Date: Thursday 28th August, 2014

Ontology, Model, and Specification Integration and Interoperability (OntoIOp)

Version 0.6

OMG Document Number:
Normative reference:
Machine readable files(s):
Normative:
Non-normative:

Copyright ©2014, Object Management Group, Inc.
Copyright ©2008, company name
Copyright ©2008, company name

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, 140 Kendrick Street, Needham, MA 02494, U.S.A.

TRADEMARKS

MDA®, Model Driven Architecture®, UML®, UML Cube logo®, OMG Logo®, CORBA® and XMI® are registered trademarks of the Object Management Group, Inc., and Object Management Group™, OMG™, Unified Modeling Language™, Model Driven Architecture Logo™, Model Driven Architecture Diagram™, CORBA logos™, XMI Logo™, CWM™, CWM Logo™, IIOP™, IMM™, MOF™, OMG Interface Definition Language (IDL)™, and OMG SysML™ 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

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. All OMG Specifications are available from the OMG website at:

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

Specifications are organized by the following categories:

- Business Modeling Specifications
- Middleware Specifications
  - CORBA/IIOP
  - Data Distribution Services
  - Specialized CORBA
- IDL/Language Mapping Specifications
- Modeling and Metadata Specifications
  - UML, MOF, CWM, XMI
  - UML Profile
- Modernization Specifications
- Platform Independent Model (PIM), Platform Specific Model (PSM), Interface Specifications
  - CORBAServices
  - CORBAFacilities

- OMG Domain Specifications
- CORBA Embedded Intelligence Specifications
- CORBA 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. 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`

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

Typographical Conventions

The type styles shown below are used in this document to distinguish programming statements from ordinary English. However, these conventions are not used in tables or section headings where no distinction is necessary.

Times/Times New Roman - 10 pt.: Standard body text

Helvetica/Arial - 10 pt. Bold: OMG Interface Definition Language (OMG IDL) and syntax elements.

`Courier - 10 pt. Bold:` Programming language elements.

Helvetica/Arial - 10 pt.: Exceptions

NOTE: Italic text represents names defined in the specification or the name of a document, specification, or other publication.

Issues

The reader is encouraged to report any technical or editing issues/problems with this specification to `http://www.omg.org/report_issue.htm`.

1. Scope

This OMG Specification specifies the Distributed Ontology, Modeling and Specification Language (DOL) designed to achieve integration and interoperability of ontologies, specifications and models (OMS for short). DOL is a language for distributed knowledge representation across multiple OMS, particularly OMS that have been formalized in different OMS languages. This OMG Specification responds to the OntoIOp Request for Proposals [**?**].

1.1. Background Information

Logical languages are used in several fields of computing for the development of formal, machine-processable texts that carry a formal semantics. Among those fields are 1) **O**ntologies formalizing domain knowledge, 2) (formal) **M**odels of systems, and 3) the formal **S**pecification of systems. Ontologies, models and specifications will (for the purpose of this document) henceforth be abbreviated as **OMS**, if all three can be treated in the same way.

An OMS provides formal descriptions the scope of which ranges from domain knowledge and activities (ontologies, models) to properties and behaviours of hardware and software systems (models, specifications). These formal descriptions can be used for the analysis and verification of domain models, system models and systems, using rigorous and effective reasoning tools. Since these models and systems become increasingly complex, usually a monolithic description is not feasible. Instead, different viewpoints on one domain or systems are modeled. Hence, interoperability becomes a crucial issue, in particular, formal interoperability, i.e. interoperability that is based on the formal semantics of the different viewpoints. Interoperability is both about interfacibility, enabling the use of several OMS in a common application scenario, as well as about coherence and consistency, ensuring at an early stage of the development that a coherent system can be reached.

In complex applications, which involve multiple OMS with overlapping concept spaces, data mapping may also be between different OMS, and is then called OMS alignment. While OMS alignment is most commonly studied for OMS formalized[1] in the same OMS language, Note(1) the different OMS used by complex applications may also be written in different OMS languages even of different expressiveness. This OMG Specification faces this diversity not by proposing yet another OMS language that would subsume all the others. Instead, it accepts the diverse reality and formulates means (on a sound and formal semantic basis) to compare and integrate OMS that are written in different formalisms. It specifies DOL (Distributed Ontology, Modeling and Specification Language), a formal language for expressing not only OMS but also mappings between OMS formalized in different OMS languages.

Thus, DOL gives interoperability a formal grounding and makes heterogeneous OMS and services based on them amenable to checking of coherence (e.g. consistency, conservativity, intended consequences, and compliance).

[2] Note(2)

---

[1] Note: spell-check everything for American English.

[2] Note: State somewhere: A DOL theory is at level M1 in MDA speak, the same level as an UML

1.2. Features within Scope

The following are within the scope of this OMG Specification:

1. heterogeneous OMS that combine parts written in different languages
2. mappings between (possibly structured and/or heterogeneous) OMS (mapping OMS symbols to OMS symbols)
3. translations between different OMS languages conformant with DOL (translating whole OMS to another language)
4. annotation and documentation of OMS, mappings between OMS, symbols, and sentences
5. recommendations of vocabularies for annotating and documenting OMS
6. a syntax for embedding the constructs mentioned under (1)–(4) as annotations into existing OMS
7. a syntax for expressing (1)–(3) as standoff markup that points into existing OMS
8. a formal semantics of (1)–(3)
9. criteria for existing or future OMS languages to conform with DOL

The following are outside the scope of this OMG Specification:

1. the (re)definition of elementary OMS languages, i.e. languages that allow the declaration of OMS symbols (non-logical symbols) and stating sentences about them
2. algorithms for obtaining mappings between OMS
3. concrete OMS and their conceptualization and application
4. mappings between services and devices, and definitions of service and device interoperability.

This OMG Specification describes the syntax and the semantics of the Distributed Ontology, Modeling and Specification Language (DOL) by defining an abstract syntax and an associated model-theoretic semantics for DOL.

---

diagram. Also a DOL distributed OMS, containing interpretations, alignments between OMS etc. is at level M1. The specification of DOL lives at M2, like the UML specification. Both are written in MOF.

2. Conformance

This clause defines conformance criteria for languages and logics that can be used with the distributed ontology, modeling and specification language DOL, as well as conformance criteria for serializations, translations and applications. This OMG Specification describes the conformance with DOL of a number of OMS languages, namely OWL 2, Common Logic, RDF and RDFS, as well as translations among these, in its informative annexes.

It is expected that DOL will be used for more languages than this normative set of DOL-conformant languages. There will be a **registry for DOL-conformant languages and translations** hosted at `http://ontohub.org`. This will ensure that this OMG Specification remains interoperable with past, present and even future OMS languages. The registry shall also include descriptions of DOL-conformant languages and translations (as well as other information needed by implementors and users) in machine-processable form.

There will be Maintenance Authority (MA)[1] established to maintain the registry as an informative resource governed by the standard. The registry contents itself will not be normative; however, it is expected to become the basis for normative activities.

2.1. Conformance of an OMS language/a logic with DOL

Rationale: for an OMS language to conform with DOL,

- its logical language aspect either needs to satisfy certain criteria about its abstract syntax or formal semantics itself, or there must be a translation (again satisfying certain criteria) to a language that already is DOL-conforming.
- its structuring language aspect (if present) must not conflict with DOL's own structuring mechanisms
- its annotation language aspect must not conflict with DOL's meta-language constructs.

We also define different conformance levels w.r.t. the usage of IRIs as identifiers for all kinds of entities that the OMS language supports.

