywords “**based on**” indicate the definition of the concept is largely derived from the given source but had some modification, as in the following example.

## language

Definition: system of arbitrary signals (such as voice sounds or written symbols) and rules for combining them as used by a nation, people or other distinct community

Source: **based on** AH

### C.3.4 Dictionary Basis

This caption labels a definition from a common dictionary that supports the use of the primary representation. The entry source reference (written in the ‘Source’ style described above) is supplied at the end of the quoted definition. A dictionary basis should not be interpreted as an adopted definition.

### C.3.5 General Concept

The ‘General Concept’ caption can be used to indicate a concept that generalizes the entry concept. This is not needed if there is a definition that starts with the general concept, but it is helpful in cases where a definition is not provided, such as is often the case for individual concepts (named things) or concepts taken from a source. Here are two examples.

## Switzerland

General Concept: [country](#)

## individual concept

Source: [ISO 1087-1 \(English\)](#) (3.2.2) ['individual concept']

General Concept: [concept](#)

### C.3.6 Concept Type

The ‘Concept Type’ caption is used to specify a type of the entry concept. This is typically not used if the concept has no particular type other than what is obvious from the primary representation.

- A name is implicitly for an [individual concept](#).
- Any term is implicitly for a [noun concept](#).
- A form of expression is implicitly for a [fact type](#).
- For a form of expression, one placeholder implies a [unary fact type](#) and two placeholders imply a [binary fact type](#). E.g., ‘[variable has type](#)’ is implicitly for a [binary fact type](#).
- Where a definition formally gives a more general concept, the concept being defined specializes that more general concept.

If more than one concept type is mentioned, then they are separated by commas. Order is insignificant.

The concept type '[role](#)' is commonly used. The example below shows that the concept '[negand](#)' is a role that is played by a logical formulation.

### [negand](#)

Concept Type: [role](#)  
Definition: [logical formulation](#) that is the operand of a given [logical negation](#)

Note that a definition of a [role](#) of a [fact type](#) can use “a given” to clearly mark any opposing roles in the definition, but sometimes an indefinite article is used without the word “given”.

Any [noun concept](#) that specializes the concept '[concept](#)' can be given as a concept type. The concept '[obligation claim](#)' is a logical formulation kind, which is defined below.

### [logical formulation kind](#)

Definition: [concept](#) that *specializes* the [concept](#) '[logical formulation](#)' and that classifies a [logical formulation](#) based on the presence or absence of a main logical operation or quantification

### [obligation claim](#)

Definition: [modal formulation](#) that *claims* the [modality](#) '[obligation](#)'  
Concept Type: [logical formulation kind](#)

## C.3.7 Symbol Type

The 'Symbol Type' caption is used to mention a category of symbol being defined. This is typically not used if the symbol has no particular type other than what is obvious from the font of the symbol being presented.

This caption is not generally used to note that a symbol is preferred. Rather, a preferred symbol is known because it is listed with a definition or a source. A symbol that is not preferred is shown only with a 'Synonym' or 'Synonymous Form' caption that introduces the preferred symbol for the concept.

## C.3.8 Necessity and Possibility

A 'Necessity' or 'Possibility' is usually supplemental to a definition. A 'Necessity' caption is used to state something that is necessarily true. A 'Possibility' caption explains that something is a possibility that is not prevented by definition. See the vocabulary entries in Part II for 'structural business rule statement' and 'unrestricted business rule possibility statement' (respectively) for more details.

The key phrase “it is necessary that” can be omitted from a statement of a structural rule captioned “Necessity” because it is implied by the caption. Here are examples -- two necessity claims and one possibility claim.

### [implication has antecedent](#)

Definition: relates the [implication](#) to its conditional operand  
Necessity: That a given [implication](#) has a given [antecedent](#) is a structural detail of the [implication](#).  
Necessity: Each [implication](#) has exactly one [antecedent](#).

### [vocabulary is mapped to target package](#)

Possibility: It is possible that a [vocabulary](#) is mapped to more than one [target package](#).

Definitions express characteristics that are necessary and sufficient to distinguish things denoted by a concept. Sometimes there are necessities beyond what is sufficient. The ‘Necessity’ caption is used to state such necessities.

### C.3.9 Reference Scheme

The ‘Reference Scheme’ caption is used to state how things denoted by the term can be distinguished from each other based on one or more facts about the things. A reference scheme is expressed by referring to at least one role of a binary fact type and indicating whether a reference involves a single instance of the role or whether it involves the extension of related instances.

An article (‘a’, ‘an’ or ‘the’) indicates a simple use of a role in which a single instance is used in a reference. The definite article ‘the’ is only appropriate where there can be at most one instance of the role. The words ‘the set of each’ indicate that the extension is used. The word ‘and’ is used to connect the expressions of use of multiple roles by a reference scheme.

The following examples of reference schemes are taken from the SBVR Vocabularies. The first one below uses a single value of a role (‘[closed logical formulation](#)’), meaning that a proposition can be identified by any closed logical formulation whose meaning is the proposition. The second uses two roles. It uses a definite article because each [role binding](#) has exactly one [bindable target](#) and is for exactly one [role](#).

#### [proposition](#)

Reference Scheme:        a [closed logical formulation](#) that means the [proposition](#)

#### [role binding](#)

Reference Scheme:        the [bindable target](#) that is referenced by the [role binding](#) and the [role](#) that has the [role binding](#)

The reference scheme for the concept of reference scheme itself uses two roles extensionally.

#### [reference scheme](#)

Reference Scheme:        the set of each [role](#) that is singularly used by the [reference scheme](#) and the set of each [role](#) that is extensionally used by the [reference scheme](#)

### C.3.10 Note

A ‘Note’ caption is used to label explanatory notes that do not fit within the other captions.

### C.3.11 Example

The ‘Example’ caption labels examples of involving the entry concept.

### C.3.12 Synonym

A synonym is another designation that can be substituted for the primary representation. It is a designation for the same concept. If the primary representation is a form of expression, then the ‘Synonymous Form’ caption is used rather than the ‘Synonym’ caption.

The examples below show two synonyms for one concept having one definition. The preferred symbol is given as the primary representation.

### **implication**

Definition: [logical formulation](#) **that** applies the logical “(MATERIALLY) IMPLIES” operation ( $\rightarrow$ ) to an [antecedent](#) and a [consequent](#)

Synonym: [material implication](#)

The meaning of two designations being synonyms is that they represent the same concept.

### **C.3.13 Synonymous Form**

A synonymous form is a form of expression for the same fact type. The order of placeholders for roles can be different.

A synonymous form can appear elsewhere as its own entry. However, this is not typically done if the synonymous form is simply a passive form of the primary representation. The following example shows a synonymous form that reverses the order of roles. Because the synonymous form is simply a passive form of the primary representation, it does not appear as a separate entry.

#### **statement expresses proposition**

Definition: **the** [proposition](#) is what is meant by **the** [statement](#)

Synonymous Form: [proposition is expressed by statement](#)

A synonymous form does not necessarily use the same designations for placeholders as are used in the primary representation. A placeholder might or might not use a designation for a role. When this happens, in order to match the placeholder of the synonymous form to the corresponding placeholder in the primary representation, the placeholder in the synonymous form is followed by the placeholder’s designation in the primary representation in square brackets.

The example below shows two entries, both for the same concept. One is expressed in terms of a role ([instance](#)) and the other is not.

#### **concept corresponds to thing**

Definition: **the** [thing](#) is in the extension of **the** [concept](#)

Synonymous Form: [concept has instance \[thing\]](#)

#### **concept has instance**

Synonymous Form: [concept corresponds to thing \[instance\]](#)

If the same term is used for multiple placeholders, then subscripts can be used to distinguish them.

#### **thing<sub>1</sub> is thing<sub>2</sub>**

Synonymous Form: [thing<sub>1</sub> equals thing<sub>2</sub>](#)

The meaning of two forms of expression being synonymous is that the two are represent the same fact type.

### **C.3.14 See**

Where the primary representation is not a preferred representation for the entry concept, the “See:” caption introduces the preferred representation. No definition is given in this case.

### C.3.15 Qualifier

Where a signifier is not unique in a vocabulary, there is a need for qualification by a symbol context. A symbol context is given using the “Qualifier” caption, as shown in the example below.

#### customer

Qualifier: [car rental responsibility](#)

See: [renter](#)

#### customer

Qualifier: [vehicle sales](#)

Definition: [person](#) who purchases a [rental car](#) from EU-Rent at the end of its rental life

## C.4 Specifying a Rule Set

A rule set is specified in a document section having several individual entries for guidance. Those entries are explained in the next section. The introduction to a rule set includes the rule set’s name and can further include any of the several kinds of details shown in the skeleton below.

### <Rule set name>

Description:

Vocabulary:

Note:

Source:

### C.4.1 The Rule Set Name

The rule set name appears in the ‘name’ font.

### C.4.2 Description

The ‘Description’ caption is used to describe the scope and purpose of the rules.

### C.4.3 Vocabulary

The ‘Vocabulary’ caption is used to identify what vocabulary (defined in terms of SBVR) is used by statements in the rule set.

### C.4.4 Source

The ‘Source’ caption is used if the rule set is based on a separately-defined work. It labels a reference to such a work, such as a legal statute.

### C.4.5 Note

The ‘Note’ caption is used to label explanatory notes that do not fit within the other captions.

## C.5 Guidance Entries

Each entry in a rule set is an element of guidance -- expressed as one of the following:

- An operative business rule statement
- A structural business rule statement
- An admonition statement
- An affirmation statement

Business rules include only those rules under business jurisdiction. Entries can also be made for structural rules that are not under business jurisdiction. Each entry includes the statement itself and optionally includes other information labeled by the captions shown below.

### <Rule Statement or Clarification Statement>

Name:

Guidance Type:

Description:

Source:

Synonymous Form:

Note:

Example:

Enforcement Level:

Use of each of the above captions is explained below.

### C.5.1 Rule Statement or Clarification Statement

A rule statement or clarification statement can be expressed formally or informally. A statement that is formal uses only formally styled text — all necessary vocabulary is available (by definition or adoption) such that no external concepts are required. Such a statement can be represented as a logical formulation.

### C.5.2 Name

The 'Name' caption is used to specify a name for the rule or clarification. The name is then part of the formal vocabulary.

### C.5.3 Guidance Type

The 'Guidance Type' caption is used to indicate the kind of element of guidance -- i.e., one of the following:

- operative business rule
- structural business rule
- admonition
- affirmation

#### **C.5.4 Description**

The ‘Description’ caption is used to capture the expression of the element of guidance informally (as supplied by a business user).

#### **C.5.5 Source**

The ‘Source’ caption is used if the guidance is from a separate source. It labels a reference to that source.

#### **C.5.6 Synonymous Form**

The ‘Synonymous Form’ caption is used to state additional, equivalent statements of the guidance. For example, a given rule can be expressed in a ‘prohibitive’ form and also in an ‘obligatory’ form. As for the primary statement of the guidance, these additional statements can be formal or informal.

#### **C.5.7 Note**

The ‘Note’ caption is used to label explanatory notes that do not fit within the other captions.

#### **C.5.8 Example**

The ‘Example’ caption labels examples of application of the element of guidance.

#### **C.5.9 Enforcement Level**

The ‘Enforcement Level’ caption labels the enforcement level that applies to an operative business rule (only).



## Annex D (informative)

### SBVR Structured English Patterns

This annex contains material compiled to aid the interpretation of ‘SBVR in SBVR Structured English’ vocabulary entries, as documented in Annex C and applied in the text and diagram forms of Part II and Annex E. This ‘language patterns’ material falls into two main categories:

- reading SBVR Vocabulary symbols
- reading fact types embedded in the definition text of SBVR Vocabulary symbols.

When there is an associated way to depict the construct in a graphic notation, a cross-reference is provided, when applicable, to the ‘Use of UML Notation in a Business Context to Represent SBVR-based Vocabularies’ (Annex H) -- referred to here as the ‘UML style’ -- and to the ‘Concept Diagram Graphic Notation (Annex G)’ -- referred to here as the ‘CDG style’.

#### D.1 Reading SBVR Vocabulary Symbols

This section presents the interpretation given to the three kinds of vocabulary symbol:

- Terms
- Names
- Designations for Fact Types

##### D.1.1 Primary Term for a General Concept

When I see a vocabulary entry as shown in Figure H-1, I know to vocalize it as:

‘community’ is a term for a general concept. And it is the ‘primary’ term used for the concept.

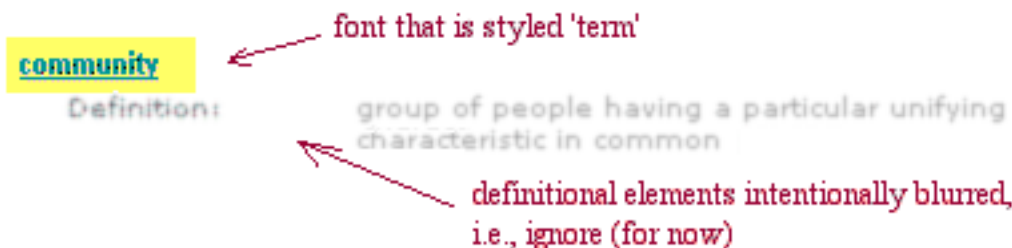


Figure D.1 - Recognizing an entry that is the primary term for a general concept.

For how to depict this in graphics, see H.1 (UML style) and G.1 (CDG style).

Commentary:

This is a typical *symbol* kind of entry presented as a ‘term’ -- the primary term for a general concept. For this kind of entry, draw a labeled box.

It is possible to have additional terms for a given general concept (i.e., terms that are synonyms). Even when documented in the text form (using the ‘Synonym’ caption), the non-primary terms of a concept are not typically reflected on the graphic. When it is considered useful to make explicit entries for the non-primary terms in a presentation of the vocabulary, the non-primary terms can appear using the ‘See’ caption to refer back to the concept’s primary term.

## D.1.2 Primary Term (Name) for an Individual Concept

When I see a vocabulary entry as shown in Figure D.2, I know to vocalize it as:

‘Real-world numerical correspondence’ is a term that is a name for an individual concept. And it is the primary name used for the concept.

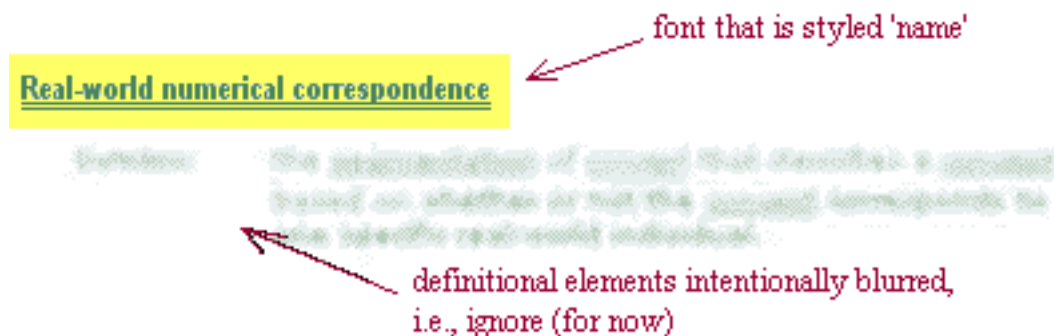


Figure D.2 - Recognizing an entry that is the primary term (name) for an individual concept

For how to depict this in graphics, see H.2 (UML style). There is no specified way to depict this in the CDG graphic notation.

Commentary:

This is a typical *symbol* kind of entry presented as a ‘name’ -- the primary name for an individual concept. For this kind of entry, draw a labeled box, with the ‘name’ underlined.

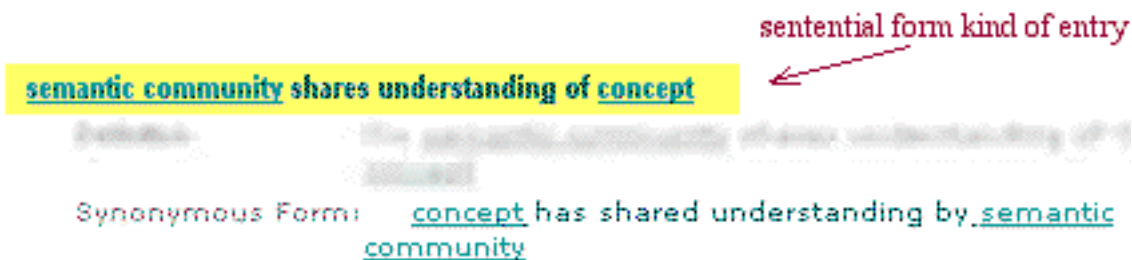
It is possible to have additional names for a given individual concept (i.e., names that are synonyms). Even when documented in the text form (using the ‘Synonym’ caption), the non-primary terms of a concept are not typically reflected on the graphic. When it is considered useful to make explicit entries for the non-primary names in a presentation of the vocabulary, the non-primary names can appear using the ‘See’ caption to refer back to the concept’s primary name.

## D.1.3 Primary Reading (‘Sentential Form’) for a Fact Type

### D.1.3.1 Primary Reading (‘Sentential Form’) for a Fact Type -- Binary Fact Type

When I see a vocabulary entry as shown in Figure D.3, I know to vocalize it as:

There is a fact type relating these two concepts and it uses the designation ‘shares understanding of’ when the concept terms are in this order. Optionally, alternative readings can be provided using the ‘Synonymous Form’ caption (as illustrated at the bottom of Figure D.3).



**Figure D.3- Recognizing an entry that is the primary reading for a binary fact type**

For how to depict this in graphics, see H.3.1 (UML style) and G.3.1 (CDG style). There is a special case of depicting a binary fact type that uses ‘has’ in the UML style. For how to depict this in graphics, see H.3.2 (UML style). There is no special way to depict this in the CDG graphic notation.

Commentary:

This is a typical *sentential form* kind of entry for a fact type -- in this case, a binary fact type. For this kind of entry, draw a labeled line between the boxes for the symbols of the participating concepts. The reading is clockwise (when the tool does not provide a graphic symbol for indicating the directionality of the reading).

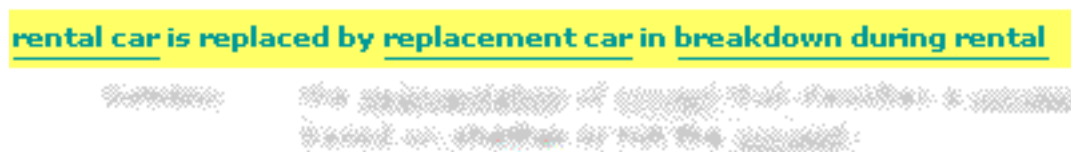
It is possible to have additional readings for a given fact type (i.e., readings that are ‘synonymous forms’ of the fact type). Additional readings are optional in both the graphic and text forms. When defined in the text form, the ‘Synonymous Form’ caption is used. Even when provided in the text, more than one reading is not typically reflected on the graphic. However, having inverse readings on an association would be an extension to UML. (This can be handled legally by defining a ‘UML profile,’ which allows additional information and custom graphics in a model.)

An alternative graphic style is to apply the n-ary graphic style (described below) for *all* fact types, including binary.

### D.1.3.2 Primary Reading (‘Sentential Form’) for a Fact Type -- N-ary Fact Type

When I see a vocabulary entry as shown in Figure D.4, I know to vocalize it as:

There is a ternary fact type relating these three concepts, using ‘is replaced by ... in’ when the fact type uses these terms for the concepts in this sequence.



**Figure D.4 - Recognizing an entry that is the primary reading for an n-ary fact type**

For how to depict this in graphics, see H.3.3 (UML style) and G.3.2 (CDG style).

Commentary:

This is a *sentential form* kind of entry for a fact type -- in this case, an n-ary fact type. For this kind of entry, there are two diagrams forms. The first diagram is the box-in-box style as defined in Annex G.3.2. The second diagram (UML-style) uses a box, given a stereotype that names the category of fact type, and a label that reflects the primary reading for the fact type. The concept terms are placed in [ ].

Note-1 -- The label in the UML form does not use the UML association 'name'; the UML association 'name' is reserved for use as a 'real' name.

Note-2 -- While suggestions have been given for depicting multiple readings on a diagram, showing additional readings for n-ary fact types is not currently part of the scope of this documentation.

### D.1.3.3 Primary Reading ('Sentential Form') for a Fact Type -- Unary Fact Type

When I see a vocabulary entry as shown in Figure D.5, I know to vocalize it as:

There is a unary fact type for this concept, with a designation of 'is damaged'.



Figure D.5 - Recognizing an entry that is the primary reading for a unary fact type

For how to depict this in graphics, see H.3.4 (UML style) and G.3.3 (CDG style).

Commentary:

This is a *sentential form* kind of entry for a fact type -- in this case, a unary fact type. For this kind of entry, the two graphic notations use different forms. The first diagram above shows the box-in-box style as defined in Annex G.3.3. For the UML-style, three alternatives are offered:

1. List the designation inside the box ('attribute' style).
2. Draw in the same style as for an n-ary fact type (above).
3. Draw using the association 'diamond'.

NOTE: The notation for unary fact type would be an extension to UML, handled legally by defining a 'UML profile'.

#### D.1.3.4 Two Vocabulary Entries (Sentential Form and Term) for a Concept

When I see a pair of vocabulary entries as shown in Figure D.6, I know to vocalize this case as:

These two entries are for coextensive concepts. I understand that, even though these are two entries in the vocabulary, they have the same instances.

rented car is recovered from non-EU-Rent site to branch

car recovery

Definition: actuality that a given rented car is recovered from a given non-EU-Rent site to a given branch

Figure D.6- Recognizing a pair of entries (sentential form and term) for a concept

For how to depict this in graphics, see H.8 (UML style) and G.3.4 (CDG style).

## D.2 Reading Embedded Fact Types

There are also fact types that are defined when the SBVR Structured English language is used to compose the definition of a vocabulary entry. The material in this section documents the most common patterns used in writing entry definitions using the elements of style defined in Annex C.

The following six patterns have been documented.

- categorization fact type
- is-role-of fact type
- partitive fact type
- instance to general concept ('predefined extension')
- categorization type
- categorization scheme

### D.2.1 Categorization Fact Type

When I see this:

semantic community

Definition: community whose unifying characteristic is a shared understanding (perception) of the things that they have to deal with

I know this is shorthand for:

semantic community

Concept Type: category

Definition: community whose unifying characteristic is a shared understanding (perception) of the things that they have to deal with

I know to vocalize it as:

The concept 'semantic community' is a 'category' of the more general concept 'community'. Furthermore, I know that what distinguishes this particular kind of community from the general case is that it is ... <distinctions brought out in the rest of the definition>

For how to depict this in graphics, see H.5 (UML style) and G.2.1 (CDG style).

## D.2.2 Is-role-of Fact Type

When I see this:

### renter

Concept Type: role  
Definition: driver who ,, ,!

I know to vocalize it as:

The concept 'renter' is a role that can be played by a driver, specifically one who ... <distinctions brought out in the rest of the definition>

For how to depict this in graphics, see H.4 (UML style) and G.4 (CDG style). The CDG style does not distinguish the various ways to depict roles as in the UML style (see treatment in H.4.1, H.4.2, and H.4.3).

## D.2.3 Partitive Fact Type

When I see this:

### vocabulary<sub>1</sub> incorporates vocabulary<sub>2</sub>

Concept Type: partitive fact type  
Definition: the vocabulary<sub>1</sub> includes each symbol that is included in the vocabulary<sub>2</sub>  
Note: When more than one vocabulary is included, a hierarchy of inclusion can provide priority for selection of definitions.

vocabulary<sub>2</sub> is incorporated into vocabulary<sub>1</sub>

### vocabulary includes symbol

Concept Type: partitive fact type  
symbol is included in vocabulary

I know to vocalize it as:

A vocabulary incorporates (another) vocabulary.

A vocabulary includes symbols.

For how to depict this in graphics, see H.7 (UML style). There is no specified way to depict this in the CDG graphic notation.

## D.2.4 Instance to General Concept Fact Type ('Predefined Extension' Entry)

When I see this:

### Canada

General Concept: [country](#)

I know to vocalize it as:

Canada is an instance of the concept 'country'

(or, 'Canada' is a designation of an individual country)

For how to depict this in graphics, see the discussion of 'Primary Term (Name) for an Individual Concept' above.

Typically, this kind of entry is simply 'indicated' (or perhaps 'adopted'), with no definition. However, when a definition is written, its styling can specify the general concept, in which case, the 'General Concept' caption can be omitted. For example, the entry below defines 'Car Rental Industry' to be an instance of 'semantic community':

### Car Rental Industry

Definition: [the semantic community](#) **that is** the group of people who work in the business of renting cars

Commentary:

When you find this pattern, draw it in the UML style using UML's arrow style for 'instantiation'. The notation has been adapted from standard UML notation to make it more 'business friendly' -- e.g., in UML, in instance ('object') would be labeled as, [Canada: country](#). Predefined extension instances are not typically depicted in the box-in-box style.

## D.2.5 Categorization Type

When I see this:

### branch type

Definition: [concept](#) **that specializes the concept** '[branch](#)' **and that classifies a** [branch](#) based on its [hours of operation](#) **and** [car storage capacity](#)

### city branch

Concept Type: [branch type](#)

Definition: [branch](#) that operates in a city

I know to vocalize it as:

The concept 'branch type' has instances that are (or are in 1:1 correspondence with) certain categories of 'branch' -- depending on the interpretation you take for this pattern.

'city branch' is a category of 'branch'.

'city branch' is (or is in 1:1 correspondence with) a 'branch type'.

For how to depict this in graphics, see H.6.2 (UML style). There is no specified way to depict this in the CDG graphic notation.

Commentary:

When you find this pattern -- a 'Definition' caption that begins,

concept that *specializes* the concept 'other-concept' and that *classifies* an other-concept based on ...

-- it is a compact, textual way to say multiple things, as follows:

1. that the mentioned *other-concept* has categories for which the *other-concept* is the more general concept, and
2. that the entry being defined is itself a category of concept, one whose instances are the categories of the mentioned more general concept.

Furthermore, the vocabulary entries for the certain category include a 'Concept Type:' caption that mentions the categorization type. For example, the vocabulary entry for 'city branch' mentions 'branch type' as its Concept Type.

(The examples that illustrate how to depict the '1:1 correspondence' interpretation have not yet been developed.)

## D.2.6 Categorization Scheme

When I see this:

### Branches by Type

Description: segmentation that *is for* branch and *subdivides* branch based on branch type  
Necessity: Branches by Type *contains* the categories 'airport branch' and 'city branch' and 'agency'.

### agency

Definition: branch that *does not have a EU-Rent location* and *has minimal car storage* and *has on-demand operation*  
Necessity: agency *is included in* Branches by Type.

### airport branch

Definition: branch that *has a EU-Rent location* and *has large car storage* and *has 24-7 operation*  
Necessity: airport branch *is included in* Branches by Type.

### city branch

Definition: branch that *has a EU-Rent location* and *has moderate car storage* and *has long business hours*  
Necessity: city branch *is included in* Branches by Type.

I know to vocalize it as:

'Branches by Type' is the name of a categorization scheme (or, in this case, a 'segmentation', which is a restricted case of categorization scheme). This scheme is for the general concept 'branch', presenting the instances of branch as divided into the categories that make up the scheme, according to the stated criteria. Each category's entry indicates being part of the scheme.

For how to depict this in graphics, see H.6.1 (UML style) and G.2.2 (CDG style).



Commentary:

When you find this pattern -- a 'Definition' caption under a 'name' symbol that begins,

the categorization scheme that *is for the* concept 'mentioned-other-concept' and *subdivides* mentioned-other-concept based on...

or

the segmentation that *is for the* concept 'mentioned-other-concept' and *subdivides* mentioned-other-concept based on...

-- it is a compact, textual way to say multiple things, as follows:

1. that the entry being defined is a categorization scheme (or a categorization scheme that is a segmentation), and
2. that the mentioned concept is the concept that is the scheme is *for*.

Furthermore, each vocabulary entry for one of the categories in the scheme identifies itself as part of the scheme using a 'Necessity' caption. (Note that a category can be part of more than one scheme.)



## **Annex E**

### **(informative)**

## **EU-Rent Example**

### **E.1 Introduction**

EU-Rent is a (fictitious) car rental company with branches in several countries.

#### **E.1.1 Overview of EU-Rent's Business Service**

EU-Rent rents cars to its customers. Customers may be individuals or companies. Different models of car are offered, organized into groups. All cars in a group are charged at the same rates. A car may be rented by a booking made in advance or by a 'walk-in' customer on the day of rental. A rental booking specifies the car group required, the start and end dates/times of the rental and the EU-Rent branch from which the rental is to start. Optionally, the reservation may specify a one-way rental (in which the car is returned to a branch different from the pick-up branch) and may request a specific car model within the required group.

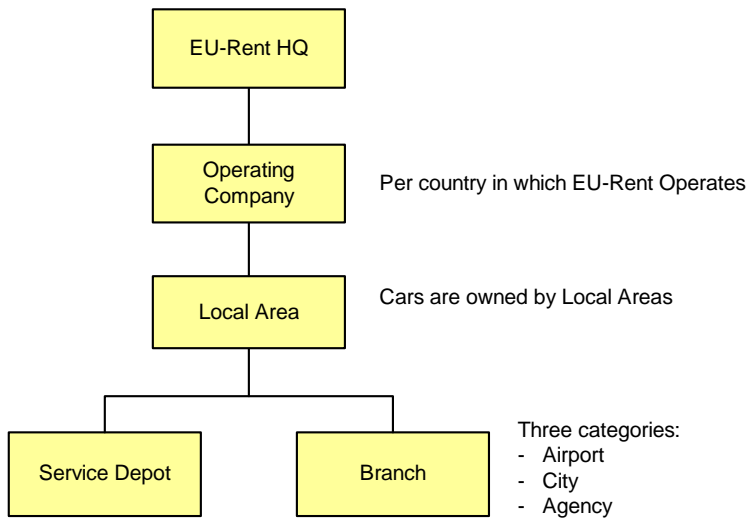
EU-Rent has a loyalty club. Customers who join accumulate points that they can use to pay for rentals

EU-Rent from time to time offers discounts and free upgrades, subject to conditions.

EU-Rent records 'bad experiences' with customers (such as unauthorized late return from rental, or damage to car during rental) and may refuse subsequent rental reservations from such customers.

#### **E.1.2 EU-Rent Organization**

EU-Rent's organization is illustrated below:



**Figure E.1- EU-Rent Organization**

EU-Rent’s world headquarters sets global policy and owns the world-wide reservations system.

In each country in which it does business EU-Rent has an Operating Company that:

- Adapts global policy to local regulation, custom and practice
- Selects which car models will be purchased for each car group
- Sets rental tariffs

Within each country EU-Rent manages its business by Local Area, which EU-Rent defines. A Local Area contains a number of Branches for Rental Car pick-up and return, and a number of Service Depots that maintain and repair EU-Rent’s cars.

Cars are stored at Branches but owned by Local Area. They are moved between Branches within a Local Area to meet rental demand.

Cars may also be moved between Local Areas within a country; if this happens, car ownership is also transferred. Car movements happen in two ways:

- A direct transfer. This is an action internal to EU-Rent: a EU-Rent employee drives the car.
- A one-way rental between different local areas in the same country: a rental customer drives the car. [Note: if the one-way rental is between different countries, car ownership is not transferred. At a time decided by the branch manager, the car is returned to the country in which it is registered].

Branches are of three types:

- ‘Airport’: large Branches at major airports and some major rail terminals. They are open 24 hours per day, 7 days per week, have storage capacity for hundreds of cars, and sufficient staff to have specialized roles in the workflow.
- ‘City’: small Branches. They usually keep extended business hours (e.g., 7.00 a.m. to 8.00 p.m.), have storage space for tens of cars, and small numbers of staff who are interchangeable in the workflow.
- ‘Agency’: service desks in hotels, travel agents, etc. They have storage space for few cars, and are operated on demand by part-time staff who will typically do the entire workflows for rental and return.

### E.1.3 Adopted Vocabularies

To illustrate vocabulary adoption, two car industry glossaries (also fictitious) have been introduced. One is in English; the other in German.

### E.1.4 Introductory Examples

Below are some elements of guidance, selected from the main part of the case study to illustrate how guidance is expressed and how fact types in the vocabulary support the formulation of guidance.

The intention is to provide an initial impression of what EU-Rent's business rules look like and how they are supported by fact types as expressed in EU-Rent's vocabulary, before going into the full detail of the case study. The examples also appear, with related elements of guidance, in the main body of the case study that follows this introduction.

The entries noted as "supporting fact types" are the fact types on which the guidance is directly based. Those noted as "related factual connections" indicate related parts of the vocabulary which would be of interest to a business person.

- |   |                          |  |
|---|--------------------------|--|
| 1 | Structural business rule | It is necessary that each <u>rental</u> has exactly one <u>requested car group</u> .   |
|   | Supporting fact type     | <u>rental</u> has <u>requested car group</u>   |
| 2 | Operative business rule  | It is obligatory that the <u>duration</u> of each <u>rental</u> is at most 90 days.  |
|   | Supporting fact type     | <u>rental</u> has <u>duration</u><br><u>duration</u> <sub>1</sub> is at most <u>duration</u> <sub>2</sub>  |
| 3 | Operative business rule  | It is obligatory that each <u>driver</u> of a <u>rental</u> is a <u>qualified driver</u> .   |
|   | Supporting fact types:   | <u>rental</u> has <u>driver</u>  |
|   | Related facts:           | the <u>noun concept</u> 'qualified driver' is a <u>category</u> of the <u>noun concept</u> 'driver'  |
| 4 | Operative business rule  | If the <u>drop-off location</u> of a <u>rental</u> is not the <u>EU-Rent site</u> of the <u>return branch</u> of the <u>rental</u> then it is obligatory that the <u>rental</u> incurs a <u>location penalty charge</u> .                                |
|   | Supporting fact types:   | <u>rental</u> has <u>drop-off location</u><br><u>rental</u> has <u>return branch</u><br><u>branch</u> is located at <u>EU-Rent site</u><br><u>rental</u> incurs <u>location penalty charge</u><br><u>thing</u> <sub>1</sub> is <u>thing</u> <sub>2</sub> |

- 5 Operative business rule *It is obligatory that the rental charge of a rental is calculated in the business currency of the rental.*
- Supporting fact types: rental *has* rental charge  
rental charge *is calculated in* business currency  
rental *has* business currency
- 6 Operative business rule *It is permitted that a rental is open only if an estimated rental charge is provisionally charged to the credit card of the renter of the rental.*
- Supporting fact types: rental *has* estimated rental charge  
estimated rental charge *is provisionally charged to* credit card  
renter *has* credit card  
rental *has* renter  
rental *is* open
- Related facts: *'being open' is a characteristic of the concept 'rental'*
- 7 Operative business rule *It is obligatory that at the actual return date/time of each in-country rental and each international inward rental the local area of the return branch of the rental owns the rented car of the rental.*
- Supporting fact types: rental *has* actual return date/time  
rental *has* return branch  
branch *is included in* local area  
local area *owns* rental car  
state of affairs *occurs at* date/time
- Related facts: *the noun concept 'rented car' is a role of the noun concept 'rental car'*  
*the noun concept 'return branch' is a role of the noun concept 'branch'*  
*the noun concept 'in-country rental' is a category of the noun concept 'rental'*  
*the noun concept 'international inward rental' is a category of the noun concept 'international rental'*  
*the noun concept 'international rental' is a category of the noun concept 'rental'*

8	Operative business rule	It is obligatory that <i>at the actual start date/time of each rental the fuel level of the rented car of the rental is full.</i>
	Supporting fact types:	<i>rental has actual start/date time</i> <i>rental has rented car</i> <i>rental car has fuel level</i> <i>state of affairs occurs at date/time</i>
	Related facts:	<i>the noun concept 'rented car' is a role of the noun concept 'rental car'</i> <i>fuel level is full or 7/8 or 3/4 or 5/8 or 1/2 or 3/8 or 1/4 or 1/8 or empty</i>
9	Structural business rule clarification	It is possible that the <i>notification date/time of a bad experience that occurs during a rental is after the actual return date/time of the rental.</i>
	Supporting fact types:	<i>bad experience occurs during rental</i> <i>bad experience has notification date/time</i> <i>rental has actual return date/time</i> <i>date/time<sub>1</sub> is after date/time<sub>2</sub></i>
	Related facts:	<i>the noun concept 'notification date/time' is a role of the noun concept 'date/time'</i> <i>the noun concept 'actual return date/time' is a role of the noun concept 'date/time'</i>
10	Operative business rule clarification	It is permitted that the <i>drop-off branch of a rental is not the return branch of the rental</i>
	Supporting fact types:	<i>rental has drop-off branch</i> <i>rental has return branch</i> <i>thing<sub>1</sub> is thing<sub>2</sub></i>

## E.2 EU-Rent Examples

The case study is presented in two parts. The first section illustrates EU-Rent's specification of its vocabulary business context, i.e., its use of the SBVR constructs to define the EU-Rent communities, bodies of shared meanings and vocabularies. The second section illustrates the contents of one of EU-Rent's vocabularies -- the EU-Rent English Vocabulary of the EU-Rent English Community (a speech community) -- along with its associated rule sets.

*Limitation of scope*

Some entries in the examples have been left informal in order to limit the overall size of the case study. They might, in a ‘real’ SBVR model, be expanded into substantial formal structures.

**E.2.1 The EU-Rent Vocabulary Business Context**

The entries in this section define the business context of EU-Rent’s several vocabularies -- i.e., its communities and subcommunities, its vocabularies and bodies of shared meanings, and how these elements inter-relate. Figure E.2 presents a partial instance diagram of the concepts and facts that express EU-Rent’s vocabulary business context.

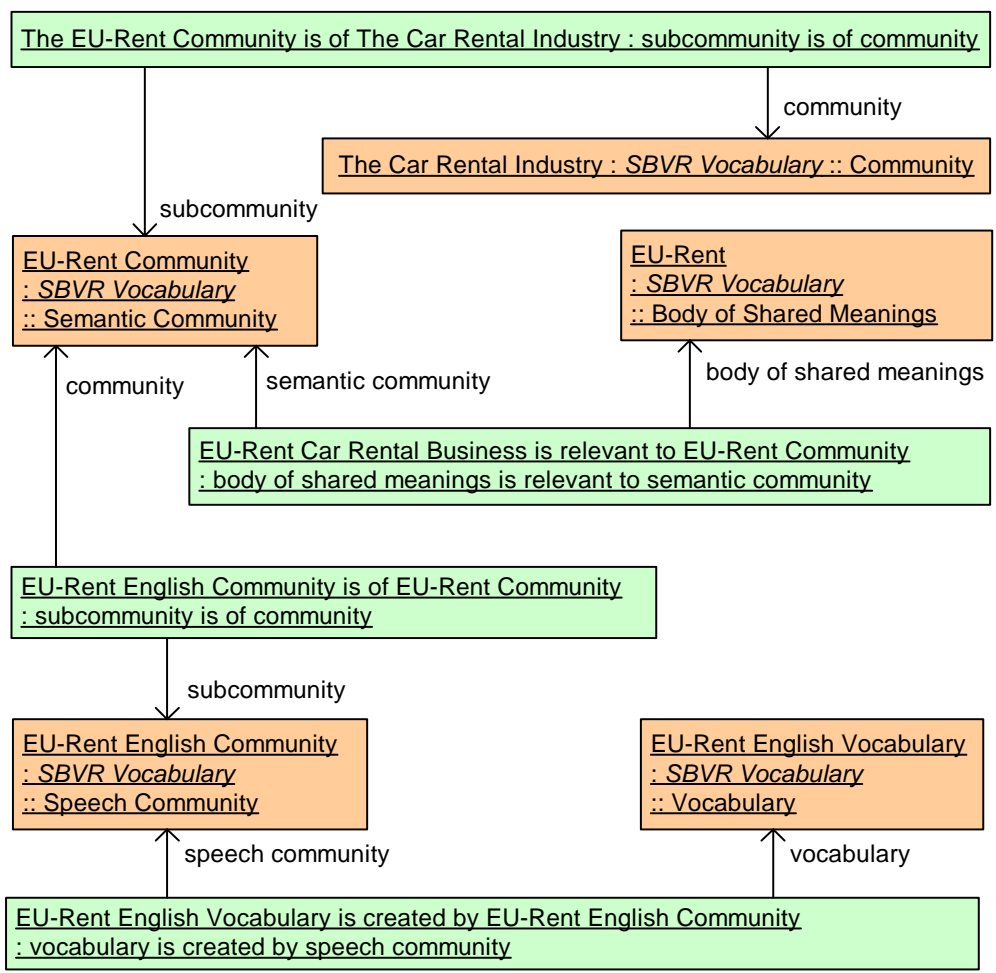


Figure E.2 - Instance diagram of concepts and facts expressing EU-Rent’s vocabulary business context

**E.2.1.1 EU-Rent Semantic Community**

The Car Rental Industry

Definition:

the semantic community that is the group of people who work in the business of renting cars



### Car Rental Business

- Definition: the body of shared meanings that is the set of concepts that are generally accepted as important across The Car Rental Industry
- Necessity: Car Rental Business is relevant to The Car Rental Industry.
- Necessity: Car Rental Business is relevant to The EU-Rent Community.

### EU-Rent

- Definition: the international car rental company that trades as “EU-Rent”.

### EU-Rent Car Rental Business

- Definition: the body of shared meanings that is the set of concepts that are important to The EU-Rent Community
- Necessity: EU-Rent Car Rental Business is relevant to The EU-Rent Community.

### The EU-Rent Community

- Definition: the semantic community that comprises all employees of EU-Rent and all others who share their body of concepts and use their vocabularies
- Necessity: The EU-Rent Community is a subcommunity of The Car Rental Industry

### EU-Rent HQ

- Definition: the organization unit that is EU-Rent’s world headquarters and management company
- Description: EU-Rent HQ sets global policy and owns the world-wide reservations system.

### EU-Rent HQ Staff

- Definition: the community that is the set of employees of EU-Rent HQ
- Necessity: The EU-Rent HQ Staff is a subcommunity of the EU-Rent community.

## **E.2.1.2 EU-Rent Speech Communities and Vocabularies**

### **E.2.1.2.1 Language-independent Vocabularies**

#### ISO Dictionary of International Symbols

- Definition: the vocabulary that is defined by ISO, of graphical symbols that have consistent meanings regardless of which natural languages they are used with
- Synonym: ISO-DIS
- Reference Scheme: ISO-DIS index
- Note: This is a fictitious standard. Work in this area is going on within ISO, but no standards have yet been published.

#### ISO-DIS

- Synonym: ISO Dictionary of International Symbols

### **E.2.1.2.2 EU-Rent English Community**

#### The EU-Rent English Community

- Definition: the speech community that is within The EU-Rent Community and has English as its primary natural language

Description: Most members of [The EU-Rent English Community](#) are employees of: [EU-Rent HQ](#), [EU-Rent CA](#), [EU-Rent GB](#), [EU-Rent IE](#), [EU-Rent US](#); trading partners of those EU-Rent companies; other EU-Rent companies who interact in English with them.

Necessity: [The EU-Rent English Community](#) *is of* [The EU-Rent Community](#).

### Car Rental Industry Standard Glossary

Definition: *the vocabulary that is defined in English by* [The Car Rental Industry](#)

Synonym: [CRISG](#)

Reference Scheme: [CRISG](#) terms

### CRISG

Synonym: [Car Rental Industry Standard Glossary](#)

### Merriam-Webster Unabridged Dictionary

Definition: *the vocabulary that* is the 2004 edition, published by Merriam-Webster

Synonym: [MWU](#)

Reference Scheme: [MWU](#) terms

### MWU

Synonym: [Merriam-Webster Unabridged Dictionary](#)

### EU-Rent English Vocabulary

Definition: *the vocabulary that is created by* [The EU-Rent English Community](#)

Necessity: [EU-Rent English Vocabulary](#) *incorporates* [MWU](#).