An OMS language is conformant with DOL if it satisfies the following conditions:

1. its abstract syntax specified as an SMOF compliant meta model or as an EBNF grammar;
2. its logical language aspect (for expressing basic OMS) is conformant, and in particular has a semantics (see below),
3. it has at least one serialization in the sense of section 2.2;
4. either there exists a translation of it into a conformant language[2], or:

---

[1] or, depending on advisability, a Registration Authority

[2] For example, consider the translation of OBO1.4 to OWL, giving a formal semantics to OBO1.4).

a) the logical language aspect (for expressing basic OMS) is conformant, and in particular has a semantics (see below),

b) the structuring language aspect (for expressing structured OMS and relations between those) is conformant (see below), and

c) the annotation language aspect (for expressing comments and annotations) is conformant (see below).

The *logical language aspect* of an OMS language is conformant with DOL if each logic corresponding to a profile (including the logic corresponding to the whole logical language aspect) is presented as an institution [10]. [3] Note that one OMS language can have several sublanguages or profiles corresponding to several logics (for example, OWL 2 has profiles EL, RL and QL, apart from the whole OWL 2 itself).

The *structuring language aspect* of an OMS language is conformant with DOL if it can be mapped to DOL's structuring language in a semantics-preserving way. The structuring language aspect **may** be empty.

The *annotation language aspect* of an OMS language is conformant with DOL if its constructs have no impact on the semantics. The annotation language aspect **shall** be non-empty; it **shall** provide the facility to express comments.

[3] Note(3)

We define the following levels of conformance of the abstract syntax of a basic OMS language with DOL, listed from highest to lowest:

Full IRI conformance The abstract syntax enforces that IRIs be used for identifying all symbols and entities.

No mandatory use of IRIs The abstract syntax does not enforce that IRIs be used for identifying all entities. Note that this includes the case of optionally supporting IRIs without enforcing their use (such as in Common Logic).

Any conforming language and logic shall have a machine-processable description as detailed in clause 2.3.

2.1.1. Conformance of language/logic translations with DOL

Rationale: a translation between logics must satisfy certain criteria in order to conform with DOL. Also, a translation between OMS languages based on such logics must be consistent with the translation between these logics. Translations should not break structuring language aspects nor comments/annotations either.

A logic translation is conformant with DOL if it is presented either as an institution morphism or as an institution comorphism.

A language translation is conformant with DOL if it is a mapping between the abstract syntaxes that restricts to a conformant logic translation when restricted to the logical language

---

[3] Institutions are necessarily monotonic; conformance criteria for non-monotonic logics are still under development. However, minimization provides non-monotonic reasoning in DOL. A further possibility to include non-monotonic logics is to construe entailments between formulas as sentences of the institution.

[3] NOTE: say something about "infrastructure theories", i.e. axiomatizations of one logic in another logic. Providers of OMS language translations MAY also provide these (given that the translation is theoroidal). Note the possible trade-off between readability and theorem proving complexity (as the infrastructure axioms may be complex) – so maybe we should encourage multiple alternative translations to co-exist.

aspect. Language translations **may** also translate the structuring language aspect, in this case, they **shall** preserve the semantics of the structuring language aspect. Furthermore, language translations **should** preserve comments and annotations. All comments attached to a sentence (or symbol) in the source **should** be attached to its translation in the target (if there are more than one sentences (resp. symbols) expressing the translation, to at least one of them).

2.2. Conformance of a serialization of an OMS language with DOL

Rationale: The main reason for the following specifications is identifier injection. DOL is capable of assigning identifiers to entities (symbols, axioms, modules, etc.) inside fragments of OMS languages that occur in a DOL document, even if that OMS language doesn't support such identifiers by its own means. Such identifiers will be visible to a DOL tool, but not to a tool that only supports the OMS language. To achieve this without breaking the formal semantics of that OMS language, we make use of annotation or commenting features that the OMS language supports, in order to place such identifiers inside annotations/comments. Depending on the nature of the concrete given serialization of the OMS language, be it plain text, some serialization of RDF, XML, or some other structured text format, we can be more specific about what the annotation/commenting facilities of that serialization must look like in order to support this identifier injection. Well-behaved XML and RDF schemas support identifier injection in a "nice" way (rather than using text-level comments). In the worst case we can't inject anything into an OMS language fragment, because the OMS language serialization simply wouldn't allow us to write suitable comments, but we'd have to point into it from the outer space by using standoff markup.

Further conformance criteria in this section are introduced to facilitate the convenient reuse of verbatim fragments of OMS language inside a DOL document.

Independently from these criteria, we distinguish different levels of conformance of a serialization w.r.t. its means of conveniently abbreviating long IRI identifiers.

We define four levels of conformance of a serialization of an OMS language with DOL.

XMI conformance An XMI serialization has been automatically derived from the SMOF specification of the abstract syntax, using MOF 2 XMI Mapping.[4] Note(4)

XML conformance The given serialization has to be specified as an XML schema, which satisfies all of the following conditions:

- The elements of the schema belong to one or more non-empty XML namespaces.[5] Note(5)
- The schema shall not forbid attributes from foreign namespaces (here: the DOL namespace) on any elements[6][7] Note(6) Note(7)

---

[4] NOTE: Christoph to all: I'm not sure how MOF and XMI works, i.e. how to inject identifiers into comments there.

[5] NOTE: FYI: That means that in a heterogeneous OMS we can recognize that a sentence is, e.g., stated in OWL, without explicitly "tagging" it as "OWL" (which we would have to do in the case of a serialization that is merely text conformant).

[6] NOTE: Christoph (2014-03-26): the rationale is that in an XML serialization we wouldn't want to inject identifiers into completely unstructured XML comments (<!– ... –>) but rather into well-structured attributes from some DOL namespace to-be-defined (think of <axiom dol:id="foo-ax">)

[7] NOTE: Maybe we also need child elements from different namespaces?

RDF conformance The given serialization has to be specified as an RDF vocabulary, which satisfies all of the following conditions:

- The elements of the vocabulary belong to one or more RDF namespaces identified by absolute URIs.
- [8]The serialization shall specify ways of giving IRIs or URIs to all structural elements of an OMS.[9] Note(8) Note(9)
- There shall be no additional rules that forbid properties from foreign namespaces (here in particular: the annotation vocabularies recommended by DOL[10]) to be stated about arbitrary subjects.[11] Note(10) Note(11)

Text conformance The given serialization has to satisfy all of the following conditions:

- The serialization conforms with the requirements for the *text/plain* media type specified in IETF/RFC 2046, section 4.1.3.
- The serialization shall provide a designated comment construct that can be placed sufficiently flexible as to be uniquely associated with any non-comment construct of the language. That means, for example, one of the following:
  - The serialization provides a construct that indicates the start and end of a comment and may be placed before/after each token that represents a structural element of an OMS.
  - The serialization provides line-based comments (ranging from an indicated position to the end of a line) but at the same time allows the flexible placement of line breaks before/after each token that represents a structural element of an OMS.

Standoff markup conformance An OMS language is standoff markup conformant with DOL if one of its serializations conforms with the requirements for the *text/plain* media type specified in IETF/RFC 2046, section 4.1.3. Note that conformance with *text/plain* is a prerequisite for using, for example, fragment URIs in the style of IETF/RFC 5147 for identifying text ranges.

[12] Note(12)

Independently from the conformance levels given above, there is the following hierarchy of conformance w.r.t. CURIEs as a means of abbreviating IRIs, listed from highest to lowest:

Prefixed CURIE conformance The given serialization allows non-logical symbol identifiers to have the syntactic form of a CURIE, or any subset of the CURIE grammar that allows named prefixes (`prefix:reference`). The serialization is **not required** to support CURIEs with no prefix.