Necessity: [EU-Rent English Vocabulary](#) *incorporates* [ISO-DIS](#).

Necessity: [EU-Rent English Vocabulary](#) *incorporates* [ISO- CRISG](#).

Necessity: [CRISG](#) has precedence over [MWU](#).

Note: The necessity above means that if a signifier used in the EU-Rent English Vocabulary is implicitly understood - i.e., does not have an owned or explicitly adopted definition - it should first be looked up in CRISG, and if it is not there, then in MWU.

## **E.2.1.2.3 EU-Rent German Community**

### The EU-Rent German Community

Definition: *the speech community that is within* [The EU-Rent Community](#) and has German as its primary natural language

Description: Most members of [The EU-Rent German Community](#) are employees of: [EU-Rent DE](#); trading partners of EU-Rent DE; other EU-Rent companies who interact, in German, with [EU-Rent DE](#)

Necessity: [The EU-Rent German Community](#) *is of* [The EU-Rent Community](#).

### Deutsches Universalwörterbuch

Definition: *the vocabulary that* is the 2003 edition published by [Duden](#)

Synonym: [DUW](#)

Reference Scheme: [DUW](#) terms

### DUW

Synonym: [Deutsches Universalwörterbuch](#)

### Glossar für Autovermietungsgeschäft

Definition:	the <a href="#">vocabulary</a> that is defined in German by <a href="#">The Car Rental Industry</a>
Synonym:	<a href="#">GFA</a>
Reference Scheme:	<a href="#">GFA</a> terms

### GFA

Synonym:	<a href="#">Glossar für Autovermietungsgeschäft</a>
----------	---

### EU-Rent German Vocabulary

Definition:	the <a href="#">vocabulary</a> that is created by the <a href="#">EU-Rent German Community</a>
Necessity:	<a href="#">EU-Rent German Vocabulary</a> incorporates <a href="#">DUW</a> .
Necessity:	<a href="#">EU-Rent German Vocabulary</a> incorporates <a href="#">GFA</a> .
Necessity:	<a href="#">EU-Rent German Vocabulary</a> incorporates <a href="#">ISO-DIS</a> .
Necessity:	<a href="#">GFA</a> has precedence over <a href="#">DUW</a> .
Note:	The necessity above means that if a signifier used in the EU-Rent German Vocabulary does not have an owned or explicitly adopted definition, it should first be looked up in GFA, and if it is not there, then in DUW.

## **E.2.2 The EU-Rent English Vocabulary and Rules**

### **E.2.2.1 Concepts and Vocabulary**

#### **E.2.2.1.1 Car Movement**

This section illustrates the creation of a ‘building block’ of concepts and related vocabulary, defined once and used in more than one context. [car movement](#) is used in both [rental](#) and [car transfer](#) (logistical movement of a car by EU-Rent staff).

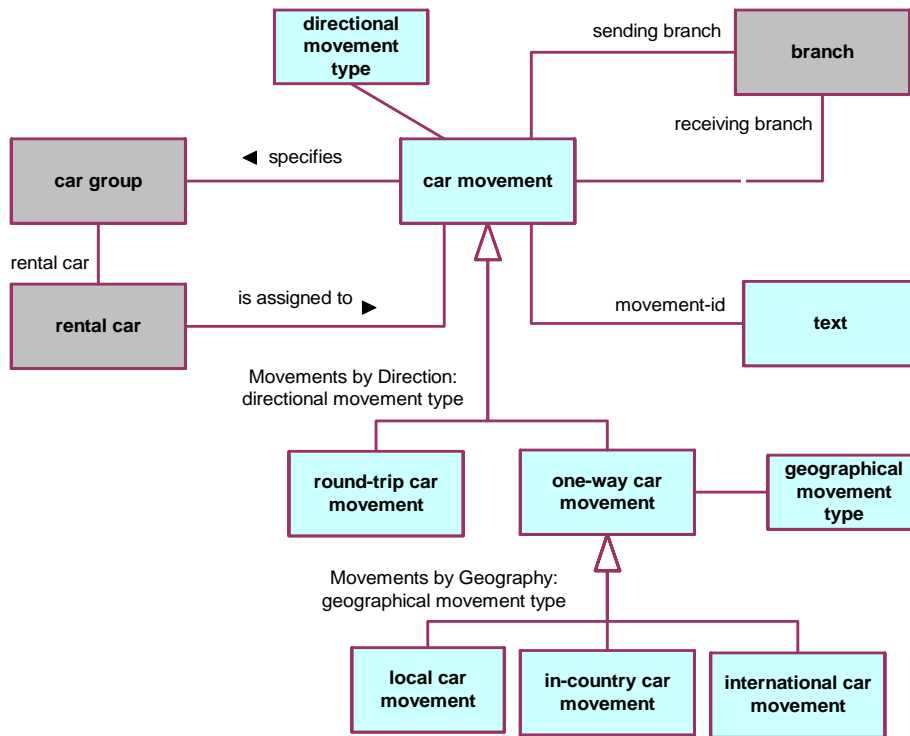


Figure E.3 - Car Movements

### car movement

- Definition: planned movement of a [rental car](#) of a specified [car group](#) from a [sending branch](#) to a [receiving branch](#)
- Reference Scheme: [movement-id](#)
- Description: A car movement meets the business requirement that a car of a given group has to be moved between branches (“we need to move a full-size car from the London City branch to the Heathrow Airport branch”). A specific car will be assigned to it at some time, not necessarily when the requirement is first identified.
- Note: ‘[car movement](#)’ plays roles in both ‘[rental](#)’ and ‘[car transfer](#)’ and car movements are scheduled in these roles.

### car movement has movement-id

Necessity: each [car movement](#) has exactly one [movement-id](#).

### car movement has receiving branch

Necessity: each [car movement](#) has exactly one [receiving branch](#).

### car movement has sending branch

Necessity: each [car movement](#) has exactly one [sending branch](#).

### car movement specifies car group

Necessity: [each car movement specifies exactly one car group](#).

### directional movement type

Concept Type: [categorization type](#)

Definition: [concept that specializes the concept 'car movement' and that classifies a car movement](#) based on whether the car is moved to a different branch or not

### geographical movement type

Concept Type: [categorization type](#)

Definition: [concept that specializes the concept 'one-way car movement' and that classifies a one-way car movement](#) based on whether it crosses local area or international boundaries

### in-country car movement

Concept Type: [geographical movement type](#)

Definition: [one-way car movement that is not in-area and is not international](#)

Note: This means that the movement is between two local areas in the same country.

Necessity: [in-country car movement is included in Movements by Geography](#).

### international car movement

Definition: [geographical movement type](#)

Definition: [one-way car movement that is international](#)

Necessity: [international car movement is included in Movements by Geography](#).

### local car movement

Definition: [geographical movement type](#)

Definition: [one-way car movement that is in-area](#)

Necessity: [local car movement is included in Movements by Geography](#).

### movement-id

Concept Type: [role](#)

Definition: [text that is assigned by EU-Rent as unique identifier of car movement](#)

Note: A given car could be moved more than once between the same two branches, perhaps even on the same day. Movement-id is needed to provide a reliable reference scheme.

### Movements by Direction

Definition: [segmentation that is for the concept 'car movement' and subdivides car movements](#) based on [directional movement type](#)

Necessity: [Movements by Direction contains the categories 'one-way movement' and 'round-trip movement'](#).

### Movements by Geography

Definition: [segmentation that is for the concept 'one-way car movement' and subdivides one-way car movements](#) based on [geographical movement type](#)

Necessity: [Movements by Geography contains the categories 'in-country car movement' and 'international car movement' and 'local car movement'](#).

### **one-way car movement**

Concept Type: [directional movement type](#)  
Definition: [car movement](#) *that is not round-trip*  
Necessity: [one-way car movement](#) *is included in* [Movements by Direction](#).

### **sending branch**

Concept Type: [role](#)  
Definition: [branch](#) *that* is the origin of a [car movement](#)

### **rental car is assigned to car movement**

Necessity: *At most one* [rental car](#) *is assigned to each* [car movement](#).  
Necessity: *The* [rental car](#) *that is assigned to a* [car movement](#) *is of the* [car group](#) *specified by the* [car movement](#).

### **receiving branch**

Concept Type: [role](#)  
Definition: [branch](#) *that* is the destination of a [car movement](#)

### **round-trip car movement**

Concept Type: [directional movement type](#)  
Definition: [car movement](#) *that is round-trip*  
Necessity: [round-trip car movement](#) *is included in* [Movements by Direction](#).

### *Characteristics*

#### **car movement being in-area**

Concept Type: [characteristic](#)  
Definition: [car movement](#) *having* [receiving branch](#) *that is included in the* [local area](#) *of the* [sending branch](#) *of the* [car movement](#)

#### **car movement being international**

Concept Type: [characteristic](#)  
Definition: [car movement](#) *having* [country of sending branch](#) *that is not the* [country of receiving branch](#) *of the* [car movement](#)

#### **car movement being round-trip**

Concept Type: [characteristic](#)  
Definition: [car movement](#) *having* [sending branch](#) *that is the* [receiving branch](#) *of the* [car movement](#)

### **E.2.2.1.2 Car Transfers**

“Car Transfer” illustrates two features of SBVR usage:

- Use of a “building block” as defined in [car transfer](#) includes [one-way car movement](#).
- A trade-off of redundancy (in the sense of defining a concept both directly and indirectly) against simplification of logical formulation and representation.

For example, EU-Rent defines transferred car as the concept ‘rental car that is assigned to the one-way car movement that is included in a car transfer.’

*Note that both of these are matters of practice, not mandated by SBVR. Decisions on reuse and redundancy are business decisions made by the semantic community (here, EU-Rent) to help it manage its body of shared meanings and vocabularies.*

Generally, derivable necessities are not restated. For example, a car movement has exactly one sending branch; a car transfer includes exactly one one-way car movement; the pick-up branch of a car transfer is the sending branch of the included one-way car movement. There is no restated necessity: car transfer has exactly one pick-up branch.

*Again, this is a matter of practice. If a semantic community would prefer the derivable necessities to be explicit, SBVR will support it.*

“Redundant” concepts are specified using structural rules (necessities). For example:

car transfer has transferred car

Necessity: The transferred car of a car transfer is the rental car that is assigned to the one-way car movement that is included in the car transfer.

One extension of the approach is carry-over of the segmentation of car transfers by geographical movement type (local, in-country and international). The segmentation and categorization type are not repeated. The categories of car transfer are specified as corresponding to the respective categories of car movement.

Note: fact types derived from inclusion of fixed period and one-way car movement (e.g., car transfer has transfer pick-up branch) are not shown on the diagram, but are defined in the text below.

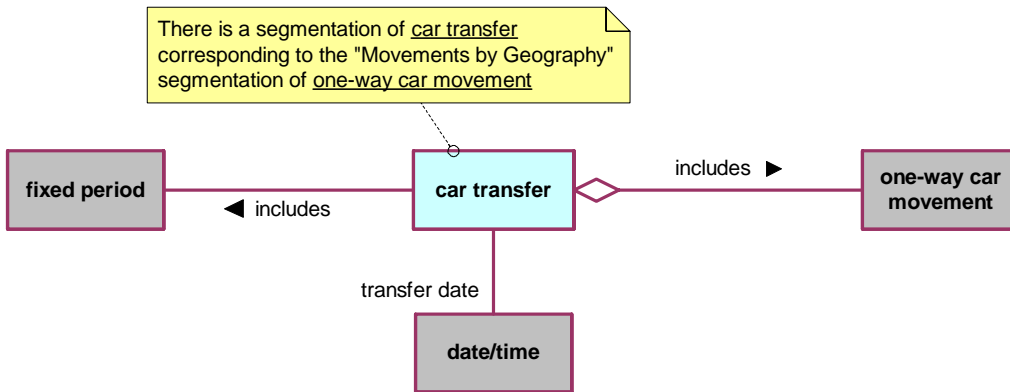


Figure E.4 - Car Transfers

car transfer

Definition: logistical action by EU-Rent, transferring a rental car between branches on a given day

car transfer has transfer date

Necessity: each car transfer has exactly one transfer date.

car transfer has transfer drop-off branch

Necessity: the transfer drop-off branch of a car transfer is the receiving branch of the one-way car movement included in the car transfer.

### car transfer has transfer drop-off date/time

Necessity: *the transfer drop-off date/time of a car transfer is the actual end date/time of the fixed period included in the car transfer.*

### car transfer has transfer pick-up branch

Necessity: *the transfer pick-up branch of a car transfer is the sending branch of the one-way car movement included in the car transfer.*

### car transfer has transfer pick-up date/time

Necessity: *the transfer drop-off date/time of a car transfer is the actual start date/time of the fixed period included the car transfer.*

### car transfer has transferred car

Necessity: *The transferred car of a car transfer is the rental car that is assigned to the one-way movement that is included in the car transfer.*

### car transfer includes car movement

Necessity: *each car transfer includes exactly one car movement.*

### car transfer includes fixed period

Necessity: *each car transfer includes exactly one fixed period.*

Note: EU-Rent does not schedule car transfers within their transfer dates. It wants to know, at the end of the transfer, the actual pick-up and drop-off times. By the time EU-Rent is interested in the period of the transfer, it is in the past - and so is fixed.

### in-country car transfer

Definition: *car transfer that includes an in-country car movement*

### international return

Definition: *car transfer that includes an international car movement*

### local car transfer

Definition: *car transfer that includes a local car movement*

### transfer date

Concept Type: *role*

Definition: *date that a car transfer is scheduled for*

Note: The transfer date is usually scheduled in advance. The pick-up date/time and drop-off date/time are the actual times during the day, notified when the transfer is completed.

### transfer drop-off branch

Concept Type: *role*

Definition: *branch from which the transferred car of a car transfer is picked up*

### transfer drop-off date/time

Concept Type: *role*

Definition: *date/time when the transferred car of a car transfer is actually dropped off*



### **transfer pick-up branch**

Concept Type: [role](#)

Definition: [branch](#) from which **the** [transferred car](#) *of a* [car transfer](#) *is picked up*

### **transfer pick-up date/time**

Concept Type: [role](#)

Definition: [date/time](#) when **the** [transferred car](#) *of a* [car transfer](#) *is actually picked up*

### **transferred car**

Concept Type: [role](#)

Definition: [rental car](#) relocated by **a** [car transfer](#)

### E.2.2.1.3 EU-Rent Locations

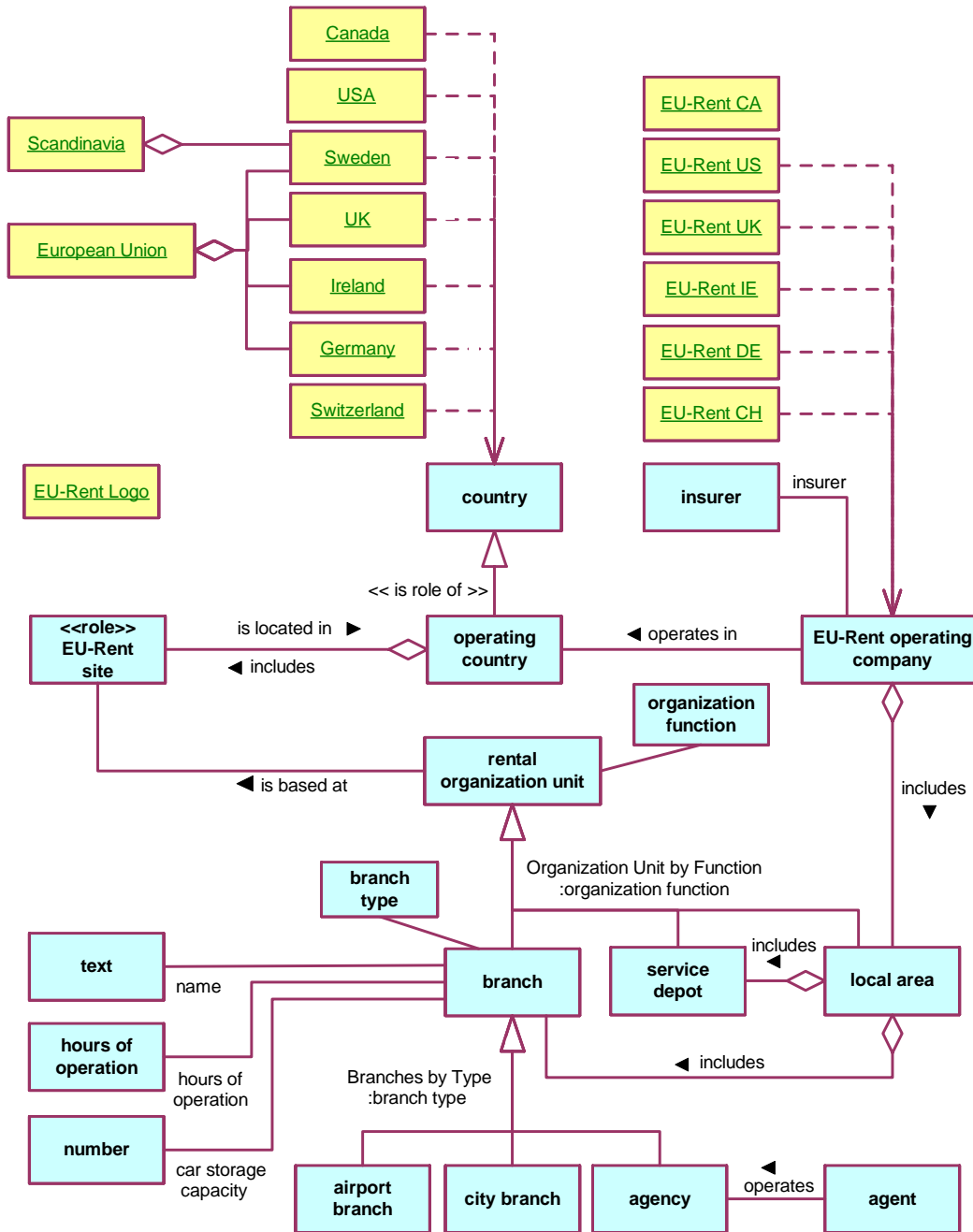


Figure E.5- EU-Rent Locations

#### agency

Concept Type:

[branch type](#)

Definition: [branch](#) that does not have a EU-Rent location and has minimal car storage and has on-demand operation

Necessity: the [concept 'agency'](#) is included in [Branches by Type](#).

### [agent](#)

Definition: organization external to EU-Rent that carries out car rental business on behalf of EU-Rent

Example: Hotel, travel agent.

### [agent operates agency](#)

Note: Operation is usually by part-time staff who will carry out the entire workflows for rental and return.

### [airport branch](#)

Concept Type: [branch type](#)

Definition: [branch](#) that has a EU-Rent location and has large car storage and has 24-7 operation

Note: This kind of [branch](#) is usually at or near a major airport or rail terminal and has sufficient staff to have specialized roles in the workflow.

Necessity: the [concept 'airport branch'](#) is included in [Branches by Type](#).

### [branch](#)

Concept Type: [organization function](#)

Definition: [rental organization unit](#) that has rental responsibility

Necessity: the [concept 'branch'](#) is included in [Organization Units by Function](#).

### [branch has car storage capacity](#)

Concept Type: [is-property-of fact type](#)

### [branch has country](#)

Necessity: The [country](#) of a [branch](#) is the [operating country](#) of the [operating company](#) that includes the [local area](#) that includes the [branch](#).

### [branch has hours of operation](#)

Concept Type: [is-property-of fact type](#)

### [branch has name](#)

Concept Type: [is-property-of fact type](#)

### [branch is included in local area](#)

Concept Type: [partitive fact type](#)

Necessity: Each [branch](#) is included in exactly one [local area](#).

### [branch type](#)

Definition: [concept](#) that specializes the [concept 'branch'](#) and that classifies a [branch](#) based on [hours of operation](#) and [car storage capacity](#)

### Branches by Type

- Definition: [segmentation](#) that is for the concept 'branch' and subdivides branches based on [branch type](#)
- Necessity: [Branches by Type](#) contains the categories 'airport branch' and 'city branch' and 'agency'.

### car storage capacity

- Concept Type: [role](#)
- Definition: [number of rental cars](#) that can be stored at the [EU-Rent site](#) that is the base for a [branch](#)
- Note: Some of the capacity at a branch's site might be taken up by cars that are not available for rental -- e.g., cars awaiting service or transfer to other branches; hotel guests' cars.

### city branch

- Concept Type: [branch type](#)
- Definition: [branch](#) that has a [EU-Rent location](#) and has moderate car storage and has long business hours
- Note: This kind of [branch](#) is usually in a city center and has small numbers of staff who are interchangeable in the workflow.
- Necessity: [city branch](#) is included in [Branches by Type](#).

### country

- Source: MWU (1,2b) ["country"]
- Note: Has pre-defined population (below)

### EU-Rent Logo

- Definition: upper-case 'EU' followed by a hyphen followed by upper-case 'R' followed by lower-case 'ent' all in the company's custom-designed font and using the company's standard color on a white ground

### EU-Rent operating company

- Definition: operating [company of EU-Rent](#)
- Note: In each [operating country](#) EU-Rent has a [EU-Rent operating company](#) that:
- adapts global policy to local regulation, custom and practice
  - selects which car models will be purchased for each car group
  - sets rental tariffs
- Note: Has pre-defined population (below)

### EU-Rent operating company includes local area

- Synonymous Form: [local area](#) is included in [EU-Rent operating company](#)

### EU-Rent operating company operates in operating country

- Necessity: Each [EU-Rent site](#) is located in exactly one [operating country](#).

### EU-Rent site

- Concept Type: [role](#)
- Definition: [location](#) used by [EU-Rent](#)

### EU-Rent site is base for rental organization unit

Synonymous Form: [rental organization unit is based at EU-Rent site](#)

### European Union

Definition: the geopolitical area that is composed of [Sweden](#) and [Germany](#) and [Ireland](#) and [UK](#) and ...

### hours of operation

Definition: the times during which a facility is open for business

Example: 24 hours per day, 7 days a week; 7:00 am to 8:00 pm; on demand.

### local area

Concept Type: [organization function](#)

Definition: [rental organization unit](#) that has area management responsibility

Description: A [local area](#) contains a number of [branches](#) for [rental car](#) pick-up and return and a number of [service depots](#) that maintain and repair EU-Rent's [rental cars](#).

Necessity: [service depot is included in Organization Units by Function](#).

### local area includes branch

Synonymous Form: [branch is included in local area](#)

### local area includes service depot

Synonymous Form: [service depot is included in local area](#)

### local area is included in operating company

Concept Type: [partitive fact type](#)

Necessity: Each [local area](#) is included in exactly one [EU-Rent operating company](#).

### location

Source: MWU (1a) ["location"]

### name

Concept Type: [role](#)

Concept Type: [text](#)

Source: MWU (1a) ["name"]

### operating company

See: [EU-Rent operating company](#)

### operating country

Concept Type: [role](#)

Definition: [country](#) in which EU-Rent does business

### organization function

Concept Type: [categorization type](#)

Definition: [concept](#) that specializes the [concept](#) '[rental organization unit](#)' and that classifies a [rental organization unit](#) by its functional role in EU-Rent

## Organization Units by Function

- Definition: [segmentation](#) that *is for the* [concept 'rental organization unit'](#) and *subdivides* [rental organization units](#) based on [organization function](#)
- Necessity: [Organization Units by Function](#) *contains* the [categories 'branch'](#) and ['local area'](#) and ['service depot'](#).

## rental organization unit

- Concept Type: [role](#)
- Definition: organization unit that operates part of EU-Rent's car rental business

## rental organization unit is based at EU-Rent site

- Concept Type: [associative fact type](#)
- Necessity: Each [rental organization unit](#) *is based at exactly one* [EU-Rent site](#).

## service depot

- Concept Type: [organization function](#)
- Definition: [rental organization unit](#) that *has servicing responsibility*
- Necessity: [service depot](#) *is included in* [Organization Units by Function](#).

## service depot is included in local area

- Concept Type: [partitive fact type](#)
- Necessity: Each [service depot](#) *is included in exactly one* [local area](#).

## Scandinavia

- Definition: the geographic area that *is composed of* [Sweden](#) and ...

## *Characteristics*

## rental organization unit having 24-7 operation

- Concept Type: [characteristic](#)
- Definition: the [rental organization unit](#) *has* [hours of operation](#) that are 24 hours per day, 7 days per week

## rental organization unit having a EU-Rent location

- Concept Type: [characteristic](#)
- Definition: the [rental organization unit](#) *is based at a* [EU-Rent site](#) that is owned by [EU-Rent](#)
- Note: [Some things](#) *are based at* [EU-Rent sites](#) that are owned by third parties such as hotels and travel agents.

## rental organization unit having area responsibility

- Concept Type: [characteristic](#)
- Definition: the [rental organization unit](#) includes organization units for which it has responsibility to coordinate operations and ensure resources

## rental organization unit having large car storage

- Concept Type: [characteristic](#)

Definition: [the rental organization unit](#) *has* [car storage](#) *that accommodates hundreds of* [rental cars](#)

### **rental organization unit *having long business hours***

Concept Type: [characteristic](#)

Definition: [the rental organization unit](#) *has* [hours of operation](#) *that correspond to an extended business day*

Example: 7:00 am to 8:00 pm, six days per week.

### **rental organization unit *having minimal car storage***

Concept Type: [characteristic](#)

Definition: [the rental organization unit](#) *has* [car storage](#) *that can accommodate a small number of* [rental cars](#)

### **rental organization unit *having moderate car storage***

Concept Type: [characteristic](#)

Definition: [the rental organization unit](#) *has* [car storage](#) *that can accommodate tens of* [rental cars](#)

### **rental organization unit *having on-demand operation***

Concept Type: [characteristic](#)

Definition: [the rental organization unit](#) *has* [hours of operation](#) *that are flexible in response to customer demand*

### **rental organization unit *having rental responsibility***

Concept Type: [characteristic](#)

Definition: [the rental organization unit](#) is responsible for operation of customer-facing rental business

### **rental organization unit *having servicing responsibility***

Concept Type: [characteristic](#)

Definition: [the rental organization unit](#) is responsible for maintenance and servicing of rental cars

*Pre-defined population:* [country](#)

### **Canada**

Concept Type: [individual concept](#)

General Concept: [country](#)

### **Germany**

Concept Type: [individual concept](#)

General Concept: [country](#)

Synonym: [DE](#)

### **Ireland**

Concept Type: [individual concept](#)

General Concept: [country](#)

### Sweden

Concept Type: [individual concept](#)  
General Concept: [country](#)

### Switzerland

Concept Type: [individual concept](#)  
General Concept: [country](#)  
Synonym: [CH](#)

### UK

Concept Type: [individual concept](#)  
General Concept: [country](#)  
Synonym: [United Kingdom](#)

### United States

Concept Type: [individual concept](#)  
General Concept: [country](#)  
Synonym: [USA](#)

*Pre-defined population:* [EU-Rent operating company](#)

### EU-Rent CA

Definition: the [EU-Rent operating company](#) that is located in [Canada](#)

### EU-Rent DE

Definition: the [EU-Rent operating company](#) that is located in [Germany](#)

### EU-Rent IE

Definition: the [EU-Rent operating company](#) that is located in [Ireland](#)

### EU-Rent UK

Definition: the [EU-Rent operating company](#) that is located in [UK](#)

### EU-Rent US

Definition: the [EU-Rent operating company](#) that is located in [United States](#)



### E.2.2.1.4 Car Specifications

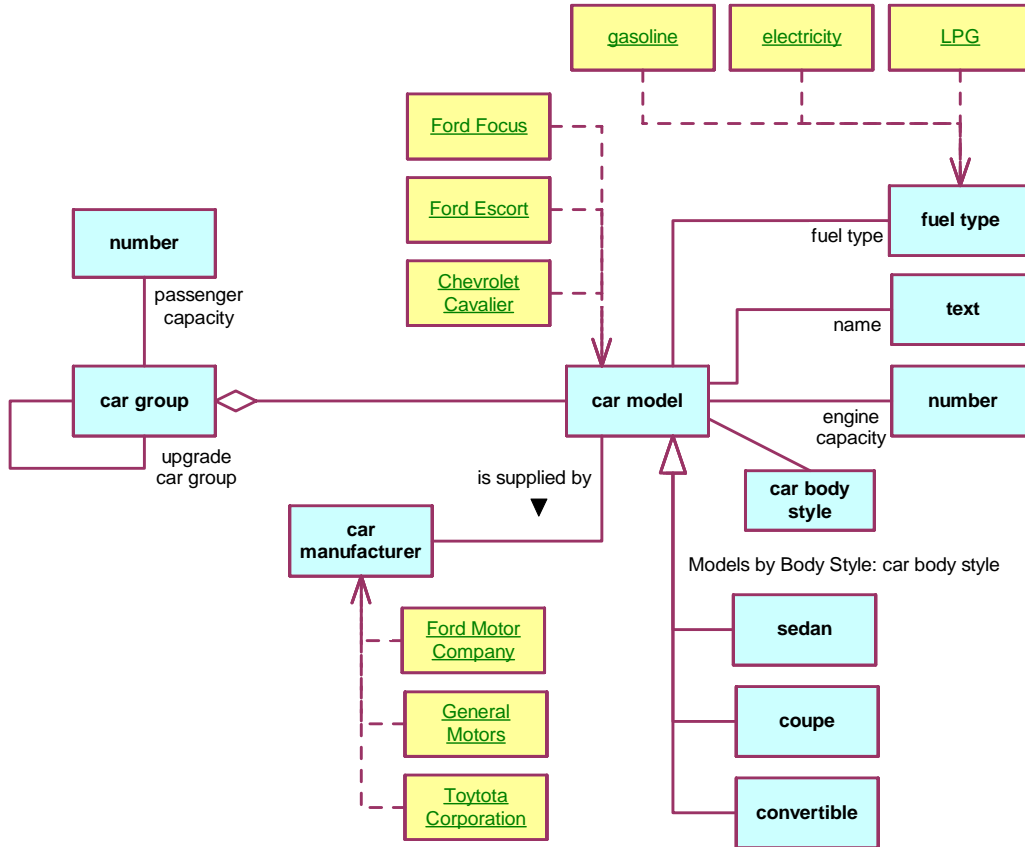


Figure E.6 - Car Specifications

#### car body style

Concept Type:

categorization type

Definition:

concept that specializes the concept 'car model' and that classifies a car model based on industry-defined criteria

Source:

CRISG (2a) ["body style"]

#### car group

Source:

CRISG ["rate group"]

Note:

Different models of car are offered for rental, organized into groups which establish a price point.

Necessity:

Each car model of a car group is charged at the rental rates of the car group.

#### car group has passenger capacity

Concept Type:

is-property-of fact type

Definition: [the car group has the passenger capacity that](#) defines the capacity of a [car](#) of [the car group](#), with [the driver](#) counting as [One](#)

### [car group has upgrade car group](#)

Definition: [the car group has an upgrade car group that](#) is used when no car of a [requested car group](#) is available

### [car manufacturer](#)

Definition: producer of cars that EU-Rent has decided to do business with

Note: Has pre-defined population (below)

### [car model](#)

Source: CRISG ["car model"]

Note: Has pre-defined population (below)

Note: Cars of a given model are all built to the same specification, e.g., body style, engine size, fuel type. EU-Rent bases its model names on those assigned by the car manufacturers, but sometimes has to extend them to distinguish models with different engine sizes and numbers of doors.

Reference Scheme: [name of car model](#)

### [car model is included in car group](#)

Necessity: [Each car model is included in exactly one car group.](#)

### [car model has engine capacity](#)

Concept Type: [is-property-of fact type](#)

### [car model is supplied by car manufacturer](#)

Synonymous Form: [car manufacturer supplies car model](#)

Necessity: [Each car model is supplied by exactly one car manufacturer.](#)

### [car model has fuel type](#)

Concept Type: [is-property-of fact type](#)

Definition: Some car models can have more than one fuel – e.g., can switch between gasoline and LPG, or between electricity and gasoline.

Necessity: [Each car model has at least one fuel type.](#)

### [car model has name](#)

Concept Type: [is-property-of fact type](#)

### [convertible](#)

Concept Type: [car body style](#)

Source: CRISG ["convertible"]

Necessity: [convertible is included in Models by Body Style.](#)

### [coupe](#)

Concept Type: [car body style](#)

Source: CRISG ["coupe"]  
Necessity: [coupe](#) *is included in* [Models by Body Style](#).

### engine capacity

Concept Type: [role](#)  
Definition: [number](#) **that** indicates the engine cylinder capacity in cubic centimeters  
Source: CRISG ["engine size"]

### fuel type

Source: CRISG ["fuel type"]  
Note: Has pre-defined population (below)

### Models by Body Style

Definition: [segmentation](#) **that is for the** [concept 'car model'](#) **and subdivides** [car models](#) based on [car body style](#)  
Necessity: [Models by Body Style](#) *contains* the [categories 'convertible' and 'coupe' and 'sedan'](#).

### passenger capacity

Concept Type: [role](#)  
Definition: [number](#) that is the count of adults, including the driver, that the car can comfortably hold

### sedan

Concept Type: [car body style](#)  
Source: CRISG ["sedan"]  
Necessity: [sedan](#) *is included in* [Models by Body Style](#).

### upgrade car group

Concept Type: [role](#)  
Definition: [car group](#) from which cars may be offered for rental if there are no cars available in another [requested car group](#)

*Pre-defined Population:* [car model](#)

### Chevrolet Cavalier

Concept Type: [individual concept](#)  
General Concept: [car model](#)

### Ford Focus

Concept Type: [individual concept](#)  
General Concept: [car model](#)

### Ford Escort

Concept Type: [individual concept](#)  
General Concept: [car model](#)

*Pre-defined Population:* [car group](#)

### Economy

Source: CRISG ["economy group"]  
General Concept: [car group](#)

### Compact

Source: CRISG ["compact group"]  
General Concept: [car group](#)

### Intermediate

Source: CRISG ["intermediate group"]  
General Concept: [car group](#)

### Full Size

Source: CRISG ["fullsize group"]  
General Concept: [car group](#)

*Pre-defined Population:* [car manufacturer](#)

### Ford Motor Company

Concept Type: [individual concept](#)  
General Concept: [car manufacturer](#)

### General Motors

Concept Type: [individual concept](#)  
General Concept: [car manufacturer](#)

### Toyota Corporation

Concept Type: [individual concept](#)  
General Concept: [car manufacturer](#)

*Pre-defined Population:* [fuel type](#)

### Electricity

Concept Type: [individual concept](#)  
Source: CRISG ["electric fuel"]

### Gasoline

Concept Type: [individual concept](#)  
Source: CRISG ["gasoline"]  
General Concept: [fuel type](#)  
Synonym: [petrol](#) [UK]  
Synonym: [benzin](#) [DE]  
Synonym: [essence](#) [FR]

LPG

Concept Type: [individual concept](#)  
General Concept: [fuel type](#)  
Source: CRISG ["liquefied petroleum gas"]

**E.2.2.1.5 Rentals**

There are some trade offs of redundancy and reuse (and, hence, a bigger vocabulary) against some simplification of formulation and expression.

Note: fact types derived from inclusion of [rental period](#) and [car movement](#) (e.g., [rental has pick-up branch](#)) are not shown on the diagram, but are defined in the text.

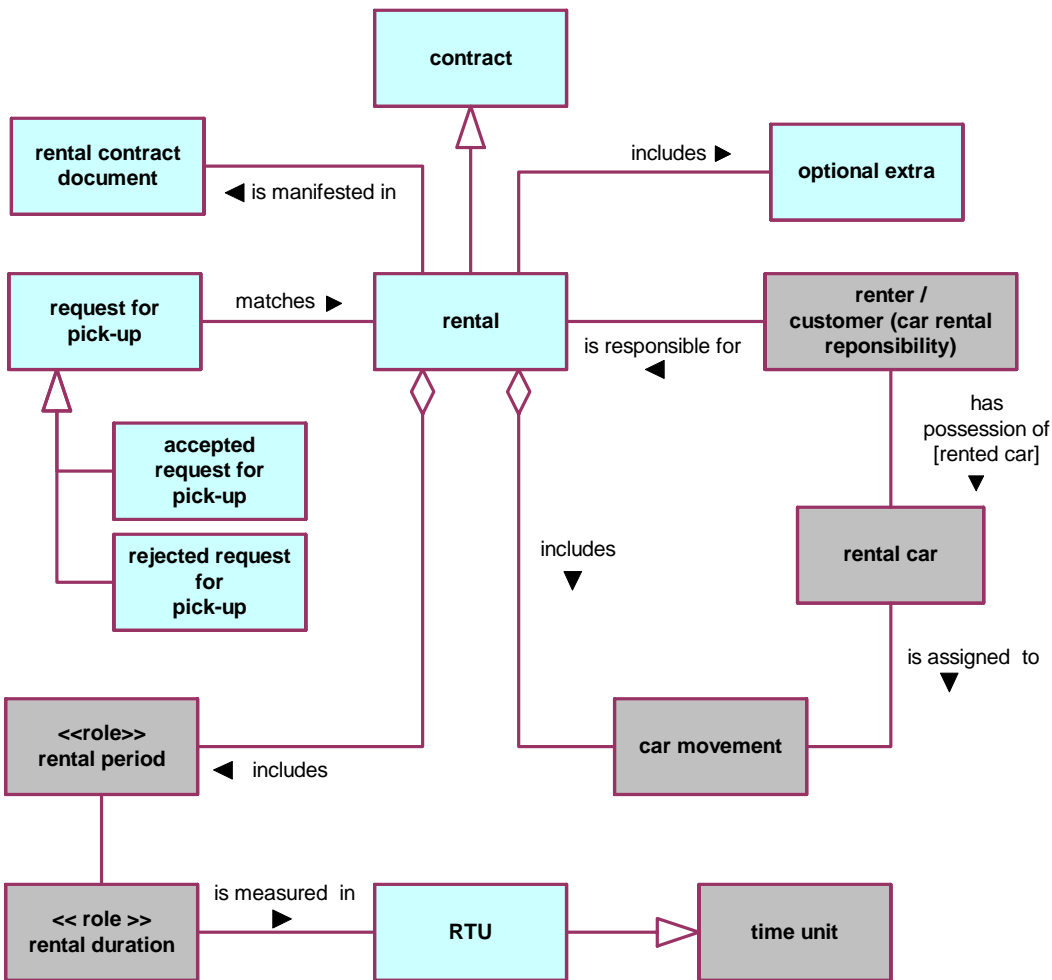


Figure E.7 - Rentals

[accepted request for pick-up](#)

Definition: [request for pick](#) up that is accepted by EU-Rent

Necessity: The [request for pick-up](#) *matches* exactly one [rental](#).  
Necessity: The [renter](#) *presents* a [valid credit card](#).  
Necessity: The [renter](#) *provides* [current contact details](#).  
Necessity: Each [driver of the rental](#) *has* a [valid driver license](#).

### actual pick-up date/ time

Concept Type: [role](#)  
General Concept: [date/time](#)  
Definition: [date/time](#) when the [rented car of a rental](#) *is picked up* by the [renter](#)

### actual return date/time

Concept Type: [role](#)  
General Concept: [date/time](#)  
Definition: [date/time](#) when a [rented car](#) *is returned* to EU-Rent

### current contact details

Concept Type: [role](#)  
Definition: [contact details](#) *that* have been confirmed as up-to-date by the [renter](#)

### optional extra

Definition: Item that may be added to a [rental](#) at extra charge if the [renter](#) so chooses  
Example: [One-way rental, fuel pre-payment, additional insurances, fittings \(child seats, satellite navigation system, ski rack\)](#)  
Source: CRISG ["optional extra"]

### pick-up branch

Necessity: The [pick-up branch of a rental](#) *may not be changed*.  
Note: If the [renter](#) wishes to change the [pick-up branch](#) of a [rental](#), EU-Rent regards it as a cancellation and a new [rental](#).

### rejected request for pick-up

Definition: [request for pick up](#) that is rejected by EU-Rent

### rental

Definition: [contract](#) *with* [renter](#) specifying use of a car of a [car group](#) for a [rental period](#) and a [car movement](#)  
Dictionary Basis: contract for use of a rental car by a renter for an agreed period under the rental company's terms and conditions for rental. [CRISG]

### rental contract document

Definition: information artifact that is the manifestation of a [rental](#)

### rental duration

Concept Type: [role](#)  
Definition: [duration](#) *used to calculate* a [rental charge](#)

### rental duration is measured in rental time unit

- Necessity: Each rental duration is measured in a whole number of rental time units.
- Example: A rental with an end date/time that was 11 days and 7 hours after its start date/time would have a rental duration of 1 x 1-week RTU plus 5 x 1-day RTU.  
If EU-Rent were to introduce a 3-day RTU, this would change to 1 x 1-week RTU plus 1 x 3-day RTU plus 2 x 1-day RTU.

### rental has actual pick-up date/time

- Necessity: The actual pick-up date/time of a rental is the actual start date/time of the rental period of the rental.

### rental has actual return date/time

- Necessity: The actual return date/time of a rental is the actual end date/time of the rental period of the rental.

### rental has pick-up branch

- Necessity: the pick-up branch of a rental is the sending branch of the car movement included in the rental.

### rental has rental duration

- Necessity: The rental duration of a rental is the duration of the variable period included in the rental.

### rental has requested car group

- Necessity: The requested car group of a rental is the car group that is specified in the car movement that is included in the rental.
- Possibility: The requested car group of an advance rental is changed before the actual pick-up date/time of the advance rental.
- Necessity: The requested car group of an advance rental is not changed after the actual pick-up date/time of the advance rental.

### rental has return branch

- Necessity: the return branch of a rental is the receiving branch of the car movement included in the rental

### rental has scheduled pick-up date/time

- Necessity: The scheduled pick-up date/time of a rental is the scheduled start date/time of the rental period of the rental.

### rental has scheduled return date/time

- Necessity: The scheduled pick-up date/time of a rental is the scheduled end date/time of the rental period of the rental.

### rental includes car movement

- Concept Type: partitive fact type
- Necessity: Each rental includes exactly one car movement
- Note: The car movement may be changed by changing the return branch.

### rental includes rental period

- Concept Type: [partitive fact type](#)
- Necessity: Each [rental](#) *includes* exactly one [rental period](#).
- Note: The [rental period](#) may be changed by rescheduling at the renter's request, by early or late arrival for rental, and by late return from rental.

### rental is manifested in rental contract document

- Concept Type: [associative fact type](#)

### rental period

- Concept Type: [role](#)
- Definition: [variable period](#) *that is included in a rental*

### rented car

- Concept Type: [role](#)
- Definition: [rental car](#) *that is assigned a rental*

### rented car is assigned to rental

- Necessity: The [rented car](#) *that is assigned to a rental* is the [rental car](#) *that is assigned to the car movement that is included in the rental*.

### renter has possession of rented car

- Definition: the [renter](#) has the [rented car](#) for use on rental

### request for pick-up

- Definition: request from a [renter](#) to pick up the [rental car](#) of a [rental](#) that has been reserved by him

### request for pick-up matches rental

- Necessity: The [rental is assigned](#) (has a rental car assigned to it).
- Necessity: The [pick-up branch of the rental](#) is the [branch](#) at which the request is made.
- Necessity: The [renter of the rental](#) is the person making the request for pick-up.
- Necessity: The [scheduled start date of the rental](#) is the day of the request for pick-up.
- Note: This entry is partly informal in order to limit the case study size.

### requested car group

- Concept Type: [role](#)
- Definition: [car group](#) that is requested for a rental
- Necessity: *At a given date/time each rental has exactly one requested car group.*

### return branch

- Concept Type: [role](#)
- Definition: [branch](#) stipulated in the [rental contract](#) for return of the [rented car](#)
- Note: If the renter does not return the car to this branch, a penalty charge will be levied.
- Necessity: Each [rental](#) *has exactly one return branch at a given date/time.*



Possibility: [The return branch of a rental is changed before the actual return date/time of the rental.](#)

### scheduled pick-up date/time

Definition: [date/time](#) at which a [rented car](#) is scheduled to be picked up from EU-Rent

Note: [The possibilities and necessities for changing the start date/time of variable period apply to scheduled pick-up date/time of rental.](#)

### scheduled return date/time

Definition: [date/time](#) at which a [rented car](#) is scheduled to be returned to EU-Rent

Note: [The possibilities and necessities for changing the end date/time of variable period apply to scheduled return date/time.](#)

### valid credit card

Concept Type: [role](#)

Definition: [credit card](#) that is acceptable for payment of the [rental charges of the rental](#) for which it is presented

Necessity: The card is of a type that EU-Rent accepts.

Necessity: “Expiry date” on the [valid credit card](#) is after the [scheduled end date/time of the rental](#).

Necessity: “Cardholder” is the person presenting the card.

Note: This entry is informally defined in order to limit the case study size.

### valid driver license

Concept Type: [role](#)

Definition: [driver license](#) that is acceptable for the [rental](#) for which it is presented

Necessity: “Expiry date” on the [valid driver license](#) is after the [scheduled end date/time of the rental](#).

Necessity: “Driver” is the person presenting the license.

Necessity: The rented car falls within “vehicle types”.

Necessity: The license is legally acceptable in the country of the pick-up branch.

Note: This entry is informally defined in order to limit the case study size.

#### **E.2.2.1.6 Rental Categorization**

This section defines some categorizations of rental, which enable some subsequent simplification of formulations and representations.

It also introduces the use of characteristic type for defining states - see [rental state](#) and [advance rental state](#).

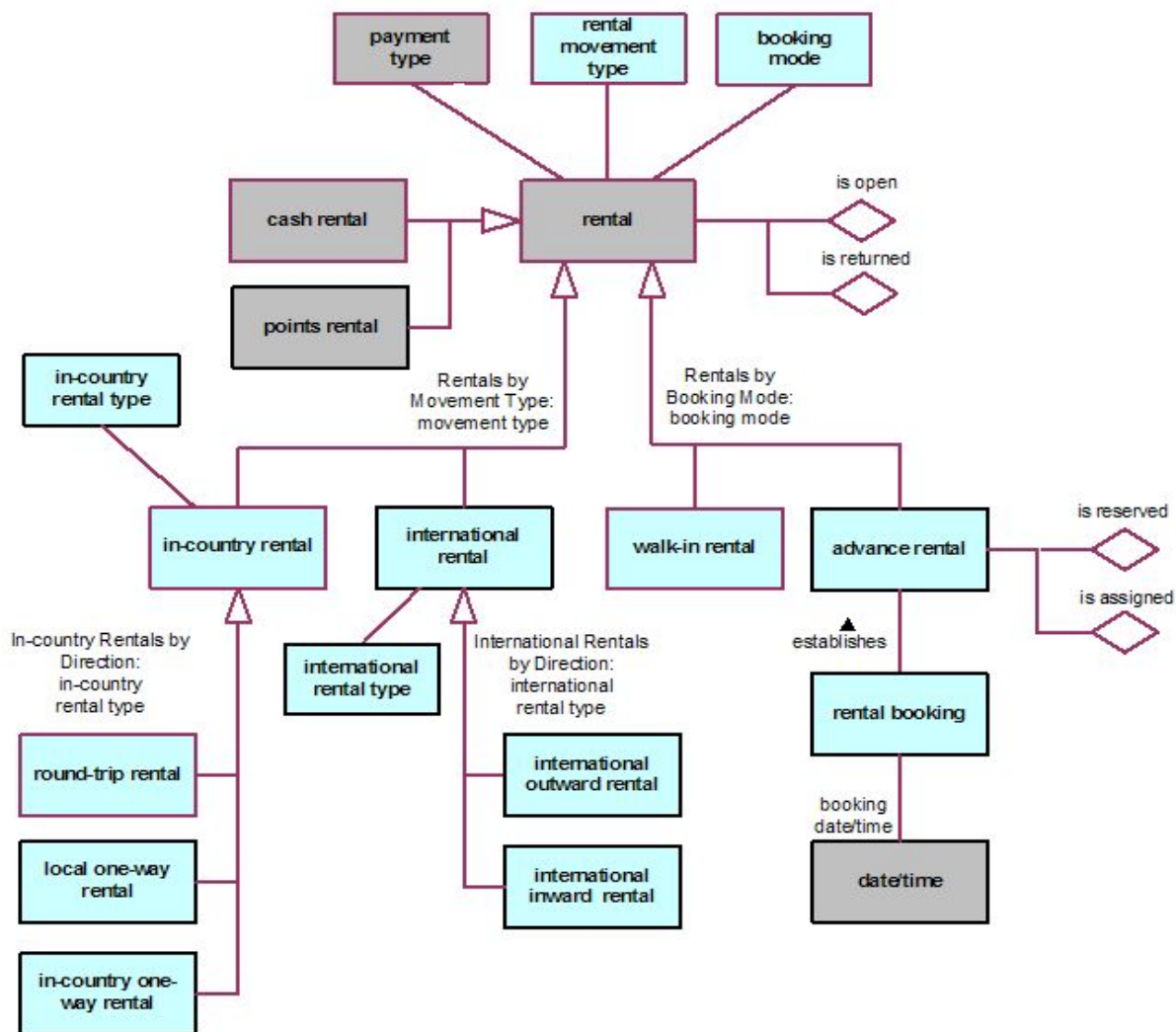


Figure E.8- Rental Categories

### advance rental

Concept Type:

booking mode

Definition:

rental that is contracted with EU-Rent on or earlier than the day before the scheduled pick-up date/time of the rental

Necessity:

Each advance rental *specifies exactly one car group at a given date/time.*

Possibility:

The car group specified for an advance rental *is changed before the actual pick-up date/time of the advance rental.*

Necessity:

The car group specified for an advance rental *is not changed after the actual pick-up date/time of the advance rental.*

Necessity:

The concept 'advance rental' *is included in* Rentals by Booking Mode.

### **advance rental being assigned**

Concept Type: [advance rental state](#)

Definition: [advance rental](#) *having* a [car movement](#) *that has an assigned* [rented car](#) *that has not yet been picked up*

### **advance rental being reserved**

Concept Type: [advance rental state](#)

Definition: [advance rental](#) *having* a [car movement](#) *that does not have an assigned* [rented car](#)

### **advance rental state**

Concept Type: [characteristic type](#)

### **booking date/time**

Concept Type: [role](#)

Definition: [date/time](#) when [rental booking is accepted by](#) [EU-Rent](#)

### **booking mode**

Concept Type: [categorization type](#)

Definition: [concept](#) *that specializes the concept* '[rental](#)' *and that classifies a* [rental](#) *whether it is booked in advance or not*

### **car model is requested for rental**

Synonymous Form: [rental requests car model](#)

Necessity: *Each* [rental](#) *requests at most one* [car model](#).

Possibility: *The* [car model](#) *specified for an* [advance rental](#) *is changed before the* [actual pick-up date/time of the](#) [advance rental](#).

Necessity: *The* [car model](#) *specified for an* [advance rental](#) *is not changed after the* [actual pick-up date/time of the](#) [advance rental](#).

### **in-country one-way rental**

Concept Type: [rental movement type](#)

Definition: [one-way rental](#) *that includes an* [in-country car movement](#)

Note: This type of [rental](#) is between [branches](#) in different [local areas](#) in the same [country](#).

Necessity: [in-country one-way rental](#) *is included in* [In-Country Rentals by Direction](#).

### **in-country rental type**

Concept Type: [categorization type](#)

Definition: [concept](#) *that specializes the concept* '[in-country rental](#)' *and that classifies an* [in-country rental](#) *based on whether it is* [round trip](#), within a [local area](#) or between [local areas](#) in the same [country](#)

### **In-country Rentals by Direction**

Definition: [segmentation](#) *that is for the concept* '[rental](#)' *and subdivides* [rentals](#) based on [movement type](#)

Necessity: [In-Country Rentals by Direction](#) *contains the categories* '[round-trip rental](#)' *and* '[local one-way rental](#)' *and* '[in-country one-way rental](#)'.

### international rental

Concept Type:	<a href="#">rental movement type</a>
Definition:	<a href="#">one-way rental</a> that <i>includes</i> an <a href="#">international car movement</a>
Note:	This type of <a href="#">rental</a> is between <a href="#">branches</a> in different <a href="#">countries</a> .
Necessity:	<a href="#">international rental</a> <i>is included in</i> <a href="#">Rentals by Movement Type</a> .

### international inward rental

Concept Type:	<a href="#">international rental type</a>
Definition:	<a href="#">international rental</a> that <i>has</i> <a href="#">country of the return branch of the rental</a> that <i>is</i> the <a href="#">country of registration of the rented car of the rental</a>

### International Rentals by Direction

Definition:	<a href="#">segmentation</a> that <i>is for</i> <a href="#">international rental</a> and <i>subdivides</i> <a href="#">rental</a> based on <a href="#">international rental type</a>
Necessity:	<a href="#">International Rentals by Direction</a> <i>contains</i> the <a href="#">categories</a> ' <a href="#">international inward rental</a> ' and ' <a href="#">international outward rental</a> '.

### international outward rental

Concept Type:	<a href="#">international rental type</a>
Definition:	<a href="#">international rental</a> that <i>has</i> <a href="#">country of the pick-up branch of the rental</a> that <i>is</i> the <a href="#">country of registration of the rented car of the rental</a>

### international rental type

Concept Type:	<a href="#">categorization type</a>
Definition:	<a href="#">concept</a> that <i>specializes</i> the <a href="#">concept</a> ' <a href="#">international rental</a> ' and that <i>classifies</i> an <a href="#">international rental</a> based on whether its direction is to or from <a href="#">the country of registration of the rented car</a>

### local one-way rental

Concept Type:	<a href="#">rental movement type</a>
Definition:	<a href="#">one-way rental</a> that <i>includes</i> a <a href="#">local car movement</a>
Note:	This type of <a href="#">rental</a> is between <a href="#">branches</a> within a <a href="#">local area</a> .
Necessity:	<a href="#">local one-way rental</a> <i>is included in</i> <a href="#">In-Country Rentals by Direction</a> .

### rental booking

Source:	CRISG ["reservation"]
Synonym:	<a href="#">reservation</a>
Definition:	acceptance by EU-Rent of a request from a <a href="#">renter</a> for an <a href="#">advance rental</a> .
Note:	The request informs EU-Rent of the <a href="#">car group</a> required, the <a href="#">scheduled pick-up date/time</a> and <a href="#">scheduled return date/time</a> , and the <a href="#">pick-up branch</a> and <a href="#">return branch</a> , and provides details of the <a href="#">renter</a> . Optionally, a specific <a href="#">car model</a> within the required <a href="#">car group</a> may be requested.

### rental booking establishes advance rental

Concept Type:	<a href="#">associative fact type</a>
Synonymous Form:	<a href="#">advance rental</a> <i>is established by</i> <a href="#">rental booking</a>

Necessity: Each advance rental is established by exactly one rental booking.

### rental booking has booking date/time

Concept Type: is-property-of fact type

Necessity: Each rental booking has exactly one booking date/time.

Necessity: The booking date/time of the rental booking that establishes a cash rental is before the scheduled pick-up date/time of the rental.

Necessity: The booking date/time of the rental booking that establishes a points rental is at least 5 days before the scheduled pick-up date/time of the rental.

### rental being open

Concept Type: rental state

Definition: rental having a rented car that is in possession of the renter and the end date/time of the grace period of the rental is in the future

### rental being returned

Concept Type: rental state

Definition: the rented car of the rental has been returned from rental to a branch

### rental movement type

Concept Type: categorization type

Definition: concept that specializes the concept 'rental' and that classifies a rental based on whether it is within a country or between countries

### rental state

Concept Type: characteristic type

### Rentals by Booking Mode

Definition: segmentation that is for the concept 'rental' and subdivides rentals based on booking mode

Necessity: Rentals by Booking Mode contains the categories 'advance rental' and 'walk-in rental'.

### Rentals by Movement Type

Definition: segmentation that is for the concept 'rental' and subdivides rentals based on rental movement type

Necessity: Rentals by Movement Type contains the categories 'in-country rental' and 'international rental'.

### reservation

Synonym: rental booking

### round-trip rental

Concept Type: rental movement type

Definition: rental that includes a round-trip car movement

Note: In this type of rental the pick-up branch is the return branch.

Necessity: round-trip rental is included in In-Country Rentals by Direction.

## walk-in rental

Concept Type:

[booking mode](#)

Definition:

[rental](#) that is contracted with EU-Rent on the day that the car is picked up

Necessity:

[walk-in rental](#) is included in [Rentals by Booking Mode](#).

### E.2.2.1.7 Rental Pricing

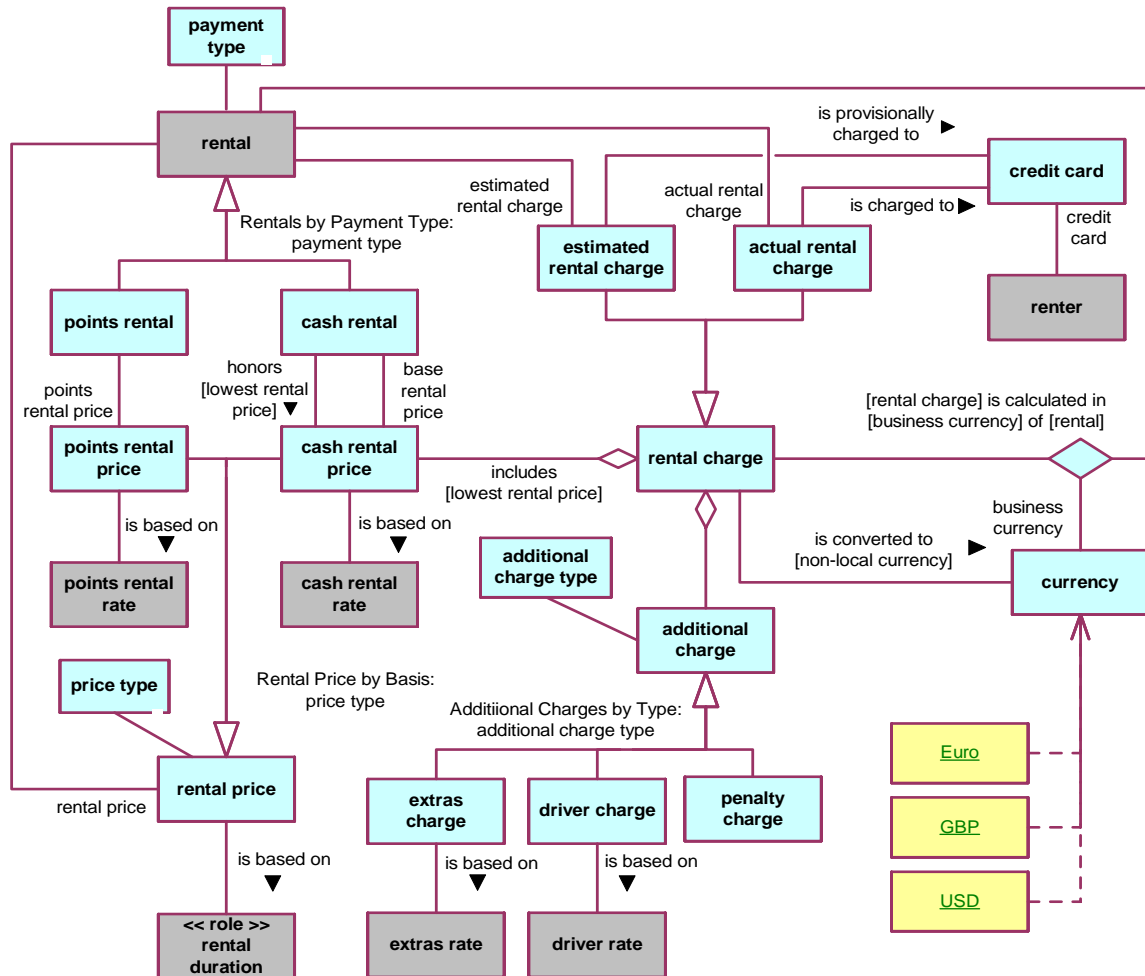


Figure E.9 - Rental Pricing

## actual rental charge

Definition:

[rental charge](#) that is calculated at end of rental

## additional charge

Definition:

[charge](#) included in [rental charge](#) in addition to [lowest rental price](#)

### **additional charge type**

Concept Type: [categorization type](#)  
Definition: [concept that specializes the concept 'additional charge' and that classifies an additional charge](#) based on why it was incurred - option selected by the renter, additional driver, or penalty charge

### **additional charge is included in rental charge**

#### **Additional Charges by Charge Type**

Definition: [segmentation that is for the concept 'additional charge' and subdivides additional charges](#) using [additional charge type](#)  
Necessity: [Additional Charges by Basis contains the categories 'extras charge' and 'driver charge' and 'penalty charge'.](#)

### **base rental price**

Concept Type: [role](#)  
Definition: [price](#) charged for the use of the [rented car](#) of a [rental](#), before any [additional charges](#) are added  
Description: The [base rental price](#) is the sum of the rental rates for the requested [car model](#) for the RTUs ([rental time units](#)) that make up the [rental duration](#). The base rental price can be calculated in money or loyalty club points.  
The [rental duration](#) is broken down into integral numbers of [RTUs](#), working from the largest [RTU](#) towards the smallest (see example).  
Necessity: If the [rental duration](#) is not for an exact number of days, the final part-day is charged as a [Rental Day](#).  
Note: Hourly tariff lines are used only for calculating late charges.  
Example: A 10-day rental is broken down into four rental time units: 1 x week + 3 x 1-day. The base rental price is the sum of the prices for the four RTUs.

### **base rental price is based on rental duration**

### **business currency**

Concept Type: [role](#)  
Definition: [currency](#) in which EU-Rent undertakes financial transactions  
Description: [currency of an operating country](#).

### **cash rental**

Concept Type: [payment basis](#)  
Definition: [rental that is charged in money](#)  
Necessity: [cash rental is included in Rentals by Payment Type](#).

### **cash rental price**

Concept Type: [price type](#)  
Definition: [base rental price that is in money](#)  
Necessity: [cash rental price is included in Rental Prices by Basis](#).

### cash rental price is based on cash rental rate

Necessity: Each cash rental price is based on exactly one cash rental rate.

### cash rental honors lowest rental price

Necessity: Each cash rental honors exactly one lowest rental price.

Necessity: The lowest rental price of a rental is honored after the booking date/time of the booking that establishes the rental.

Necessity: The lowest rental price of a rental is honored before the actual return date/time of the rental.

### charge

Source: MWU 5b1 ["charge"]

### credit card

Dictionary Basis: MWU, 1: a small card (as one issued by hotels, restaurants, stores, or petroleum companies) authorizing the person or company named or its agent to charge goods or services

### currency

Source: MWU 2a ["currency"]

Note: Has predefined population (see below)

### driver charge

Concept Type: additional charge type

Definition: additional charge that is for additional drivers authorized for a rental

Necessity: drivers charge is included in Additional Charges by Basis

### driver charge is based on driver rate

### estimated rental charge

Definition: rental charge estimated at start of rental

### estimated rental charge is provisionally charged to credit card

### extras charge

Concept Type: additional charge type

Definition: additional charge that is for optional extra

Necessity: extras charge is included in Additional Charges by Basis

### extras charge is based on extras rate

### lowest rental price

Concept Type: role

Definition: cash rental price that is most favorable to the renter of a cash rental

Description: Between the booking date/time of a rental and its actual return date/time, pricing changes (e.g., tariff changes, discounts, promotions) may occur.  
The lowest rental price is the most favorable price for the renter that results from any such changes.



Honoring the lowest rental price applies only while the car group and duration of the rental remain unchanged.

Necessity: A cash rental price of a rental that is calculated because of EU-Rent price changes and that is less than the lowest rental price of the rental replaces the lowest rental price of the rental.

Necessity: A cash rental price of a rental that is calculated because of changes to the car group or rental duration of the rental replaces the lowest rental price of the rental.

Necessity: The lowest rental price of a rental is not replaced after the actual return date/time of the rental.

### lowest rental price is included in rental charge

#### non-local currency

Concept Type: role

Definition: currency that is not the currency of a rental

#### payment type

Concept Type: categorization type

Definition: concept that specializes the concept 'rental' and that classifies a rental based on whether it is paid for by credit card or loyalty club points

#### penalty charge

Concept Type: additional charge type

Definition: additional charge that is for non-compliance with the terms of a rental

Necessity: penalty charge is included in Additional Charges by Basis

#### points rental

Concept Type: payment basis

Definition: rental that is charged in loyalty club points

Necessity: Each points rental has a points rental price.

Necessity: The renter of each points rental is a club member.

Necessity: points rental is included in Rentals by Payment Type.

#### points rental price

Concept Type: price type

Definition: base rental price that is in loyalty club points

Necessity: points rental price is included in Rental Prices by Basis.

### points rental price is based on points rental rate

#### price

Source: MWU (1) ["price"]

#### price type

Concept Type: categorization type

Definition: concept that specializes the concept 'base rental price' and that classifies a base rental price based on whether it is calculated in money or loyalty club points

### rental charge

Concept Type: [role](#)  
Definition: [charge](#) that is the total amount estimated or charged for a [rental](#)

### rental has business currency

Necessity: [The business currency of a rental is the currency of the operating country of the operating company that includes the local area that includes the pick-up branch of the rental.](#)

### rental charge is calculated in business currency

Necessity: [Each rental charge of each rental is calculated in the business currency of the rental.](#)

### rental charge is converted to non-local currency

Description: If a renter requests it, the rental charge for a rental can be shown on the contract and/or the invoice in a currency other than the currency in which it is calculated. This is done by converting the rental charge to the non-local currency.

### Rental Prices by Basis

Definition: [segmentation that is for the concept 'base rental price' and subdivides base rental prices based on price type](#)  
Necessity: [Rentals by Payment Type contains the categories 'cash rental price' and 'points rental price'.](#)

### Rentals by Payment Type

Definition: [segmentation that is for the concept 'rental' and subdivides rentals based on payment basis](#)  
Necessity: [Rentals by Payment Type contains the categories 'cash rental' and 'points rental'.](#)

### renter has credit card

*Pre-defined Population:* [currency](#)

### Euro

Concept Type: [individual concept](#)  
General Concept: [currency](#)  
Synonym: [EUR](#)

### GBP

Concept Type: [individual concept](#)  
General Concept: [currency](#)  
Synonym: [British Pound](#)

### USD

Concept Type: [individual concept](#)  
General Concept: [currency](#)  
Synonym: [United States Dollar](#)

### E.2.2.1.8 Tariff

To keep the EU-Rent case study to a manageable size, the relationship of tariff to operating country has been greatly simplified. In reality there would be a standard tariff structure, replicated for each operating country and each populated with a different set of values. This is a data design issue, and not much is lost from the illustration of concepts and vocabulary by omitting it.

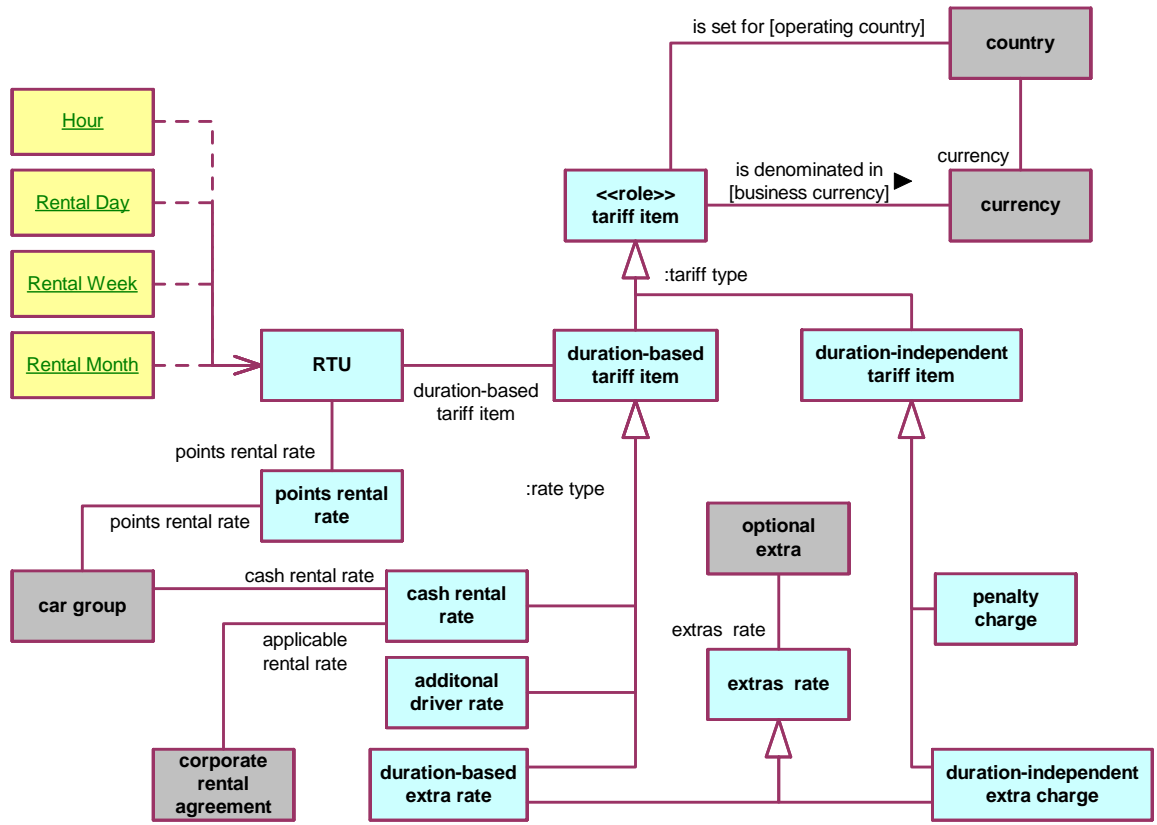


Figure E.10- Rental Tariffs

#### additional driver rate

Concept Type: [rate type](#)  
 Definition: [duration-based tariff item](#) that is for [each additional driver](#) of a [rental](#)

#### applicable rental rate

Concept Type: [role](#)  
 Definition: [cash rental rate](#) that is [applicable to a corporate rental agreement](#)

#### cash rental rate

Concept Type: [rate type](#)  
 Definition: [duration-based tariff item](#) that is for rental of a car of a given [car group](#)

**cash rental rate *is for* car group**

**corporate rental agreement *has* applicable rental rate**

**duration-based extra rate**

Concept Type: [rate type](#)

Definition: [duration-based tariff item](#) *that is for an* [optional extra](#)

**duration-based tariff item**

Concept Type: [tariff type](#)

Definition: [tariff item](#) *that* is charged to a rental per RTU in the duration of the rental

Example: daily cash rental rate for car group, daily cost of child seat.

**duration-independent extra charge**

Definition: [duration-independent tariff item](#) *that is for an* [optional extra](#)

**duration-independent tariff item**

Concept Type: [tariff type](#)

Definition: [tariff item](#) *that* is the basis for a charge to a rental regardless of the duration of the rental

Example: charge for fuel consumed; charge for one-way rental

Note: The tariff item may be the basis for calculation rather than a fixed amount. For example, a charge for fuel is calculated per liter or per gallon.

**extras rate**

Definition: [generalization](#) of [duration-dependent extra charge](#) and [duration-independent extra charge](#)

**extras rate *is for* optional extra**

**penalty charge**

Definition: [duration-independent tariff item](#) *that* is a penalty charge for some breach of the conditions of [a rental](#)

**points rental rate**

Concept Type: [role](#)

Definition: [number](#) *that* represents the loyalty club points charged per RTU to rent a car of a given car group

**points rental rate *is for* car group**

**points rental rate *is for* RTU**

**rate type**

Concept Type: [categorization type](#)

Definition: [concept](#) *that specializes the* [concept](#) '[duration-based tariff item](#)' and *that classifies a* [duration-based tariff item](#) based on what type of service is being charged for

**rental time unit**

See: [RTU](#)

## RTU

Definition:	time unit that is an atomic (integer) unit of time for which a car can be rented
Synonym:	<a href="#">rental time unit</a>
Dictionary Basis:	CRISG ["RTU"]
Note:	Has pre-defined population - see below.

## RTU has duration-based tariff item

## RTU has points rental rate

Synonymous Form:	<a href="#">points rental rate</a> <i>is for</i> <a href="#">RTU</a>
------------------	--

## tariff item

Concept Type:	<a href="#">role</a>
Definition:	<a href="#">number</a> that represents the price in some <a href="#">business currency</a> of some element of a <a href="#">rental</a>
Source:	MWU 2b ["tariff"]
Example:	weekly rate for a car of a given car group; cost of additional insurance; penalty charge for drop-off at location other than the return branch
Note:	This entry is informally defined in order to limit the case study size. In a 'real' SBVR model, tariff items would have validity periods, and would include special offers that would override standard rates for limited periods.

## tariff item is denominated in business currency

## tariff item is set for operating country

## tariff type

Concept Type:	<a href="#">categorization type</a>
Definition:	<a href="#">concept</a> that <i>specializes the concept</i> ' <a href="#">tariff item</a> ' and that <i>classifies a</i> <a href="#">tariff item</a> based on whether it is a per-RTU change or not

## *Predefined Population -* [RTU](#)

## Hour

Definition:	"the 24th part of a mean solar day : 60 minutes of mean solar time"
General Concept:	<a href="#">RTU</a>
Source:	MWU 2b ["hour"]

## Rental Day

Definition:	24-hour period, starting at actual pick-up time of rental
Note:	Not the scheduled pick-up date/time
General Concept:	<a href="#">RTU</a>
Example:	Day beginning at 3:45 p.m.

## Rental Week

Definition:	7 consecutive <a href="#">rental days</a>
General Concept:	<a href="#">RTU</a>

## Rental Month

Definition: 28 consecutive [rental days](#)

General Concept: [RTU](#)

### E.2.2.1.9 Rental Problems

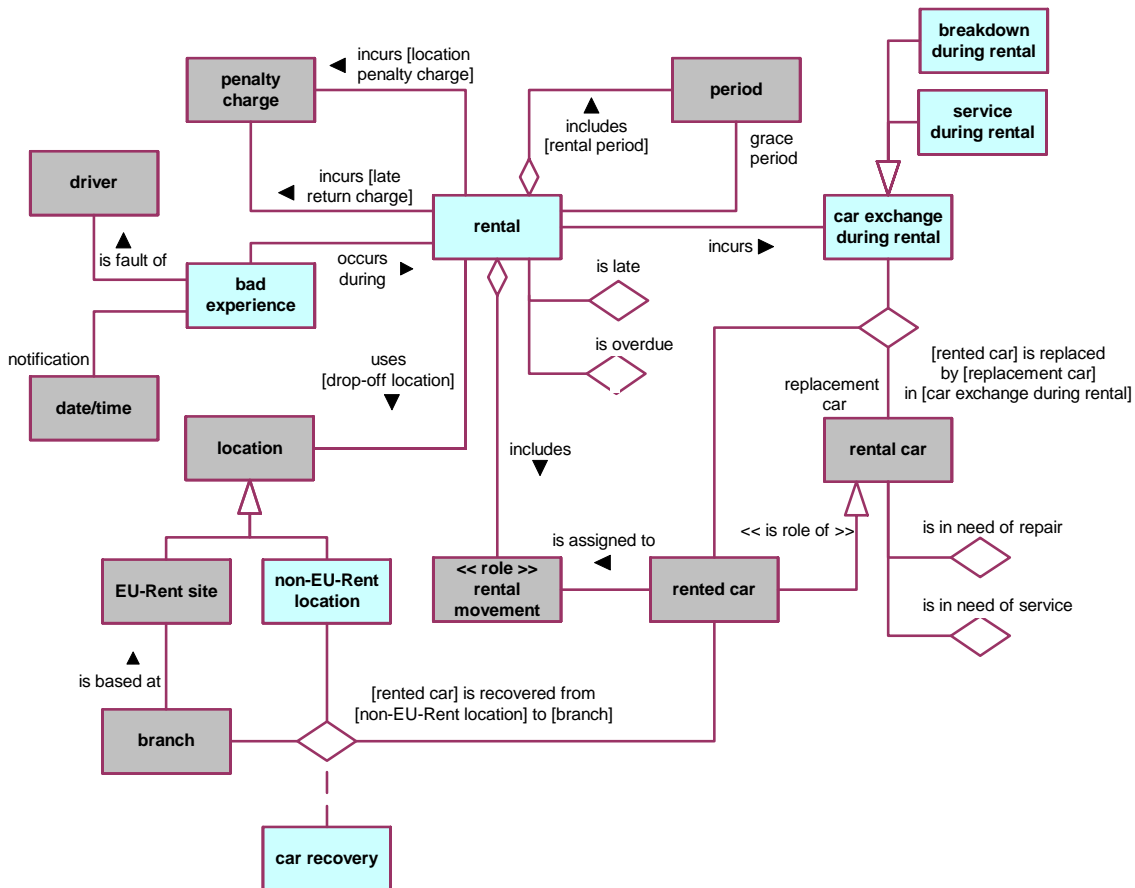


Figure E.11- Rental Problems

## actual return branch

See: [drop-off branch](#)

## bad experience

Definition: an undesirable occurrence during a [rental](#) that is the fault of one of the [drivers](#)

Example: Speeding offence, unpaid parking fine, damage to car caused by careless driving.

Note: This entry is informally defined in order to limit the case study size.

### bad experience has notification date/time

### bad experience is fault of driver

### breakdown during rental

Definition: car exchange during rental of a rented car that has operational problems

### car exchange during rental

Definition: situation where the rented car of a rental cannot be used for the remainder of the rental duration

### car recovery

Definition: actuality that a given rented car is recovered from a given non-EU-Rent site to a given branch

### charge

Source: CRISG ["charge"]

### drop-off branch

Concept Type: role

Definition: branch to which a rented car is actually returned

Possibility: A car may be returned to a branch other than the one agreed in the rental. EU-Rent will accept the car, but will charge a location penalty.

Synonym: actual return branch

### drop-off location

Concept Type: role

Definition: location where the rented car of a rental is dropped off

### grace period

Concept Type: role

Definition: period that has start date/time that is scheduled end date/time of rental period and end date/time that is the earlier of (scheduled end date/time of rental period plus one hour, closing time of return branch of rental)

### late return charge

Concept Type: role

Definition: penalty charge that is made for a rental that is late

Description: The late charge is calculated using the hourly tariff for the car group to which the car belongs, for durations of up to 5 hours after the end of the grace period. Part-hours are rounded up. The daily tariff is used for durations between 5 and 24 hours.

If, after the end of the grace period, the renter contacts EU-Rent to extend the rental, the late return charge is calculated from the end of the grace period to the date/time when the rental extension is agreed.

Note: If the car is not returned within 48 hours after the end of the grace period, and the renter has not contacted EU-Rent to extend the rental, the insurance lapses and EU-Rent will report the car to the police as stolen and uninsured.

### **location penalty charge**

Concept Type: [role](#)  
Definition: [penalty charge](#) that is made for each rental that has a [drop-off location](#) that is not the [EU-Rent site of the return branch of the rental](#)  
Description: The location penalty charge is calculated in three parts: a fixed penalty; cost of retrieving the car if the location is a non-EU-Rent site (e.g., an airport car park); cost of moving the car to the return branch specified in the rental. Car movement costs are taken from a standard scale based on the distance between branches and per-mile (or per-kilometer) costs for car groups.

### **non-EU-Rent location**

Concept Type: [role](#)  
Definition: [location](#) that is not the [location](#) of a [rental organization unit](#)

### **notification date/time**

Concept Type: [role](#)  
Definition: [date/time](#) at which something is notified to EU-Rent

### **rental being late**

Concept Type: [rental state](#)  
Definition: [rental](#) having a [rented car](#) that is in possession of the [renter](#) and the [end date/time of the grace period of the rental](#) is in the past and is less than 24 hours in the past

### **rental being overdue**

Concept Type: [rental state](#)  
Definition: [rental](#) having a [rented car](#) that is in possession of the [renter](#) and the [end date/time of the grace period of the rental](#) is more than 24 hours in the past

### **rental car being in need of repair**

Concept Type: [rental state](#)  
Definition: [rental car](#) having damage or breakdown that renders it unusable for rental

### **rental car being in need of service**

Concept Type: [rental state](#)  
Definition: [rental car](#) having [service reading](#) that is at least 5000 [miles](#).

### **rental car state**

Concept Type: [characteristic type](#)

### **rental has drop-off branch**

Necessity: The [drop-off branch](#) of a [rental](#) is the [branch](#) that is based at the [EU-Rent site](#) that is the [drop-off location](#) of the [rental](#).



rental *incurs* car exchange during rental

rental *incurs* late return charge

rental *incurs* location penalty charge

rental *has* grace period

Note: late return charges are not applied until after the grace period.

rental *uses* drop-off location

rented car *is recovered from* non-EU-Rent location *to* branch

rented car *is replaced by* replacement car *in* car exchange during rental

replacement car

Concept Type: role

Definition: rental car *in* a car exchange during rental *that* is used after the exchange

service exchange

Definition: car exchange during rental *of* a rented car *that is due for service*

unauthorized drop-off location

Definition: location *that is used to drop off the* rented car *of* a rental *and that is not the* EU-Rent site *of the* return branch *of the* rental

### E.2.2.1.10 Rental Cars

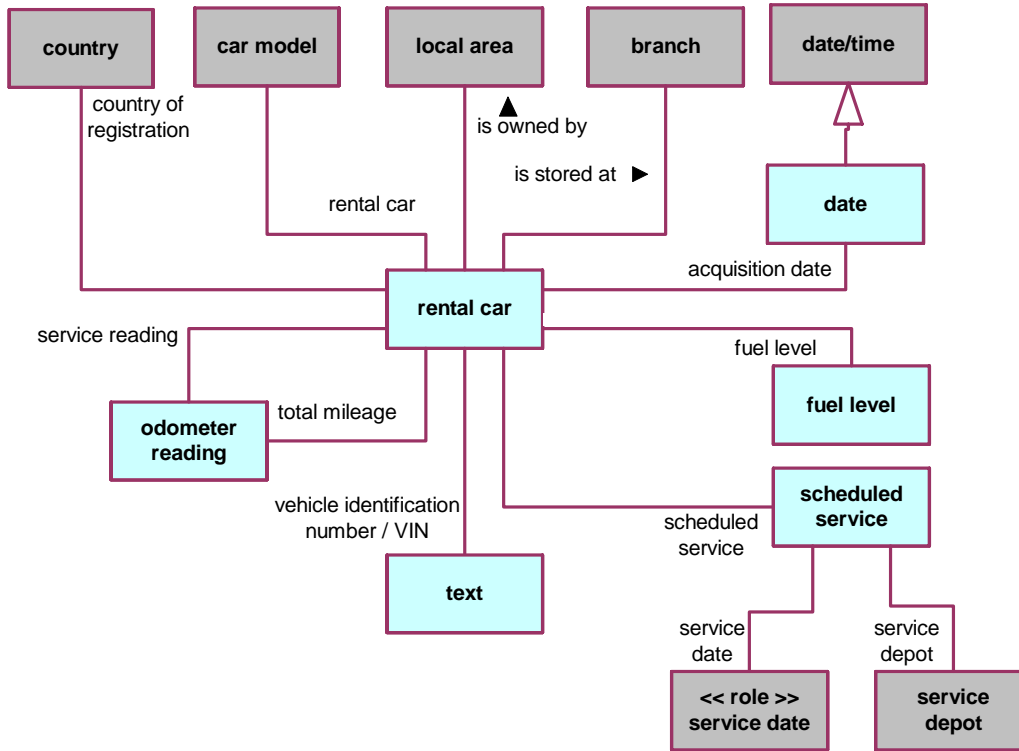


Figure E.12 - Rental Cars

#### acquisition date

Concept Type: [role](#)  
 Definition: [date](#) on which EU-Rent took ownership of some thing

#### car

See: [rental car](#)

#### country of registration

Concept Type: [role](#)  
 Definition: [country](#) in which something is registered with the relevant authorities

#### fuel level

Definition: [full](#) or [7/8](#) or [3/4](#) or [5/8](#) or [1/2](#) or [3/8](#) or [1/4](#) or [1/8](#) or [empty](#)  
 Source: CRISG ["fuel level"]

### odometer reading

Concept Type: [role](#)  
General Concept: [number](#)  
Source: CRISG ["odometer reading"]

### rental car

Source: MWU (1/1d) ["car"], CRISG ("rental car")  
Definition: vehicle owned by EU-Rent and rented to its customers  
Synonym: [car](#)

### rental car has acquisition date

Concept Type: [is-property-of fact type](#)  
Synonymous Form: [rental car is acquired on acquisition date](#)

### rental car has country of registration

Concept Type: [is-property-of fact type](#)

### rental car has odometer reading

Concept Type: [is-property-of fact type](#)

### rental car has service reading

Concept Type: [is-property-of fact type](#)

### rental car has vehicle identification number

Concept Type: [is-property-of fact type](#)  
Necessity: Each [rental car](#) has exactly one [vehicle identification number](#).

### rental car has fuel level

Definition: [is-property-of fact type](#)

### rental car is of car model

Concept Type: [is-property-of fact type](#)  
Necessity: Each [rental car](#) is of exactly one [car model](#).

### rental car is owned by local area

Necessity: Each [rental car](#) is owned by exactly one [local area](#).  
Synonymous Form: [local area owns rental car](#)

### rental car is stored at branch

Necessity: Each [rental car](#) is stored at at most one [branch](#).

### scheduled service

Definition: maintenance service for a [rental car](#) that is scheduled at a (EU-Rent) [service depot](#)

### service date

General Concept: [role](#)  
Definition: [date](#) of [scheduled service](#)

[scheduled service](#) *has* [service date](#)

[scheduled service](#) *has* [service depot](#)

### service reading

Concept Type: [role](#)  
Definition: [odometer reading](#) since the car was last serviced  
Note: When the [service reading](#) reaches 5000 miles (8000 km), the car will be scheduled for service.