Informative comment: In this case, a prefix map with multiple prefixes **may** be used to

---

[8] NOTE: Christoph (2014-03-26): The rationale is that RDF in principle allows for identifying everything, so an RDF-based serialization of an OMS language should not forbid making use of such RDF constructs that do allow for identifying arbitrary things.

[9] NOTE: Q-AUT: And what if it doesn't? e.g. OWL doesn't specify IRIs for import declarations, so we can, e.g., not annotate them when using the RDF serialization of OWL. We could only do it via RDF reification, or by using an XML serialization.

[10] NOTE: Christoph (2014-03-26): I think we are no longer explicity recommending annotation vocabularies such as OMV, but nevertheless an RDF serialization of an OMS language must allow annotations to things, using RDF properties that do not belong to the RDF vocabulary of the OMS language. It should treat them like OWL treats annotation properties, i.e. as not changing the formal semantics.

[11] NOTE: FYI: No well-behaved RDF vocabulary would do so, but we'd better be safe.

[12] NOTE: FYI: The latter two seem trivial, but we need them to rule out ad hoc diagrams drawn on a napkin

map the non-logical symbol identifiers of a basic OMS to IRIs in multiple namespaces (cf. clause 9.5.3)

Unprefixed names only The given serialization only supports CURIEs with no prefix, or any subset of the grammar of the `REFERENCE` nonterminal in the CURIE grammar. Informative comment: In this case, a binding for the empty prefix **has to** be declared, as this is the only possibility of mapping the identifiers of the basic OMS to IRIs, which are located in one flat namespace.

CURIEs that have a prefix may not be acceptable identifiers in every serialization of a basic OMS language, as the standard CURIE separator character, the colon (`:`), may not be allowed in identifiers. [13] Therefore, the declaration of DOL-conformance of the respective serialization (cf. clause 2.2) **may** define an *alternative CURIE separator character*, or it **may** forbid the use of prefixed CURIEs altogether. Note(13)

Any conforming serialization of an OMS language shall have a machine-processable description as detailed in clause 2.3.

2.3. Machine-processable description of conforming languages, logics, and serializations

Rationale: When a parser processes a DOL OMS found somewhere, which refers to modules in OMS languages, or includes them verbatim, the parser needs to know what language to expect; further DOL-supporting software needs to know, e.g., what other DOL-conforming languages the module in the given OMS language can be translated to. Therefore we require that all languages/logics/serializations that conform with DOL describe themselves in a machine-comprehensible way.

For any conforming OMS language, logic, and serialization of an OMS language, it is required that it be assigned an HTTP IRI, by which it can be identified. It is also required that a machine-processable description of this language/logic/serialization be retrievable by dereferencing this IRI, according to the linked data principles. At least there has to be an RDF description in terms of the vocabulary specified in annex C, which has to be made available in the RDF/XML serialization when a client requests content of the MIME type *application/rdf+xml*. Descriptions of the language/logic/serialization in further representations, having different content types, may be provided.[14] Note(14)

2.4. Conformance of a document with DOL

Rationale: for exchanging DOL documents with other users/tools, nothing that has a formal semantics must be left implicit. One DOL tool may assume that by default any OMS fragments inside a DOL document are in some fixed OMS language unless specified otherwise, but another DOL tool can't be assumed to understand such DOL documents. Defaults are, however, practically convenient, which is the reason for having the following section about the conformance of an *application*.

---

[13] NOTE: Q-ALL: I recall that in the 2012-04-18 teleconference we agreed on this – but does it really make sense? Are there any relevant OMS language serializations that do not allow : in identifiers (or that do allow it theoretically but discourage it in practice) but allow some other non-letter character?

[14] NOTE: FYI: that opens the door for, e.g., OMDoc

A document conforms with DOL if it contains a DOL text that is well-formed according to the grammar. That means, in particular, that any information related to logics has to be made explicit (as foreseen by the DOL abstract syntax specified in clause 9), such as:

- the logic of each OMS that is part of the DOL document,
- the translation that is employed between two logics (unless it is one of the default translations specified in annex G)

However, details about aspects of an OMS that do not have a formal, logic-based semantics, may be left implicit. For example, a conforming document may omit explicit references to matching algorithms that have been employed in obtaining an alignment.

2.5. Conformance of an application with DOL

In practice, DOL-aware *applications* may also deal with documents that are not conforming with DOL according to the criteria established in clause 2.4. However, an application only *conforms* with DOL if it is capable of producing DOL-conforming documents as its output when requested.

We expect most DOL-aware applications to support a fixed (possibly extensible) set of OMS languages conforming with DOL. It is, for example, possible that a DOL-aware application only supports OWL and Common Logic. In that case, the application may process documents that mix OWL and Common Logic ontologies *without* explicitly declaring the respective logics, as the respective syntaxes of OWL and Common Logic can be distinguished by examining the different keywords. However, for DOL conformance, that application has to be capable of exporting documents with explicit references to the logics used.

[15] Note(15)

[16] Note(16)

---

[15] NOTE: applications need to strip DOL annotations from embedded fragments in other OMS languages

[16] NOTE: applications need to be able to expand CURIEs into IRIs when requested

3. Normative References

1. W3C/TR REC-owl2-syntax:2009OWL 2 Web Ontology Language: Structural Specification and Functional-Style Syntax. W3C Recommendation, 27 October 2009. `http://www.w3.org/TR/2009/REC-owl2-syntax-20091027/`
2. ISO/IEC 14977:1996Information technology – Syntactic metalanguage – Extended BNF
3. W3C/TR REC-xml:2008Extensible Markup Language (XML) 1.0 (Fifth Edition). W3C Recommendation, 26 November 2008. `http://www.w3.org/TR/2008/REC-xml-20081126/`
4. W3C/TR REC-owl2-profiles:2009OWL 2 Web Ontology Language: Profiles. W3C Recommendation, 27 October 2009. `http://www.w3.org/TR/2009/REC-owl2-profiles-20091027/`
5. ISO/IEC 24707:2007Information technology – Common Logic (CL): a framework for a family of logic-based languages
6. OMG Document ptc/2010-11-14:OMG Unified Modeling Language™ (OMG UML), Superstructure, Version 2.4. `http://www.omg.org/spec/UML/2.4/Superstructure.` Section 7 (Classes)
7. IETF/RFC 3986Uniform Resource Identifier (URI): Generic Syntax. January 2005. `http://tools.ietf.org/html/rfc3986`
8. IETF/RFC 3987Internationalized Resource Identifiers (IRIs). January 2005. `http://tools.ietf.org/html/rfc3987`
9. IETF/RFC 5147URI Fragment Identifiers for the text/plain Media Type. April 2008. `http://tools.ietf.org/html/rfc5147`
10. W3C/TR REC-rdf-concepts:2004Resource Description Framework (RDF): Concepts and Abstract Syntax. W3C Recommendation, 02 February 2004. `http://www.w3.org/TR/2004/REC-rdf-concepts-20040210/`
11. W3C/TR REC-xml-names:2009Namespaces in XML 1.0 (Third Edition). W3C Recommendation, 8 December 2009. `http://www.w3.org/TR/2009/REC-xml-names-20091208/`
12. W3C/TR REC-rdfa-core-20120607RDFa Core 1.1. Syntax and processing rules for embedding RDF through attributes. W3C Recommendation, 07 June 2012. `http://www.w3.org/TR/2012/REC-rdfa-core-20120607/`
13. ISO/IEC 10646Information technology – Universal Multiple-Octet coded Character Set (UCS)
14. W3C/TR REC-rdf-schema:2004RDF Vocabulary Description Language 1.0: RDF Schema. W3C Recommendation, 10 February 2004. `http://www.w3.org/TR/2004/REC-rdf-schema-20040210/`
15. W3C/TR REC-rdf-mt:2004RDF Semantics. W3C Recommendation, 02 February 2004. `http://www.w3.org/TR/2004/REC-rdf-mt-20040210/`

[17] [18] [19]

Note(17)

Note(18)

Note(19)

---

[17] Note: more, see RFP

[18] Note: introduce a separate reference scheme for normative references

[19] Note: Q-ALL: I have listed them roughly in the order of occurrence: OK?