### vehicle identification number

Concept Type: [role](#)  
Definition: [text](#) that is the unique identifier of a particular vehicle  
Synonym: [VIN](#)

### VIN

Synonym: [vehicle identification number](#)

### E.2.2.1.11 Customers

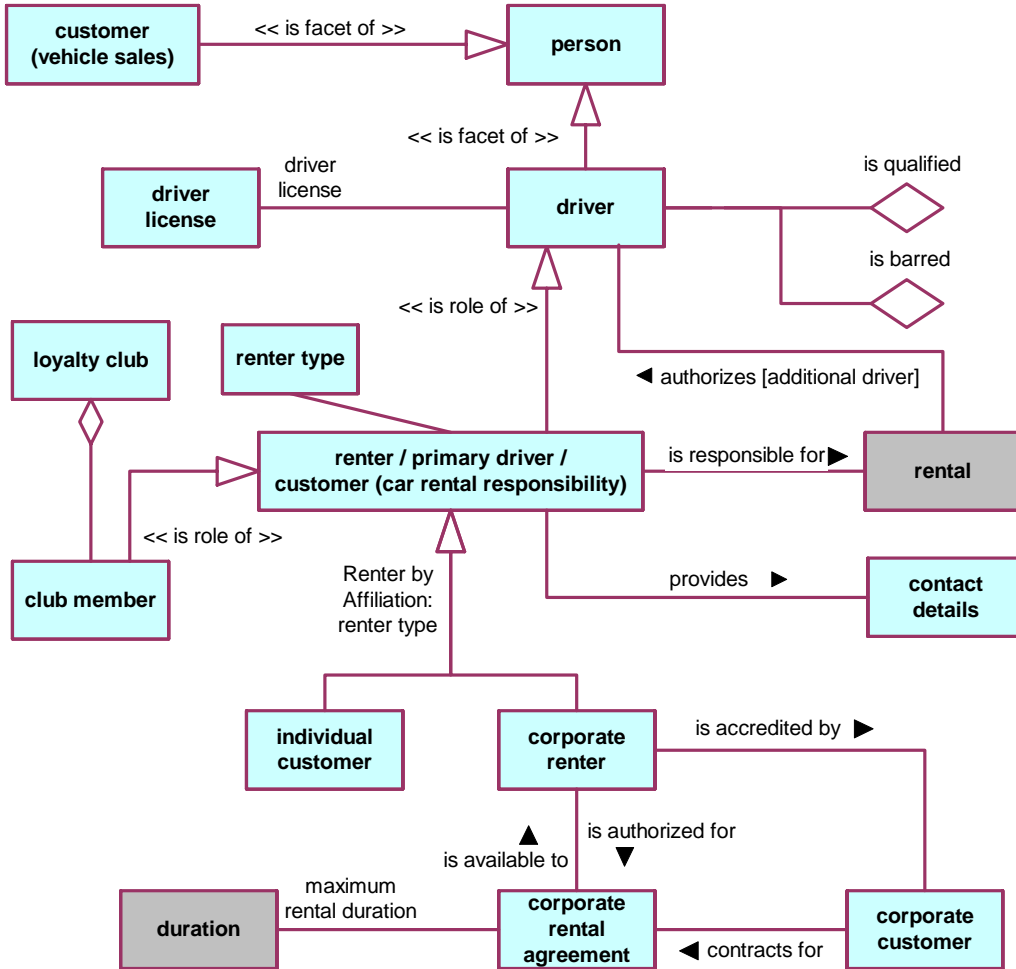


Figure E.13 - Customers

#### additional driver

Source: CRISG ["additional driver"]  
 Concept Type: [role](#)  
 Definition: [driver](#) of a [rental](#) who is not the [renter](#) of the [rental](#)

#### additional driver is authorized in rental

Necessity: Each [rental](#) [authorizes](#) at most 3 [additional drivers](#).  
 Synonymous Form: [rental](#) [authorizes](#) [additional driver](#)

### car rental responsibility

Source: CRISG ["rental responsibility"]

### club member

Definition: renter who has joined EU-Rent's loyalty club

### contact details

Definition: address, telephone number and (if available) email address

### corporate customer

Dictionary Basis: relating or belonging to a corporation MWU ["corporate"]

Dictionary Basis: person or company who buys goods and services MWU ["customer"]

### corporate customer contracts for corporate rental agreement

### corporate rental agreement

Definition: contract under which a corporate renter can rent a car at a negotiated set of rates

Note: Each rental under a corporate rental contract is an individual contract, and the corporate renter is personally responsible for it.

Note: This entry is informally defined in order to limit the case study size.

### corporate rental agreement has maximum rental duration

### corporate renter

Concept Type: renter type

Definition: renter who is a representative of a corporate customer, accredited to rent cars under the terms of its corporate rental agreement, who has booked at least one rental

Necessity: Each corporate renter is a person who is accredited by a corporate customer and who is responsible for at least one rental.

Necessity: corporate renter is included in Renters by Affiliation

### corporate renter is accredited by corporate customer

### corporate renter is authorized for corporate rental agreement

Synonymous Form: corporate rental agreement is available to corporate renter

Necessity: Each corporate renter is authorized for at least one corporate rental agreement.

### customer

Qualifier: car rental responsibility

Concept Type: role

See: renter

### customer

Qualifier: vehicle sales

Concept Type: facet

Definition: person who purchases a rental car from EU-Rent at the end of its rental life

### driver

Concept Type: [facet](#)  
Definition: [person](#) who has been identified as one who can drive the [rented car](#) of a [rental](#)

### driver being barred

Concept Type: [driver state](#)  
Definition: [driver being prohibited from renting a car from EU-Rent](#)  
Note: A barred driver is a person known to EU-Rent as a driver (either a renter or an additional driver), who has **at least 3** [bad experiences](#).

### driver being qualified

Concept Type: [driver state](#)  
Definition: [driver being over 21 years old and having a valid driver's license and not being under any pending legal action that could adversely affect his driver's license or insurability](#).

### driver has driver license

#### driver license

Source: CRISG ["driver license"]

#### driver state

Concept Type: [characteristic type](#)

#### individual customer

Concept Type: [renter type](#)  
Definition: [renter](#) who is not a [corporate renter](#), who meets at least one of the following criteria: has completed a [rental](#) within the last 5 years; has a [rental](#) currently in progress; has made a [rental booking](#)  
Necessity: **Each individual customer is a given person who is not a corporate renter and who is responsible for at least one rental that is a Reserved Rental or an Assigned Rental or an Open Rental or a Returned Rental that has an end date that is less than 5 years earlier than the current day date.**  
Necessity: [individual customer](#) *is included in* [Renters by Affiliation](#).

#### loyalty club

Definition: EU-Rent's incentive scheme for its frequent renters  
Note: A [customer](#) who joins the [loyalty club](#) accumulates points that s/he can use to pay for a [rental](#).

#### loyalty club includes club member

Concept Type: [partitive fact type](#)

#### maximum rental duration

Concept Type: [role](#)  
Definition: [duration](#) that is the upper limit for [rental duration](#) of each [rental](#) made under the terms of a [corporate rental agreement](#)

## person

Source: MWU (1a) ["person"]

## primary driver

See: [renter](#)

## renter

Source: CRISG ["renter"]

Concept Type: [role](#)

Definition: [person](#) contractually responsible for a [rental](#)

Synonym: [customer](#) ([car rental responsibility](#))

Synonym: [primary driver](#)

## renter is responsible for rental

Concept Type: [associative fact type](#)

Necessity: **The [renter of a rental](#) may not be changed.**

Note: If the renter wishes to change the [rental](#) to a different [renter](#), EU-Rent regards it as a cancellation and a new [rental](#).

## renter provides current contact details

## renter type

Concept Type: [categorization type](#)

Definition: [concept](#) that specializes the [concept](#) 'renter' and that classifies a [rental](#) based on whether it is an individual customer or a corporate renter

## Renters by Affiliation

Definition: [segmentation](#) that is for the [concept](#) 'renter' and subdivides renters based on [renter type](#)

Necessity: [Renters by Affiliation](#) contains the categories 'individual customer' and 'corporate renter'.

## vehicle sales

Source: CRISG ["car sales"]

### E.2.2.2 EU-Rent Guidance expressed using the EU-Rent English Vocabulary

This section presents elements of guidance (business rules, admonitions, and affirmations) that accompany the "EU-Rent English Vocabulary" -- as described in Annex C ('C.4 Specifying a Rule Set').

NOTE: The guidance in this section is expressed in the EU-Rent English Vocabulary; a working subset of this is provided in the preceding section. If the statements are difficult to understand at face value – e.g., seem ambiguous, or don't quite fit everyday understanding of the words used – it is important to check the definitions in the vocabulary before challenging the guidance statements.

Many of the guidance statements are supported by descriptions, which reflect EU-Rent users' informal statements of the guidance.



The examples in this section are generally presented in the form “It is obligatory that ...”, “It is necessary that ...”, etc. This emphasizes the application of the modal claim (obligation, necessity, etc.) to the underlying fact type, but sometimes provides a cumbersome representation. SBVR Structured English (see Annexes H and C) also supports more direct representation. For example, the operative business rule:

It is obligatory that each driver of a rental is a qualified driver.

can be represented as

Each driver of a rental must be a qualified driver.

For a treatment of these examples in RuleSpeak<sup>®</sup>, a widely-used business rule notation, see Annex F.

### E.2.2.2.1 Introduction

#### <EU-Rent English Vocabulary Rules>

Vocabulary: EU-Rent English Vocabulary

#### <EU-Rent English Vocabulary Levels of Enforcement>

Level of enforcement is a categorization scheme for business rules defined (or adopted) by the organization that owns the rules. EU-Rent’s categories are listed below.

##### Enforcement Level: strict

Definition: strictly enforced: if the rule is violated, the sanction or other consequences always ensue.

##### Enforcement Level: deferred

Definition: deferred enforcement: strictly enforced, but enforcement may be delayed — e.g., waiting for resource with required skills.

##### Enforcement Level: pre-authorized

Definition: pre-authorized override: enforced, but exceptions allowed, with prior approval for actors with before-the-fact override authorization.

##### Enforcement Level: post-justified

Definition: post-justified override: if not approved after the fact, the sanction or other consequences will ensue.

##### Enforcement Level: override

Definition: override with explanation: comment must be provided when the violation occurs.

##### Enforcement Level: guideline

Definition: guideline: suggested, but not enforced.

### E.2.2.2.2 Rule Set -- Rental Rules

It is necessary that each rental has exactly one requested car group.

Guidance Type: structural business rule

Description: The renter may request a change of car group up to pick-up time, but a car group must always be specified

Supporting fact type: rental has requested car group

It is necessary that each rental *has* exactly one rental period.

Guidance Type: structural business rule

Description: The renter may request a change to the start and/or end of the rental period, or cause a de-facto change by late return of the car, but a rental period must always be specified

Supporting fact type: rental *has* rental period

It is necessary that each rental *has* exactly one return branch.

Guidance Type: structural business rule

Description: The renter may request a change to the return branch, or cause a de-facto change by return of the car to an unauthorized branch, but a return branch must always be specified

Supporting fact type: rental *has* return branch

It is necessary that the scheduled pick-up date/time of each advance rental *is after* the booking date/time of the rental booking that *establishes* the advance rental.

Guidance Type: structural business rule

Description: When a rental reservation is made (establishing an advance rental) the rental scheduled pick-up date/time must be later than the actual date/time of reservation.

Supporting fact types: rental booking *has* booking date/time  
rental booking *establishes* advance rental  
rental *has* scheduled pick-up date/time  
date/time<sub>1</sub> *is after* date/time<sub>2</sub>

Related facts: the noun concept 'cash rental' *is a category of* the noun concept 'rental'  
the noun concept 'advance rental' *is a category of* the noun concept 'rental'

#### E.2.2.2.3 Rule Set -- Charging / Billing / Payment Rules

It is permitted that a rental *is open* only if an estimated rental charge *is provisionally charged to* the credit card of the renter of the rental.

Guidance Type: operative business rule

Description: While a renter has possession of a car, there is a provisional charge to EU-Rent against his credit card. This will be replaced by an actual charge at the end of the rental

Enforcement Level: Strict

Supporting fact types: rental *has* estimated rental charge  
estimated rental charge *is provisionally charged to* credit card  
renter *has* credit card  
rental *has* renter

Related facts: 'being open' *is a characteristic of* the noun concept 'rental'

It is necessary that the rental charge of each rental *is calculated in* the business currency of the rental.

Guidance Type: structural business rule

Note: This is a constraint imposed by credit card issuers.

Supporting fact types: rental *has* rental charge  
rental charge *is calculated in* business currency of rental  
rental *has* business currency

If the renter of a rental requests a price conversion then it is obligatory that the rental charge of the rental is converted to the currency of the price conversion.

Guidance Type: operative business rule  
Description: EU-Rent will provide the customer with a bill in another currency, but the actual billing is done in the business currency, and converted.  
Enforcement Level: strict  
Supporting fact types: rental has renter  
rental has rental charge  
Related fact: a price conversion is the rental charge of a rental denominated in a currency requested by the renter

It is necessary that each cash rental honors the lowest rental price of the cash rental.

Guidance Type: structural business rule  
Description: Between the booking date/time of a cash rental and its actual return date/time, pricing changes (e.g., tariff changes, discounts, promotions) may occur.  
The lowest rental price is the most favorable price for the renter that results from any such changes.  
Honoring the lowest rental price applies only while the car group and duration of the rental remain unchanged.

The structural business rule above can be elaborated as three detailed structural business rules.

It is necessary that a cash rental price for a cash rental that is calculated because of EU-Rent price changes and that is less than the lowest rental price honored by the rental replaces the lowest rental price honored by the rental.

It is necessary that a cash rental price for a cash rental that is calculated because of changes to the car group or rental duration of a rental replaces the lowest rental price honored by the rental.

It is necessary that the lowest rental price honored by a rental is not replaced after the actual return date/time of the rental.

There is no need for a structural business rule that the lowest rental price is not replaced before the booking date, because the rental does not exist before that date.

Supporting fact types: cash rental honors lowest rental price  
rental has base rental price  
rental has actual return date/time  
rental has car group  
rental has rental duration  
state of affairs occurs after date/time  
Related facts: the noun concept 'cash rental' is a category of the noun concept 'rental'  
the noun concept 'lowest rental price' is a role of the noun concept 'cash rental price'  
the noun concept 'cash rental price' is a category of the noun concept 'base rental price'

#### E.2.2.2.4 Rule Set -- Driver Rules

It is permitted that a rental is open only if each driver of the rental is not a barred driver.

Synonymous Form:	It is prohibited that a <u>rental is open</u> if a <u>driver of the rental is a barred driver</u> .
Guidance Type:	<u>operative business rule</u>
Description:	While a rented car is in possession of the renter, no driver for the rental can be a barred driver.
Enforcement Level:	<u>pre-authorized</u>
Supporting fact types:	<u>rental has primary driver</u> <u>rental has additional driver</u>
Related facts:	<i>'being open' is a characteristic of the noun concept 'rental'</i> <i>'being barred' is a characteristic of the noun concept 'driver'</i> <i>the noun concept 'primary driver' is a role of the noun concept 'driver'</i> <i>the noun concept 'additional driver' is a role of the noun concept 'driver'</i>

It is obligatory that each driver of a rental is a qualified driver.

Guidance Type:	<u>operative business rule</u>
Description:	To be accepted as a driver on a rental, a person must comply with EU-Rent's definition of "driver is qualified".
Enforcement Level:	<u>Strict</u>
Supporting fact types:	<u>rental has primary driver</u> <u>rental has additional driver</u>
Related facts:	<i>'being qualified' is a characteristic of the concept 'driver'</i> <i>the noun concept 'primary driver' is a role of the noun concept 'driver'</i> <i>the noun concept 'additional driver' is a role of the noun concept 'driver'</i>

#### E.2.2.2.5 Rule Set -- Pick-up / Return Rules

This section illustrates a trade-off of a larger body of shared concepts, and corresponding vocabulary, against simpler formulation of business rules.

The business rules here could have been stated more elaborately; e.g., one of the examples below is:

It is obligatory that the country of the return branch of each international inward rental is the country of registration of the rented car of the rental.

It could have been stated as

*"If the country of the pick-up branch of a rental is not the country of registration of the rented car of the rental then it is obligatory that the country of the return branch of the rental is the country of registration of the rented car."*

Defining categories of rental, as used below, simplifies the expression of rules at the expense of additional concepts and larger vocabulary to be managed.

This kind of trade-off is a business choice of the semantic community.

It is obligatory that *at the actual return date/time of each in-country rental and each international inward rental the local area of the return branch of the rental owns the rented car of the rental.*

Guidance Type:	<a href="#">operative business rule</a>
Description:	When a car is moved between branches in different local areas in the same country, or is returned to its country of registration after being dropped off abroad, ownership moves between local areas with it. This is so whether it is a one-way rental or a transfer made by EU-Rent.
Note:	Ideally, this would be a structural rule, defining ownership at the end of rentals, but EU-Rent cannot always control car movements as it would like to.
Enforcement Level:	<a href="#">pre-authorized</a>
Supporting fact types:	<a href="#">rental has actual return date/time</a> <a href="#">rental has return branch</a> <a href="#">branch is included in local area</a> <a href="#">local area owns rental car</a> <a href="#">state of affairs occurs at date/time</a>
Related facts:	<a href="#">the noun concept 'rented car' is a role of the noun concept 'rental car'</a> <a href="#">the noun concept 'return branch' is a role of the noun concept 'branch'</a> <a href="#">the noun concept 'in-country rental' is a category of the noun concept 'rental'</a> <a href="#">the noun concept 'international inward rental' is a category of the noun concept 'international rental'</a> <a href="#">the noun concept 'international rental' is a category of the noun concept 'rental'</a>

It is obligatory that the *country of the return branch of each international inward rental is the country of registration of the rented car of the rental.*

Guidance Type:	<a href="#">operative business rule</a>
Description:	When a one-way rental has dropped a car off in a different country, that car may then be used for only one kind of rental – a one-way rental back to its country of registration.
Note:	If a one-way rental back to country of registration does not occur within a short time, the branch manager will have a EU-Rent employee transfer the car.
Enforcement Level:	<a href="#">pre-authorized</a>
Supporting fact types:	<a href="#">branch has country</a> <a href="#">rental has return branch</a> <a href="#">rental car has country of registration</a>
Related facts:	<a href="#">the noun concept 'rented car' is a role of the concept 'rental car'</a> <a href="#">the noun concept 'international inward rental' is a category of the noun concept 'rental'</a> <a href="#">the noun concept 'return branch' is a role of the noun concept 'branch'</a> <a href="#">the noun concept 'country of registration' is a role of the noun concept 'country'</a>

It is necessary that if a *rental is open and the rental is not an international inward rental then the rented car of the rental is owned by the local area of the pick-up branch of the rental.*

Guidance Type:	<a href="#">structural business rule</a>
Note:	This ensures that the local area that owned the car at the start of a rental retains responsibility for it until it is dropped off at a EU-Rent branch. It also ensures that a car's ownership is retained within its country of registration.

Supporting fact types for the three business rules above:

[rental](#) *has* [pick-up branch](#)  
[local area](#) *includes* [branch](#)  
[rental car](#) *is owned by* [local area](#)

Related facts: [the noun concept 'rented car' is a role of the noun concept 'rental car'](#)  
['international inward rental' is a category of 'international rental'](#)  
['international rental' is a category of 'rental'](#)  
['being open' is a characteristic of 'rental'](#)  
['pick-up branch' is a role of 'branch'](#)  
['return branch' is a role of 'branch'](#)

If the [actual return date/time of a rental](#) *is after* the [end date/time of the grace period of the rental](#) then it is obligatory that the [rental](#) *incurs* a [late return charge](#).

Guidance Type: [operative business rule](#)

Note: The grace period of a rental ends one hour after the rental's scheduled return date/time or at close of business of the return branch, whichever is earlier.

Enforcement Level: [Strict](#)

Supporting fact types: [rental](#) *has* [actual return date/time](#)  
[rental](#) *has* [grace period](#)  
[period](#) *has* [end date/time](#)  
[date/time<sub>1</sub>](#) *is after* [date/time<sub>2</sub>](#)  
[rental](#) *incurs* [late return charge](#)

Related facts: [the noun concept 'actual return date/time' is a role of the noun concept 'date/time'](#)  
[the noun concept 'grace period' is a role of the noun concept 'period'](#)  
[the noun concept 'end date/time' is a role of the noun concept 'date/time'](#)

If the [drop-off location of a rental](#) *is not* the [EU-Rent site of the return branch of the rental](#) then it is obligatory that the [rental](#) *incurs* a [location penalty charge](#).

Guidance Type: [operative business rule](#)

Description: If a rented car is returned to a location that is not the specified return branch of the rental, that branch will accept the car but a location penalty charge will be applied to the rental.

Enforcement Level: [Strict](#)

Supporting fact types: [rental](#) *has* [drop-off location](#)  
[rental](#) *has* [return branch](#)  
[branch](#) *is located at* [EU-Rent site](#)  
[rental](#) *incurs* [location penalty charge](#)

If a [rental](#) *is assigned* then it is obligatory that the [rented car of the rental](#) *is stored at* the [pick-up branch of the rental](#).

Synonymous Form: [It is prohibited that the rented car of an assigned rental is not stored at the pick-up branch of the rental.](#)

Guidance Type: [operative business rule](#)

Description: A rental car must physically be at the pick-up branch when it is assigned to a rental.

Note: This is an example of a rule created to ensure that real-world influences do not cause problems in EU-Rent's business. In this case, EU-Rent knows that sometimes cars are not brought to branches when they are supposed to be, so it insists that cars assigned to rentals are physically present. It does not permit cars that are "due to be returned to this branch tomorrow" to be assigned.

After assignment to a rental, the car must stay at the branch until pick-up time.

This doesn't mean that the car can't be moved. It means that if a car is to be moved, it must be unassigned from any rental and another car assigned in its place.

Enforcement Level: [Override](#)

Supporting fact types: [rental car is stored at branch](#)

Related facts: [the noun concept 'rented car' is a role of the noun concept 'rental car' 'being assigned' is a characteristic of the noun concept 'advance rental'](#)

[the noun concept 'advance rental' is a category of the noun concept 'rental'](#)

[the noun concept 'pick-up branch' is a role of the noun concept 'branch'](#)

[At the actual start date/time of each rental it is obligatory that the fuel level of the rented car of the rental is full.](#)

Guidance Type: [operative business rule](#)

Description: A rented car must have a full tank of fuel at the rental pick-up time.

Note: This is an example of a rule created to ensure that real-world influences do not cause problems in EU-Rent's business. In this case, two requirements are met. First, a car must have some fuel in it for the customer to drive it away.

Second, starting fully-fuelled means that EU-Rent can easily estimate how much fuel is to be charged for at the end of the rental.

Enforcement Level: [post-justified](#)

Supporting fact types: [rental has start/date time](#)

[rental has rental car](#)

[rental car has fuel level](#)

[state of affairs occurs at date/time](#)

Related facts: [the concept 'rented car' is a role of the concept 'rental car'](#)

[fuel level is full or 7/8 or 3/4 or 5/8 or 1/2 or 3/8 or 1/4 or 1/8 or empty](#)

#### E.2.2.2.6 Rule Set -- Points Rental Rules

[It is necessary that the booking date/time of a points rental is at least 5 days before the scheduled start date/time of the rental.](#)

Guidance Type: [structural business rule](#)

Description:

Supporting fact types: [rental has booking date/time](#)

[rental has scheduled start date/time](#)

[date/time<sub>1</sub> is before date/time<sub>2</sub>](#)

Related facts: [the noun concept 'points rental' is a category of the noun concept 'rental'](#)

[the noun concept 'scheduled start date time' is a role of the noun concept 'date/time'](#)

[the noun concept 'booking date/time' is a role of the noun concept 'date/time'](#)

It is necessary that the renter of each points rental is a club member.

- Guidance Type: structural business rule
- Note: Only club members have points balances against which points rentals can be charged.  
Bookings for points rentals are not accepted from non-members.
- Supporting fact type: rental has renter
- Related facts: the noun concept 'points rental' is a category of the noun concept 'rental'  
the noun concept 'club member' is a role of the noun concept 'renter'

#### E.2.2.2.7 Rule Set -- Rental Period Rules

It is obligatory that the start date of each reserved rental is in the future.

- Synonymous Form: It is prohibited that the start date of a reserved rental is in the past.
- Guidance Type: operative business rule
- Description: A rental should not be booked or rescheduled with a start date/time earlier than the actual date/time of the booking or rescheduling.
- Note: On any given day, rentals that are due to be picked up that day should not be “reserved”, but “assigned” - i.e., they should have cars assigned to them.
- Enforcement Level: pre-authorized
- Supporting fact types: rental has start date  
date/time is in the future
- Related facts: the noun concept 'reserved rental' is a category of the noun concept 'rental'  
the noun concept 'start date' is a role of the noun concept 'date/time'

It is obligatory that the duration of each rental is at most 90 days.

- Guidance Type: operative business rule
- Description: EU-Rent doesn't allow rentals to be reserved for longer than 90 days or be extended beyond 90 days.
- Enforcement Level: pre-authorized
- Supporting fact type: rental has duration

If rental<sub>1</sub> is not rental<sub>2</sub> and the renter of rental<sub>1</sub> is the renter of rental<sub>2</sub> then it is obligatory that the rental period of rental<sub>1</sub> does not overlap the rental period of rental<sub>2</sub>.

- Guidance Type: operative business rule
- Description: A renter can have at most one open rental – i.e., can have only one rental car at a time.
- Enforcement Level: pre-authorized
- Supporting fact types: rental has renter  
rental has rental period  
period<sub>1</sub> overlaps period<sub>2</sub>



### E.2.2.2.8 Rule Set -- Servicing Rules

It is obligatory that each rental car in need of service has a scheduled service.

Guidance Type:	<u>operative business rule</u>
Description:	A rental car that has done more than 5000 miles since its last service is in need of service and has to be scheduled for service.
Note:	For countries that measure distance in kilometers, the figure is 8000
Enforcement Level:	<u>Deferred</u>
Supporting fact type:	<u>rental car has scheduled service</u>
Related fact:	' <i>being in need of service</i> ' is a characteristic of ' <u>rental</u> '

It is obligatory that the service reading of a rental car is at most 5500 miles.

Guidance Type:	<u>operative business rule</u>
Description:	A car must not be run for more than 5500 miles without being serviced.
Note:	For countries that measure distance in kilometers, the figure is 8800
Enforcement Level:	<u>pre-authorized</u>
Supporting fact types:	<u>rental car has service reading</u>

If the rented car of an open rental is in need of service or is in need of repair then it is obligatory that the rental incurs a car exchange during rental.

Guidance Type:	<u>operative business rule</u>
Description:	During a rental, if the rental car's service reading exceeds 5000 miles (8000 km), the renter must take the car to a branch
Enforcement Level:	<u>pre-authorized</u>
Supporting fact types:	<u>rental has rented car</u> <u>rental incurs car exchange during rental</u>
Related facts:	the noun concept ' <u>rented car</u> ' is a role of the noun concept ' <u>rental car</u> ' ' <i>being open</i> ' is a characteristic of the concept ' <u>rental</u> '

### E.2.2.2.9 Rule Set -- Transfer Rules

At the transfer drop-off date/time of a car transfer it is obligatory that the transferred car of the car transfer is owned by the local area that includes the transfer drop-off branch of the car transfer.

Guidance Type:	<u>operative business rule</u>
Description:	When a car is moved between branches in different local areas in the same country, ownership moves to the local area of the receiving branch.
Enforcement Level:	<u>Strict</u>
Supporting fact types:	<u>car transfer has transfer drop-off date/time</u> <u>car transfer has transfer drop-off branch</u> <u>car transfer has transferred car</u> <u>local area includes branch</u> <u>rental car is owned by local area</u> <u>state of affairs occurs at date/time</u>
Related facts:	the noun concept ' <u>transfer drop-off date/time</u> ' is a role of the noun concept ' <u>date/time</u> '

the noun concept 'transfer drop-off branch' is a role of the noun concept 'branch'  
the noun concept 'transferred car' is a role of the noun concept 'rental car'

It is obligatory that the country of the transfer drop-off branch of an international return is the country of registration of the transferred car of the international return.

Synonymous Form:	It is prohibited that the <u>country of the transfer drop-off branch of an international return is not the country of registration of the transferred car of the international return.</u>
Guidance Type:	<u>operative business rule</u>
Description:	When, as a result of a one-way rental, a car has been dropped off in a different country, it can be moved only back to its country of registration
Enforcement Level:	<u>pre-authorized</u>
Supporting fact types:	<u>car transfer has transfer drop-off branch</u> <u>car transfer has transferred car</u> <u>branch has country</u> <u>rental car has country of registration</u> <u>thing<sub>1</sub> is thing<sub>2</sub></u>
Related facts:	the noun concept 'transferred car' is a role of the noun concept 'rental car' the noun concept 'international return' is a category of the noun concept 'car transfer' the noun concept 'transfer drop-off branch' is a role of the noun concept 'branch'

At the drop-off date/time of an international return it is obligatory that the transferred car of the international return is owned by the local area that includes the transfer drop-off branch of the international return.

Guidance Type:	<u>operative business rule</u>
Description:	When a car is moved between branches in different local areas in the same country, ownership moves to the local area of the receiving branch.
Enforcement Level:	<u>pre-authorized</u>
Supporting fact types:	<u>car transfer has transfer drop-off branch</u> <u>car transfer has transferred car</u> <u>local area includes branch</u> <u>state of affairs occurs at date/time</u>
Related facts:	the noun concept 'transferred car' is a role of the noun concept 'rental car' the noun concept 'international return' is a category of the noun concept 'car transfer' the noun concept 'transfer drop-off branch' is a role of the noun concept 'branch'

#### E.2.2.2.10 EU-Rent Admonitions and Affirmations expressed in EU-Rent's English Vocabulary

It is possible that the notification date/time of a bad experience that occurs during a rental is after the actual return date/time of the rental.

Guidance Type:	<u>affirmation</u>
Note:	This is an unconditional expression - "after" has no business intent to imply there is a prohibition on "on or before"
Description:	A 'bad experience' may not be known at rental return.

The notification of, say, police action for a moving traffic offense may be received by EU-Rent some time after rental return.

Supporting fact types: [bad experience occurs during rental](#)  
[bad experience has notification date/time](#)  
[rental has actual return date/time](#)  
[date/time<sub>1</sub> is after date/time<sub>2</sub>](#)

Related facts: [the noun concept 'notification date/time' is a role of the noun concept 'date/time'](#)  
[the noun concept 'return date/time' is a role of the noun concept 'date/time'](#)

It is permitted that the [rental car that is moved by a car transfer is in need of service](#).

Guidance Type: [admonition](#)

Description: A car that is in need of service (i.e., has a service reading higher than 5000 miles) may be transferred between branches.  
All relevant rules apply. One that is important is that the car's service reading must not go over 5500 miles. So, if the distance between the branches would take the service reading over this limit, the transfer would not be allowed.

Note: Such a transfer would require that any service scheduled for the car be cancelled and a service scheduled at a service depot in the local area of the receiving branch.

Supporting fact types: [rental car is moved by car movement](#)  
[car transfer includes one-way car movement](#)

Related facts: [the concept 'one-way transfer movement' is a role of the concept 'car movement'](#)  
['being in need of service' is a characteristic of the concept 'rental car'](#)

It is permitted that a [renter has more than one advance rental](#).

Guidance Type: [admonition](#)

Description: A renter may make multiple rental bookings, each establishing an advance rental.

Note: There is an operative business rule that governs this permission. A renter is allowed to have only one car at a time in his possession, so the rental periods of his advance rentals must not overlap.

Supporting fact type: [renter has rental](#)

Related facts: [the concept 'advance rental' is a category of the concept 'rental'](#)

It is permitted that the [renter of a rental is an additional driver of a rental](#).

Guidance Type: [admonition](#)

Description: The person who is the renter for a rental may be an additional driver for another rental – even if both rentals are open at the same time.

Note: There is an operative business rule that governs this permission. A person cannot be the renter and an additional driver on the same rental.

Supporting fact types: [rental has renter](#)  
[rental has additional driver](#)

Related facts: [the concept 'renter' is a role of the concept 'driver'](#)  
[the concept 'additional driver' is a role of the concept 'driver'](#)

It is permitted that the drop-off branch of a rental is not the return branch of the rental.

Guidance Type:	<u>admonition</u>
Description:	There is no rule that allows an EU-Rent branch to refuse a rental return because it is not the return branch for the rental.
Note:	EU-Rent wants its cars back at the end of rental. It will accept return at any branch. It will charge a location penalty if the branch is not the return branch of the rental -- but, in any case, it wants its car back.
Supporting fact types:	<u>rental has drop-off branch</u> <u>rental has return branch</u> <u>thing<sub>1</sub> is thing<sub>2</sub></u>

#### E.2.2.2.11 EU-Rent Business Rules related to Business Processes

Business processes are outside the scope of SBVR and are the subject of another OMG RFP, “Business Process Definition Metamodel” (BPDM).

In practice, however, business rules are closely related to business processes. This section suggests how some process-related rules could be formulated, without encroaching on BPDM territory.

For example, “business process” and “event” are not explicitly defined in SBVR. Of course, an enterprise-specific vocabulary could define them.

In this section, “process” and “event” are implied in the business rules and vocabulary. This area is likely to change with further development of BPDM, and agreement within the OMG on how BPDM and SBVR should be integrated. It is suggested that these examples should be revisited then.

#### E.2.2.2.12 Example illustrating pre-conditions

It is obligatory that a request for pick-up is accepted only if an assigned rental matches the request for pick-up and the renter that is responsible for the assigned rental has a valid credit card and the renter provides current contact details and each driver of the rental has a valid driver license.

Guidance Type:	<u>operative business rule</u>
Enforcement Level:	<u>strict</u>
Note:	The (implied) process is “rental pick-up”. If the conditions are not met, the request is not accepted and the procedure is not started.
Note:	This rule in could be (and, in practice, probably should be) split into four simpler rules, each giving one precondition.
Supporting fact types:	<u>request for pick-up matches rental</u> <u>renter has credit card</u> <u>renter provides contact details</u> <u>driver has driver license</u>
Related facts:	<u>the noun concept ‘assigned rental’ is a role of the noun concept ‘rental’</u> <u>the noun concept ‘valid credit card’ is a role of the noun concept ‘credit card’</u> <u>the noun concept ‘current contact details’ is a category of the noun concept ‘contact details’</u> <u>the noun concept ‘valid driver license’ is a role of the noun concept ‘driver license’</u> <u>‘being accepted’ is a characteristic of ‘request for pick-up’</u>

### E.2.2.2.13 Example illustrating post-conditions

*At the actual start date/time of a rental it is obligatory that the estimated rental charge is provisionally charged to the credit card of the renter that is responsible for the rental and that the renter has possession of the rented car of the rental.*

Guidance Type: operative business rule

Enforcement Level: strict

Note: The actual start/time date of a rental is the expected end event of an (implied) “rental pick-up” process - the process whose pre-conditions were illustrated in the preceding example. If the pick-up is successful - so that the rental actually starts - the provisional charge will have been made and the renter will have the car.

Note: If the procedure fails (say, the charge to the credit card is not accepted, the renter is not able to use the controls of the car, or the car is not working and no substitute is available) the estimated charge is not made, the renter does not have possession of a EU-Rent car, and the rental does not start.

Note: As for the previous example, the business rule in the example could be split into two simpler rules, each giving one postcondition.

Supporting fact types: estimated rental charge is provisionally charged to credit card  
renter has credit card  
renter is responsible for rental  
renter has possession of rented car  
rental has rented car

### E.2.2.2.14 Examples illustrating Invariants provided by Structural Rules

*It is necessary that the cash rental price of each rental that is the responsibility of a corporate renter is based on the cash rental rates of the corporate rental agreement that is available to the corporate renter.*

Guidance Type: structural business rule

Supporting fact types: rental has base rental price  
cash rental price is based on cash rental rate  
corporate rental agreement has applicable rental rate  
renter is responsible for rental  
corporate rental agreement is available to corporate renter

Related facts: the noun concept ‘applicable rental rate’ is a role of the noun concept ‘cash rental rate’  
the noun concept ‘corporate renter’ is a category of the noun concept ‘renter’  
the noun concept ‘cash rental price’ is a category of the noun concept ‘base rental price’

*It is necessary that the rental duration of each rental that is the responsibility of a corporate renter is not greater than the maximum rental duration of the corporate rental agreement that is available to the corporate renter.*

Guidance Type: structural business rule

Supporting fact types: rental has rental duration  
corporate rental agreement has maximum rental duration  
renter is responsible for rental

Related facts: corporate rental agreement is available to corporate renter  
the noun concept 'maximum rental duration' is a role of the noun concept 'duration'  
the noun concept 'rental duration' is a role of the noun concept 'duration'  
the noun concept 'corporate renter' is a category of the noun concept 'renter'

#### E.2.2.2.15 Examples illustrating Business Rules that cause actions related to Events

It is obligatory that the insurer of each operating company is notified of each overdue rental that has a pick-up branch that is in the operating company.

Guidance Type: operative business rule  
Enforcement Level: deferred  
Note: The (implied) event is “scheduled return date/time + 24 hours”. As well as changing the status of a rental, it would cause the action “notify insurer”.  
Supporting fact types: operating company has insurer  
rental has pick-up branch  
pick-up branch is in operating company  
Related facts: 'being overdue' is a characteristic of 'rental'

If the drop-off location of a rental is not the EU-Rent site of a branch then it is obligatory that the rented car of the rental is recovered from the drop-off location to some branch.

Guidance Type: operative business rule  
Enforcement Level: deferred  
Note: The (implied) event is “notification of rental drop-off at a location that is not a branch” - it would cause the action “recover car”  
Supporting fact types: rental has drop-off location  
rental organization unit is based at EU-Rent site  
rented car is recovered from non-EU-Rent location to branch  
thing<sub>1</sub> is thing<sub>2</sub>  
Related facts: the noun concept 'drop-off location' is a role of the noun concept 'location'  
the noun concept 'non-EU-Rent-location' is a category of the noun concept 'location'  
the noun concept 'EU-Rent-site' is a role of the noun concept 'location'  
the noun concept 'branch' is a role of the noun concept 'rental organization unit'

## E.2.3 Common Vocabulary

This section illustrates some common SBVR vocabulary that could be adopted by enterprise-specific vocabularies.

In reality this vocabulary would be larger, containing many common terms and forms of expression useful in describing enterprises. Here we have included some extracts that are directly relevant to the EU-Rent example in this annex.

### E.2.3.1 General

#### text

Source: [Unicode 4.0.0 Glossary](#) ['Character Sequence']

General Concept: [expression](#)

#### thing<sub>1</sub> is thing<sub>2</sub>

Definition: The [thing<sub>1</sub>](#) and the [thing<sub>2</sub>](#) are the same [thing](#)

#### thing [individual concept] is changed

Definition: the extension of the [individual concept](#) is different at one point in time from what it is at a subsequent point in time

### E.2.3.2 Numbers

#### integer

Definition: number with no fractional part

Note: The [Integer Namespace](#), in the [Namespace Registration Vocabulary](#), has designations for all of the integers

#### integer<sub>1</sub> is less than integer<sub>2</sub>

Definition: The [integer<sub>1</sub>](#) is numerically less than the [integer<sub>2</sub>](#)

Synonymous Form: [integer<sub>1</sub> < integer<sub>2</sub>](#)

Synonymous Form: [integer<sub>2</sub> is greater than integer<sub>1</sub>](#)

Synonymous Form: [integer<sub>2</sub> > integer<sub>1</sub>](#)

#### nonnegative integer

Definition: [integer](#) that is greater than or equal to zero

Synonym: [whole number](#)

Synonym: [nonnegative integer](#)

### E.2.3.3 Time

#### actual date/time

Concept Type: [role](#)

Definition: [date/time](#) at which a [state of affairs](#) occurs

Description: Used in business rules such as “the rental start date requested on a rental reservation must not be earlier than the actual date/time of submission of the reservation”.

## date

Definition: date/time that is to the precision of year-month-day

## date/time

Source: the point of time at which a transaction or event takes place or is appointed to take place:  
a given point of time MWU ["date" 2,2]

Source: a point or period when something occurs : the moment of an event, process, or condition  
MWU ["time" 2,2A]

## date/time<sub>1</sub> *is after* date/time<sub>2</sub>

## date/time<sub>1</sub> *is before* date/time<sub>2</sub>

## date/time *is in the future*

Definition: date/time *being after* the date/time of the current moment

Example: Each reserved rental (rental that does not yet have a car assigned) should have a scheduled pick-up date/time *that is in the future*.

## date/time *is in the past*

Definition: date/time *being before* the date/time of the current moment

Each returned rental (rental for which the car has been returned to EU-Rent) should have an actual return date/time *that is in the past*.

## duration

Concept Type:

Definition: quantity of elapsed time of a period, measured in some time unit(s)

## duration<sub>1</sub> *is at most* duration<sub>2</sub>

## duration *is measured in* time unit

Definition: Each duration *is measured in* at least one time unit.

## end date/time

Concept Type: role

Definition: date/time at which period concludes

## period

Concept Type:

Definition: A time interval measured from a start date/time to an end date/time

Necessity: The start date/time of each period is before the end date/time.

Example: "From 23-April-2004/11:30 to 27-April-2004/17:50"

Note: period is related to, but different from, duration. For the example above, the duration is "4 days, six hours and 20 minutes". Different periods can have the same duration.

## period *has* duration

Concept Type: is-property-of fact type



### period has end date/time

Concept Type: [is-property-of fact type](#)

### period has start date/time

Concept Type: [is-property-of fact type](#)

### period<sub>1</sub> overlaps period<sub>2</sub>

Definition: (the [start date/time of period<sub>1</sub>](#) is after the [start date/time of period<sub>2</sub>](#) and before the [end date/time of period<sub>2</sub>](#)) or (the [end date/time of period<sub>1</sub>](#) is after the [start date/time of period<sub>2</sub>](#) and before the [end date/time of period<sub>2</sub>](#)).

### start date/time

Concept Type: [role](#)

Definition: [date/time](#) at which [period](#) begins

### state of affairs occurs after date/time

Concept Type: [associative fact type](#)

### state of affairs occurs at date/time

Concept Type: [associative fact type](#)

### state of affairs occurs before date/time

Concept Type: [associative fact type](#)

### state of affairs<sub>1</sub> occurs before state of affairs<sub>2</sub> occurs

Concept Type: [associative fact type](#)

## E.2.3.3.1 Example of a reusable structure in common vocabulary

Fixed and variable periods, described below, are structures that can play roles included in other concepts. For example, “variable period” (with all its necessities and possibilities) is included in EU-Rent’s rental, with the role name “rental period”.

### fixed period

Definition: [period](#) that cannot be changed

Example: Period in the past, e.g., the OMG Burlingame meeting time.

Example: Period defined by clock or calendar, e.g., “first ten days in May”.

Example: Period in the future fixed by fiat, e.g., trip for which you have bought air tickets that cannot be rescheduled or refunded.

### fixed end date/time

Concept Type: [role](#)

Definition: [date/time](#) that is the end of a [fixed period](#)

### fixed start date/time

Concept Type: [role](#)

Definition: [date/time](#) that is the start of a [fixed period](#)

### fixed period has fixed end date/time

Necessity: The end date/time of a fixed period is not changed.

### fixed period has fixed start date/time

Necessity: The start date/time of a fixed period is not changed.

### variable period

Definition: period that can be rescheduled

Example: period of a EU-Rent rental

### variable period has actual start date/time

Necessity: Each variable period has at most one actual start date/time

Necessity: The actual start date time of a variable period is not changed.

### variable period has actual end date/time

Necessity: Each variable period has at most one actual end date/time

Necessity: The actual end date time of a variable period is not changed.

### variable period has scheduled start date/time

Necessity: Each variable period has exactly one scheduled start date/time

Possibility: The scheduled start date/time of a variable period is changed before the actual start date/time of the variable period.

Necessity: The scheduled start date/time of a variable period is not changed after the actual start date/time of the variable period.

Note: Additional constraints may be added in specific contexts - e.g., in EU-Rent the cut-off for changing the start date of a points rental is 5 days before its scheduled start date/time.

### variable period has scheduled end date/time

Necessity: Each variable period has exactly one scheduled end date/time

Possibility: The scheduled end date/time of a variable period is changed before the actual end date/time of the variable period.

Necessity: The scheduled end date/time of a variable period is not changed after the actual end date/time of the variable period.

Note: Additional constraints may be added in specific contexts - e.g., EU-Rent won't allow the scheduled end date of a rental to be changed so that the rental would have duration of more than 90 days.

### variable period has duration

Description: Duration of a variable period is measured in one of three ways, depending on what is known at the time of measurement:

(1) Before the actual start date/time the duration of a variable period is measured from scheduled start date/time to scheduled end date/time.

(2) At any date/time between actual start date/time and actual end date/time the duration of a variable period is measured from actual start date/time to scheduled end date/time.