4. Terms and Definitions

[20] Note(20)

For the purposes of this document, the following terms and definitions apply.

4.1. Distributed OMS and the Distributed Ontology, Modeling and Specification Language

distributed OMS,
hyperontology

collection of named *OMS,* possibly written in different *OMS languages*, linked by named *OMS mappings* and named *OMS graphs*

distributed ontology, modeling and specification language,
DOL

language for formalizing *distributed OMS*, whose syntax and semantics are specified in this OMG Specification

NOTE When viewed as an *OMS language*, DOL has *OMS* as its *non-logical symbols*, and *OMS mappings* as its *sentences.*

4.2. OMS

OMS (ontology, specification or model)

set of expressions (like *non-logical symbols*, *sentences* and *structuring* elements) in a given *OMS language* (or several such languages)
NOTE An *OMS* can be written in different *OMS language serializations.*

OMS language

language equipped with a formal, declarative, logic-based semantics, plus non-logical *annotations*
NOTE An OMS language is used for the formal specification of *OMS.*
EXAMPLE OMS languages include OWL, Common Logic, F-logic, UML class diagrams, RDFS, and OBO. [21] Note(21)

---

[20] NOTE: OMG specifications shall not contain glossaries, hence always refer to this section if definitions of terms are needed.

[21] NOTE: Are query languages, like SPARQL, considered by OntoIOp to be OMS languages? That is an interesting point. I tend to say "yes", because we definitely want to have them in. However, query

non-logical symbol,
OMS symbol

atomic expression or syntactic constituent of an *OMS* that requires an interpretation through a *model*

NOTE The notion of "atomic sentence" used in logic is different, it usually may involve several non-logical symbols.

EXAMPLE Non-logical symbols in OWL W3C/TR REC-owl2-syntax:2009 (there called "entities") comprise

- individuals (denoting objects from the domain of discourse),
- classes (denoting sets of objects; also called concepts), and
- properties (denoting binary relations over objects; also called roles).

This is opposed to logical symbols in OWL, e.g. those for intersection and union of classes.
EXAMPLE Non-logical symbols in Common Logic ISO/IEC 24707:2007 comprise

- names (denoting objects from the domain of discourse),
- sequence markers (denoting sequences of objects).

This is opposed to logical symbols in Common Logic, e.g. logical connectives and quantifiers.

vocabulary,
signature

set (or otherwise structured entity) of *non-logical symbols* of an *OMS*
NOTE The signature of a term is the set of all non-logical symbols occuring in the *term*. The signature of an *OMS language* is the set of all non-logical symbols possible in that language.
NOTE The signature of an OMS is usually uniquely determined.

model

semantic interpretation of all *non-logical symbols* of a *signature*
NOTE A model of an OMS is a model of the signature of the OMS that moreover *satisfies* all the *axioms* of the OMS.
NOTE This term is not to be confused with model in the sense of modeling (i.e., the "M" in OMS).

term

syntactic expression either consisting of a single *non-logical symbol* or recursively composed of other terms (a.k.a. its subterms)

---

languages do not fit exactly into the scheme of the OntoIOp notion of OMS language (and the underlying notion of institution), because the latter is about satisfaction of sentences in models, whereas a query language is about computing answer substitutions to queries. However, there is research about how to relate and reconcile both (also in the institution community). I think we should devote an OntoIOp telecon to this topic.

sentence

term that is either true or false in a given *model*, i.e. which is assigned a truth value in this *model*.[22] Note(22)

NOTE In a *model*, on the one hand, a sentence is always true or false. In an *OMS*, on the other hand, a sentence can have several logical statuses: it can be an axiom, if postulated to be true; a theorem, if proven from other axioms and theorems; a conjecture, if expecting to be proven from other axioms and theorems; or have another of many possible statuses.

NOTE A sentence can conform to one or more signatures (namely those signatures containing all non-logical symbols used in the sentence).

NOTE It is quite common that sentences are required to be closed (i.e. have no free variables). However, this depends on the OMS language at hand.

axiom

sentence postulated to be valid (i.e. true in every *model*)

theorem

sentence that has been proven from other *axiom*s and *theorem*s

satisfaction relation

relation between models and sentences indicating which sentences hold true in the model

query

sentence containing *query variables* that can be instantiated by a *substitution*

query variable

symbol that will be used in a *query* and a *substitution*

NOTE From an abstract point of view, query variables are just symbols; they are used in a way that they will be substituted using a substitution. Many OMS languages have special notations for (query) variables.

NOTE Usually, query variables are the free variables of a sentence, whereas there can be other (bound) variables.

NOTE If there are no variables in an OMS language, constants can be used as query variables.

substitution

logical OMS mapping that maps *query variables* of one *OMS* to complex *terms* of another OMS

answer substitution

substitution that, when applied to a given *query*, turns the latter into a logical consequence of a given *OMS* [23] Note(23)

---

[22] NOTE: FYI: From Common Logic, I changed “unit of logical text” to “term”.

[23] NOTE: we have to revisit this once we design the abstract syntax of queries etc.

4.3. Semantic Web

resourceweb

something that can be globally identified
NOTE IETF/RFC 3986:2005, Section 1.1 deliberately defines a resource as "in a general sense [...] whatever might be identified by [an IRI]". The original source refers to URIs, but DOL uses the compatible IRI standard IETF/RFC 3987:2005 for identification.
EXAMPLE Familiar examples include an electronic document, an image, a source of information with a consistent purpose (e.g., "today's weather report for Los Angeles"), a service (e.g., an HTTP-to-SMS gateway), and a collection of other resources. A resource is not necessarily accessible via the Internet; e.g., human beings, corporations, and bound books in a library can also be resources. Likewise, abstract concepts can be resources, such as the operators and operands of a mathematical equation, the types of a relationship (e.g., "parent" or "employee"), or numeric values (e.g., zero, one, and infinity). IETF/RFC 3986:2005, Section 1.1

element (of an OMS)

any *resource* in an *OMS* (e.g. a *non-logical symbol*, a *sentence*, a *correspondence*, the *OMS* itself, ...) or a named set of such *resources*.

linked data

structured data that is published on the Web in a machine-processable way, according to principles specified in [28, 5]
NOTE The linked data principles (adapted from [28] and its paraphrase at [40]) are the following:

1. Use IRIs as names for things.
2. Use HTTP IRIs so that these things can be referred to and looked up ("dereferenced") by people and user agents.[1]
3. Provide useful machine-processable (plus optionally human-readable) information about the thing when its IRI is dereferenced, using standard formats.
4. Include links to other, related IRIs in the exposed data to improve discovery of other related information on the Web.

NOTE RDF, serialized as RDF/XML [19], is the most common format for publishing linked data. However, its usage is not mandatory.
NOTE Using HTTP content negotiation [16] it is possible to serve representations in different formats from the same URL.

---

[1] I.e., the IRI is treated as a URL (uniform resource locator).

4.4. OMS Annotation and Documentation

annotation

additional information without a logical semantics that is attached to an *element* of an *OMS*
NOTE Formally, an annotation is given as a (subject, predicate, object) triple as defined by SOURCE: W3C/TR REC-rdf-concepts:2004, Section 6. The subject of an annotation is an *element* of an OMS. The predicate is an RDF property defined in an external OMS and describes in what way the annotation object is related to the annotation subject.
NOTE According to note 4.4 it is possible to interpret annotations under an RDF semantics. "Without a logical semantics" in this definition means that annotations to an OMS are not considered sentences of that OMS.

OMS documentation

set of all *annotations* to an *OMS*, plus any other documents and explanatory comments generated during the entire OMS building process
NOTE Adapted from [39]

4.5. Structured OMS

basic OMS

signature equipped with a set of *sentences* and *annotations*, which may be used as a building block for a larger *OMS*
NOTE The sentences must use only those non-logical symbols that are present in the signature.

structured OMS

OMS that results from other *OMS* by *import*, *union*, *combination*, *renaming* or other structuring operations

subOMS

OMS whose sets of *non-logical symbols* and *sentences* are subsets of those present in a given larger *OMS*

extension

OMS whose sets of *non-logical symbols* and *sentences* are supersets of those present in a given smaller *OMS*

consequence-theoretic conservative extension

extension that does not add new *theorems* (in terms of the unextended *signature*)
NOTE An extension O_2 of an OMS O_1 is a consequence-theoretic conservative extension, if all properties formulated in the signature of O_1 hold for O_1 whenever they hold for O_2.

model-theoretic conservative extension

extension that does not lead to a restriction of class of *model*s of an *OMS*
NOTE An extension O_2 of an OMS O_1 is a model-theoretic conservative extension, if all properties formulated in the signature of O_1 hold for O_1 whenever they hold for O_2.
NOTE Any model-theoretic conservative extension is also a consequence-theoretic one.

conservative extension

consequence-theoretic or *model-theoretic conservative extension*
NOTE If used without qualification, the consequence-theoretic version is meant.

monomorphic extension

extension whose newly introduced *non-logical symbols* are interpreted in a way unique up to isomorphism
NOTE An *extension* O_2 of an *OMS* O_1 is a monomorphic extension, if each model of O_1 can be expanded to a model of O_2 that is unique up to isomorphism.
NOTE Each monomorphic extension is also a model-theoretic conservative extension but not vice versa.

definitional extension

extension whose newly introduced *non-logical symbols* are interpreted in a unique way
NOTE An *extension* O_2 of an *OMS* O_1 is a definitional extension, if each model of O_1 can be uniquely expanded to a model of O_2.
NOTE O_2 being a definitional extension of O_1 implies a bijective correspondence between the classes of models of O_2 and O_1.
NOTE Each definitional extension is also a monomorphic extension but not vice versa.

weak definitional extension

extension whose newly introduced *non-logical symbols* can be interpreted in at most one way
NOTE An *extension* O_2 of an *OMS* O_1 is a weak definitional extension, if each model of O_1 can be expanded to at most one model of O_2.
NOTE An extension is definitional if and only if it is both weakly definitional and model-theoretically conservative.

implied extension

model-theoretic conservative extension that does not introduce new *non-logical symbols*
NOTE A *conservative extension* O_2 of an *OMS* O_1 is an implied extension, if and only if the signature of O_2 is the signature of O_1. O_2 is an implied extension of O_1 if and only if the model class of O_2 is the model class of O_1.
NOTE Each implied extension is also a definitional extension but not vice versa.

module

subOMS that *conservatively extends to* the whole *OMS*
NOTE The conservative extension can be either model-theoretic or consequence-theoretic;

without qualification, the consequence-theoretic version is used. [24] Note(24)

module extraction

activity of obtaining from an *OMS* concrete *modules* to be used for a particular purpose (e.g. to contain a particular sub-*signature* of the original *OMS*)
NOTE Cited and slightly adapted from [39]
NOTE The goal of module extraction is "decomposing an OMS into smaller, more manageable modules with appropriate dependencies" [38]

EXAMPLE Consider an OWL DL ontology about wines, from which we would like to extract a module about white wines. That module would contain the declaration of the non-logical symbol "white wine", all declarations of non-logical symbols related to "white wine", and all sentences about all of these non-logical symbols.

approximant

approximation (in the sense of a logically implied theory, possibly after suitable translation) of an OMS in a smaller *signature* or *OMS language*

maximum approximant

best possible (in the sense of a maximum set of logical consequences) *approximant* of an OMS in a smaller *signature* or *OMS language*
NOTE Technically, a maximum approximant is a uniform interpolant, see [31].

closed world assumption

presumption that what is not known to be true, is false

minimization, circumscription

way of implementing the *closed world assumption* by restricting the *models* to those that are minimal
NOTE See [33], [29].

4.6. Mappings Between OMS

correspondence

relationship between an *non-logical symbol* e_1 from an *OMS* O_1 and an *non-logical symbol* e_2 from an OMS O_2, or between an *non-logical symbol* e_1 from O_1 and a *term* t_2 formed from *non-logical symbols* from O_2
NOTE A correspondence is given as a quadruple $(e_1, R, \left\{ \begin{array}{c} e_2 \\ t_2 \end{array} \right\}, c)$, where R denotes the type of relationship that is asserted to hold between the two non-logical symbols/terms, and $0 \leq c \leq 1$ is a confidence value. R and c may be omitted: When R is omitted, it is implied

---

[24] NOTE: this is about coverage only. Should we also care about safety?

from the context ("equivalence" for alignments, and "equality" for logical OMS mappings)[25] ; Note(25) when c is omitted, it defaults to 1.

NOTE A confidence value of 1 does not imply logical equivalence (cf. [27] for a worked-out example).

OMS mapping , linkOMS[26]

Note(26)

relationship between two *OMS*, typically given as a set of *correspondences*

logical OMS mapping

OMS mapping that has a formal, logic-based semantics
NOTE Logical OMS mappings are given as sets of correspondences, which are required to be *signature morphisms.*
NOTE Some specific kinds of logical OMS mappings will be introduced below.

interpretation, view

logical OMS mapping that postulates a relation between two *OMS*
NOTE An interpretation typically leads to proof obligations, i.e. one has to prove that axioms of the source OMS of the mapping are theorems in the target OMS.
NOTE When an interpretation is given as a set of correspondences, these are given as tuples, where the type of relationship is given by the specific kind of interpretation.

equivalence

logical OMS mapping ensuring that two *OMS* share the same definable concepts
NOTE Two OMS are equivalent if they have a common definitional extension. The OMS may be written in different OMS languages.

interface signature

signature mediating between an *OMS* and a *module* of that *OMS* in the sense that it contains those *non-logical symbols* that the *sentences* of the *module* and the *sentences* of the *OMS* have in common
NOTE Adapted from [15]

module relation

logical OMS mapping stating that one *OMS* is a *module* of the other one.

---

[25] NOTE: Q-AUT: For interpretations that is the only viable way, but for alignments? Is there any reasonable "implied default", or should we let R default to something like owl:sameAs?