(3) At any date/time after the actual end date/time the duration of a variable period is measured from actual start date/time to actual end date/time (i.e., the period is then fixed).

## Annex F (informative)

### The RuleSpeak<sup>®</sup> Business Rule Notation

RuleSpeak<sup>®</sup> is an existing, well-documented<sup>1</sup> business rule notation developed by Business Rule Solutions, LLC (BRS) that has been used with business people in actual practice in large-scale projects since the second half of the 1990s.

Annex C presented a business rule notation within SBVR Structured English that features prefixing rule keywords onto appropriate propositions. RuleSpeak can also use the constructs of SBVR Structured English, but embeds equivalent keywords within the propositions themselves (mixfix).

As discussed in Annex A, more than one notation for expressing business rules is possible using SBVR Structured English. (This is probably also true for other notations compliant with SBVR). Regardless of how expressed, equivalent semantics can be captured<sup>2</sup> and formally represented as logical formulations.

The following selected examples using the EU-Rent case study illustrate use of RuleSpeak. The complete set of examples for EU-Rent in RuleSpeak is provided in section F.3, with additional comments.

- |   |                          |   |
|---|--------------------------|---|
| 1 | Structural business rule | It is necessary that each <u>rental</u> <i>has</i> exactly one <u>requested car group</u> .   |
|   | <b>RuleSpeak version</b> | Each <u>rental</u> always <i>has</i> exactly one <u>requested car group</u> .   |
| 2 | Operative business rule  | It is obligatory that the <u>duration</u> of each <u>rental</u> <i>is at most</i> 90 days.  |
|   | <b>RuleSpeak version</b> | The <u>duration</u> of a <u>rental</u> must not <i>be more than</i> 90 days.  |
| 3 | Operative business rule  | It is obligatory that each <u>driver</u> of a <u>rental</u> <i>is a</i> <u>qualified driver</u> .   |
|   | <b>RuleSpeak version</b> | A <u>driver</u> of a <u>rental</u> must <i>be a</i> <u>qualified driver</u> .   |
| 4 | Operative business rule  | If the <u>drop-off location</u> of a <u>rental</u> <i>is not the</i> <u>EU-Rent site</u> of the <u>return branch</u> of the <u>rental</u> then it is obligatory that the <u>rental</u> <i>incurs a</i> <u>location penalty charge</u> . |
|   | <b>RuleSpeak version</b> | A <u>rental</u> must <i>incur a</i> <u>location penalty charge</u> if the <u>drop-off location</u> of the <u>rental</u> <i>is not the</i> <u>EU-Rent site</u> of the <u>return branch</u> of the <u>rental</u> .                        |

---

1. [Ross2003], Chapters 8-12. Versions of RuleSpeak have been available on the Business Rule Solutions, LLC website ([www.BRSolutions.com](http://www.BRSolutions.com)) since the late 1990s. Public seminars have taught the syntax to thousands of professionals starting in 1996 ([www.AttainingEdge.com](http://www.AttainingEdge.com)). The original research commenced in 1985, and was originally published in 1994 [Ross1997].

2. For a business-oriented, SBVR-compliant approach, see [Ross2005], Chapters 4-5.

5	Structural business rule	It is necessary that the <u>rental charge of a rental is calculated in the business currency of the rental.</u>
	<b>RuleSpeak version</b>	The <u>rental charge of a rental is always calculated in the business currency of the rental.</u>
6	Operative business rule	It is permitted that a <u>rental is open only if an estimated rental charge is provisionally charged to the credit card of the renter of the rental.</u>
	<b>RuleSpeak version</b>	A <u>rental may be open only if an estimated rental charge is provisionally charged to the credit card of the renter of the rental.</u>
7	Operative business rule	It is obligatory that <u>at the actual return date/time of each in-country rental and each international inward rental the local area of the return branch of the rental owns the rented car of the rental.</u>
	<b>RuleSpeak version</b>	The <u>local area of the return branch of an in-country rental or international inward rental must own the rented car of the rental at the actual return date/time of the rental.</u>
8	Operative business rule	It is obligatory that <u>at the actual start date/time of each rental the fuel level of the rented car of the rental is full.</u>
	<b>RuleSpeak version</b>	The <u>fuel level of the rented car of a rental must be full at the actual start date/time of the rental.</u>
9	Affirmation	It is possible that the <u>notification date/time of a bad experience that occurs during a rental is after the actual return date/time of the rental.</u>
	<b>RuleSpeak version</b>	The <u>notification date/time of a bad experience that occurs during a rental is sometimes after the actual return date/time of the rental.</u>
10	Admonition	It is permitted that the <u>drop-off branch of a rental is not the return branch of the rental.</u>
	<b>RuleSpeak version</b>	The <u>drop-off branch of a rental need not be the return branch of the rental.</u>

## F.1 Expressions in RuleSpeak

RuleSpeak builds on the same expression forms described in Annex C (C.1), with the minor difference that distinct keywords are used for the Modal Operations related to business rules. The following section presents the RuleSpeak alternative rule keywords for Rules and Affirmations and Admonitions.<sup>3</sup>

---

3. It is important to note that use of these keywords must be in a context that is clearly indicated to be for Rules and Affirmations/Admonitions only.

### F.1.1 Modal Operations in RuleSpeak

Modality claim type	Statement form	SBVR Structured English keywords	RuleSpeak keywords
obligation claim	‘obligative statement’ form	it is obligatory that $p$	$r$ must $s$
obligation claim embedding a logical negation	‘prohibitive statement’ form	it is prohibited that $p$	$r$ must not $s$
	‘restricted permission statement’ form	it is permitted that $p$ only if $q$	$r$ may $s$ only $t$
permissibility claim	‘unrestricted permission statement’ form	it is permitted that $p$	$r$ may $s$ $r$ need not $s$
necessity claim	‘necessity statement’ form	it is necessary that $p$	$r$ always $s$
obligation claim embedding a logical negation	‘impossibility statement’ form	it is impossible that $p$	$r$ never $s$
	‘restricted possibility statement’ form	it is possible that $p$ only if $q$	$r$ can $s$ only $t$
possibility claim	‘unrestricted possibility statement’ form	it is possible that $p$	$r$ sometimes $s$ $r$ can $s$

NOTES:

- $p$  and  $q$ , and  $r$ ,  $s$ , and  $t$ , are all parts of the same proposition, say  $u$ .
- In a permissibility claim or a possibility claim, the ‘only’ is always followed immediately by one of the following:
  - an ‘if’ (yielding ‘only if’).
  - a preposition.

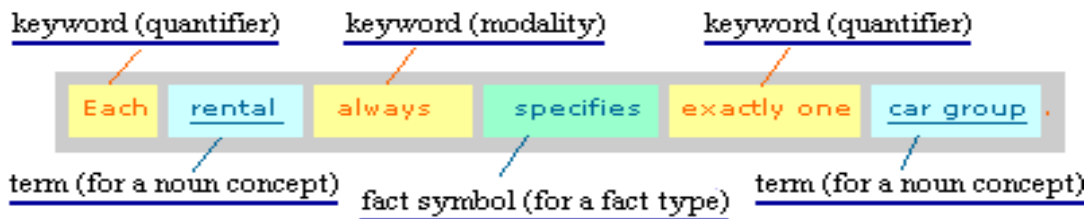
An example of a business rule statement using the ‘only [preposition]’ form is the following:

A spot discount for a rental may be given **only** *by* a branch manager.

### F.1.2 Example in RuleSpeak

Each **rental** **always** *specifies* exactly one **car group**.

The example above includes three keywords or phrases, two terms, and one fact symbol, as illustrated below.



As noted above, every Operative Business Rule or Affirmation or Admonition can be stated by using one of the following embedded keywords.

must	or should	rule keyword
must not	or should not	rule keyword
only	often as in only if	rule keyword
may	or need not	affirmation/admonition keyword

Every Structural Rule or Affirmation or Admonition can be stated by using one of the following embedded keywords.

always		rule keyword
never		rule keyword
can ... only	often as in can ... only if	rule keyword
sometimes	or not always	affirmation/admonition keyword

Special-purpose keywords for indicating specific kinds of Structural Rules include the following. In these forms, “always” is assumed implicit.

is to be considered		for derivation or inference
is to be computed as		for computation
is to be fixed at [number]	or is to be [number]	for establishing constants

Among the most basic usage rules and guidelines of RuleSpeak are the following. (Note that these usage rules and affirmations/admonitions are given using proper RuleSpeak notation.)

1. ‘Should’ may be used in place of ‘must’ in expressing a business rule *only if* one of the following is true:
  - The business rule does not have an enforcement level.
  - The business rule has an enforcement level, and that enforcement level is consistent with the English sense of ‘should’.

Comment: To say this differently:

‘Should’ *must not* be used in place of ‘must’ in expressing a business rule if all of the following are true:

- The business rule has an enforcement level.
- The enforcement level of the business rule is inconsistent with the English sense of ‘should’.

2. ‘May’ *must* be used in the sense of “permitted to” in RuleSpeak. ‘May’ *must not* be used in the sense of “might”.<sup>4</sup>

4. [Ross2003], p. 130.

3. An affirmation or admonition **must not** include a rule keyword.
4. A statement expressing a rule or affirmation or admonition **should not** begin with a condition.

Comment: ‘Condition’ as used here means a qualification set off by ‘if’, ‘while’, ‘when’, etc. (e.g., if a rental is open...).

5. A double negative **should** be avoided in expressing a rule in RuleSpeak.

Comment: Double negatives, especially using two ‘not’s, are generally undesirable in good English usage, and often prove particularly troublesome in rule statements.

Example.

Rule: A withdrawal from an account must not be made if the account is not active.

Revised rule: A withdrawal from an account may be made only if the account is active.

Comment: The revised rule is expressed in the form of a ‘restricted permissive statement.’

## F.2 Concepts, Definitions and Rules: RuleSpeak Practices

SBVR is very flexible in supporting alternative practices with respect to rules and definitions. This flexibility is enabled by the underlying logical formulations and their underpinning in formal logic.

Two core RuleSpeak practices with respect to definitions are the following.<sup>5</sup>

1. **“Essence” by Definitions.** A definition should always focus on the core essence of a concept – that is, on *fundamental* meaning that is unlikely to change. Such meaning is expressed as naturally as possible. The form of language used in common dictionaries is strongly preferred.
2. **“Boundaries” by Rules.** All constraints should be expressed as rules separate from definitions. Such rules generally define the ‘boundary conditions’ of a concept; that is, when something is or is not an instance of the concept. Since specific boundaries for a concept (e.g., “gold customer”) can change over time, they should not be embedded in definitions. An additional advantage – crucial for communication with and among business people – is that the underlying vocabulary can be kept as compact and as focused as possible.

Experience in large-scale projects indicates that these core practices:

- Ensure good business communication.
- Produce friendly and highly stable definitions.
- Scale extremely well for complex business problems featuring hundreds or thousands of rules.

RuleSpeak might therefore be characterized as more ‘rule-ish’ than the approach described in Annex E. RuleSpeak is well-suited for practitioners who want to:

- Move faster to rule capture.
- Use more natural (less formal) wordings for definitions.

These issues are pragmatic concerns for business rule projects. It is important to remember, of course, that under SBVR either approach (and others) can produce identical semantics ‘under the covers’ (i.e., in logical formulations).

5. [Ross2005], Chapter 4, pp 51-52.

## F.2.1 Example in RuleSpeak

A EU-Rent definition and set of related specifications taken from Annex E concerning “agency” (a type of “branch”) serve to illustrate. The RuleSpeak approach is outlined subsequently.

### F.2.1.1 Sample Definition and Related Specifications for “Agency” from Annex E

‘Agency’: service desks in hotels, travel agents, etc. They have storage space for few cars, and are operated on demand by part-time staff who will typically do the entire workflows for rental and return.

#### agency

Concept Type: [branch type](#)

Definition: [branch](#) that does not have a [EU-Rent location](#) and has minimal car storage and has on-demand operation

#### rental organization unit [having a EU-Rent location](#)

Concept Type: [characteristic](#)

Definition: [rental organization unit](#) that is based at a [EU-Rent site](#) that is owned by [EU-Rent](#)

Note: [Some things](#) are based at [EU-Rent sites](#) that are owned by third parties such as hotels and travel agents.

#### rental organization unit [having minimal car storage](#)

Concept Type: [characteristic](#)

Definition: [rental organization unit](#) that has [car storage](#) that can accommodate a small number of rental cars

#### rental organization unit [having on-demand operation](#)

Concept Type: [characteristic](#)

Definition: [rental organization unit](#) that has [hours of operation](#) that are flexible in response to customer demand

### F.2.1.2 RuleSpeak Approach for the “Agency” Example

1. Find a suitable definition from a standard dictionary, or if available, an industry glossary, to serve as the basis for the definition. The Merriam-Webster Unabridged Dictionary offers the following for “agency”, an appropriate basis for an ‘essence’ definition.

*4a: an establishment engaged in doing business for another \*an advertising agency\* \*an employment agency\**

#### agency

Concept Type: [branch type](#)

Definition: another company engaged in conducting EU-Rent business operations

2. Define Fact Types for ‘agency’. For this example, assume an agency has the following (binary) fact types by virtue of being a branch. These fact types would probably be indicated as properties.



- [branch has location](#)
- [branch has car storage capacity](#)
- [branch has operating mode](#)

Comments:

- RuleSpeak does not emphasize using characteristics for building definitions.
  - In practice, fact types are generally not given definitions in RuleSpeak. If all noun concepts represented by roles for a fact type are well-defined, a definition for the fact type itself generally adds very little.
  - For the sake of simplicity, assume that location, car storage, and operating mode already have suitable definitions.
3. Define the appropriate structural rule(s) to establish (current) ‘boundaries’ for the concept ‘agency’. Note that these ‘boundaries’ might be modified, expanded, or contracted over time.

All of the following are always true for an [agency](#):

- It [has a third-party location](#).
- It [has a minimal car storage capacity](#).
- Its [operation mode is on-demand](#).

4. Specify structural rules for derived terms (e.g., “third-party location”, “minimal”, etc.).

[A location is to be considered a third-party location if located at an EU-Rent site that is owned by a third party.](#)

[The car storage capacity of a branch is to be considered minimal if less than ... \[condition\(s\)\]](#)

5. Ensure all non-derived terms have “essence” definitions.

### **on-demand**

Definition: flexible in response to customer demand

Comment: Derived concepts are generally not given definitions in RuleSpeak since the structural rule(s) for them are, literally, *definitive*.

## **F.2.2 Structural Rules vs. Operative Rules**

In RuleSpeak, the distinction between structural rules and operative rules is viewed as follows.<sup>6</sup>

- *Structural rules* prescribe criteria for how the business chooses to organize (“structure”) its business semantics. Such rules express criteria for correct decisions, derivations, or business computations. Structural rules supplement definitions.
- *Operative business rules* focus directly on the propriety of conduct in circumstances (business activity) where willful or uninformed actions can fall outside the boundaries of behavior deemed acceptable. Unlike structural rules, operative rules can be violated *directly*.

The distinction is clear-cut in most cases; in some, it is more difficult. For example, consider “booking” in the EU-Rent case study. “Booking” (like “order”, “reservation”, “registration”, etc.) is essentially a ‘made-up’ device of the business. It is an artifact of knowledge that exists ‘simply’ to help manage complex, expensive resources.

6. [Ross2005], Chapters 5 and 6.

Therefore, rules about creating bookings (e.g., that the requested pick-up date-time is to be **after** the booking date-time) are to be viewed as structural. If not followed (applied) in creating a booking, the booking is simply invalid. In other words, since bookings are a knowledge 'thing', the business can establish definitive rules for them. These are the “boundary” rules discussed earlier.

Now consider "actual pick-up date-time", the date-time when possession of a rental car is actually handed over to a rental customer (or is *said* to have been anyway). EU-Rent might want to avoid post-dating handovers -- i.e., have a rule that the actual pick-up date-time is to be after the booking date-time.

This case is quite different. "Actual pick-up date-time" reflects activity (or the communication thereof) outside the realm of knowledge artifacts -- i.e., conduct that takes place in the 'real world'. Because such rules can be broken (by people), they are operative.

In borderline cases, Rule Speak best practice is the following:<sup>7</sup>

RuleSpeak Best Practice: Carefully distinguish what should be (according to the structural rules) vs. what really is, based on actual business decisions / actions. One or more operative rules are then specified to constrain 'what really is' against 'what should be'. These latter rules, being operative, are the ones that can be violated.

### F.3 Complete Set of EU-Rent Examples in RuleSpeak

This section provides one-by-one RuleSpeak counterparts for the EU-Rent guidance (business rules and affirmations/admonitions) presented in Annex E<sup>8</sup>. This restatement provides semantically equivalent expression that is more business friendly.

Comment: Many of the guidance statements in Annex E are supported by descriptions, which reflect EU-Rent users' informal statements of the guidance, and by fact types and levels of enforcement. That material has been removed from here for the sake of brevity. Refer to Annex E for details.

#### F.3.1 Rule Set -- Rental Rules

It is necessary that each rental *has* exactly one requested car group.

A rental always *has* exactly one requested car group.

Comment: “A” may be used in place of “each” with no change in meaning, as follows<sup>9</sup>.  
(This note will not be repeated subsequently.)

Each rental always *specifies* exactly one car group

Guidance Type: structural business rule

7. [Ross2005], Chapter 6, pp. 107-108.

8. As of the time of this writing.

9. [Ross2003]

It is necessary that each rental *has* exactly one rental period.

A rental always *has* exactly one rental period.

Guidance Type: structural business rule

It is necessary that each rental *has* exactly one return branch.

A rental always *has* exactly one return branch.

Guidance Type: structural business rule

It is necessary that the scheduled pick-up date/time of each advance rental *is after* the booking date/time of the rental booking that *establishes* the advance rental.

The scheduled pick-up date/time of an advance rental *is always after* the booking date/time of the rental booking that *establishes* the advance rental.

Guidance Type: structural business rule

### F.3.2 Rule Set -- Charging / Billing / Payment Rules

It is permitted that a rental *is open* only if an estimated rental charge *is provisionally charged* to the credit card of the renter of the rental.

A rental may *be open* only if an estimated rental charge *is provisionally charged* to the credit card of the renter of the rental.

Guidance Type: operative business rule

It is necessary that the rental charge of each rental *is calculated in* the business currency of the rental.

The rental charge of a rental *is always calculated in* the business currency of the rental.

Guidance Type: structural business rule

If the renter of a rental requests a price conversion then it is obligatory that the rental charge of the rental is converted to the currency of the price conversion.

The rental charge of a rental must be converted to the currency of a price conversion requested by the renter of the rental.

Comment: RuleSpeak does not recommend the “If ...then...” syntax for operative business rules<sup>10</sup>.  
(This note will not be repeated subsequently.)

Guidance Type: operative business rule

It is necessary that each cash rental honors its lowest rental price.

A cash rental always honors its lowest rental price.

Guidance Type: structural business rule

[From Annex E] “The structural business rule above can be elaborated as three detailed structural business rules:”

It is necessary that a cash rental price for a cash rental that is calculated because of EU-Rent price changes and that *is less than* the lowest rental price honored by the rental replaces the lowest rental price honored by the rental.

A cash rental price for a cash rental that is calculated because of EU-Rent price changes and that *is less than* the lowest rental price honored by the rental always replaces the lowest rental price honored by the rental.

It is necessary that a cash rental price for a cash rental that is calculated because of changes to the car group or rental duration of a rental replaces the lowest rental price honored by the rental.

A cash rental price for a cash rental that is calculated because of changes to the car group or rental duration of a rental always replaces the lowest rental price honored by the rental.

---

10. [Ross2003].

It is necessary that the lowest rental price honored by a rental is not replaced *after the actual return date/time of the rental*.

The lowest rental price honored by a rental is never replaced *after the actual return date/time of the rental*.

### F.3.3 Rule Set -- Driver Rules

It is permitted that a rental is open only if each driver of the rental is not a barred driver.

A rental may be open only if each driver of the rental is not a barred driver.

Synonymous Form: It is prohibited that a rental is open if a driver of the rental is a barred driver.

A rental must not be open if a driver of the rental is a barred driver.

Guidance Type: operative business rule

It is obligatory that each driver of a rental is a qualified driver.

Each driver of a rental must be a qualified driver.

Guidance Type: operative business rule

### F.3.4 Rule Set -- Pick-up / Return Rules

It is obligatory that *at the actual return date/time of each in-country rental and each international inward rental the local area of the return branch of the rental owns the rented car of the rental*.

The local area of the return branch of an in-country or international inward rental always *owns the rented car of the rental at the actual return date/time of the rental*.

NOTE: RuleSpeak treats this rule as structural, rather than operative, for the reasons given earlier.

Guidance Type: structural business rule

It is obligatory that the country of the return branch of each international inward rental is the country of registration of the rented car of the rental.

The country of the return branch of an international inward rental is always the country of registration of the rented car of the rental.

NOTE: RuleSpeak treats this rule as structural, rather than operative, for the reasons given earlier.

Guidance Type: structural business rule

It is necessary that if a rental is open and the rental is not an international inward rental then the rented car of the rental is owned by the local area of the pick-up branch of the rental.

The rented car of a rental is always owned by the local area of the pick-up branch of the rental if the rental is open and the rental is not an international inward rental.

Guidance Type: structural business rule

If the actual return date/time of a rental is after the end date/time of the grace period of the rental then it is obligatory that the rental incurs a late return charge.

A rental must incur a late return charge if the actual return date/time of the rental is after the end date/time of the grace period of the rental.

Guidance Type: structural business rule

If the drop-off location of a rental is not the EU-Rent site of the return branch of the rental then it is obligatory that the rental incurs a location penalty charge.

A rental must incur a location penalty charge if the drop-off location of the rental is not the EU-Rent site of the return branch of the rental.

Guidance Type: operative business rule

If a rental is assigned then it is obligatory that the rental car that is assigned to the rental is stored at the pick-up branch of the rental.

The rental car assigned to a rental must be stored at the pick-up branch of the rental if the rental is assigned.

Guidance Type: operative business rule

At the actual start date/time of a rental it is obligatory that the fuel level of the rented car of the rental is “full”.

The fuel level of a rental car assigned to a rental must be “full” at the actual start date/time of the rental.

Guidance Type: operative business rule

### F.3.5 Rule Set -- Points Rental Rules

It is necessary that the booking date/time of a points rental is at least 5 days before the scheduled start date/time of the rental.

The booking date/time of a points rental is always at least 5 days before the scheduled start date/time of the rental.

Guidance Type: structural business rule

It is necessary that the renter of each points rental is a club member.

The renter of a points rental is always a club member.

Guidance Type: structural business rule

### F.3.6 Rule Set -- Rental Period Rules

It is obligatory that the start date of each reserved rental is in the future.

The start date of a reserved rental must be in the future.

Guidance Type: operative business rule

It is prohibited that the start date of a reserved rental is in the past.

The start date of a reserved rental must not be in the past.

Guidance Type: operative business rule

It is obligatory that the duration of a rental is at most 90 days.

The duration of a rental must be at most 90 days.

Guidance Type: operative business rule

If rental<sub>1</sub> is not rental<sub>2</sub> and the renter of rental<sub>1</sub> is the renter of rental<sub>2</sub> then it is obligatory that the rental period of rental<sub>1</sub> does not overlap the rental period of rental<sub>2</sub>.

The rental period of rental<sub>1</sub> must not overlap the rental period of rental<sub>2</sub> if all the following are true:

- rental<sub>1</sub> is not rental<sub>2</sub>
- the renter of rental<sub>1</sub> is the renter of rental<sub>2</sub>.

Guidance Type: operative business rule

### F.3.7 Rule Set -- Servicing Rules

It is obligatory that each rental car that is in need of service has a scheduled service.

A rental car in need of service must have a scheduled service.

Guidance Type: operative business rule

It is obligatory that the service reading of a rental car is at most 5500 miles.

The service reading of a rental car is always at most 5500 miles.

NOTE: RuleSpeak treats this rule as structural, rather than operative, for the reasons given earlier.

Guidance Type: structural business rule

If the rented car of an open rental is in need of service or is in need of repair then it is obligatory that the rental incurs a car exchange during rental.

An open rental must incur a car exchange during rental if the rented car of the rental is in need of service or is in need of repair.

Guidance Type: structural business rule



### F.3.8 Rule Set -- Transfer Rules

At the transfer drop-off date/time of a car transfer it is obligatory that the transferred car of the car transfer is owned by the local area that includes the transfer drop-off branch of the car transfer.

The transferred car of a car transfer is always owned at the transfer drop-off date/time of a car transfer by the local area that includes the transfer drop-off branch of the car transfer.

NOTE: RuleSpeak treats this rule as structural, rather than operative, for the reasons given earlier.

Guidance Type: [structural business rule](#)

It is obligatory that the country of the transfer drop-off branch of an international return is the country of registration of the transferred car of the international return.

The country of the transfer drop-off branch of an international return is always the country of registration of the transferred car of the international return.

NOTE: RuleSpeak treats this rule as structural, rather than operative, for the reasons given earlier.

Guidance Type: [structural business rule](#)

Synonymous Form: [It is prohibited that the country of the transfer drop-off branch of an international return is not the country of registration of the transferred car of the international return.](#)

NOTE: A RuleSpeak expression of the Synonymous Form has been intentionally omitted. The form “prohibited ... not” (or impossible ... not”) is actually a double negative, which as noted earlier, RuleSpeak always discourages because it inevitably causes confusion.

- Prohibited that ... not” is equivalent to “obligatory that not ... not”, a double negative, which is shown more clearly in RuleSpeak, “must not ... not”.
- Impossible that ... not” is equivalent to “Necessary that not ... not”, which would be more clearly a double negative in RuleSpeak, “never ... not”.

At the drop-off date/time of an international return it is obligatory that the transferred car of the international return is owned by the local area that includes the transfer drop-off branch of the international return.

The transferred car of an international return is always owned at the drop-off date/time of an international return by the local area that includes the transfer drop-off branch of the international return.

NOTE: RuleSpeak treats this rule as structural, rather than operative, for the reasons given earlier.

Guidance Type: [structural business rule](#)

### F.3.9 EU-Rent Affirmations and Admonitions expressed in EU-Rent's English Vocabulary

It is possible that the notification date/time of a bad experience that occurs during a rental is after the actual return date/time of the rental.

The notification date/time of a bad experience that occurs during a rental is sometimes after the actual return date/time of the rental.

Guidance Type: affirmation

It is permitted that the rental car that is moved by a car transfer is in need of service.

The rental car moved by a car transfer may be in need of service.

Guidance Type: admonition

It is permitted that a renter has more than one advance rental.

A renter may have more than one advance rental.

Guidance Type: admonition

It is permitted that the renter of a rental is an additional driver of a rental.

The renter of a rental may be an additional driver of a rental.

Guidance Type: admonition

It is permitted that the drop-off branch of a rental is not the return branch of the rental.

The drop-off branch of a rental need not be the return branch of the rental.

Guidance Type: admonition

## Annex G (informative)

### Concept Diagram Graphic Notation

A business vocabulary can be presented to a business audience using four simple main conventions, described in this Annex. These conventions have been purposely kept neutral of any particular modeling notations, and have been selected to be largely self-explanatory and visually intuitive. Note that a diagram using these conventions is only one view of a vocabulary and is intended to help in understanding some particular aspects of the vocabulary and the conceptual schema that underlies it.

Various graphic constructs are used to provide visual clarity (e.g., color, shading, font, font size, etc.). Unless explicitly stated, none of these carry any semantic or syntactic meaning. A diagram can be viewed in grayscale without losing information.

#### G.1 Boxes -- Concepts

A box of any size represents a core concept. The name in the box is the preferred term (name) given to that concept. Refer to the Vocabulary for the precise meaning of each term. Because of the need to format within realistic bounds, some concepts re-appear in several diagrams.

For example, Figure G-1 depicts two concepts, termed ‘concept a’ and ‘concept s.’



Figure G.1 - [concept a](#) and [concept s](#) are core concepts within the model

#### G.2 Box-Within-A-Box -- Categories

##### G.2.1 Simple Categories

Straightforward categorization — where one element is a category of another element — is represented as a box within a box. Another way to think about this is that the inner box (the category) represents a specific kind or variation of the concept represented by the outer box (the more general concept).

There is no assumption in this graphic representation that box-within-a-box implies mutually exclusive categories, or represents an exhaustive or mandatory list of categories. When categories in the SBVR model are mutually exclusive, this constraint is documented in the Vocabulary. When the categories of a scheme are completely enumerated and required as shown, these constraints are documented in the Vocabulary.

For example, Figure G.2 depicts two concepts, termed ‘concept s’ and ‘concept t’ that are categories of a more general concept, termed ‘concept a.’

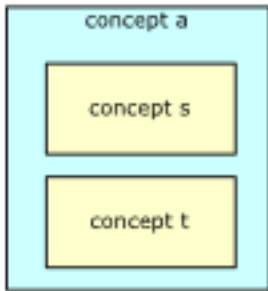


Figure G.2 - [concept s](#) and [concept t](#) are categories of [concept a](#)

## G.2.2 Categorization Schemes and Segmentations

In some cases, categories form part of a designated categorization scheme. For these, a dashed-line box is used to depict the categorization scheme within the concept box. The scheme box surrounds the categories that make up the scheme. The categorization scheme's name is shown at the top of the scheme box that the scheme is for. Note that a category may appear in more than one scheme.

When the categorization scheme depicts a 'segmentation' -- a categorization scheme in which the set of categories are mutually exclusive and complete -- these constraints are documented in the Vocabulary. This may also be shown on the diagram as '[segmentation]' after the categorization scheme name.

For example, Figure G.3 depicts a categorization scheme, named 'Scheme X' that is for a concept (termed 'concept a'). Two concepts (termed 'concept s' and 'concept t') make up the scheme.

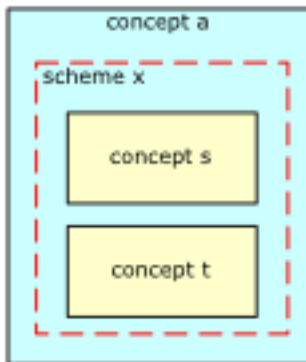


Figure G.3 - [concept s](#) and [concept t](#) are mutually exclusive categories of [concept a](#) within the categorization scheme ['scheme x'](#)

## G.3 Connections Between Boxes -- Fact Types

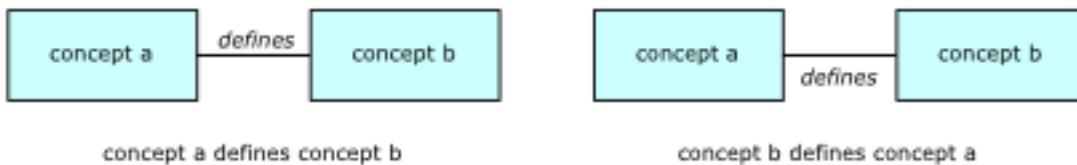
### G.3.1 Binary Fact Types

A line connecting any two boxes (or the same box twice) indicates a connection between core concepts. Such a line represents a fact type. The labels adjacent to the lines are written as verbs or verb phrases so that the facts of the SBVR model can be read as simple sentences. These sentences convey the meaning of the connections in the context of the SBVR model; however, more explanation is given in the Vocabulary, along with the definitions for each of the terms involved.

The rules that apply to these constructs are also part of the SBVR model. However, these rules are not expressed in the model graphics. For example, the connection lines represent simple unconstrained facts (i.e., ‘many-to-many’ and ‘optional’ in both directions). While the diagram may suggest some rules, the final word on any rule is documented in the Vocabulary.

To avoid clutter, only one reading of a fact type is shown in the graphics. The fact type is read clockwise around the line, from participating concept, to verb phrase, to (other) participating concept. Additional readings, as useful, are provided in the Vocabulary.

Figure G.4 depicts two fact types, with one reading for each.

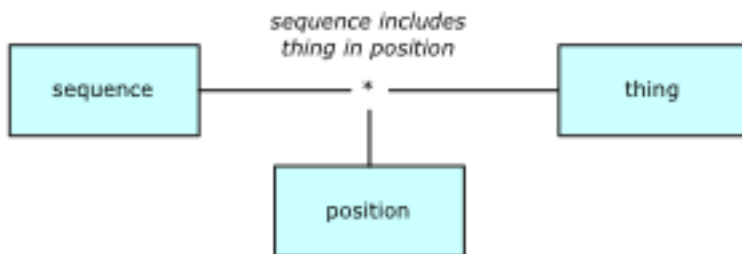


**Figure G.4 - Reading two fact types, using ‘defines’ as a typical verb phrase**

### G.3.2 N-ary Fact Types

Where a connection involves more than two core concepts, a simple line cannot be used to represent the fact type. In this case, the fact type is shown as \* with the fact type lines radiating from it to the participating concepts. The reading is placed adjacent to the \* and no verbs are written on the lines.

Figure G.5 illustrates a ternary fact type and one of its readings.



**Figure G.5 - An n-ary fact type**

### G.3.3 Unary Fact Types

Unary fact types are shown using a similar \* notation. A unary fact type is drawn as a line coming out of the concept box and ending with \*. The fact type verb phrase is placed adjacent to the \* symbol.

Figure G.6 illustrates a unary fact type.

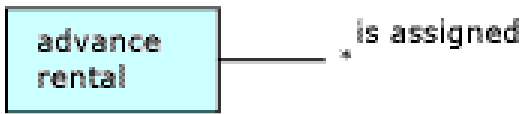


Figure G.6 - A unary fact type

### G.3.4 'Objectified' Fact Types

When a noun concept is defined using objectification such that it is coextensive with a fact type it is shown as a box labeled with the primary term for the noun concept. The reading of the fact type is provided in a legend (or glossary). To aid in visually distinguishing these fact type-objectifying noun concepts from other concepts, the concept name is marked with \* which provides the visual clue to look in the legend/glossary.

No verb phrase labels are written on the lines to the concepts that participate in the fact type. This permits the fact type itself to participate in other fact types without visual ambiguity.<sup>1</sup>

Figure G.7 depicts a fact type (rented car is rented from non-EU-Rent site to branch) and its objectification as the noun concept termed 'car recovery.'

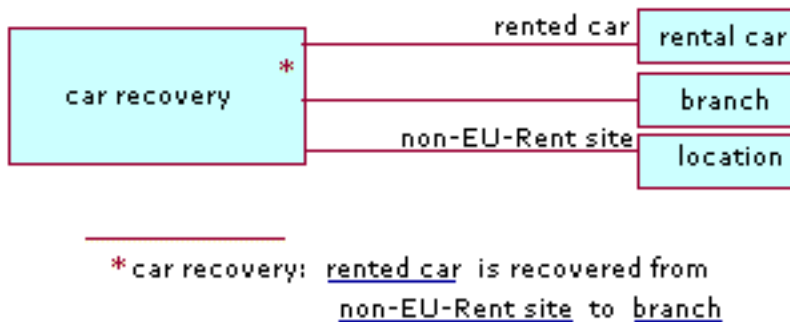


Figure G.7 - 'Objectified' fact type

## G.4 Roles

A role name may be given to a concept's participation in a fact type. This is reflected as a term (the role name) adjacent to the box for the concept playing the role in the fact type. There is no syntactic or semantic significance to the side of the line on which the role name is placed, other than careful placement to avoid confusion between the verb phrase and any role names.

1. There is a potential for confusion if the objectified fact type then participates in another fact type that is objectified, but this case is so rare that these conventions have elected simplicity for the typical cases over excruciating precision and the associated complexity.

Figure G.8 depicts a role name 'part' given to the concept termed 'concept b' in this fact type.



**Figure G.8 - Role name**





## Annex H (informative)

### Use of UML Notation in a Business Context to Represent SBVR-Style Vocabularies

The purpose of the UML diagrams in Part II and Annex E is to display a vocabulary graphically. This kind of UML model is commonly called a ‘Business Object Model’ (BOM). Note that these diagrams are not the MOF metamodel that is generated from a vocabulary, and these two uses of UML should not be confused.

A BOM is commonly used to convey business vocabulary (e.g., the SBVR vocabulary) so its use should be familiar. The diagrams do not show any special stereotypes as long as conventions are explained. This Annex provides that explanation.

#### H.1 General Concept (Noun Concept)

The primary term for a concept that is not a role, individual concept, or fact type is shown as a class (rectangle). The rectangle is labeled with the concept’s primary term, written just as the entry term appears in the vocabulary.

If there are additional terms for the concept they can be added within the rectangle, labeled as such -- e.g., “*also: is-category-of fact type*” as depicted in Figure H.1.

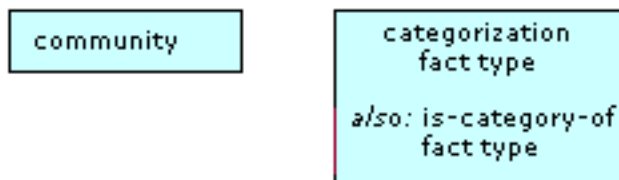


Figure H.1 - Two general concepts

#### H.2 Individual Concept

The name given to an individual concept is shown as an instance specification (rectangle). The name is followed by a colon and then by the term for its general concept. This text string is underlined within the rectangle.

While it is possible to have additional names for a given individual concept (i.e., names that are synonyms), the non-primary names of an individual concept are not typically reflected on the diagram. Figure H.2 depicts two individual concepts.

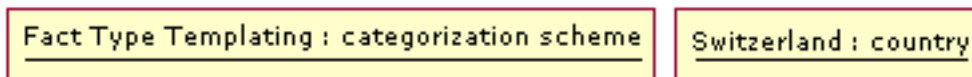


Figure H.2 - Two individual concepts

Alternatively, an individual concept can be depicted as an instance of its related general concept (noun concept), as in Figure H.3.

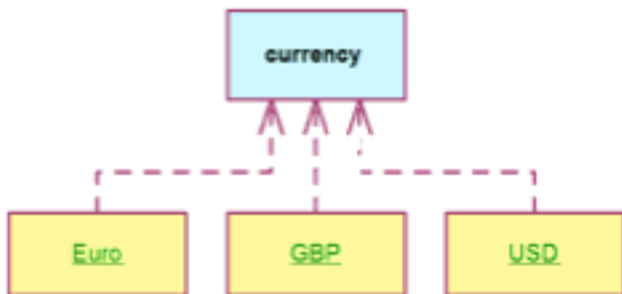


Figure H.3- Three individual concepts as instances of the related general concept

## H.3 Fact Types

Use of the UML association notation works well for representing fact types in an SBVR-based vocabulary diagram. However, it is important to remember that an SBVR fact type is not an association. A fact type is a classifier that has particular semantics.

### H.3.1 Binary Fact Types

The form of expression of a binary fact type, other than one using 'has', is shown as an association (a line between rectangles). If there is another form of expression for the fact type that reads in the opposite direction, only the active form is needed if the other form is the normal passive form for the same verb.

Alternatively, both forms can be shown, one above the line and the other below. Either the 'clockwise reading rule' or a solid triangle as an arrow can be used to show the direction of reading. Figure H.4 illustrates three alternative presentations of a binary fact type.

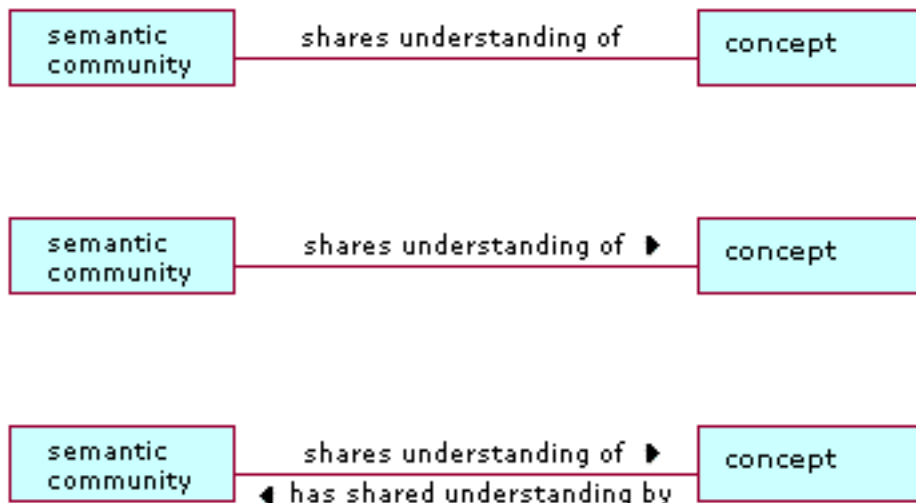


Figure H.4 - Three alternatives for presenting a binary fact type

### H.3.2 Binary Fact Types using 'has'

For each form of expression using 'has', the second role name is shown as an association end name. The verb 'has' is not shown on the diagram when giving an association end name. An end name implies 'has' as shown in Figure H.5. Any verb phrase shown is assumed to be usable without the end name.

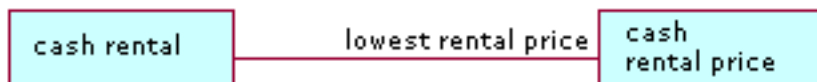


Figure H.5- Depicting the fact type 'cash rental has lowest rental price'

When a binary fact type's form of expression uses 'has' and there is no specialized role, the second role name is still reflected on the diagram in this consistent way (on the line adjacent to the rectangle) and 'has' is not displayed. This is illustrated in Figure H.6.



Figure H.6- Depicting the fact type 'branch has country'

### H.3.3 Fact Types with Arity of 3 or more

For fact types with more than two roles, the UML association notation is used. The primary form of expression is shown, with the placeholders included in square brackets as shown in Figure H.7.

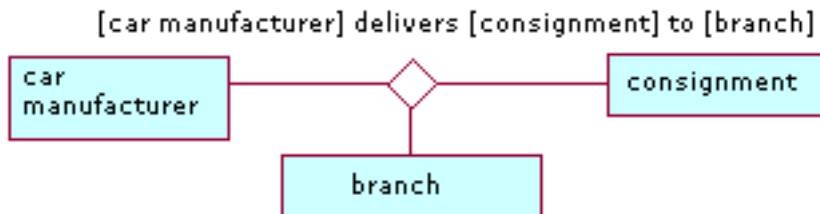


Figure H.7- Depicting a fact type with arity of three

### H.3.4 Unary Fact Types

UML associations only apply to binary and higher-arity. Ordinarily a unary fact type is transformed into a UML Boolean attribute, as shown in Figure H.8.

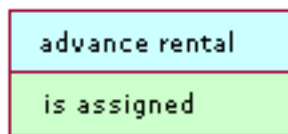


Figure H.8- Depicting the unary fact type 'advance rental is assigned' as a Boolean attribute

However, the SBVR unary fact type is more accurately modeled in UML using an alternative style, which applies the same conventions described in section F.3.3, adapted for the unary case shown in Figure H.9.



Figure H.9- Depicting the unary fact type 'advance rental is assigned' using association notation

## H.4 Roles

Note that a 'role' in SBVR is a concept in its own right.

### H.4.1 Role depicted as an Association End Name

A term for a role is typically shown as an association end name. Multiple appearances of the same role name coming into the same class imply a more general 'role' concept as well as the specific roles shown.

NOTE: Figure H.10 shows two forms of expression for the same fact type (see also section H.3.2).

[speech community](#) *uses* [vocabulary](#)  
[vocabulary](#) *has* [audience](#)

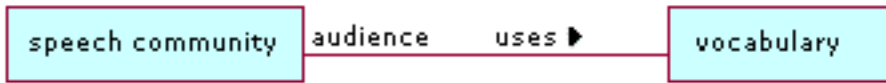


Figure H.10- Depicting a role as an association end name

### H.4.2 Role depicted using UML Stereotyping

Since a ‘role’ in SBVR is a concept in its own right it can also be depicted as a class (rectangle), with UML stereotyping used to denote the specialization. As illustrated in Figure H.11, the stereotype <<role>> can be reflected for the class or the generalization line can use the stereotype <<is-role-of>>.

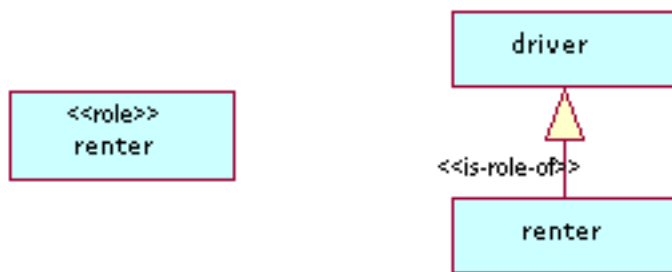


Figure H.11- Depicting a role as a class, with stereotyping

### H.4.3 Term for a Role in a Form of Expression

When a term for a role is used in a form of expression, and that form is not an attributive form (e.g., “a has b”), then the term for the role needs to be shown. It is not shown as an association end because that would imply an attribute form (e.g., “has”). Instead, the term for the role is shown in square brackets, along with the verbal part of the form of expression, as shown in Figure H.12.

In the first example: for the fact type “concept incorporates characteristic”, ‘characteristic’ is a term for a role -- the general concept is unary fact type. Rather than put “incorporates” on the association line connecting “concept” to “unary fact type”, the text on the line incorporates the term for the role and reads, “incorporates [characteristic]”. A second example (from EU-Rent) is given.

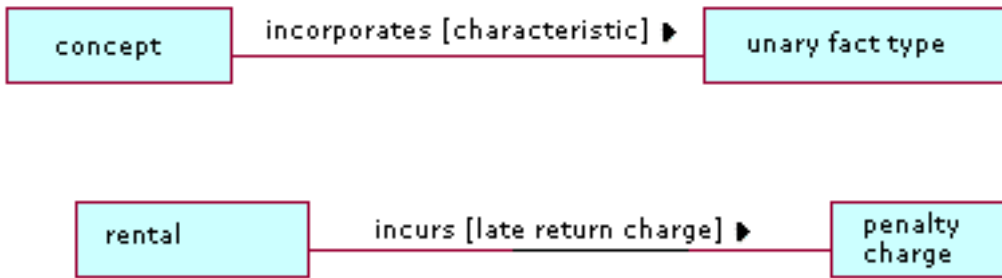


Figure H.12- Two examples of a term for a role in a form of expression

## H.5 Generalizations

Generalizations are shown in the normal UML way as shown in Figure H.13.

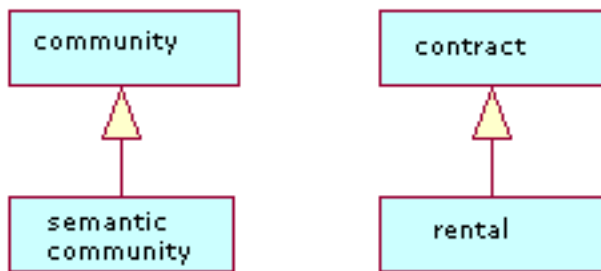


Figure H.13- Two examples of generalization

## H.6 Categorization

### H.6.1 Categories and Categorization Schemes

A set of mutually-exclusive categories can be depicted by bringing the generalization lines together, as shown on the left diagram in Figure H.14. Contrast that with the diagram on the right which reflects two independent specializations -- i.e., a community can be both a semantic community and a speech community.

Optionally, the name of a categorization scheme can be assigned to the set of categories, e.g., 'Rentals by Payment Type'.

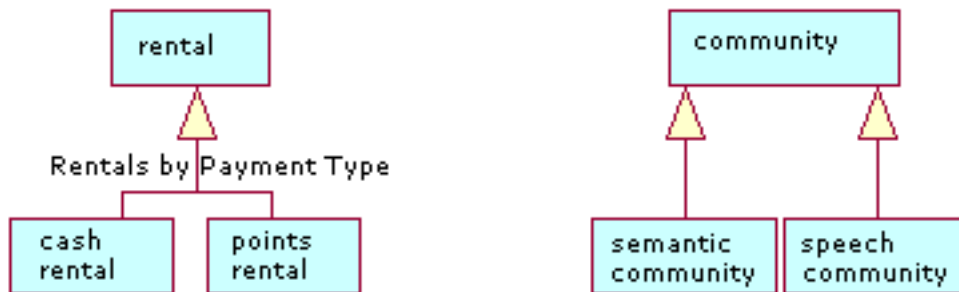


Figure H.14- Depicting mutually-exclusive categories vs. independent specializations

## H.6.2 Categories and Categorization Types (Concept Types)

Use of UML powertype notation is not typical, but it can be used to show the categories specified by a categorization type (concept type). Note that the second diagram in Figure H.15 illustrates a named categorization scheme ('Branches by Type') which is related to the categorization type 'branch type'.

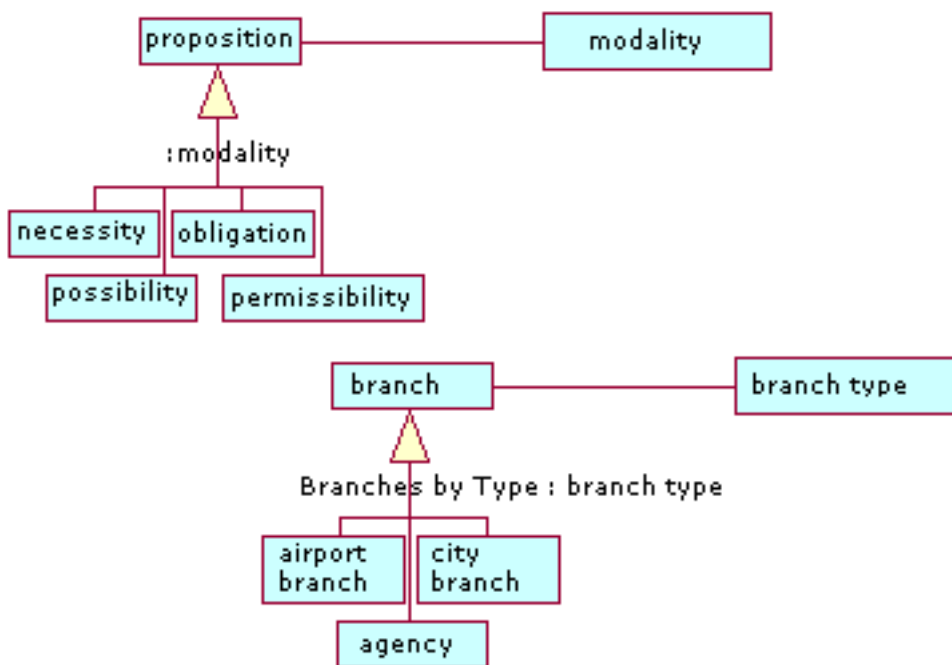


Figure H.15- Two examples of depicting the categories specified by a categorization type

## H.7 Partitive Fact Type

UML aggregation notation is used to represent partitive fact types.

The diagram on the left of Figure H.16 shows the forms of expression for the vocabulary-to-symbol and vocabulary-to-form-of-expression fact types:

symbol is included in vocabulary

form of expression is included in vocabulary

The right diagram of Figure H.16 shows the form of expression for the partitive fact type between vocabulary and vocabulary, as follows:

vocabulary<sub>1</sub> incorporates vocabulary<sub>2</sub>

Note that the subscripts in the form of expression are not reflected on the diagram.

As the diagrams of Figure H-16 illustrate, reflecting the verb phrase of a partitive fact type on the diagram is optional.

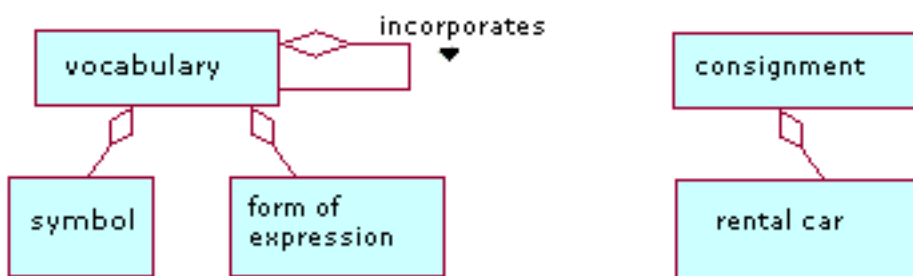


Figure H.16- Two examples of partitive fact type

## H.8 Fact Type ‘Objectification’

Where a noun concept is defined using objectification such that it is coextensive with a fact type, an association class is used to depict the noun concept, as shown in Figure H.17. A dashed line connects the association line for the fact type with the box for the noun concept. A binary fact type is shown in a similar fashion, with the dashed line connecting to the binary association line.

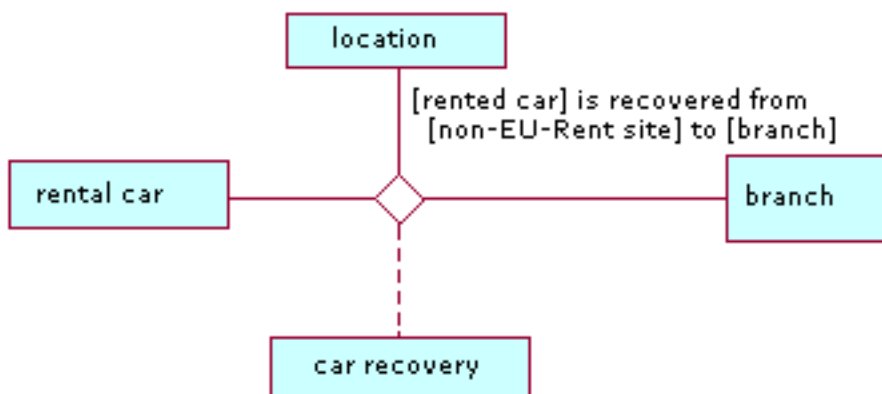


Figure H.17- Depicting fact type ‘objectification’



## H.9 Multiplicities

Multiplicities are typically not shown.

However, display of UML multiplicity is a diagram-level option. When UML multiplicity is used on a diagram (as a whole), this element is used to depict a formally-stated alethic necessity of a particular multiplicity. UML multiplicity is used for no other case. In a diagram that uses UML multiplicity, the default assumption for an unannotated association end is '\*' (which is interpreted as '0 or more' -- i.e., unconstrained).



# Annex I

## (informative)

### The ORM Notation for Verbalizing Facts and Business Rules

Annexes C and F discussed how to verbalize facts and business rules in the SBVR Structured English and RuleSpeak notations. This annex briefly presents a third approach to verbalization that is based on *Object-Role Modeling (ORM)* [Halp1998, Halp2001], a conceptual modeling approach that has been used productively in industry for over 30 years. While this approach has been localized to other languages (including Japanese, German and French), we restrict ourselves here to the English version.

Business rules may be specified in ORM using graphical and/or textual languages. We confine ourselves here to just part of ORM's textual language. We regard a *static business rule* to be a constraint or derivation rule that applies to each individual state of the business, taken one state at a time. This annex focuses on the *verbalization* of static business rules, ignoring dynamic rules relating to state transitions or workflows. In the interests of brevity, only a few of ORM's rule verbalization patterns are illustrated here, mainly using examples from the EU-Rent case study. A detailed discussion may be found in the references [Halp2003a, Halp2003b, Halp2003c, Halp2003d, Halp2004c, Halp2004d, Halp2004e, Halp2004f, Halp2004g, Halp2004b].

#### I.1 Criteria for Business Rule Verbalization in ORM

Static business rules are best applied to a fact model that identifies the fact types of interest to the business. Table I-1 shows some fact types with arities from 1 to 4. Each role corresponds to an object placeholder (depicted here as an ellipsis "...") in the predicate. Here predicates are displayed in *mixfix* notation, allowing object terms to be placed in a sentence at any position. Higher arity predicates (quinary etc.) are also possible.

Table I.1 - Examples of fact types of different arity

Fact Type	Predicate	Arity
Person smokes	... smokes	1 (Unary)
Person was born in Country	... was born in ...	2 (Binary)
Person played Sport for Country	... played ... for ...	3 (Ternary)
Person introduced Person to Person on Date	... introduced ... to ... on ...	4 (Quaternary)

The ORM textual language for verbalizing fact instances, fact types and business rules is based on the following criteria:

- *expressibility* - the language is able to express a wide range of business rules
- *clarity* - the rules are understandable by non-technical domain experts
- *flexibility* - the language directly supports predicates of any arity
- *localizability* - the language constructs are expressible in different native languages
- *formality* - the rules are unambiguous, and should ideally be executable

Apart from its graphical language, ORM uses a textual language that is both formal and conceptual, so that it can serve for communication and validation with domain experts, as well as being executable. Relevant dimensions used in ORM for rule verbalization are listed in Table I.2, along with the choices available. For detailed discussion of these criteria, see the references.

**Table I.2 - Classification schemes for rule verbalization**

Dimension	Choices
Form	Positive Negative Default
Modality	Alethic Deontic
Style	Relational Attribute Mixed
Context	Local Global
Formality	Informal Semiformal Formal

ORM’s verbalization language applies to mixfix predicates of any arity, with predefined patterns to cater for a very wide range of constraints found in business domains. Unlike some other approaches, ORM leaves the verbalization of the underlying fact model unchanged (e.g., no need to pluralize noun phrases and related verb phrases).

Every constraint has an associated *modality*, determined by the logical modal operator that functions explicitly or implicitly as its main operator. In practice, the modality is typically either *alethic* or *deontic* (see Table I.3). Logical negation may be used to obtain the usual equivalences (e.g., not necessary  $\equiv$  possible, not obligatory  $\equiv$  permitted, not permitted  $\equiv$  forbidden).

**Table I.3 - Alethic and deontic modal operators**

Alethic	Deontic
It is necessary that	It is obligatory that
It is possible that	It is permitted that
It is impossible that	It is forbidden that

The next two sections present some simple examples. Far more complex examples may be found in the references.

## I.2 Some Basic Rule Examples in ORM

### Simple uniqueness constraint:

Positive form:

Each rental car *is owned by* at most one branch.

In positive verbalizations, the modality is often assumed (as above), but may be explicitly prepended (“It is obligatory that” for deontic modality; “It is necessary that” for alethic modality).

Negative form, deontic modality:

It is forbidden that the same rental car *is owned by* more than one branch.

Negative form, alethic modality:

It is impossible that the same rental car *is owned by* more than one branch.

### Composite uniqueness constraint:

Positive, deontic form of a uniqueness constraint over two fact roles from the ternary fact type room at hour slot is booked for course.

It is obligatory that  
given any room and hour slot  
that room at that hour slot is booked for at most one course

### Composite Exclusion constraint:

Relational style: No person *directed and reviewed* the same movie.

Attribute style: For each movie:

no director is a reviewer.

### Join Subset constraint:

Each advisor who *serves in* a country  
also *speaks a language that is used by* that country.

### Derivation Rule:

Relational style: Define person<sub>1</sub> *is an uncle of* person<sub>2</sub> as

person<sub>1</sub> *is a brother of* person<sub>3</sub> who *is a parent of* person<sub>2</sub>

Attribute style: For each person: uncle = brother of parent.

## I.3 Some EU-Rent Rule Examples

This section provides restatements in ORM of some EU-Rent rules presented in earlier annexes. The ORM rewording is displayed after the original formulation, assuming that the fact types used in the verbalization are defined in the model.

It is obligatory that a rental have a car group..

It is obligatory that each rental *has a* car group.

It is obligatory that a rental car with a service reading greater than 5000 miles be scheduled for service..

It is obligatory that each rental car that *has* a service reading greater than 5000 miles *is scheduled for service*.

It is obligatory that in an international return the country of the local area of the receiving branch be the country of registration of the rental car..

It is obligatory that  
if a rental car *is in* an international return that *is to* a receiving branch  
that *is in* a local area that *is in* a country  
then that rental car *is registered in* that country.

It is permitted that a renter books multiple rentals..

It is permitted that each renter *books* more than one rental.

## Annex J (informative)

### ORM Examples Related to the Logical Foundations for SBVR

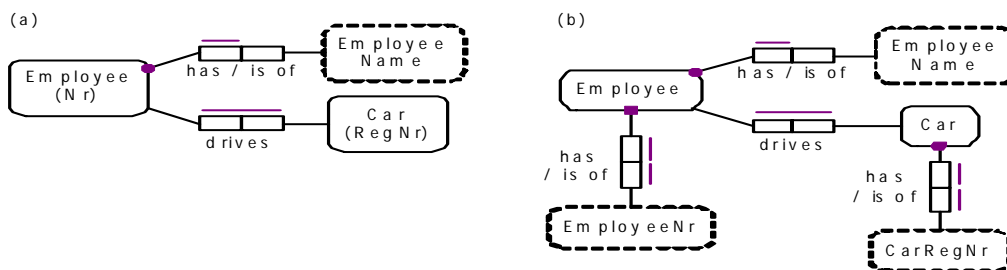
#### J.1 Introduction

This annex provides some detailed examples to illustrate how foundational concepts described in Chapter 10.1.1 can be captured in an existing logic-based approach. The examples use Object-Role Modeling (ORM), which has a well-defined mapping to formal logic [Halp1989]. A basic introduction to ORM may be found in [Halp2000] and a detailed treatment in [Halp2001]. ORM takes a fact-based approach to modeling business scenarios that is compatible with the SBVR approach.

#### J.2 Simple Database Example

Figure J.1 shows an ORM schema for the simple Employee/Car database example discussed in Chapter 10.1.1. In ORM, *objects* are either *entities* (non-lexical objects that are identified by definite descriptions, and that typically change state) or *values* (lexical constants that identify themselves, such as character strings). In ORM 2 (the latest version of ORM, used here), entity types and value types are depicted as named, soft rectangles with solid or dotted lines respectively (previous versions of ORM used ellipses instead of soft rectangles). Logical *predicates* are depicted as named sequences of role boxes, where each *role* is a part played in the relationship. For binary fact types, if forward and inverse predicate readings are displayed on the same side of the role boxes, they are separated by a slash “/”. By default, predicates are read left-to-right and top-to-bottom.

A large dot on a role connector indicates that the attached *role is mandatory* (i.e., for each state of the fact model, each instance in the population of the object type must play that role). The object type’s population in the fact model is not necessarily the same as the real world population in that state, and is typically far smaller than the extension of the object type (which covers all possible states). For example, each employee has an employee name, but it is optional whether an employee drives a car.



**Figure J.1 - ORM schema for the simple Employee/Car database example**

A bar beside a role box depicts a *uniqueness constraint*, indicating that for each state of the fact model each object that instantiates that role does so only once. For example, each employee has at most one employee name. A bar that spans two or more roles depicts a uniqueness constraint over that role combination, indicating that for each state of the fact model each object sequence that instantiates that role sequence does so only once. For example, the fact type Employee drives Car is many:many, and in each state any instance of this fact type appears at most once.

Figure J-1(b) displays simple injective (mandatory, 1:1 into) reference schemes explicitly as binary relationships. Employees are referenced by their employee numbers, and cars by their registration numbers. Figure J-1(a) displays these reference schemes compactly as parenthesized reference modes.

### J.3 Open/Closed World

Consider the populated unary fact type in Figure J.2(a). For simplicity, we omit reference schemes, and assume people may be identified by their first names. We know that Fred smokes. If we use open world semantics, then it is unknown whether Sue or Tom smoke. If the ORM schema is mapped to a UML class, then the open world interpretation leads to an optional isSmoker attribute with only one possible value ('Y' for yes), as shown in Figure J.2(b). If we apply closed world semantics, then the absence of facts that Sue or Tom smoke entails that they don't smoke; this leads to a mandatory, Boolean isSmoker attribute, as shown in Figure J.2(c).

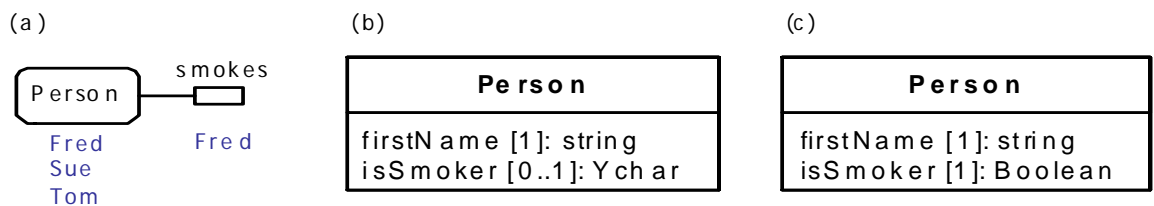


Figure J.2 - An ORM model (a), and UML classes based on (b) open world and (c) closed world semantics.

Currently most ORM tools adopt the closed world assumption for unaries. However for the next generation of ORM tools that are designed to interoperate with SBVR tools, it is anticipated that unaries will be treated as open by default.

For many fact types in a business domain, especially those without functional roles, it is impractical to include all the negative instances as base facts. For example, for the fact type Employee drives Car, there might be many thousands of cars, so one would normally not explicitly include negated facts such as Employee 1 does **not** drive Car 'AAA246'. In some cases however, especially with functional roles or when the population is small, it is practical to include negated facts as base facts.

Figure J.3 shows two ways to model a business domain where for each person in the population of the domain it is known whether that person smokes or not. In each case, negated facts are explicitly treated as base facts, and the predicates are given open world semantics. Semi-closure is implied because of the constraints. In Figure J.3(a) the xor constraint (circled mandatory dot overlaid by 'X' for exclusion) declares that each person referenced in the fact model population plays exactly one of the two roles (smoking or not smoking). In Figure J.3(b) the mandatory, uniqueness and value constraints collectively ensure the same thing. When either of the ORM schemas is mapped to a UML class, a mandatory Boolean isSmoker attribute results, as shown in Figure J.3(c).

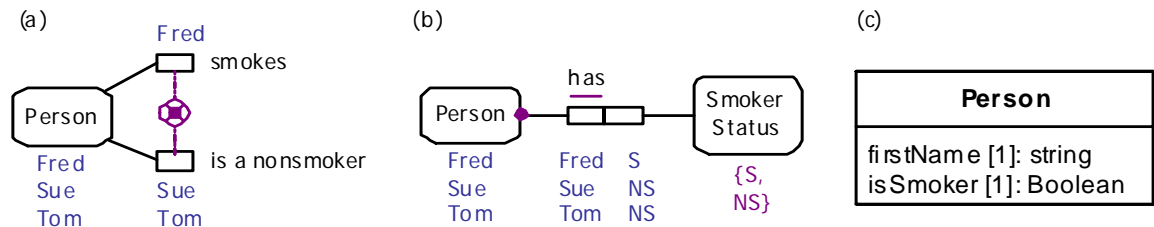


Figure J.3 - Open world semantics plus negated facts and constraints that ensure semi-closure.



Now consider a business domain where we know that Fred smokes, and that Sue doesn't smoke, but are unsure whether Tom smokes. To model this at all, we need open world semantics. Figure J.4 shows three ways to model this in ORM, as well as the equivalent UML class. Figure J.4(a) uses an exclusion constraint, Figure J.4(b) uses an optional binary, and Figure J.4(c) uses a mandatory binary and a special value (here shown as "?") to indicate that the smoking status is unknown. We treat this special value like any other value, using 2-valued logic, rather than adopt a generic null based on 3-valued logic (as in SQL). The equivalent UML class notation is shown in Figure J.4(d).

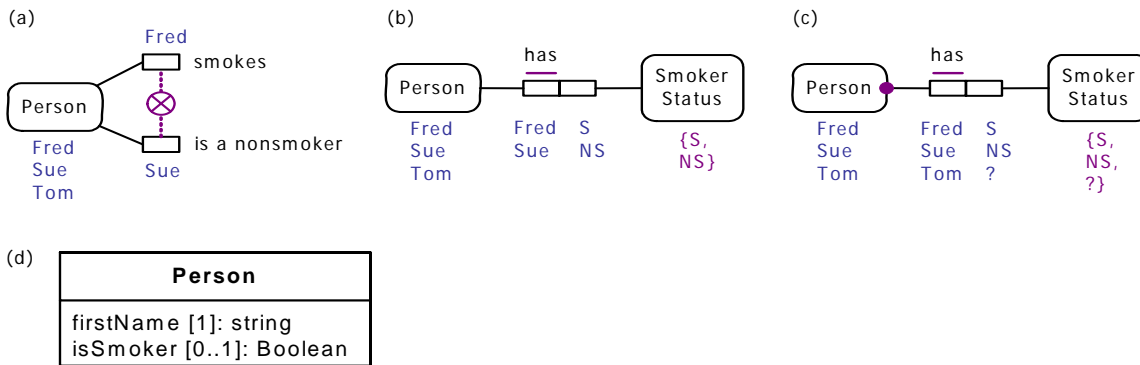


Figure J.4 - We may indicate whether a person smokes or not, or that this is unknown.

## J.4 Deontic Constraints

In the ORM schema shown in Figure J.5, the fact type Person is a husband of Person is declared to be many to many, as shown by the alethic, spanning uniqueness constraint over the top of the predicate. In addition a deontic uniqueness constraint has been added (depicted by a bar starting with an "o" for "obligatory") to each role to indicate that the fact type *ought* to be 1:1. The leftmost deontic constraint verbalizes as: **It is obligatory that each Person is a husband of at most one Person**. The other deontic constraint (each wife should have at most one husband) may be handled in a similar way.

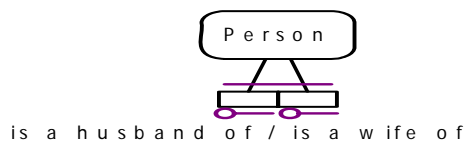
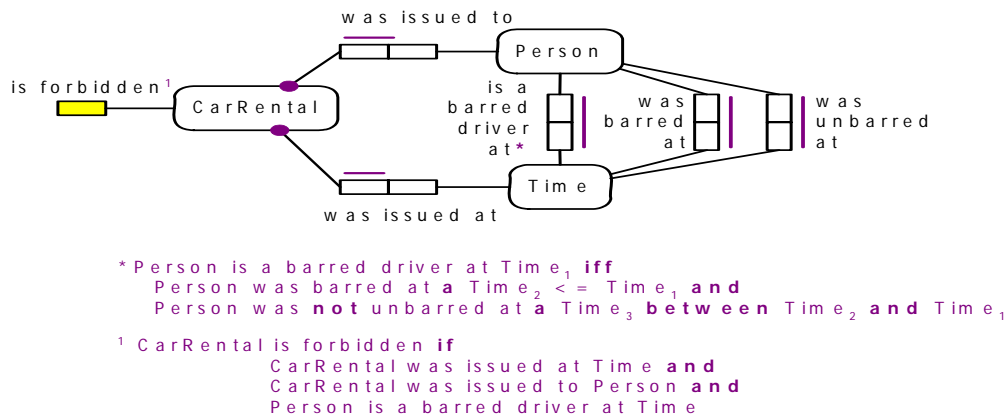


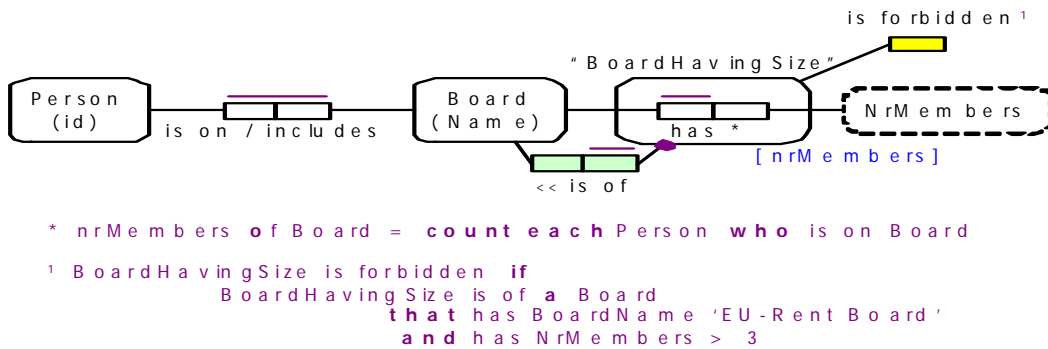
Figure J.5 - Deontic constraints obligate the marriage relationship to be 1:1.

The deontic constraint "Car rentals ought not be issued to people who are barred drivers at the time the rental was issued." may be captured by the textual constraint on the domain fact type CarRental is forbidden, as shown in the ORM schema in Figure J.6. The fact type Person is a barred driver at Time is derived from other base fact types (Person was barred at Time, Person was unbarred at Time) using the derivation rule shown.



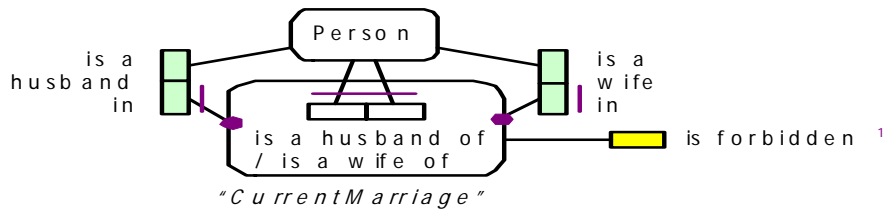
**Figure J.6 - Specifying a deontic constraint forbidding rentals to barred drivers using a domain level predicate.**

The deontic constraint “It is forbidden that more than three people are on the EU-Rent Board.” is captured by the textual constraint on the derived fact type BoardHavingSize is forbidden in the ORM schema shown in Figure J.7. The derivation rule is stated in attribute style, but its underlying relational style is used in invoking the derivation rule within the body of the deontic constraint.



**Figure J.7 - Specifying a deontic constraint on the size of the EU-Rent board using a domain level predicate.**

The deontic constraints that require each person to have at most one spouse may be formulated as textual constraints on the fact type CurrentMarriage is forbidden, as shown in Figure J.8.

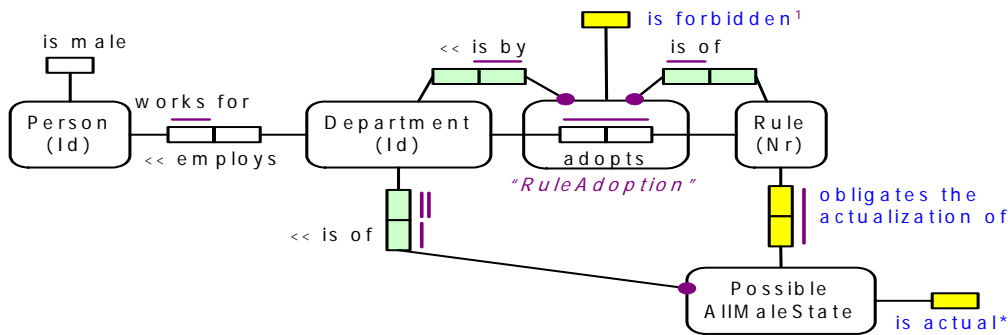


<sup>1</sup> CurrentMarriage is forbidden if  
 a Person<sub>1</sub> who is a husband in CurrentMarriage  
 is a husband of more than one Person<sub>2</sub>.  
 CurrentMarriage is forbidden if  
 a Person<sub>1</sub> who is a wife in CurrentMarriage  
 is a wife of more than one Person<sub>2</sub>.

**Figure J.8 - An alternative way to capture the deontic constraints in Figure 5.**

The ORM schema in Figure J.9 relates to the following deontic constraint: "It is not permitted that some department adopts a rule that says it is obligatory that each employee of that department is male". This example includes the mention (rather than use) of an open proposition in the scope of an embedded deontic operator. The schema uses the special predicates "obligates the actualization of" and "is actual", as well as an object type "PossibleAllMaleState" which includes all conceivable all-male-states of departments, whether actual or not.

The formalization of the deontic constraint works, because the relevant instance of PossibleAllMaleState exists, regardless of whether or not the relevant depart actually is all male. The "obligates the actualization of" and "is actual" predicates embed a lot of semantics, which is left implicit. While the connection between these predicates is left informal, the derivation rule for PossibleAllMaleState is actual provides enough semantics to enable human readers to understand the intent.



\* PossibleAllMaleState is actual iff  
PossibleAllMaleState is of a Department and  
each Person who works for that Department is male

<sup>1</sup> RuleAdoption is forbidden if  
RuleAdoption is by a Department  
and is of a Rule  
that obligates the actualization of a PossibleAllMaleState  
that is of the same Department

\*  $\forall x:\text{PossibleAllMaleState}$   
[x is actual  $\equiv \exists y:\text{Department} (x \text{ is of } y \ \& \ \forall z:\text{Person} (z \text{ works for } y \supset z \text{ is male}))]$

<sup>1</sup>  $\forall x:\text{RuleAdoption}$   
[ $\exists y:\text{Department} \ \exists z:\text{Rule} \ \exists w:\text{PossibleAllMaleState}$   
(x is by y & x is of z & z obligates the actualization of w & w is of y)  
 $\supset$  x is forbidden]

aliter:

<sup>1</sup>  $\forall x:\text{RuleAdoption} \ \forall y:\text{Department} \ \forall z:\text{Rule} \ \forall w:\text{PossibleAllMaleState}$   
[(x is by y & x is of z & z obligates the actualization of w & w is of y)  $\supset$  x is forbidden]

Figure J.9 - A complex case involving embedded mention of propositions.

## **Annex K**

### **(informative)**

## **Design Rationale Details for the Use of MOF and XMI**

The following sections explain the goals that led to the Vocabulary-to-MOF/XMI Mapping Rule Set, questions (and answers) about SBVR use of MOF/XMI, the steps to using MOF and XMI, and recommended usage of interchange capabilities.

### **K.1 Goals of Mapping Vocabularies to MOF and XMI**

The rules for mapping a vocabulary to MOF and XMI aim at accomplishing several goals listed below. Given a vocabulary defined in terms of SBVR:

1. The rules guide the creation of a MOF model (based on MOF 2) of repositories that can hold a representation of any fact that can be meant by an atomic formulation using the concepts represented in the vocabulary.
2. The rules guide the creation of an XML schema (based on OMG's XMI for MOF 2) for XML documents that can convey any facts that can be meant by an atomic formulation using the concepts represented in the vocabulary.
3. A vocabulary namespace is the only necessary input to a complete rule-guided specification of the MOF model and the XMI-based XML schema.
4. Rules that guide what names are given to UML and XML elements can be overridden with exceptions for special cases as might be needed or desired.
5. Separate instances of following the rules for the same vocabulary result in effectively identical MOF models and XML schemas.
6. Writers and readers of XML documents do not need to have used the exact same XML schema. If the vocabulary elements used by the writer are in the vocabulary of the reader, then the XML document should be understandable by the reader.
7. XML documents should remain stable and readable with respect to updates to an XML schema that result from the normal course of a vocabulary growing. That is, new versions of a vocabulary and new versions of rules related to that vocabulary should not invalidate an XML document as long as representations relevant to the XML document are not removed from the vocabulary (they can be deprecated, but not removed).
8. Readers and writers of an XML document can use different reference schemes for the same concepts.
9. The rules preserve the signifiers of the vocabulary with minimum change in deriving names for the MOF model and XML schema in order to make the model, schema and XML documents readable and understandable to people familiar with the vocabulary.
10. To the degree possible, the rules do not lead to using MOF capabilities beyond those of EMOF in order to facilitate working with the widest possible range of MOF implementations.
11. The rules are suitable for any metalevel and are therefore suitable for vocabularies about vocabularies and for vocabularies concerning business rules.
12. The rules are suitable for vocabularies concerning any business domain.

13. An XML schema for a vocabulary can be used as an imported or included schema in another XML schema for another vocabulary that includes the first vocabulary.
14. A tool that reads documents based on an old XML schema can also read new documents that are based on a newer XML schema and can understand every fact that is expressed according to the old XML schema — everything that is based on the vocabulary of the old XML schema — as long as representations of the old vocabulary continue to be in the new vocabulary.
15. The rules for mapping representations of a vocabulary to UML and XMI are reversible such that the source representations within the vocabulary can be identified from the UML model and XMI-based XML schema.
16. The rules for forming UML and XMI names from symbols are isomorphic.

The following are NOT goals:

1. Generation of an implementation of business rules based on MOF and OCL.
2. Generation of UML models of vocabulary and/or rules meant for presenting vocabulary and rules in UML notation. This is not a mapping to a business object model.
3. Reduction of synonyms and synonymous forms or symbols for the same concepts to a single normal form. A semantic community always has the option of choosing a single language vocabulary with synonyms removed if it so desires.
4. Automatic translation of content of MOF repositories or XML documents across languages (crossing speech communities within a single semantic community). A model-driven tool could potentially do such a thing, but this is not a goal for these mapping rules.
5. Matching MOF-based identity to reference schemes defined for terms in a vocabulary. It would require more of MOF than is reasonably possible or practical.

## **K.2 Questions and Answers about SBVR use of MOF/XMI**

This section answers questions asked about SBVR's use of MOF/XMI.

1. Why has EMOF been chosen over full MOF?
2. Why doesn't the SBVR Metamodel Inherit from the UML Infrastructure?
3. What is a 'Fact-Oriented' Approach?
4. Why is SBVR Vocabulary Interchange based on a Fact-Oriented Approach?
5. What is Essential SBVR?

### **K.2.1 Why has EMOF been chosen over MOF?**

1. EMOF is more widely supported and more easily supported than full MOF.
2. The open source development tool 'Eclipse' supports EMOF and not MOF.
3. MOF associations do not support fact types with more than two roles in a simple consistent way and therefore are not useful for SBVR vocabulary interchange purposes.
4. XMI-based XML generated from EMOF is simpler and more straightforward than XMI-based XML generated from full MOF.

## K.2.2 Why doesn't the SBVR Metamodel Inherit from the UML Infrastructure?

1. There are many precedents of OMG standards that do not inherit from the UML infrastructure (e.g., CWM, CORBA, EAI, and EDOC).
2. The primary SBVR use of MOF is interchange of semantics of business vocabularies and rules between tools. It is not aimed at being an internal model of any tool.
3. It is required that the model support partial representations — that any amount of information be representable in a MOF repository or XML document, even if only a single fact. When you inherit from the UML Infrastructure you include many extra kinds of facts beyond what is needed for communicating business semantics.

## K.2.3 What is a 'Fact-Oriented' Approach?

The fact-oriented approach used to generate the proposed metamodel puts the granularity of objects at the level of individual facts. Each fact is represented by its own object, and not by an attribute of some other object.

The Fact-Oriented Approach is exemplified below for a small part of the proposed vocabulary that includes the following designations:

concept	role (specialization of concept)
projection	closed projection (specialization of projection)

A closed projection is a projection of a specific type. The example vocabulary also includes verbs for expressing facts in these forms:

closed projection *defines* concept  
concept *specializes* concept

Figure K.1 shows a metamodel based on the fact-oriented approach. As explained above, the fact-oriented approach makes a separate class for each type of fact that can be expressed. The different types of facts are divided into two kinds, as follows:

1. Facts about instantiations of concepts (the classes shown on the left). Each of these classes has one attribute for referring to what is being classified.
2. Facts based on fact types (the classes shown on the right). These classes typically represent facts about relationships between things, such as the relationship between a concept and one that it specializes. Each of these classes has one or more attributes: one for each role of the fact type.

## Classes For Instantiation Facts

<b>is concept</b>
concept : Referent

<b>is role</b>
object type : Referent

<b>is projection</b>
projection : Referent

<b>is closed projection</b>
closed projection : Referent

## Classes For Fact Types

<b>concept specializes concept</b>
concept1 : Referent concept2 : Referent

<b>closed projection defines concept</b>
closed projection : Referent concept : Referent

**Figure K.1 - Fragment of a fact-oriented metamodel**

The fact-oriented approach depends on a general class called Referent (not shown above). An object of that class is a reference point for stating facts. A subclass of Referent is Thing, which is used to represent an individual thing that is a subject or object of a fact. For example, suppose we want to represent the fact that a closed projection defines a particular role and the fact that the role specializes another concept. The following objects are used to represent these facts:

1. an object of the 'closed projection defines concept' class with its 'closed projection' attribute set to a first object of the Thing class and its 'concept' attribute set to a second object of the Thing class.
2. an object of the 'is role' class with its 'role' attribute set to the first object of the Thing class.
3. an object of the 'concept specializes concept' class with its 'concept1' attribute set to the first object of the Thing class and its 'concept2' attribute set to a third object of the Thing class.

### **K.2.4 Why is SBVR Vocabulary Interchange based on a Fact-Oriented Approach?**

There are many reasons for choosing the fact-oriented approach for the particular purpose of standardizing business-level data interchange. The most important reasons are these:

1. Fine control over exactly what is communicated
2. Facts about facts
3. Multidimensional categorization
4. Things changing over time



5. Communication for many purposes that cannot be predicted
6. Semantic stability of interchange documents
7. Extensibility and reuse in other vocabularies

#### **K.2.4.1 Fine control over exactly what is communicated**

A primary reason that a fact-oriented approach is best for a model intended for data interchange at the business level is that people communicate facts. The fact is the atomic unit of communication. Controlling what specific facts are expressed in a communication is important with regard to privacy and intellectual property. Different messages must include different facts about the same subject depending on the purpose of the message and who is receiving it.

With a fact-oriented approach, communicated content is controlled at the level of individual facts. With more common object-oriented approaches facts are bundled together. If a provider of information does not have all of the facts for a whole object, default values commonly appear in the content. Miscommunication results because a receiver cannot tell that a default value was not explicitly given by the sender.

The rules of XMI do not require all details of an object to be shown in an XML document, but the standard MOF programming interface provides only for writing whole objects to XML. Note that the standard MOF programming interface is the Java Metadata Interface (JSR 40) of the Java Community Process (see [www.jcp.org](http://www.jcp.org)). Whether communicating using XMI or using objects directly, the fact-oriented approach gives fine control over what is communicated.

#### **K.2.4.2 Facts about facts**

The fact-oriented approach allows for facts themselves, any and all of them, to be the subjects of other facts. A fact is itself a kind of thing, and can therefore be the subject of other facts.

Why would anyone need facts about facts? At the level of business rules that guide business activities, facts are things of great interest, as the following questions illustrate.

1. Which facts are confidential and which are not?
2. What facts must be disclosed when negotiating a contract?
3. When and how were certain facts disclosed?
4. Which facts are expressed in what business documents and in what formal communications?
5. Which facts about customers are private as a general rule?
6. Which specific private facts has a customer authorized to be shared with whom?
7. When did a fact become true?
8. What is the source of knowing a specific fact?
9. What version of a document states a certain fact?

Business users often need to communicate facts about facts. The fact-oriented approach lets them do this without going under the covers and without going outside of standard XMI-based interchange capabilities.

### **K.2.4.3 Multidimensional categorization**

Business vocabularies, including the proposed vocabularies for defining business vocabularies and rules, include designations for various categories of things. Categorization occurs in many dimensions. For example, projections are categorized by whether they are closed and are separately categorized set projections or bag projections.

The fact-oriented approach encounters no difficulty with things being categorized in many different ways because each instantiation of a concept is a separate fact. For example, to represent that a particular projection is a set projection, there is simply an object of type 'is set projection'.

### **K.2.4.4 Things changing over time**

In a business, things change with time. Suppose a rental location becomes a regional headquarters. With the fact-oriented approach there is no disruption. A new object of type 'is regional headquarters' is created to represent the new fact.

Approaches not based on facts fail because MOF does not support an object of one class being transformed into an object of another class. In such approaches a 'Rental Location' object cannot be transformed into a 'Regional Headquarters' object, so the 'Rental Location' object must be deleted and replaced by a 'Regional Headquarters' object. Due care must be given not only to replicating all attributes, but also to the likely impossible task of finding all references to the 'Rental Location' object in order to replace each one with a reference to the new 'Regional Headquarters' object. A simple change becomes a big change, and change management becomes much more complicated.