[26] NOTE: Q-ALL: Is this the correct way of stating that I mean "the term link, when used in the context of ontologies"?

import

logical OMS mapping between two *OMS* such that one OMS behaves as if it were included into the other
NOTE Semantically, an import of O_2 into O_1 is equivalent to the verbatim inclusion of O_2 in place of the import declaration
NOTE The purpose of O_2 importing O_1 is to make non-logical symbols and sentences of O_1 available in O_2.
NOTE Importing O_1 into O_2 turns O_2 into an extension of O_1.
NOTE An owl:import in OWL is an import.

renaming

assignment of new names to some *non-logical symbols* of an *OMS*
NOTE A renaming results in a *logical OMS mapping* between the original and the renamed OMS.

reduction

logical OMS mapping reducing an *OMS* to a smaller *signature*

alignment

flexible, relational *OMS mapping* that does not always have a formal, logic-based semantics

matching

algorithmic procedure that generates an *alignment* for two given *OMS*
NOTE For both matching and alignment, see [9, 25].

union

aggregation of several *OMS* to a new *OMS*, without any renaming

OMS graph

graph with *OMS* as nodes and *OMS mappings* as edges, showing how the OMS are interlinked [27] Note(27)
NOTE An OMS graph is a diagram of OMS in the sense of category theory, but different from a diagram in the sense of model-driven architecture. [28] Note(28)

combination

aggregation of all the OMS in a *OMS graph*, where *non-logical symbol*s are shared according to the *OMS mapping*s in the OMS graph
EXAMPLE Consider an ontology involving a concept `Person`, and another one involving `Human being`, and an *alignment* that relates these to concepts. In the combination of the ontologies along the alignment, there is only one concept, representing both `Person` and `Human being`.

---

[27] NOTE: add note with examples of what can be these links
[28] NOTE: clarify the difference to distributed OMS

sharing

property of *OMS symbols* being mapped to the same symbol when computing a *combination* of a *OMS graph*
NOTE Sharing is always relative to a given OMS graph that relates different OMS. That is, two given OMS symbols can share w.r.t. one OMS graph, and not share w.r.t. some other OMS graph.

4.7. Features of OMS Languages

OMS language translation

mapping from constructs in the source *OMS language* to their equivalents in the target *OMS language*

NOTE An OMS language translation shall satisfy the property that the result of a translation is a well-formed text in the target language.

sublanguage

syntactically specified subset of a given language, consisting of a subset of its terminal and nonterminal symbols and grammar rules

language aspect

set of language constructs of a given language, not necessarily forming a sublanguage

logical language aspect

the (unique) *language aspect* of an *OMS language* that enables the expression of *non-logical symbols* and *sentences* in a logical language

structuring language aspect

the (unique) *language aspect* of an *OMS language* that covers *structured OMS* as well as the relations of *basic OMS* and *structured OMS* to each other, including, but not limited to *imports*, *OMS mappings*, *conservative extensions*, and the handling of prefixes for CURIEs

annotation language aspect

the (unique) *language aspect* of an *OMS language* that enables the expression of comments and annotations

profile

(syntactic) *sublanguage* of an *OMS language* interpreting according to a particular *logic* that targets specific applications or reasoning methods
EXAMPLE Profiles of OWL 2 include OWL 2 EL, OWL 2 QL, OWL 2 RL, OWL 2 DL, and OWL 2 Full.
NOTE Profiles typically correspond to *sublogics*.

NOTE Profiles can have different *logics*, even with completely different semantics, e.g. OWL 2 DL versus OWL 2 Full.
NOTE The logic needs to support the language.

4.8. OMS Language Serializations

serialization

specific syntactic encoding of a given *OMS language*
NOTE Serializations serve as standard formats for exchanging OMS between tools.
EXAMPLE OWL uses the term "serialization"; the following are standard OWL serializations: OWL functional-style syntax, OWL/XML, OWL Manchester syntax, plus any standard serialization of RDF (e.g. RDF/XML, Turtle, . . .). However, RDF/XML is the only one tools are required to implement.
EXAMPLE Common Logic uses the term "dialect"; the following are standard Common Logic dialects: Common Logic Interchange Format (CLIF), Conceptual Graph Interchange Format (GCIF), eXtended Common Logic Markup Language (XCL).

document

result of serializing an OMS using a given serialization

standoff markup

way of providing *annotations* to subjects in external resources, without embedding them into the original resource (here: *OMS*)

4.9. Logic

logic

specification of valid reasoning that comprises *signatures*, *sentences*, *models*, and a *satisfaction relation* between *models* and *sentences*
NOTE Most OMS languages have an underlying logic.
EXAMPLE $\mathcal{SROIQ}(D)$ is the logic underlying OWL 2 DL.
NOTE See annex C for the organization of the relation between OMS languages and their logics and serializations.

institution

metaframework mathematically formalising the notion of a *logic*
NOTE See clause 10 for a formal definition.

logic translation

mapping of a source *logic* into a target *logic* (mapping *signatures*, *sentences* and *models*) that keeps or encodes the logical content of *OMS*

logic reduction

mapping of a source *logic* onto a (usually less expressive) target *logic* (mapping *signatures*, *sentences* and *models*) that simply forgets those parts of the logical structure not fitting the target *logic*

logic approximation

mapping of a source *logic* onto a (usually less expressive) target *logic* that tries to approximate the OMS expressed in the source *logic* with means of the expressivity of the target *logic*
NOTE A unique maximal approximation need not exist.

sublogic

a *logic* that is a syntactic restriction of another logic, inheriting its semnatics

heterogeneous OMS

OMS whose parts are supported by different *logics*
EXAMPLE [29] Note(29)

4.10. Interoperability

[30] Note(30)
[31] Note(31)
[32] Note(32)

---

[29] NOTE: todo. Maybe take it from section 7?

[30] NOTE: TODO: possibly define some notion of "interoperability" that is tailored to this OMG Specification. At least we need to be able to speak about overall consistency, alignments, etc.

[31] NOTE: FYI: Definitions in earlier drafts were not quite helpful:
- OMS integration := "combination of different OMS into a coherent whole, via alignments"
- OMS interoperability := "relation among OMS (via OMS alignments) with the goal of using them jointly in an application scenario"

AENOR commented on the latter: "The definition of this term needs some revision and more precision in the document as for the real criteria that shall be applied to evaluate the degree of interoperability between OMS."

[32] NOTE: Frank Farance cited the following from ISO/IEC 2381-1 Information Technology Vocabulary – Part 1: Fundamental Terms:
01.01.47
interoperability
The capability to communicate, execute programs, or transfer data among various functional units in a manner that requires the user to have little or no knowledge of the unique characteristics of those units.
01.01.40
functional unit
An entity of hardware or software, or both, capable of accomplishing a specified purpose.
... and the following from the FDIS 20944-1 Information technology – Metadata Registries Interoperability and Bindings (MDR-IB)– Part 1: Framework, common vocabulary, and common provisions for conformance
3.21.12.4
data interoperability
interoperability concerning the creation, meaning, computation, use, transfer, and exchange of data
3.21.12.5

logically interoperable

property of *structured OMS*, which may be written in different *OMS languages* [33]based on different *logics*, of being usable jointly in a coherent way (via suitable *OMS language translations*), such that the notions of their overall consistency and logical entailment have a precise logical semantics Note(33)

NOTE TODO Michael: explain the relationship to other notions of interoperability (from existing standards)

---

metadata interoperability

interoperability concerning the creation, meaning, computation, use, transfer, and exchange of descriptive data

[33] NOTE: TODO: phrase this more precisely, based on the previously introduced terms

5. Symbols

[34] Note(34)

As listed below, these symbols and abbreviations are generally for the main clauses of the OMG Specification. Some annexes may introduce their own symbols and abbreviations which will be grouped together within that annex.

CASL	Common Algebraic Specification Language, specified by the Common Framework Initiative
CGIF	Conceptual Graph Interchange Format
CL	Common Logic
CLIF	Common Logic Interchange Format
CURIE	Compact URI expression
DDL	Distributed description logic
DOL	Distributed Ontology, Modeling and Specification Language
EBNF	Extended Backus-Naur Form
E-connections	a modular ontology language (closely related to DDL)
F-logic	frame logic, an object-oriented ontology language
IRI	Internationalized Resource Identifier
OWL 2	Web Ontology Language (W3C), version 2: family of knowledge representation languages for authoring ontologies
OWL 2 DL	description logic profile of OWL 2
OWL 2 EL	a sub-Boolean profile of OWL 2 (used often e.g. in medical ontologies)
OWL 2 Full	the language that is determined by RDF graphs being interpreted using the OWL 2 RDF-Based Semantics [18]
OWL 2 QL	profile of OWL 2 designed to support fast query answering over large amounts of data
OWL 2 RL	fragment of OWL 2 designed to support rule-based reasoning
OWL 2 XML	XML-based serialization of the OWL 2 language
P-DL	Package-based description logic
RDF	Resource Description Framework, a graph data model
RDFa	a set of XML attributes for embedding RDF graphs into XML documents
RDF/XML	an XML serialization of the RDF data model
RIF	Rule Interchange Format
UML	Unified Modeling Language
URI	Uniform Resource Identifier
URL	Uniform Resource Locator
W3C	World Wide Web Consortium
XML	eXtensible Markup Language

---

[34] Note: add OMG stuff

6. Additional Information

[35] [36] [37] Note(35) Note(36) Note(37)

An ontology is a formal description of the concepts and relationships that are of interest to an agent or a community of agents. Today, ontologies are applied in eBusiness, eHealth, eGovernment, eInclusion, eLearning, smart environments, ambient assisted living (AAL), and virtually all other information-rich endeavours. Ontologies have been used initially and principally for data and database integration through providing a common representation of the subject domain onto which the data sources can be mapped meaningfully. Over the years, the purpose has broadened beyond data and services interoperability to include a wide range of tasks and ontologies are used in information systems at run-time, such as being a component in *in silico* scientific workflows, used for natural language processing, in ontology-driven querying of digital libraries, user profiling in recommender systems, adaptive e-Learning tools, and more.[38] Note(38)

[This page is left intentionally blank.]

---

[35] Note: have a look at ODM's Additional Information

[36] Note: this could be put in section 1 (if possible with OMG), or turned into informative notes, or deleted

[37] Note: this is only about ontologies, generalize to OMS

[38] Note: Terry Longstreth: Interoperability in this context seems to be mutual consistency? Fostering Mutual Consistency among disjoint ontological formalisms (intensions) and their realisations (extensions). TM: yes, but more than that: also interfacability, such that the joint use in a common application scenario is enabled.

7. Goals and Usage Scenarios

There are many domains in which multiple OMS are needed, in some cases axiomatized in the same language, and in other cases axiomatized in different logical languages. This leads to several challenges in the design and deployment of OMS, which have been addressed by current research in ontological engineering, formal software specification and formal modeling:

- How can we support sharability and reusability of OMS within the same domain?
- How can we merge OMS in different domains, particularly in the cases in which the OMS are axiomatized in different logical languages?
- What notions of modularity play a role when only part of an OMS is being shared or reused?
- What are the relationships between versions of an OMS axiomatized in different logical languages?

These challenges can be illustrated by the following set of use cases. These use cases illustrate that in ontology design, in formal specification, and in model-driven development, the same problem arises: the use of heterogeneous formal representations leads to interoperability challenges. There are ad-hoc solutions to these challenges, and specialized languages and tools are used in practice. However, there is no standardized approach or representation metalanguage to enable more accurate and consistent alignment, integration, and mapping among these specifications and the tools that implement them.

7.1. Use case Onto-1: Interoperability between OWL and FOL ontologies

In order to achieve interoperability, during ontology development it is often necessary to describe concepts in a language more expressive than OWL. Therefore, it is common practice to informally annotate OWL ontologies with FOL axioms (e.g., Keet's mereotopological ontology [Part-Whole], Dolce Lite [Dolce-lite], BFO-OWL). OWL is used because of better tool support, FOL because of greater expressiveness. However, relegating FOL axioms to informal annotations means that these are not available for machine processing. Another example of this problem is the following: For formally representing concept schemes (including taxonomies, thesauri and classification schemes) and provenance information there are the two W3C standards SKOS (Simple Knowledge Organization System) and PROV, as well as [39]ISO and other domain-specific standards for metadata representation. The semantics for the SKOS and PROV languages are largely specified as OWL ontologies; however, as OWL cannot capture the full semantics, the rest is specified using some informal first-order rules. In other words, valid instance models that use SKOS or PROV may be required to satisfy both OWL and FOL axioms. When solving reasoning tasks over either SKOS or PROV ontologies, Note(39)

---

[39] NOTE: CL: Did we mean something like "ISO 12345" here, i.e. some specific ISO standard that we reference by number?

OWL reasoners are not able to consider the FOL axioms. Hence, the information contained in these axioms is lost.

DOL allows the user to replace such informal annotations by formal axioms in a suitable ontology language. The relation between the OWL ontology and the FOL axioms is that of a heterogeneous import. In the result, both the OWL and the FOL axioms are amenable to, e.g., automated consistency checks and theorem proving. Hence, all available information can be used in the reasoning process.

7.2. Use Case Onto-2: Ontology integration by means of a foundational ontology

One major use case for ontologies in industry is to achieve interoperability and data integration. However, if ontologies are developed independently and used within the same domain, the differences between the ontologies may actually impede interoperability. One strategy to avoid this problem is the use of a shared foundational ontology (e.g., DOLCE or BFO), which can be used to harmonize different domain ontologies. One challenge for this approach is that foundational ontologies typically rely on expressive ontology languages (e.g., Common Logic), while domain ontologies may be represented in languages that are optimized for performance (e.g., OWL EL). For this reason, currently the role of the foundational ontology is mainly to provide a conceptual framework that may be reused by the domain ontologies; further, watered-down versions of the foundational ontologies in OWL (like DOLCE-lite or the OWL version of BFO) are used as basis for the development of domain ontologies, be this as is, in an even less expressive version (e.g., a DOLCE-lite in OWL 2 EL), or only a relevant subset thereof (e.g., only the branch of endurants). A sample orchestration of interactions between the foundational and domain ontologies in various languages is depicted in Figure 8.1 below.

DOL provides the framework for integrating different domain ontologies, aligning these to foundational ontologies [Alignment1-2] and combining the aligned ontologies into a coherent integrated ontology – even across different ontology languages. Thus, DOL enables ontology developers to utilize the complete, and most expressive, foundational ontologies for ontology integration and validation purposes.

7.3. Use Case Onto-3: Module extraction from large ontologies

Especially in the biomedical domain, ontologies tend to become very large (e.g., SNOMED CT, FMA) with over 100000 concepts and relationships. Yet, none of these ontologies covers all aspects of a domain, and frequently provide coverage at various levels of specificity, with excessive detail in some areas that may not be required for all usage scenarios. Often, for a given knowledge representation problem in industry, only relevant knowledge from two such large reference ontologies needs to be integrated, so a comprehensive integration would be both unfeasible and unwieldy. Hence, parts (modules) of these ontologies are obtained by selecting the concepts and relationships (roles) relevant for the intended application. An integrated version will then be based on these excerpts from the original ontologies (i.e., modules). For example, the Juvenile Rheumatoid Arthritis ontology JRAO has been created using modules from the NCI thesaurus and GALEN medical ontology. (See [40]Figure 7.1) DOL supports the Note(40)

---

[40] Note: CL: can we please have a vector graphic here?

description of such subsets (modules) of ontologies, as well as their alignment and integration.