As with multidimensional categorization, various other design choices are used to cope with things changing over time. But again, design choices are driven by factors that often change as a model grows or is put to different uses. Individual requirements to support change are often discovered late when design changes cause substantial disruption.

### **K.2.4.5 Communication for many purposes that cannot be predicted**

Object-oriented models tend to be designed for a particular purpose. The purpose drives the design such that changes in purpose often lead to substantial redesign.

For example, when a group of experts in a field such as UML modeling work together to define an object-oriented metamodel such as the UML metamodel, much design and redesign results from the various participants' intended uses of the metamodel. Consider the metamodel design changes to existing elements occurring in only a minor revision of UML from version 1.3 to 1.4. A UML 1.3 based XML encoding of a simple UML model of one class with one attribute is invalid with respect to the UML 1.4 XML format. The diagram of the class and attribute did not change, and the English words that describe the class and attribute did not change, but the metamodel changed. New requirements cause redesign in each revision of a metamodel. Many man-years of redesign went into the core parts of UML 2 that concern classes and attributes.

Metamodel changes introduced by redesign cause substantial work for tool vendors. Vendors often choose to skip compliance for some versions because of time and cost. This noncompliance disables the intended interchange capabilities between tools from different vendors.

The fact-oriented approach addresses the problem of constant design churn that results from changes in purpose. The fact-oriented approach is vocabulary-driven. The words used to state a fact almost always remain valid from one version of a vocabulary to the next, so the generated metamodel and resulting XML format with respect to those words continue to be the same. New purposes often lead to adding words to a vocabulary for saying things that could not previously be said. But additions to a vocabulary result only in additions to a generated metamodel, not changes. The fact-oriented approach leads to a metamodel for business vocabulary and rules well suited to multiple and unpredicted uses.

#### **K.2.4.6 Semantic stability of interchange documents**

The fact-oriented approach provides semantic stability in the face of changing business rules and expanding business vocabularies so that document contents and queries are not impacted. Semantic loss is minimized over time.

XMI documents based on the fact-oriented approach remain stable and readable as vocabularies and the resulting XML schemas grow. Adding to a vocabulary causes no problem in reading XML documents created before the vocabulary was expanded. Adding to a vocabulary does not invalidate existing documents. Of course, removing elements from a vocabulary does invalidate existing documents, so it is important to keep designations in a vocabulary, even if they become deprecated.

#### **K.2.4.7 Extensibility and reuse in other vocabularies**

The SBVR vocabulary must support not only defining business vocabularies and rules, but it must also be able to be adopted into other vocabularies and to be extended to meet the variety of needs that arise in managing and using vocabularies and rules. The proposed metamodel must be able to be incorporated into other models and extended in step with inclusion and extension of the proposed vocabulary.

An object-oriented metamodel is extended by defining additional classes in a separate package. Those classes can specialize existing classes in the metamodel, but neither attributes nor superclasses can be added to existing classes.

Expansion of business vocabularies introduces new types of facts about things for which concepts are already defined. It also introduces new concepts without any regard to their being more specific or more general than what is already defined. The fact-oriented approach imposes no restrictions on extensibility because, unlike the other approaches, adding something new does not change anything already in a model.

The fact-oriented approach to creating an EMOF model from a vocabulary works well for vocabularies in any subject area. This approach is especially important where a subject area vocabulary incorporates and expands any of the SBVR Vocabularies. The reason is that the EMOF model created from the subject area vocabulary includes the metamodel elements of the proposed SBVR vocabulary, so any XML document that conforms to the proposed SBVR XML format is implicitly conformant to the resulting subject area model XML format. This extensibility is likely to lead to further use of the simple pattern for using EMOF, and therefore to much more widespread use of EMOF and XMI outside of their current niche in defining OMG metamodels.

It has been suggested that even though a fact-oriented approach is important for models representing business domain vocabularies, it would be reasonable to have a traditional OMG metamodel for business vocabulary and rules that does not follow the fact-oriented approach. This suggestion neglects the important requirement that the metamodel must be able to be included in other models created from business vocabularies and to be expanded by other models representing various business domains. For such inclusion and extensibility to operate smoothly, it is essential that the metamodel follows the same modeling approach as the domain models, which is a fact-oriented approach.

#### **K.2.4.8 Some Questions and Answers about the Fact-Oriented Approach**

Since the fact-oriented approach is new in OMG, questions about the approach have naturally arisen. Answers to some of the most important questions are detailed below.

##### **Can objects defined by fact-oriented models be interrelated with objects defined by object-oriented models?**

Yes they can, and in both directions. A fact-oriented model created from a vocabulary can support facts about things in other models as long as there are designations in the vocabulary for referring to those things. Conversely, any object-oriented model can use the general classes ‘fact’ and ‘thing’ as types of properties in order to refer to objects defined by a fact-oriented model. The ability to interrelate models and to represent facts about things in many kinds of models is strategically important to many vendors, including Unisys.

### **What about type checking and class hierarchies?**

Since all attributes in a fact-oriented model are of type ‘Referent,’ some nonsense facts could be represented. The fact-oriented approach does not use MOF to check for semantic consistency, but only for consistency in representation of facts in EMOF. Other typical MOF models use MOF capabilities for a small part of semantic consistency checking concerning types of things in the cases where types are represented by classes. But fallacies can be expressed using either approach.

Consistency checking of facts with respect to definitions and rules is a large task and is outside the scope of this specification. Differences in consistency checking capabilities are to be expected from different software tools that implement the SBVR. Semantic analysis is best kept separate from using MOF for representation for the following reasons:

- A much fuller semantic analysis than MOF provides is generally required.
- Having consistency errors found in two different ways adds unnecessary complexity.
- Separation of semantic analysis from representation is part of what gives the fact-oriented approach its semantic stability. Class hierarchies produced by other approaches are too prone to design changes.

The BRT proposal uses MOF to represent all definitions and rules that are the basis of semantic analysis, including type checking. But the metamodel does not use MOF for semantic consistency enforcement.

### **Do the MOF and XMI mapping specifications for creating programming interfaces and XML schemas change when using the fact-oriented approach?**

No. MOF and XMI are used without change. Also, the more complicated features of MOF and XMI are not used, so the proposed metamodel will be more easily implemented in software tools.

Software engineers who build tools involving business vocabularies and business rules will accrue significant advantage from the difference in approach. The model they use for data interchange will look less like the internal model of the tool they are building and more like the expressions of facts that can be communicated. Future extensions and changes will require less work for both the engineers and their customers.

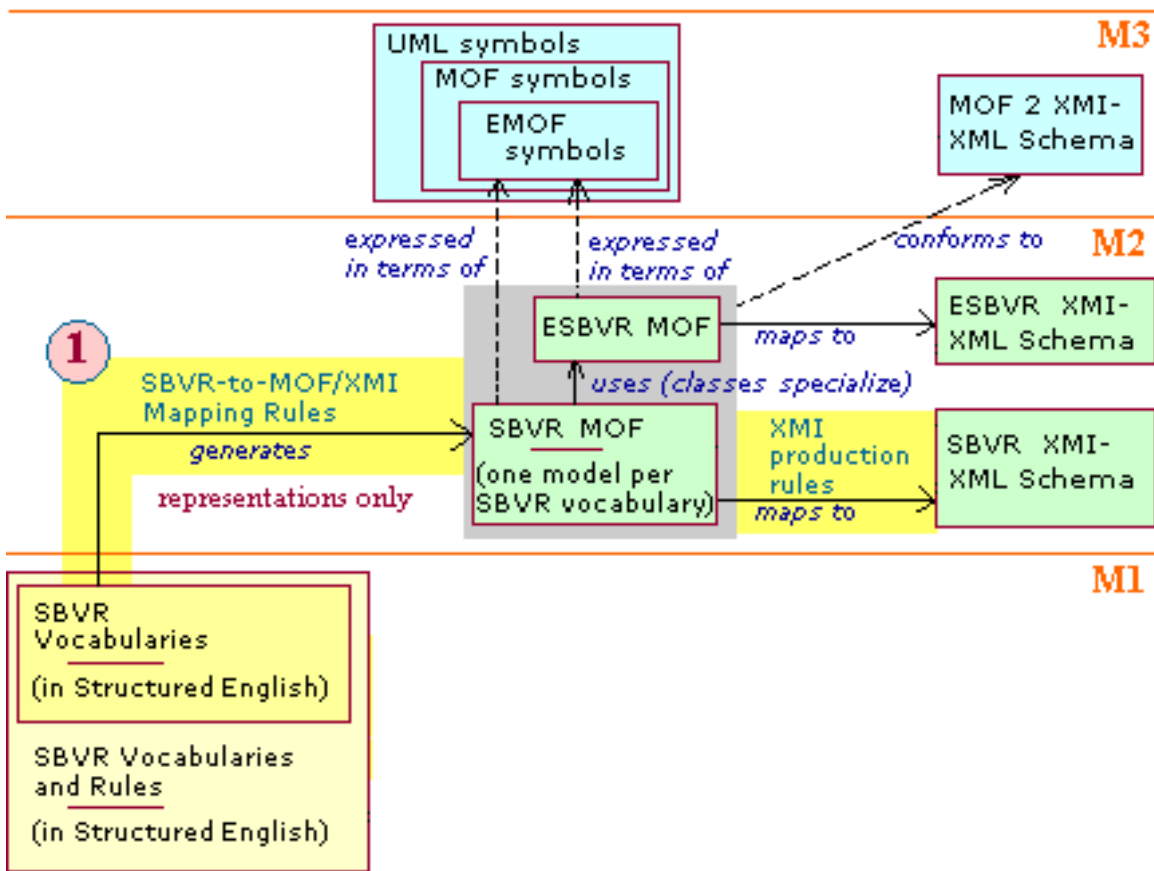
### **K.2.5 What is Essential SBVR?**

The [Essential SBVR Package](#) is a UML/MOF package that is a foundation for all models created following the [Vocabulary-to-MOF/XMI Rule Set](#). Classes within these models inherit from classes in the [Essential SBVR Package](#) in order to consistently implement the fact-oriented approach.

## K.3 Steps to Using MOF and XMI

### K.3.1 Generate SBVR MOF models from the SBVR Vocabularies in SBVR Structured English and, from that, a corresponding SBVR XMI-XML Schema

- Input:** SBVR Vocabularies in SBVR Structured English. Only the vocabulary elements (representations) are used. SBVR rules and definitions are not considered in this step.
- Uses:** ESBVR MOF; SBVR-to-MOF/XMI Vocabulary and Mapping Rule Set; XMI production rules
- Produces:** SBVR MOF models (as XML that conforms to the MOF 2 XMI-XML Schema), ESBVR XMI-XML Schema, and SBVR XMI-XML Schema

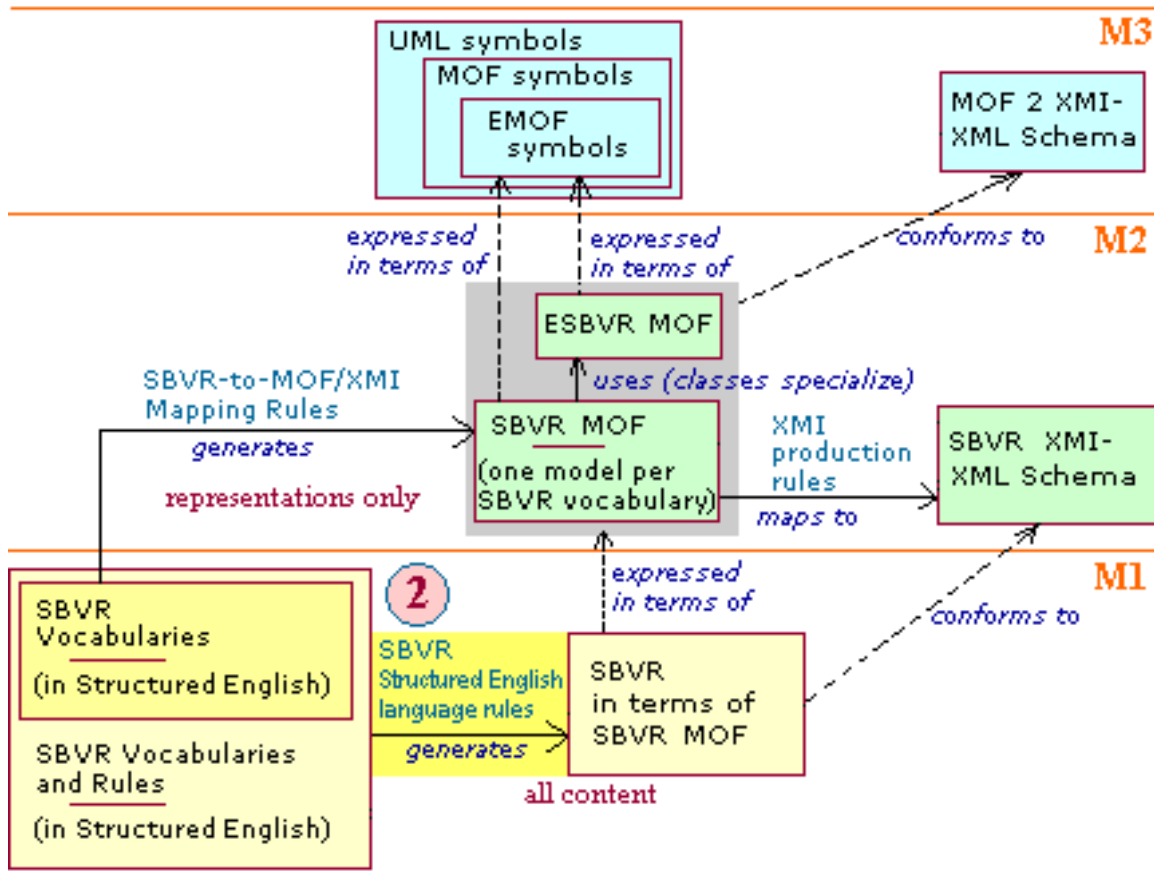


### K.3.2 Represent the whole SBVR Business Vocabularies and Rules in terms of the SBVR MOF model

Input: SBVR Vocabularies and Rules in SBVR Structured English. The entire formal content of the SBVR definitions and rules (all content) are taken.

Uses: SBVR Structured English language rules (Annex C)

Produces: SBVR in terms of SBVR MOF (as XML that conforms to the SBVR XMI-XML Schema)

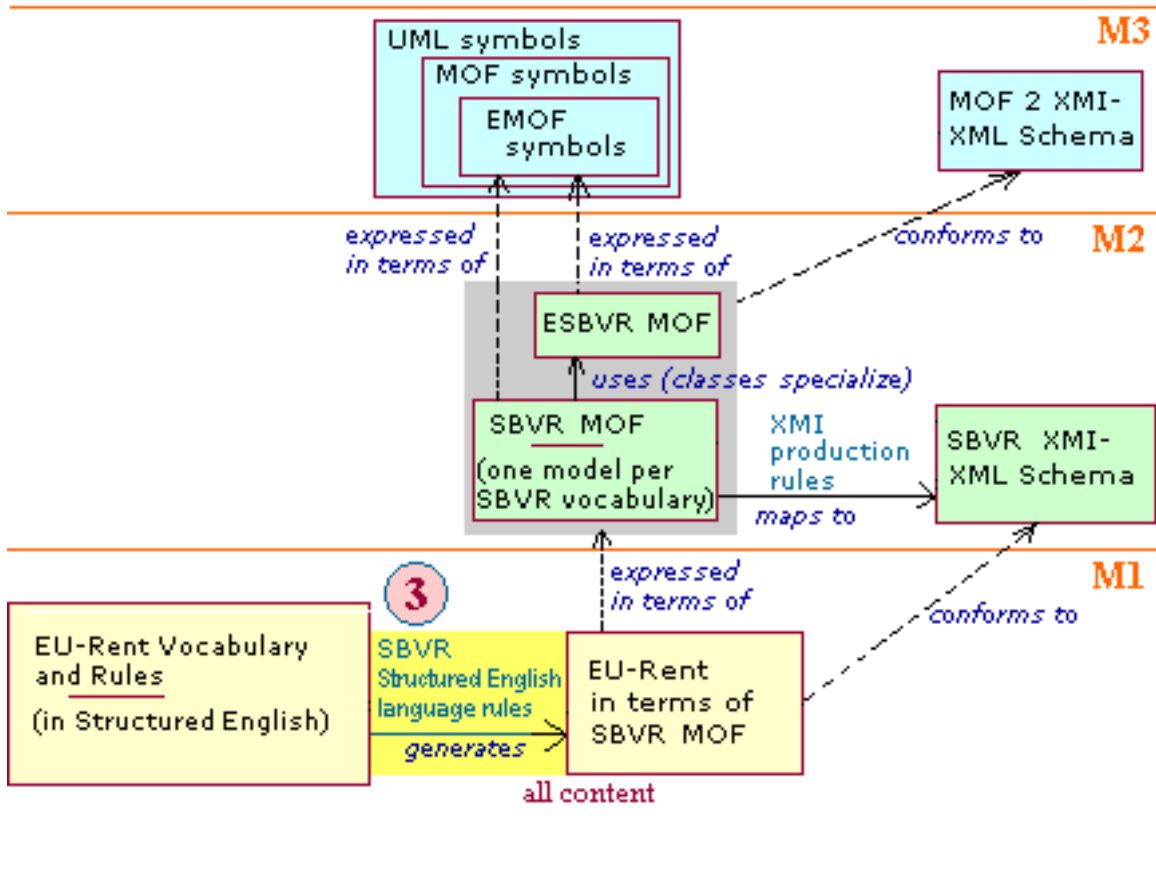


### K.3.3 Represent another Business Vocabulary and Rules in terms of the SBVR MOF model

Input: EU-Rent English Business Vocabulary and Rules in SBVR Structured English. The entire formal content of the EU-Rent definitions and rules (all content) are taken.

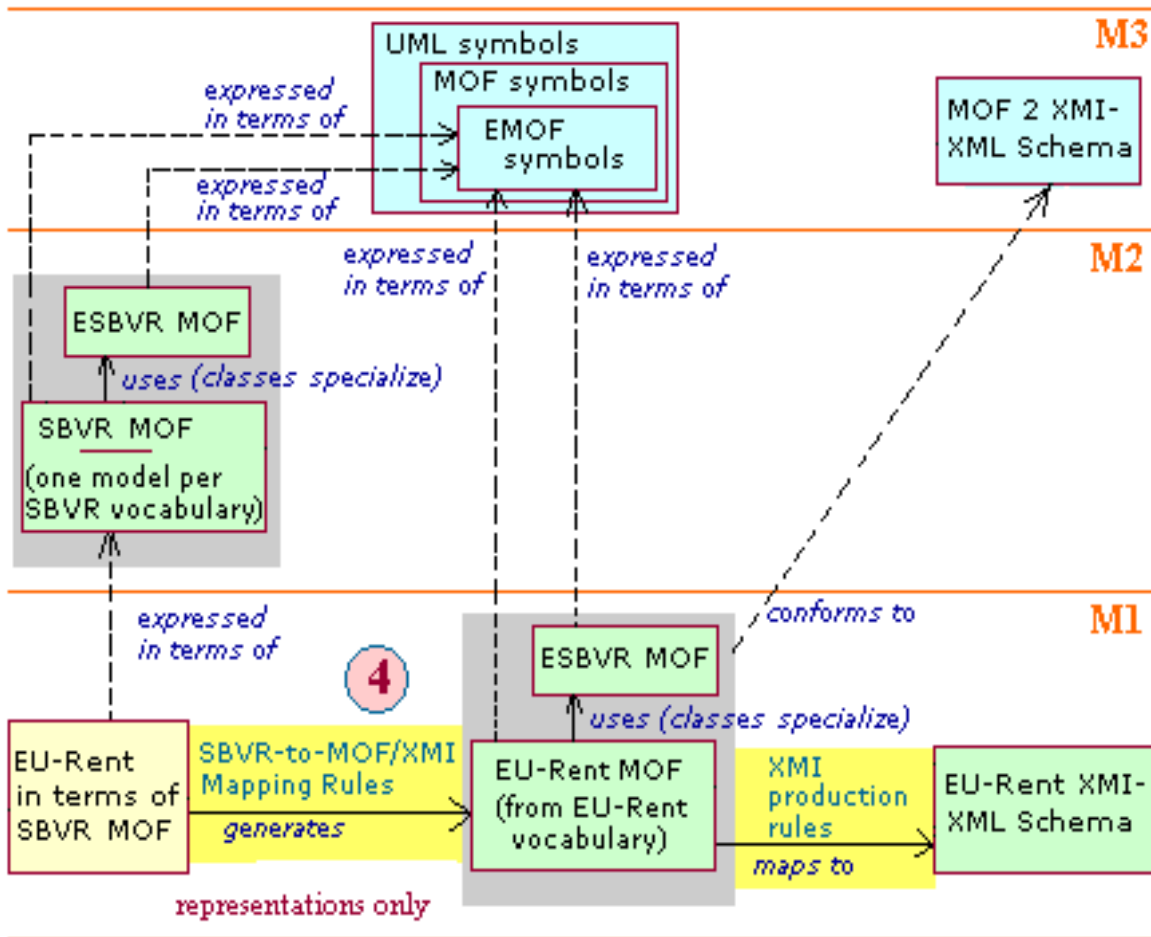
Uses: SBVR Structured English language rules (Annex C)

Produces: EU-Rent in terms of SBVR MOF (as XML that conforms to the SBVR XMI-XML Schema)



### K.3.4 Generate the business' MOF model and XML Schema from its business vocabulary

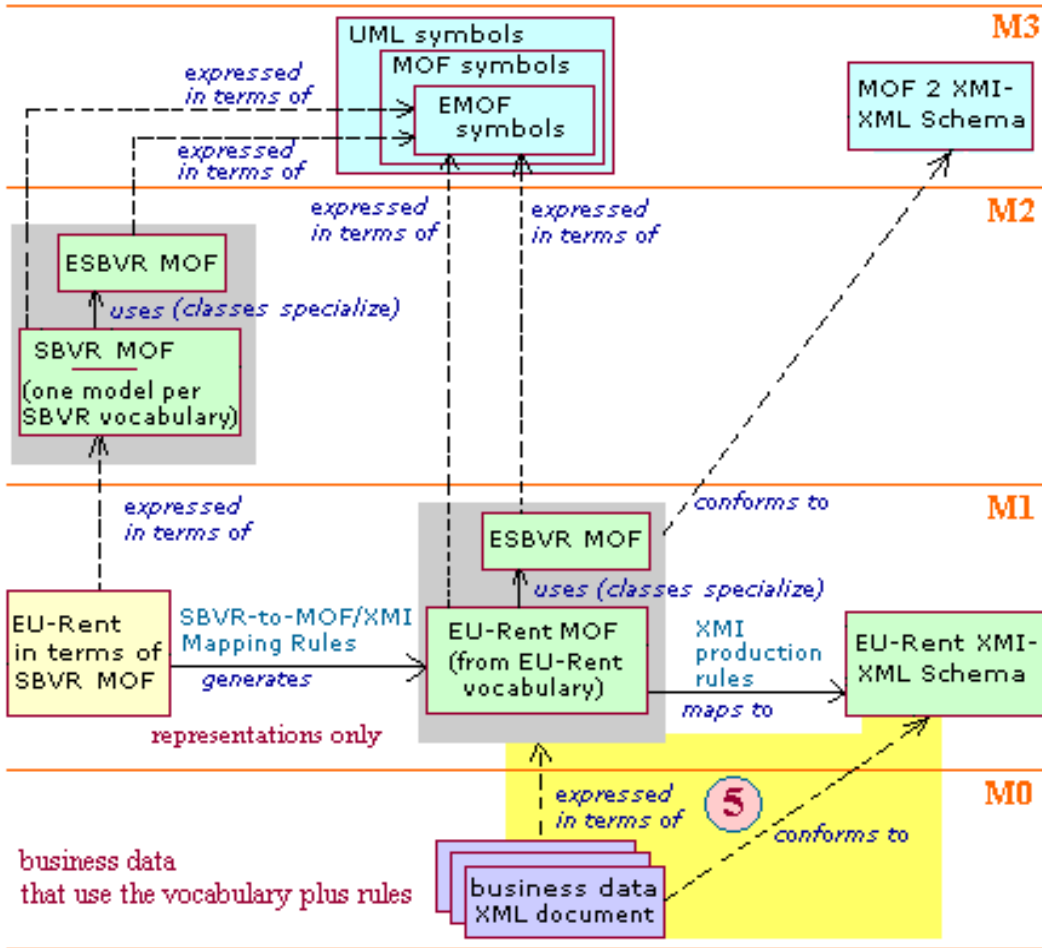
- Input:** EU-Rent in terms of SBVR MOF. Only vocabulary elements (representations) are used. EU-Rent rules and definitions are not considered in this step.
- Uses:** ESBVR MOF; SBVR-to-MOF/XMI vocabularies and mapping rules; XMI production rules
- Produces:** EU-Rent MOF model (as XML that conforms to the MOF 2 XMI-XML Schema) and EU-Rent XMI-XML Schema





### K.3.5 Represent business facts in terms of the business' XMI-XML Schema

Business data in XML files can be verified to conform to the EU-Rent XMI-XML Schema. The data are in terms of the EU-Rent MOF model and are therefore understood with respect to EU-Rent concepts.



### K.4 Recommended Usage of Interchange Capabilities

When communicating via XML, it is important to choose an appropriate XML schema that will accommodate all facts to be communicated. Consider the facts used to demonstrate XML content:

- A given car has the vehicle identification number 'ABC123'.
- A given branch has the name 'London Downtown'.
- The car is owned by the branch.
- The branch is a regional headquarters.

All of the facts above can be represented as atomic formulations. They don't involve modalities, quantifiers or other things requiring use of an SBVR vocabulary. They are just facts. In this case, an XML schema based on EU-Rent is appropriate.

On the other hand, consider the following rule:

- It is obligatory that each open rental *has* an assigned car.

The rule statement uses EU-Rent vocabulary, but describing the semantics of the rule as a set of facts requires the SBVR vocabulary in order to state the rule's modality, quantifications and so on.

In summary, use an XML schema derived from a business vocabulary for facts expressible in the business vocabulary. Use an SBVR XMI-XML Schema where SBVR concepts and formulations are required, such as for communicating business vocabularies or rules, even though the rule statements also use a business vocabulary.

When communicating content via XML, it is important to consider reference schemes (see 'Reference Schemes' section in Part II). For example, to communicate that a certain car is owned by a certain branch (the third fact listed in the example above) facts must be communicated to identify which car and which branch. Reference schemes indicate what types of facts to use to identify things. The example above assumes that a reference scheme for car uses its vehicle identification number and a reference scheme for a branch uses the branch's name.

In some cases the sender and receiver of an XML document use different reference schemes. A sender should provide sufficient content in an XML document to satisfy the reference schemes of both himself and the receiver. Satisfying the receiver's scheme is most important so that the receiver can understand the document. Satisfying the sender's scheme is important if there is to be any return communication – the receiver must have a way to satisfy the sender's scheme when their roles are reversed.

## K.5 Using SBVR with Other MOF-Based Metamodels

SBVR's MOF-based metamodel differs from typical MOF-based metamodels in being vocabulary-driven and fact-based. How a typical MOF-based metamodel can build on and take advantage of SBVR has been a subject of discussion and interest.

This subject has become particularly important in developing a metamodel for organizational structure. The initial submitters to the Organizational Structure Metamodel (OSM) RFP have come together to form a joint submission. OSM provides a specific example of a general approach. Relevant concepts from SBVR are being incorporated into a proposed OSM in a way that fits well into an overall OSM design. OSM concepts have corresponding elements in a business vocabulary. A model of an organization must be able to combine instances of OSM concepts (e.g., "Organization") with instances of SBVR concepts (e.g., "logical formulation", "definition"). Also, it is possible that some OSM concepts are shared with or adopted from SBVR.

The following two steps facilitate a reasonable integration of SBVR with a custom designed OSM MOF-based class model. The same steps can be applied when combining SBVR with other metamodels.

1. Each MOF class in OSM should be a subclass (usually indirectly) of Essential SBVR::Thing. This allows SBVR-based facts to be represented regarding instances of all OSM classes.
2. Each class in OSM should be mapped to a concept represented by a term in an OSM vocabulary, and reference schemes should be defined. Attributes and associations in the OSM class model should be mapped to fact types represented in the vocabulary. These mappings must be documented.

Step 1 can be ignored if only a mapping is wanted. Using both steps provides maximum integration.

Based on step 1, for each MOF class in OSM, each object of the class is inherently of type Essential SBVR::Thing, so facts modeled using SBVR about corresponding instances can be captured based on SBVR's MOF-based metamodel. The same objects can also have OSM attributes and participate in OSM associations.

Based on step 2 above, for each fact type modeled in terms of SBVR that is also mapped to the OSM class model (e.g., as an attribute or association), the mapping is documented such that facts represented by an instantiation of the class model can be expressed using the OSM vocabulary.

Following the two steps supports sharing and specializing of concepts between SBVR and designed metamodels. For example, OSM's concept "organizational role" can specialize SBVR's concept "role". An object of type OSM::OrganizationalRole in an OSM repository thereby represents an SBVR role such that SBVR can be used to formalize an organizational role's definition and give it one or more designations that can be used when stating rules. Note that an OSM::OrganizationalRole object is also, by inheritance based on step 1 above, an Essential SBVR::Thing object, so anyone with a combined SBVR/OSM repository can include SBVR-based facts involving the same object.

Where SBVR is being combined with another metamodel that has already been defined independently of SBVR, the UML package merge capability can be used for step one above.

The two steps above provide a general answer to the question of how to use SBVR with traditional subject-based metamodels while preserving the benefits of both approaches to modeling. Traditional metamodels benefit from custom design. The SBVR metamodel and other SBVR-based metamodels benefit from satisfying the goals listed in K.1 and from the advantages of a fact-oriented approach listed K.2.4.

As the application of MDA involves more and more models coming from a business perspective, and therefore more vocabulary-driven models, the simple steps above will provide consistent alignment and semantic integration between traditional modeling and SBVR's fact-based, vocabulary-driven modeling, all using MOF and XMI.



## Annex L (informative)

### Examples of SBVR's Use of MOF

While the first purpose of the Vocabulary-to-MOF/XMI Mapping Rule Set is to produce a MOF model and XML schema from the SBVR vocabulary, the rules are best illustrated using a more simple vocabulary. The example below produces a MOF model and XML schema from a small, simplified subset of the EU-Rent vocabulary.

Consider the following subset of the EU-Rent English Vocabulary:

[branch](#)  
[branch](#) *has* [name](#)  
[branch](#) *owns* [car](#)  
[car](#)  
[car](#) *has* [vehicle identification number](#)  
[name](#)  
[regional headquarters](#)  
[vehicle identification number](#)

#### L.1 Example MOF/UML Model

The [Vocabulary-to-MOF/XMI Mapping Rule Set](#) guides creation of the following UML/MOF model.

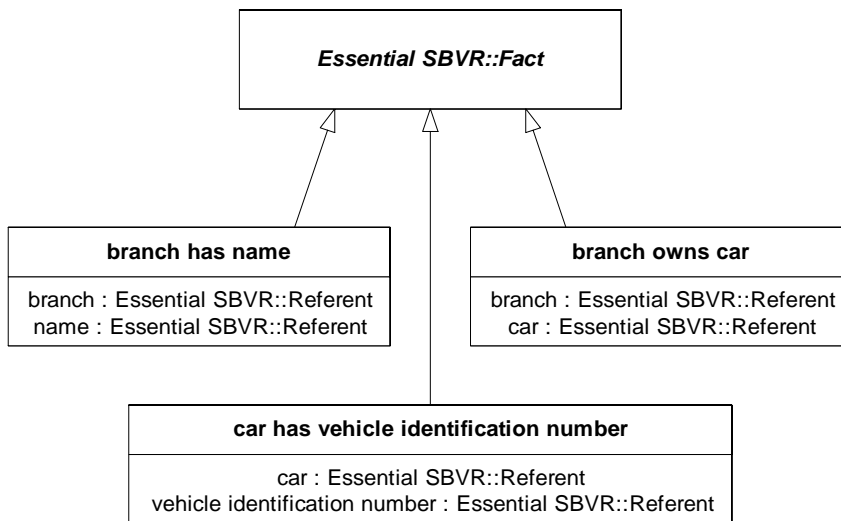
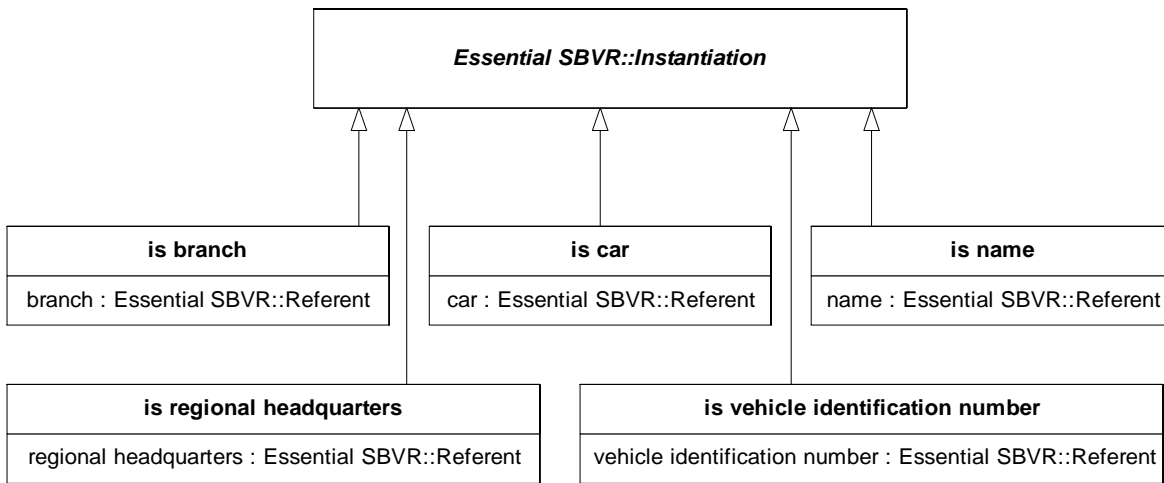


Figure L.1 - UML/MOF model created for part of EU-Rent English Vocabulary

The subclasses of the [Instantiation Class](#) shown in the diagram above are used to represent that a thing is an instance of the corresponding concept. Note that an instance of the ‘is car’ class is not used to represent a car – it represents a fact that a thing is a car. The subclasses of [Fact Class](#) shown above are used to represent facts based on the sentential forms in the vocabulary.

## L.2 Example Instances

Assume we want to express the following related facts, each of which is expressible using a single sentential form.

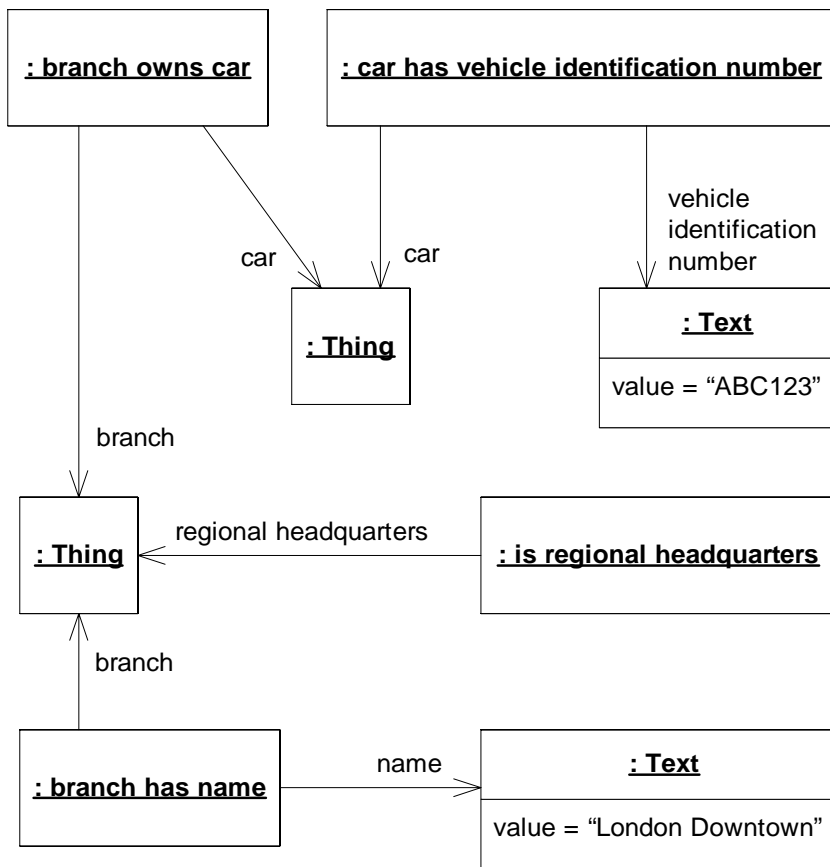
A given [car](#) *has* the [vehicle identification number](#) ‘ABC123’.

A given [branch](#) *has* the [name](#) ‘London Downtown’.

The [car](#) *is owned by* the [branch](#).

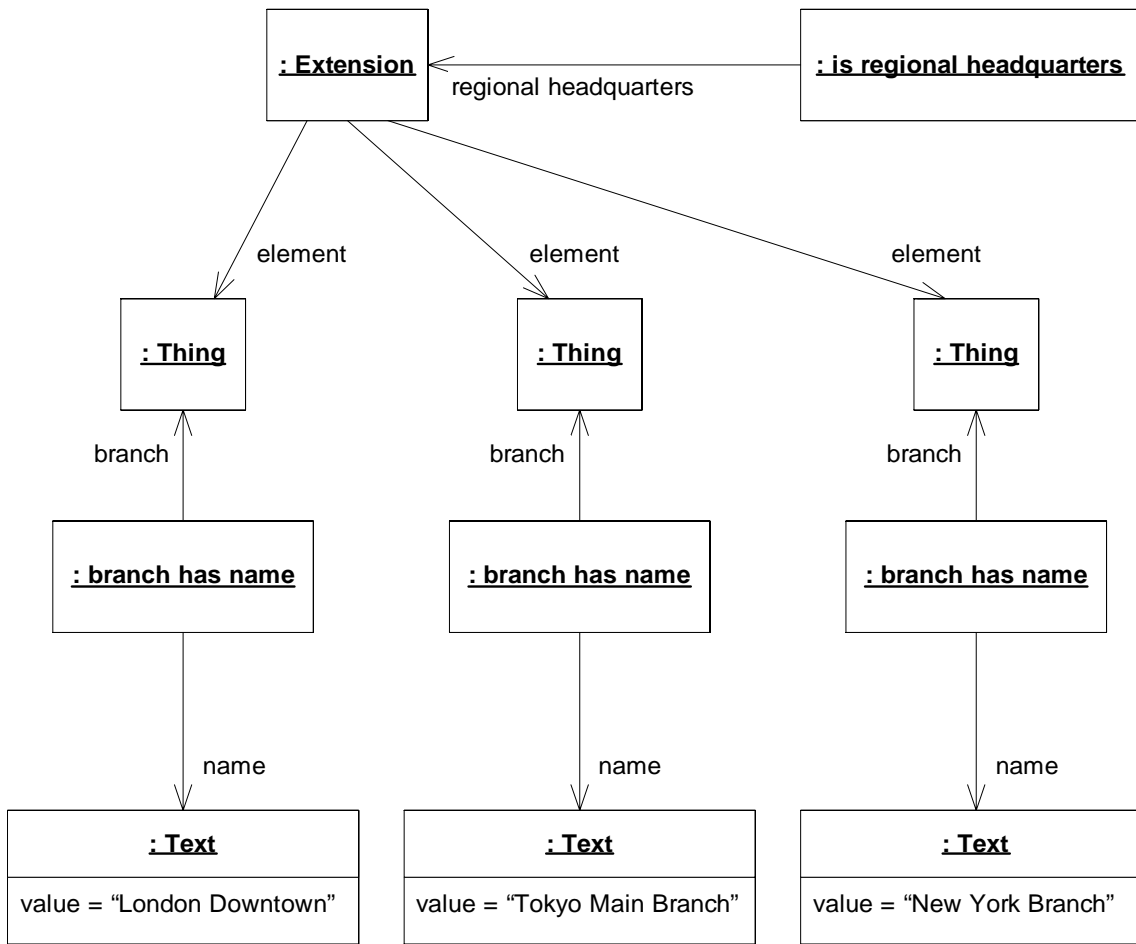
The [branch](#) *is a* [regional headquarters](#).

These facts are represented in terms of the created UML/MOF model by the following instance diagram.



**Figure L.2 - Instance diagram of facts expressed using EU-Rent English Vocabulary**

The example above represents some individual facts. Sometimes it is desirable to represent an entire extension of a concept. The example in Figure L.3 represents that all of the regional headquarters are the branches named "London Downtown," "Tokyo Main Branch," and "New York Branch:"



**Figure L.3 - Instance diagram of facts expressed using EU-Rent English Vocabulary**

The Extension class can also be used to represent everything in relation to something else. For example, Figure L.4 represents that all of the cars owned by the London Downtown branch are the ones with vehicle identification numbers ABC123 and XZY987.



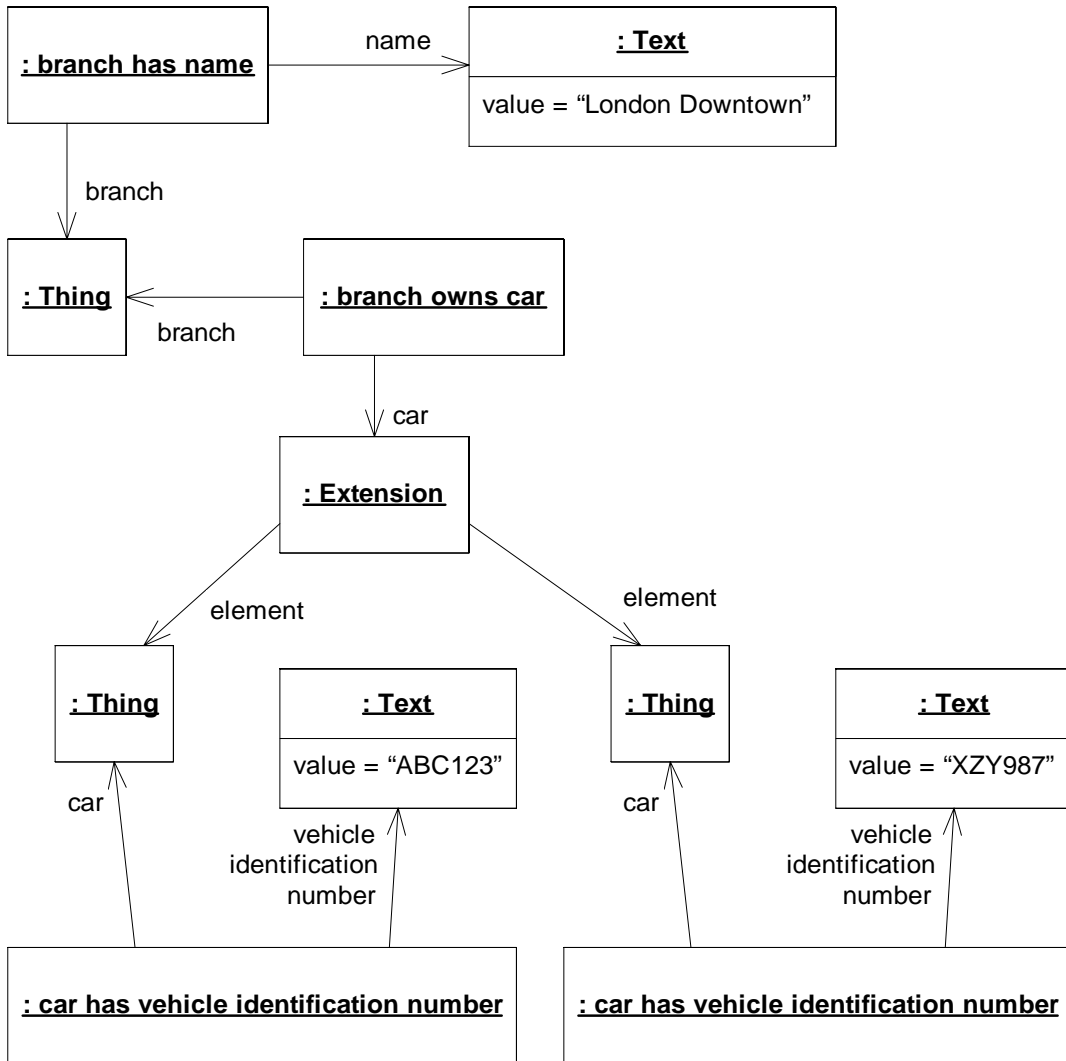


Figure L.4 - Instance diagram of facts expressed using EU-Rent English Vocabulary

### L.3 XML Schema Example

An XML schema can be created from the MOF model made for the partial EU-Rent vocabulary based on the XMI specification for MOF 2.<sup>1</sup> The Vocabulary-to-MOF/XMI Mapping Rule Set guides the creation of tags in the model so that valid names are created for XML elements and attributes. Here is the body of the XML schema.

1. This specification is still in finalization, so the schema below could change based on a final specification.

```

<xsd:complexType name="is-branch" >
  <xsd:complexContent >
    <xsd:extension base="esbvr:Instantiation">
      <xsd:choice minOccurs="0" maxOccurs="unbounded">
        <xsd:element name="branch" type="esbvr:Referent"/>
      </xsd:choice>
      <xsd:attribute name="branch" type="xsd:IDREF" use="optional"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:element name="is-branch" type="is-branch"/>
<xsd:complexType name="branch-has-name" >
  <xsd:complexContent >
    <xsd:extension base="esbvr:Fact">
      <xsd:choice minOccurs="0" maxOccurs="unbounded">
        <xsd:element name="branch" type="esbvr:Referent"/>
        <xsd:element name="name" type="esbvr:Referent"/>
      </xsd:choice>
      <xsd:attribute name="branch" type="xsd:IDREF" use="optional"/>
      <xsd:attribute name="name" type="xsd:IDREF" use="optional"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:element name="branch-has-name" type="branch-has-name"/>
<xsd:complexType name="branch-owns-car" >
  <xsd:complexContent >
    <xsd:extension base="esbvr:Fact">
      <xsd:choice minOccurs="0" maxOccurs="unbounded">
        <xsd:element name="branch" type="esbvr:Referent"/>
        <xsd:element name="car" type="esbvr:Referent"/>
      </xsd:choice>
      <xsd:attribute name="branch" type="xsd:IDREF" use="optional"/>
      <xsd:attribute name="car" type="xsd:IDREF" use="optional"/>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>
<xsd:element name="branch-owns-car" type="branch-owns-car"/>
<xsd:complexType name="is-car" >
  <xsd:complexContent >
    <xsd:extension base="esbvr:Instantiation">
      <xsd:choice minOccurs="0" maxOccurs="unbounded">
        <xsd:element name="car" type="esbvr:Referent"/>
      </xsd:choice>
    </xsd:extension>
  </xsd:complexContent>
</xsd:complexType>

```

```

        <xsd:attribute name="car" type="xsd:IDREF" use="optional"/>
    </xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:element name="is-car" type="is-car"/>
<xsd:complexType name="car-has-vehicle-identification-number" >
    <xsd:complexContent >
        <xsd:extension base="esbvr:Fact">
            <xsd:choice minOccurs="0" maxOccurs="unbounded">
                <xsd:element name="car" type="esbvr:Referent"/>
                <xsd:element name="vehicle-identification-number" type="esbvr:Referent"/>
            </xsd:choice>
            <xsd:attribute name="car" type="xsd:IDREF" use="optional"/>
            <xsd:attribute name="vehicle-identification-number" type="xsd:IDREF"
                use="optional"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
<xsd:element name="car-has-vehicle-identification-number"
    type="car-has-vehicle-identification-number"/>
<xsd:complexType name="is-name" >
    <xsd:complexContent >
        <xsd:extension base="esbvr:Instantiation">
            <xsd:choice minOccurs="0" maxOccurs="unbounded">
                <xsd:element name="name" type="esbvr:Referent"/>
            </xsd:choice>
            <xsd:attribute name="name" type="xsd:IDREF" use="optional"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
<xsd:element name="is-name" type="is-name"/>
<xsd:complexType name="is-regional-headquarters" >
    <xsd:complexContent >
        <xsd:extension base="esbvr:Instantiation">
            <xsd:choice minOccurs="0" maxOccurs="unbounded">
                <xsd:element name="regional-headquarters" type="esbvr:Referent"/>
            </xsd:choice>
            <xsd:attribute name="regional-headquarters" type="xsd:IDREF" use="optional"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
<xsd:element name="is-regional-headquarters" type="is-regional-headquarters"/>
<xsd:complexType name="is-vehicle-identification-number" >

```

```

<xsd:complexContent >
  <xsd:extension base="esbvr:Instantiation">
    <xsd:choice minOccurs="0" maxOccurs="unbounded">
      <xsd:element name="vehicle-identification-number" type="esbvr:Referent"/>
    </xsd:choice>
    <xsd:attribute name="vehicle-identification-number" type="xsd:IDREF"
      use="optional"/>
  </xsd:extension>
</xsd:complexContent>
</xsd:complexType>
<xsd:element name="is-vehicle-identification-number" type="is-vehicle-identification-number"/>

```

## L.4 XML Document Example

Shown below is one of the various ways the facts from the instance diagram in Figure L.2 can be validly represented based on the XML schema produced from the MOF model created based on the Vocabulary-to-MOF/XMI Mapping Rule Set.

```

<car-has-vehicle-identification-number car="c" vehicle-identification-number="v"/>
<branch-has-name branch="b" name="n"/>
<branch-owns-car branch="b" car="c"/>
<is-regional-headquarters regional-headquarters="b"/>
<esbvr:Thing xmi:id="b"/>
<esbvr:Thing xmi:id="c"/>
<esbvr:Text xmi:id="v">ABC123</esbvr:Text>
<esbvr:Text xmi:id="n">London Downtown</esbvr:Text>

```

## **Annex M** (informative)

### **Mappings and Relationships to Other Initiatives**

#### **M.1 Mapping to Other Standards and Metamodels**

##### **M.1.1 For Rule Representation**

There are several existing metamodels for representing rules. Only OWL has become a standard. The scopes of these metamodels differ from SBVR. The discussion in this section gives an overview of the most well-known, their use and characteristics.

- No standard is yet widely used on a commercial basis.
- With respect to rules, these metamodels focus on representation. SBVR focuses on unique, discrete meanings independent of form or representation.
- SBVR includes a Formal Model theory and semantic formulations. There is only partial coverage in other metamodels.
- Uniquely, SBVR provides necessity and obligation claims, which are critical to the formal representation of business rules.
- SBVR places special emphasis on obligation claims. In the real world of business activity, people can break such operative business rules, a crucial fact other metamodels do not address.

It is possible to create transformations from SBVR to any of the metamodels or vice versa. Any of the transformations, especially those moving from information systems specifications back to SBVR may require manual input to provide missing semantics or to transform decisions not automatable.

Development of transformations should consider the following points:

##### **Transformation from SBVR to the other metamodels**

- A decision should be made how to treat necessity and obligation claims. One option is to translate these to predicates.
- Some of the non-SBVR representations do not have an equivalent operator for the ‘whether or not’ and ‘equivalence’ operators.
- Some of the non-SBVR representations do not have equivalent operators for quantifiers like ‘each’, ‘some’, ‘at least one,’ etc. In that case might be possible to create special predicates or functions to deal with this semantics.

##### **Transformation from other metamodels to SBVR**

- The non-SBVR representations can have primitive types or primitive functions that do not exist in SBVR. By extending the SBVR Vocabularies with additional vocabulary, one can create a mapping from another metamodel to the extended SBVR. SBVR is self-extensible.

**Metamodels at the Business Level Used to Talk about Real Business Things  
 – Optimally Conceptualized to be ‘Business Friendly’ for Business People**

Name	Type	Developed by	Used by	Form	Reference
N458 Topic Map Constraint Language	Proposed Standard	ISO/IEC	Topic Maps	document	<a href="http://www.isotopicmaps.org/tmcl/tmcl-2005-02-12.html">http://www.isotopicmaps.org/tmcl/tmcl-2005-02-12.html</a>

**Topic Map Constraint Language** is designed to allow users to constrain any aspect of the topic map data model. TMCL adopts TMQL [Topic Map Query Language] as a means to express both the topic map constructs to be constrained and topic map structures that must exist in order for the constraint to be met.

Development of transformations should consider the following points:

**Transformation between SBVR and Topic Map Constraint Language**

- The only transformation required, in addition to the generally applicable ones mentioned above, would be where semantics conceptualized into SBVR metamodel constructs differently from the way it is conceptualized into metamodel constructs in Topic Maps as they both talk about real business things in business friendly terms.

**Metamodels that can be Used at the Business Level Used to Talk about Real Business Things – Optimally Conceptualized for Logicians and/or Machine Processing Efficiency**

Name	Type	Developed by	Used by	Form	Reference
24707 Common Logic	Proposed Standard	ISO	KIF, CGIF, XCL, PSL	Document	<a href="http://www.iso.org">www.iso.org</a>
OWL	Standard	W3C	Semantic Web	DTD or XML schema	<a href="http://www.w3c.com">www.w3c.com</a>

**ISO Common Logic** is a first order logic language for knowledge interchange. It provides a core semantic framework for logic and the basis for a set of syntactic forms (dialects) all sharing a common semantics. **ISO Common Logic** can also be used at the Information System Specification level to talk about information and information system components as it is a general-purpose first-order predicate logic standard.

**OWL** is a Web Ontology language. Where earlier languages have been used to develop tools and ontologies for specific user communities (particularly in the sciences and in company-specific e-commerce applications), they were not defined to be compatible with the architecture of the World Wide Web in general, and the Semantic Web in particular.

OWL uses both URIs for naming and the description framework for the Web provided by RDF to add the following capabilities to ontologies:

- Ability to be distributed across many systems
- Scalability to Web needs
- Compatibility with Web standards for accessibility and internationalization
- Openness and extensibility

OWL builds on RDF and RDF Schema and adds more vocabulary for describing properties and classes: among others, relations between classes (e.g., disjointness), cardinality (e.g., "exactly one"), equality, richer typing of properties, characteristics of properties (e.g., symmetry), and enumerated classes.

Development of transformations should consider the following points:

**Transformation from SBVR to the above standards**

- In general, formal logic-based entries in SBVR-based conceptual schemas and models will be transformable into ISO Common Logic or into OWL.

**Transformation from the above standards to SBVR**

- Any ISO Common Logic sentences and Owl entries that can be expressed in ISO Common Logics that
  - talk about real business things (and not data about real business things or information system buckets that hold such data), and
  - are limited to the SBVR ‘restricted higher order logic’ can be transformed into SBVR if the semantic equivalences of different representations and different semantic formulations are provided by the transformation as these are not kept track of in ISO Common Logic.
- Some contents of SBVR-based conceptual schemas and models which do not have counterparts in OWL or ISO Common Logics might need to be provided manually.

**Metamodels that Specify Information Systems at the PIM/PSM Levels**

Name	Type	Developed by	Used by	Form	Reference
13211 Prolog	Standard	ISO		document	www.iso.org
Production Rules Representation	Proposed Specification	OMG		XMI	www.omg.org
RuleML	Metamodel	Consortium (see reference)	Mandarax, the website contains a list of 40 participants (mostly academics)	DTD	www.ruleml.org
SWRL	Metamodel	DAML		XML Schema	www.daml.org

Proprietary Metamodels					
Name	Type	Developed by	Used by	Form	Reference
RBML	Metamodel	LibRT	VALENS, Artis, Power	XML Schema	www.librt.com
SRML	Metamodel	Ilog	Ilog Jrules	DTD	www.ilog.com
SRL	Metamodel	Fair Isaac	Blaze Advisor	DTD	www.fairisaac.com
BRML	Metamodel	IBM	IBM CommonRules	XML Schema	www.ibm.com

Development of transformations should consider the following points:

**Transformation from SBVR to the above metamodels**

- Alignment of SBVR with the above metamodels requires a transform from SBVR whose entries talk about real things in the business to specifications of data about the real business things, and the design specifications for the buckets used to store that data within various components of the information system.

**Transformation from the above metamodels to SBVR**

- Requires the (re-)introduction, probably manually, of whatever business semantics (or pointers to them) are not within the information systems specifications.

**M.1.1.1 For Vocabulary Representation**

Today there are several standards and models for representing a vocabulary. It must be noted, however, that none of these provides an adequate extension to formal logics to fully support business rules. The following list gives an overview of the most well-known, their use and characteristics:

**Metamodels at the Business Level Used to Talk about Real Business Things – Optimally Conceptualized to be ‘Business Friendly’ for Business People**

Name	Type	Developed by	Used by	Form	Reference
1087-1, 704-2000, 10241, & 12620 Terminology	Standard	ISO		document	www.iso.org
17115 Health Informatics -- Vocabulary for Terminological Systems	Standard	ISO/DIS		document	www.iso.org
2788 & 5964 Thesaurus	Standard	ISO		document	www.iso.org
12620 & 13250-2 Topic Maps	Standard	ISO		document	www.iso.org
<b>Public Domain De Facto Industry Standard</b>					
ORM	Metamodel	Terry Halpin, et al	Microsoft, Case talk, Infagon		www.orm.net www.demo.nl www.mattic.com/ Infagon.html

- SBVR is based on the **ISO standards 1087-1 and 704-2000** for terminology and information science. These standards describe a methodology but do not provide a product metamodel that can be used to store and interchange business vocabularies.
- **Health Informatics -- Vocabulary for Terminological Systems** supplements the ISO 1087-1 and 704-2000 standards to provide a more formal structuring of terminology. From the standard: “The purpose of this International Standard is to define a set of basic concepts required to describe formal concept representation systems, especially for health sciences, and to describe representation of concepts and characteristics, for use especially in formal computer based concept representation systems. A main motivation is to make it possible to precisely describe content models described in other International Standards.”



- **ISO 2788 & 5964 Documentation - Guidelines for the establishment and development of monolingual/multilingual thesauri** is about creating indexes for books and other documents by identifying the subjects or topics (concepts) discussed in the document. From the standard: “The recommendations set out in this International Standard are intended to ensure consistent practice within a single indexing agency, or between different agencies (for example members of a network)”.
- **ISO/IEC 13250 Topic Maps** is about Topics (Concepts) and connections between them (Facts). From the standard: “This International Standard provides a standardized notation for interchangeably representing information about the structure of information resources used to define topics, and the relationships between topics. A set of one or more interrelated documents that employs the notation defined by this International Standard is called a topic map.”
- **ORM** is a modeling method originally intended for database design. SBVR is highly influenced by the way ORM defines and verbalizes fact types and facts. Transformations between a vocabulary of SBVR and ORM tools can be established although not all SBVR concepts have an equivalent in ORM.

Development of transformations should consider the following points:

#### Transformation between SBVR to the above standards

- With the following exceptions the only transformation required, in addition to the generally applicable ones mentioned above, would be where the semantics in SBVR was conceptualized into metamodel constructs differently from the way it was conceptualized into metamodel constructs in Topic Maps as they both talk about real business things in business friendly terms.
  - (except for ORM) none of the above standards are based on formal logics so there will need to be some manual decisions going from them to SBVR. The other direction should be automatic except for constructs not in the SBVR metamodel.
  - Constructs in SBVR not in those standards would be lost
- ORM is very similar to SBVR so that two-way transformations are minimal. However, SBVR is more comprehensive than ORM so some semantics would be lost going from SBVR and have to be provided manually going to SBVR.

#### Metamodels at the Business Level Used to Talk about Real Business Things – Optimally Conceptualized for Machine Processing Efficiency

Name	Type	Developed by	Used by	Form	Reference
RDF(S)	Standard	W3C		DTD or XML schema	www.w3c.com
OWL	Standard	W3C		DTD or XML schema	www.w3c.com

- **RDF(S)** not only talks about real business things but also contains pointers (URLs) to the storage locations where information about those business things is kept. Thus RDF(S) also includes mappings across the transformation from the Business Level to the Information System specification levels. RDF(S) does provide a metamodel that can be used to store and interchange business vocabularies. It is expected that lossless bidirectional transformations between SBVR and RDF(S) can be established.
- *OWL Web Ontology Language* is intended to be used when the information contained in documents needs to be processed by applications, as opposed to situations where the content only needs to be presented to humans. OWL can be used to explicitly represent the meaning of terms in vocabularies and the relationships between those terms. This representation of terms and their interrelationships is called an ontology. OWL has more facilities for expressing meaning and semantics than XML, RDF, and RDF-S, and thus OWL goes beyond these languages in its ability to

represent machine interpretable content on the Web

Development of transformations should consider the following points:

### Transformation from SBVR to the above standards

- All and only formal logic-based entries in SBVR will be able to be transformed into RDF(S) and/or OWL because both are also mapped to ISO Common Logic.

### Transformation from the above standards to SBVR

- Any RDF(S) and Owl entries that can be expressed in ISO Common Logics that
  - talk about real business things (and not data about real business things or information system buckets that hold such data), and
  - are limited to the SBVR ‘restricted higher order logic’ can be transformed into SBVR if the semantic equivalences of different representations and different semantic formulations are provided manually as these are not kept track of in ISO Common Logic.
- All of the vocabulary related entries not part of the SBVR formal logics section will have to be provided manually.

### M.1.1.2 Standards for database and system modeling

Today there are several standards and models for representing a database or a systems object model. Most well known and widely used are UML for Object Oriented models and Entity Relationship diagram for relational databases.

A vocabulary that is developed using SBVR may contain concepts and fact types that should also be represented in a database or object model. For those concepts a transformation to UML or Entity Relationship diagrams can be created. Be aware that the SBVR model and a PIM level UML model or ER diagram have a different perspective. That is the reason that not all elements of SBVR may be relevant in a PIM perspective and the PIM model may need to be augmented after a transformation from SBVR.

### Metamodels that Specify Information Systems at the PIM/PSM Levels

Name	Type	Developed by	Used by	Form	Reference
11179 Metadata Registry	Standard	ISO/IEC		document	www.iso.org
UML	Specification	OMG		document	www.omg.org
Entity Relationship (CWM)	Specification	OMG		document	www.omg.org

[See Section 1.9.3.3.1 for description of SBVR mapping to PIM standards, specifications, and models.]

- **ISO/IEC 11179 - Metadata Registries (MDR)**, addresses the semantics of data, the representation of data and the registration of the descriptions of that data. It is through these descriptions that an accurate understanding of the semantics and a useful depiction of the data are found.

The purposes of ISO/IEC 11179 are to promote the following:

- Standard description of data
- Common understanding of data across organizational elements and between organizations
- Re-use and standardization of data over time, space, and applications
- Harmonization and standardization of data within an organization and across organizations

- Management of the components of data
- Re-use of the components of data
- **The Unified Modeling Language™ - UML** is an OMG (Object Management Group) specification for modeling application structure, behavior and architecture.
- **Entity Relationship – CWM (Common Warehouse Metamodel)** The Common Warehouse Metamodel (CWM™) is a specification that describes metadata interchange among data warehousing, business intelligence, knowledge management and portal technologies. Entity Relationship (ER) models are used frequently as a means of describing business processes and the data on which they operate. Because of its importance as a design and tool model, the CWM includes a foundational ER model from which individual tool models may derive their specific extensions. Doing so will improve the extent to which ER models can be interchanged between various tooling environments.

Development of transformations should consider the following points:

### **Transformation from SBVR to the above standards**

- Inputs to this transformations are:
  - An extract from an SBVR model that fits the scope of the application software to be designed
  - Additional Business requirements for the application software
- The transformation is effectively the design process of the class-of-platform independent PIM model. It includes, among others, such design choices as:
  - Design of generalized data storage structures e.g., Hierarchies, Data-driven generalizations
  - Class / Attribute / Association / Association Class decisions
  - One Concept of a Business Thing implemented in two Attributes
  - Store vs. derive decisions
  - Design of time constructs
  - ‘State’ implementation design decisions
  - Surrogate Keys design choices

### **Transformation from the above standards to SBVR**

- This is a reverse engineering transformation which is made possible only by adding back in, as part of the transformation process, any SBVR business semantics that were not stored with the model when it was created, and maintained since then.

#### **M.1.1.3 Standards for Business Modeling Vocabularies + Rules**

There are a number of standards that provide vocabularies and rules for subjects commonly used to specify in business models the way businesses are to be operated. These standards can be imported into the SBVR metamodel to become general-purpose SBVR Business Vocabularies+Rules.

- Country names and codes (ISO/IEC 3166)
- Dates and times (ISO/IEC 8601)
- Currency codes (ISO/IEC 4217)
- Addresses (ISO/IEC 11180)

- Information and documentation (ISO/IEC 5127)
- Business Agreement Semantic Descriptive Techniques (ISO/IEC 15944)
- Process Specification Language (ISO 18629 series of standards)
- ...many others from ISO and other standards bodies

In turn these general-purpose SBVR can be adopted by business-specific vocabularies and used to specify a given business.

## **M.1.2 Use of UML Notation in a Business Context to Represent SBVR Vocabulary Concepts**

UML Notation can be used to represent SBVR-based vocabularies. Details of the mapping of SBVR concepts to UML Notation are provided in Annex H.

## **M.1.3 Reuse of other OMG Standards**

This SBVR specification reuses the MOF 2.0, XMI 2.1 and the UML 2 Infrastructure for its model repository and for interchange of SBVR Vocabularies and rules.

## **M.1.4 Relationship of SBVR with other OMG RFPs**

### **M.1.4.1 SBVR and Business Modeling**

SBVR is only one of several BEIDTF initiatives in the business modeling arena. Others include:

#### **Business Process Definition Metamodel (BPDM)**

The revised submission deadline for RFP responses is in August 2005.

SBVR and BPDM are complementary. SBVR specify the meaning and representation of Business Vocabulary and Rules. BPDM specifies the use of Business Vocabulary and Rules by various BPDM model elements.

The primary relationship of SBVR and BPDM is the roles Business Rules play in a BPDM. The definition of the relationship between Business Concepts, Business Facts and Business Rules in SBVR and the various model elements in BPDM is scheduled to be called for in a separate RFP, the adoption of whose response will integrate both SBVR and BPDM.

Secondarily, SBVR can be used to provide formal logics-based definitions for all the model elements in BPDM (see Section 1.9.4.2).

#### **Organization Structure Metamodel (OSM)**

The initial submission deadline for RFP responses is in November 2005.

#### **Business Rules Management (BRM)**

The RFP is being drafted.

SBVR is about the meaning and representation of Business Vocabulary and Rules and only that. BRM focuses on all other information about Business Vocabulary and Rules needed to effectively manage and use them to operate the business and as part of information system requirements. SBVR provides the Business Vocabulary and Rules that are managed by BRM. BRM manages the contents of SBVR.

## Business Motivation Model (BMM)

The Business Rules Group (BRG) has been encouraged to submit its Business Motivation Model: *Business Governance in a Volatile World* [BMM]<sup>1</sup> to the BEIDTF under the OMG's Request for Comment (RFC) process. This model addresses business goals, strategies and policies.

SBVR and BMM are complementary. SBVR adopts the BMM definition of Business Policy, and BMM adopts the SBVR definition of Business Rule.

## SBVR and Need for Integration among Business Modeling Specifications

These BEIDTF developments are related. For example, BPDM and SBVR have strongly related central concepts:

- From the BPDM perspective, Business Rules deliver 'factored out', flexible detail to support Business Processes.
- From the SBVR perspective, Business Processes provide the specific contexts in which Business Rules need to be evaluated. (In a PIM view, this might mean 'fired' or 'triggered', for example.)

Whether the BRG's Business Motivation Model is accepted or not, the BEIDTF will need a metamodel for its domain. Business processes are better defined when a business knows where it wants to go (its goals and objectives), and what it needs to do to get there (its strategies, tactics and policies). Business processes realize the strategies and tactics. Business rules realize the business policies, and both support and constrain the business processes.

Business processes, supported by business rules, are associated with organization roles and structure. Some business rules apply to organization structure and roles, independently of processes.

There is clearly a need for integration. This has been recognized. For example, the BPDM submission included 'hooks' for business rules and organization roles.

## Need for Common Vocabulary

An important first step towards integration is to ensure common vocabulary. Within a business, 'customer' and 'product' should mean the same everywhere they are intended to be the same, no matter what aspects of the business people are discussing or defining -- processes, rules, organizational responsibilities, locations, etc.

It is suggested that the BEIDTF consider *integration by adoption*, a loose coupling of metamodels by adoption of concepts and terms. This would mean:

- Shared concepts would be defined once in an 'owner' standard, and adopted by other standards as 'users'
- Benefits would be consistency across standards and reduction of replication
- The implication would be that when an 'owner' standard is revised, all the 'users' have to be considered (note: this would be a good thing!)

Concepts could also be adopted from outside the OMG; for example, this submission for SBVR adopts from ISO, standard dictionaries, and other authoritative sources.

What is important for OMG Business Modeling Standards is to ensure a shared body of meanings, largely by use of accepted vocabularies and diligent examination of the similarities and differences in the vocabularies of BEI and ADPTF standards. By definition, there are different communities and contexts involved, and the term-concept relationships may be different. Synonyms and homonyms need to be recognized, and definitions brought up to a formal logics quality.

---

1. The BRG released version 1.0 in 2000, entitled *Organizing Business Plans*.

## For the Future – A Common Vocabulary Model?

This specification for SBVR incorporates a well-developed approach to vocabulary. The SBVR view is that the concepts should be consistent across the business, and the terms used for them should be unambiguously understood. This includes management of synonyms, homonyms, and resolution of ambiguity by providing contexts.

This is important for practical application of SBVR to real businesses. People in different operational areas, in different geographical locations and in parts of businesses that have been merged or acquired, will use their familiar terminology. They can be encouraged into standard terminology, but they cannot be forced. Major customers, partner organizations, outsourcers and trade groups will also share concepts, even if they use different words.

This need to support this is not specific to business rules. It is relevant to all types of business description, from mission statements to scripts for help lines.

A next step from *integration by adoption* across OMG business modeling standards would be to create a common metamodel for business vocabulary. If the BEIDTF decided to do this, it would be reasonable to propose a subset of the SBVR model as a candidate. The part of the model that supports concepts, fact types, and vocabulary has been separated from the business rules part, and can be reused to support other aspects of business modeling.

### M.1.4.2 SBVR and Platform Independent Modeling

As discussed above, the SBVR standard should be integrated with other OMG standards for business modeling. This would help ensure that coherent business models are developed and supported consistently with tools and methodologies based on these standards.

Such business models (or substantial parts of them) will be used as bases for specification of information system models. In MDA, this would require mappings and transformations from a business model to a Platform Independent Model (PIM).

#### Mapping to a PIM

The current MDA practice is for a PIM to be defined using UML models. Two kinds of transformation will be used:

- From business concepts (including fact types) to a UML class model. Some concepts will map to classes, others to attributes. Some fact types will map to associations in the class model. Some structural business rules will map to constraints on cardinality, optionality and mutual exclusion.
- From business rules to operations and constraints in the UML models formed from business concepts. There are several possible approaches for this, and further investigation would be needed.

The transformation of business rules would provide only part of a PIM, which would also support transformed content from other business model aspects, including business process, user interfaces and workflow. This reinforces the case made above for a common business vocabulary model.

See Section M.1.1.2 for adopted PIM non-OMG standards and OMG Specifications.

#### Other submissions for SBVR

Other submissions for SBVR have presented PIM-oriented metamodels that would support a rule-based approach more directly than the general mapping to PIM described above.

They are based on extensions of UML such that many types of business rule (as described in this proposal) could be expressed in OCL. Two kinds of transformation would be required:

- From a subset of the business vocabulary to a UML class model, as described above.

- From a subset of business rules to OCL, using the vocabulary of the class model. Additional guidance would be needed for types of business rules that would not map directly.

This is an important piece of the architectural jigsaw, especially with regard to transformation to a Platform-Specific Model (PSM), and the BEIDTF might consider issuing another RFP to address it.

### **Production Rule Representation**

The BEIDTF has issued an RFP for Production Rule Representation. The RFP requests a model and XML interchange format for rules executed in an inference engine. Initial responses were submitted in August 2004, and the proposers have since agreed to collaborate on a joint proposal.

Production rules have the general form “if condition, do action”, and would use the vocabulary of a PIM’s class model. They may be grouped into rule sets that can be invoked en bloc.

Business rules in this SBVR specification have the declarative form “the following proposition should/must always be true”, and use a business vocabulary.

As with other approaches for mapping to PIM, a UML class model consistent with the business vocabulary is assumed. Some transformations from declarative business rules to production rule form are already well-understood at the level of individual rules, but substantial work will be required to develop a full mapping that includes making all conditions and actions explicit, and grouping rules into rule sets.

### **Ontology Definition Metamodel**

As well as BEIDTF initiatives, SBVR is also related to the Ontology Definition Metamodel (ODM), which is being developed in response to an RFP issued by the OMG Analysis and Design Task Force.

The OMG Ontology Definition Metamodel (ODM) intends to provide an integrated family of metamodels for a variety of knowledge representation techniques, to assist in defining and interchanging ontologies, with a key objective of supporting semantic technologies. Most of the metamodels in this family reflect the abstract syntax of an existing standard, rather than inventing a new representation paradigm. The term “ontology” refers to a machine-processable representation of knowledge, particularly for automated inferencing. In general, the audience for the ODM is the developers of inference engines, tools that capture and prepare ontologies for inference engines from other declarative forms, such as UML models and structural business rules, and tools that convert ontologies into other forms of implementation model. A key concept in ontologies is that knowledge is “monotonic”: Over time we can add to our knowledge, but we won’t learn anything that contradicts something we already know for sure, so that knowledge from multiple sources can be combined.

The ODM is being developed concurrently with SBVR. The draft proposed ODM includes metamodels of several popular knowledge representation languages, with mappings between them.

The draft proposed ODM as of October 2004 includes proposed MOF metamodels for:

- Resource Description Framework Schema (RDFS – W3C Recommendation),
- Web Ontology Language (OWL – W3C Recommendation),
- ISO Common Logic (CL, defined in ISO 24707),
- Topic Maps (TM – ISO 13250),
- Unified Modeling Language (UML), and
- Description Logic (DL).

For business rules, monotonic logic is only applicable to a small fraction of the concerns. In many areas, the business is not interested in what is true for all time, but rather in what is true now and may change in the next hours or days. And in some cases, it is the objective of certain business rules to change currently true but unfavorable situations into future favorable situations. So there is a significant difference in the purposes of these standards. And this gives rise to significant differences in the interpretation of the logic models. (In the Semantic Web work, the distinction is made between class-based reasoning, which matches a subset of SBVR structural rules capabilities and is monotonic, and instance-based reasoning, which deals with actual facts about specific objects and may not be “safe” for monotonic reasoning.)

To handle operative business rules, which involve obligations and permissions, SBVR supports logic elements beyond those of the ODM languages, including CL. Lossless bi-directional transformations between the SBVR rules metamodel and the ODM metamodels are not guaranteed. A partial mapping between SBVR and the ODM metamodels could be developed. With proper care, SBVR could be used in ontology development.

ISO is considering extending CL to include modal and other logics and is planning a natural language surface syntax for CL. Both of these ISO initiatives may be important to SBVR and ODM in the future, but they are out of scope for the current ODM and SBVR work.



## Annex N (informative)

### Additional References

#### N.1 Bibliography / Normative References

[AH] *American Heritage Dictionary*.

[Anto2001] Antonelli, A. *Non-Monotonic Logic*, Stanford Encyclopedia of Philosophy, 2001. Available from <http://plato.stanford.edu/entries/logic-nonmonotonic/>

[Blo96] Bloesch, A.C., and Terry A. Halpin. "ConQuer: a Conceptual Query Language." In *Proc. ER'96: 15th Int. Conf. on Conceptual Modeling*, 121-133: Springer LNCS, 1996. Available from <http://www.orm.net/pdf/ER96.pdf>

[Blo97] \_\_\_\_\_. "Conceptual Queries using ConQuer-II." In *Proc. ER'97: 16th Int. Conf. on Conceptual Modeling*, 113-126: Springer LNCS, 1997. Available from <http://www.orm.net/pdf/ER97-final.pdf>

[BMM] Business Rules Group. *The Business Motivation Model ~ Business Governance in a Volatile World*. 1.2 ed., Sept. 2005. Originally published as *Organizing Business Plans ~ The Standard Model for Business Rule Motivation*, Nov. 2000. Available from <http://www.BusinessRulesGroup.org>

[BRM] Business Rules Group. Ronald G. Ross, ed. *Business Rules Manifesto ~ The Principles of Rule Independence*. 1.2 ed. The Business Rules Group, 2003. Updated Jan. 8, 2003. PDF. Available from <http://www.BusinessRulesGroup.org/brmanifesto.htm>

[BRG2002] Business Rules Group. *Defining Business Rules ~ What Are They Really?* 4th ed., July 2002. Originally published as *GUIDE Business Rules Project Report*, 1995. Available from <http://www.BusinessRulesGroup.org>

[BRJ2005] Editors of BRCommunity.com. "A Brief History of the Business Rule Approach." *The Business Rules Journal* 6, no. 1 (2005). Available from <http://www.BRCommunity.com/a2005/b216.html>

[CDP] *The Cambridge Dictionary of Philosophy*. 2nd ed.: Cambridge University Press, 1999.

[CSILL] *Cognitive Science Initiative: Language Lexicon*. University of Houston. Available from <http://www.hfac.uh.edu/COGSCI/lang/Entries/>

[Dean1997] Dean, Neville. *The Essence of Discrete Mathematics*, The Essence of Computing Series: Prentice-Hall, 1997.

[Fitt2002 (or TTGG)] Fitting, Melvin. *Types, Tableaus, and Gödel's God*, Trends in Logic, Studia Logica Library. Dordrecht, the Netherlands: Kluwer Academic Publishers, 2002.

[Gir12000 (or MLP)] Girle, Rod A. *Modal Logics and Philosophy*: McGill-Queen's University Press, 2000.

[Halp1989 (or HALT89)] Halpin, Terry A. "A Logical Analysis of Information Systems: Static Aspects of the Data-oriented Perspective." PhD thesis, Department of Computer Science, University of Queensland, 1989.

[Halp1998] \_\_\_\_\_. "Object-Role Modeling (ORM/NIAM)." In *Handbook on Architectures of Information Systems*. Heidelberg: Springer, 1998.

[Halp2000] \_\_\_\_\_. *Object-Role Modeling: An Overview*. San Francisco: Springer, 2000. Available from <http://www.orm.net/pdf/springer.pdf>

[Halp2001 (or IMRD)] \_\_\_\_\_. *Information Modeling and Relational Databases*. San Francisco: Morgan Kaufmann, 2001.

[Halp2003a] \_\_\_\_\_. "Verbalizing Business Rules: Part 1." *The Business Rules Journal* 4, no. 4 (2003). Available from <http://www.BRCommunity.com/a2003/b138.html>

[Halp2003b] \_\_\_\_\_. "Verbalizing Business Rules: Part 2." *The Business Rules Journal* 4, no. 6 (2003). Available from <http://www.BRCommunity.com/a2003/b152.html>

[Halp2003c] \_\_\_\_\_. "Verbalizing Business Rules: Part 3." *The Business Rules Journal* 4, no. 8 (2003). Available from <http://www.BRCommunity.com/a2003/b163.html>

[Halp2003d] \_\_\_\_\_. "Verbalizing Business Rules: Part 4." *The Business Rules Journal* 4, no. 10 (2003). Available from <http://www.BRCommunity.com/a2003/b172.html>

[Halp2004 (or HALT2004)] \_\_\_\_\_. "Information Modeling and Higher-Order Types." In *Proc. CAiSE'04 Workshops*, eds. J. Grundspenkis and M. Kirkova, 1, 233-248: Riga Tech. University, 2004. Available from <http://www.orm.net/pdf/EMMSAD2004.pdf>

[Halp2004b] \_\_\_\_\_. "Business Rule Verbalization." In *Lecture Notes in Informatics*, eds. A. Doroshenko, Terry A. Halpin and S. Liddle, P-48, 39-52. Salt Lake City: Proc. ISTA-2004, 2004.

[Halp2004c] \_\_\_\_\_. "Verbalizing Business Rules: Part 5." *The Business Rules Journal* 5, no. 2 (2004). Available from <http://www.BRCommunity.com/a2004/b179.html>

[Halp2004d] \_\_\_\_\_. "Verbalizing Business Rules: Part 6." *The Business Rules Journal* 5, no. 4 (2004). Available from <http://www.BRCommunity.com/a2004/b183.html>

[Halp2004e] \_\_\_\_\_. "Verbalizing Business Rules: Part 7." *The Business Rules Journal* 5, no. 7 (2004). Available from <http://www.BRCommunity.com/a2004/b198.html>

[Halp2004f] \_\_\_\_\_. "Verbalizing Business Rules: Part 8." *The Business Rules Journal* 5, no. 9 (2004). Available from <http://www.BRCommunity.com/a2004/b205.html>

- [Halp2004g] \_\_\_\_\_. "Verbalizing Business Rules: Part 9." *The Business Rules Journal* 5, no. 12 (2004). Available from <http://www.BRCommunity.com/a2004/b215.html>
- [Halp2005a] \_\_\_\_\_. "Verbalizing Business Rules: Part 10." *The Business Rules Journal* 6, no. 4 (2005). Available from <http://www.BRCommunity.com/a2005/b229.html>
- [Halp2005b] \_\_\_\_\_. "Verbalizing Business Rules: Part 11." *The Business Rules Journal* 6, no. 6 (2005). Available from <http://www.BRCommunity.com/a2005/b238.html>
- [Halp2005c] \_\_\_\_\_. "Verbalizing Business Rules: Part 12." *The Business Rules Journal* 6, no. 10 (2005). Available from <http://www.BRCommunity.com/a2005/b252.html>
- [Halp2005d] \_\_\_\_\_. "Verbalizing Business Rules: Part 13." *The Business Rules Journal* 6, no. 12 (2005). Available from <http://www.BRCommunity.com/a2005/b261.html>
- [Halp1981 (or DL)] Halpin, Terry A., and Rod A. Girle. *Deductive Logic*. 2nd ed. Brisbane: Logiqpress, 1981.
- [Hunt1971 (or META)] Hunter, Geoffrey. *An Introduction to the Metatheory of Standard First Order Logic*: University of California Press, 1971.
- [IETF RFC 2396] Berners-Lee, Tim, R. Fielding, and L. Masinter. *Uniform Resource Identifiers (URI): Generic Syntax*. The Internet Society, 1998. Updated August 1998. Available from <http://www.ietf.org/rfc/rfc2396.txt>
- [ISO704] International Organization for Standardization (ISO). *Terminology work - Principles and Methods*. English ed.: ISO, 2000.
- [ISO1087-1] \_\_\_\_\_. *Terminology work - Vocabulary - Part 1: Theory and Application*. English/French ed.: ISO, 2000.
- [ISO860] \_\_\_\_\_. *Terminology work - Harmonization of Concepts and Terms*. ISO, 1996.
- [ISO639-2] \_\_\_\_\_. *Codes for the Representation of Names of Languages-- Part 2: Alpha-3 Code*. Library of Congress, 2002. Available from <http://www.loc.gov/standards/iso639-2/langcodes.html>
- [ISO/IEC CD 24707] \_\_\_\_\_. *Information technology -- Common Logic (CL) -- A Framework for a Family of Logic-Based Languages*: ISO, 2005. Available from <http://www.iso.org/iso/en/CatalogueDetailPage.CatalogueDetail?CSNUMBER=39175>
- [Levi1983 (or LEVS)] Levinson, Stephen C. *Pragmatics*, Cambridge Textbooks in Linguistics: Cambridge University Press, 1983.
- [MATH] *PlanetMath.org*. Available from <http://planetmath.org/encyclopedia>
- [Mend1997 (or MEN97)] Mendelson, Elliott. *Introduction to Mathematical Logic*. 4th ed.: Chapman & Hall, 1997.
- [MWCD] *Merriam-Webster Collegiate Dictionary*.
- [MWDS] *Merriam-Webster Dictionary of Synonyms*.

[MWU] *Merriam-Webster Unabridged*.

[NODE] *The New Oxford Dictionary of English*.

[Nolt1998 (or LSO)] Nolt, John, Dennis Rohatyn, and Achille Varzi. *Logic*. 2nd ed., Schaum's Outlines. New York: McGraw-Hill, 1998.

[OSM] *Organizational Structure Metamodel*: OMG, 2005.

[Peik (or PEIL)] Peikoff, Leonard. "The Analytic-Synthetic Dichotomy." In *Rand1990*, 88-121.

[Rand1990 (or RANA90)] Rand, Ayn. *Introduction to Objectivist Epistemology*. expanded 2nd ed. New York: Meridian, 1990.

[Ross1997] Ross, Ronald G. *The Business Rule Book -- Classifying, Defining and Modeling Rules*. 2nd ed. Houston, TX: Business Rule Solutions, Inc., 1997. Originally published as *The Business Rule Book (1st Ed.)*, 1994. Available from <http://www.BRSolutions.com>

[Ross2003] \_\_\_\_\_. *Principles of the Business Rule Approach*. Boston, MA: Addison-Wesley, 2003. Available from <http://www.BRSolutions.com>

[Ross2005] \_\_\_\_\_. *Business Rule Concepts: Getting to the Point of Knowledge*. 2nd ed.: Business Rule Solutions, LLC, 2005. Available from <http://www.BRSolutions.com>

[RuleSpeak] Business Rule Solutions. *BRS RuleSpeak® Practitioner's Kit*. Business Rule Solutions, LLC, 2001-2004. PDF. Available from [http://BRSolutions.com/p\\_rulespeak.php](http://BRSolutions.com/p_rulespeak.php)

[SEP] *Stanford Encyclopedia of Philosophy*. Edward N. Zalta, ed. The Metaphysics Research Lab, Center for the Study of Language and Information, Stanford University. Available from <http://plato.stanford.edu/>

[SOED] *Shorter Oxford Dictionary of English*.

[SubePLTS (or PLTS)] Suber, Peter. *Propositional Logic Terms and Symbols*. Philosophy Department, Earlham College, 1997. Available from <http://www.earlham.edu/~peters/courses/log/terms2.htm>

[SubeGFOL (or GFOL)] \_\_\_\_\_. *Glossary of First-Order Logic*. Philosophy Department, Earlham College, 1999-2002. Available from <http://www.earlham.edu/~peters/courses/logsys/glossary.htm>

[UML2infr] Object Management Group (OMG). *Unified Modeling Language: Infrastructure*. Ver. 2.0: OMG.

[Unicode4] "The Unicode Standard, Version 4.0.0." In *The Unicode Standard, Version 4.0*. Boston, MA: Addison-Wesley, 2003. Available from <http://www.unicode.org/versions/Unicode4.0.0/b1.pdf>

[USG] The Unicode Consortium. *Glossary of Unicode Terms*. 1991-205. Updated Nov. 17 2004. Available from <http://www.unicode.org/glossary/>

[W3ID] *Webster's 3rd New International Dictionary*.

[WD] *Webster's Dictionary*.

[XMI2.1] *XML Metadata Interchange (XMI)*. Ver. 2.1: OMG.