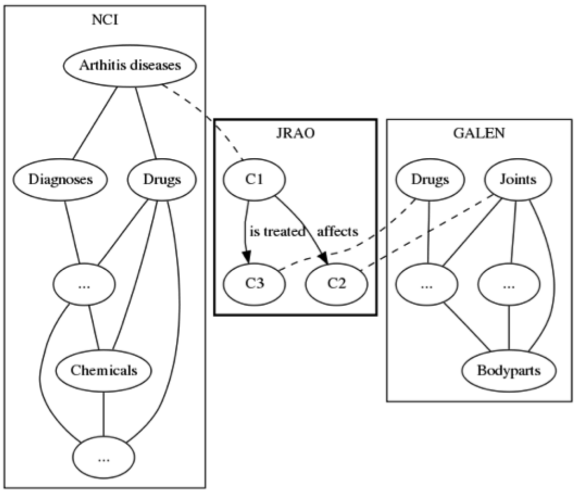

Figure 7.1.: JRAO – Example for Module Extraction

7.4. Use case Onto-4: Interoperability between closed-world data and open-world metadata

Data collection has become easier and much more widespread over the years. This data has to be assigned a meaning somehow, which occurs traditionally in the form of metadata annotations. For instance, consider geographical datasets derived from satellite data and raw sensor readings. Current implementations in, e.g., ecological economics[41] require manual annotation of datasets with the information relevant for their processes. While there have been attempts to standardize such information, metadata for datasets of simulation results are more difficult to standardize. Moreover, it is resource-consuming to link the data to the metadata, to ensure the metadata itself is of good quality and consistent, and to actually exploit the metadata when querying the data for data analysis. Note(41)

The data is usually represented in a database or RDF triple store, which work with a closed world assumption on the dataset, and are not expressive enough to incorporate the metadata ‘background knowledge’, such as the conditions for validity of the physical laws in the model of the object of observation. These metadata require a more expressive language, such as OWL or Common Logic, which operate under an open-world semantics. However, it is unfeasible to translate the whole large dataset into OWL or first-order logic. To ‘meet in the middle’, it is possible to declare bridge rules (i.e., a mapping layer) that can link the

---

[41] NOTE: Tara: add reference. Tara says: e.g. ecological economics http://www.ariesonline.org/docs/ARIESModelingGuide1.0.pdf attempts to standardize http://inspire-geoportal.ec.europa.eu/

metadata to the data. This approach can be used for intelligent data analysis that combines the data and metadata through querying the system. It enables the analysis of the data on the conceptual layer, instead of users having to learn the SQL/SPARQL query languages and how the data is stored. There are various tools and theories to realize this, which is collectively called Ontology-Based Data Access/Management, see also [OBDA].

The languages for representing the metadata or ontology, for representing the bridge rules or mapping assertions, and for representing the data are different yet they need to be orchestrated and handled smoothly in the system, be this for data analytics for large enterprises, for formulating policies, or in silico biology in the sciences.

DOL provides the framework for expressing such bridge rules in a systematic way, maintaining these, and building tools for them.

7.5. Use Case Onto-5: Verification of rules translating Dublin Core into PROV

The Dublin Core Metadata terms, which have been formalized as an RDF Schema vocabulary, developed initially by the digital library community, are less comprehensive but more widely used than PROV (cf. Use Case Onto-1). The rules for translating Dublin Core to the OWL subset of PROV (and, with restrictions, vice versa) are not known to yield valid instances of the PROV data model, i.e. they are not known to yield OWL ontologies consistent w.r.t. the OWL axioms that capture part of the PROV data model. This may disrupt systems that would like to reason about the provenance of an entity, and thus the assessment of the entity's quality, reliability or trustworthiness. The Dublin Core to PROV ontology translation[1] is expressed partly by a symbol mapping and partly by FOL rules. These FOL rules are implemented by CONSTRUCT patterns in the SPARQL RDF query language.[2] SPARQL has a formal specification of the evaluation semantics of its algebraic expressions, which is different from the model-theoretic semantics of the OWL and RDFS languages; nevertheless SPARQL CONSTRUCT is a popular and immediately executable syntax for expressing translation rules between ontologies in RDF-based languages in a subset of FOL. DOL not only supports the reuse of the existing Dublin Core RDFS and PROV OWL ontologies as modules of a distributed ontology, but it is also able to support the description of the FOL translation rules in a sufficiently expressive ontology language, e.g. Common Logic, and thus enable formal verification of the translation from Dublin Core to PROV.

7.6. Use case Spec-1: Specification Refinements

Especially in safety-critical areas such as medical systems, the automotive industry, avionics and the aerospace industry, but also for microprocessor design, often a formal software and hardware development process is used in order to ensure the correct functioning of systems. Typically, a requirement specification is refined into a design specification and then an implementation, often involving several intermediate steps (see, e.g. the V-model [V-model], although this does not require formal specification). There are numerous specification formalisms in use, including the OMG's SysML language; moreover, often during development, the formalism needs to be changed (e.g. from a specification to a programming language,

---

[1] `http://www.w3.org/TR/2013/NOTE-prov-dc-20130430/`

[2] E.g., `http://www.w3.org/TR/2013/NOTE-prov-dc-20130430/#dct-creator`

or from a temporal logic to a state machine). For each of these formalisms, notions of refinement have been defined and implemented. However, the lack of a standardized, logically sound language and methodology for such refinement hinders interoperability among different development efforts and the reuse of refinements. DOL provides the capability to represent refinement that is equally applicable to all DOL-conforming logical languages, and that covers at least the most relevant of the industrial use cases of specification refinement.

7.7. Use case Spec-2: Modularity of Specifications

In the context of use case Spec-1, often specifications become so large that it is necessary to structure them in a modular way, both for human readability, maintainability, and for more efficient tool support. The lack of a standard for such modular structuring hinders interoperability among different development efforts and the reuse of specifications. DOL provides a notion of structured modular specification that is equally applicable to all DOL-conforming logical languages.

7.8. Use case Model-1: Coherent semantics for multi-language models

[42] Note(42)

Often a single problem area within a given domain must be described using several formalisms, due to user community requirements, expressiveness, tool support and usage, and so forth. A challenge is that typically the different formalizations are written by different people using different logics, and, thus, their overall consistency is hard to maintain. The need for the use of multiple ontology languages, even within the OMG community, is also reflected by the OMG Ontology Definition Metamodel (ODM), which provides a number of syntactic transformations between such languages. One example is the OMG Date-Time Vocabulary (DTV). DTV has been formulated in different languages, each of which addresses different audiences:

- SBVR: business users
- UML (class diagrams and OCL): software implementers
- OWL: ontology developers and users
- Common Logic: (foundational) ontology developers and users

With DOL, one can, e.g.,

- formally relate the different formalizations used for DTV, relate the different formalizations using translations,
- check consistency across the different formalizations (using suitable tools),
- extract sub-modules covering specific aspects, and
- specify the OWL version to be an approximation of the Common Logic version (using a heterogeneous interpretation of OMS).

Note that the last point does not specify what information is lost in the approximation. Indeed, DOL provides the means to specify requirements on the approximation, e.g., that it maximally preserves the information.

---

[42] Note: Isn't this one about ontologies,too?

7.9. Use case Model-2: Consistency among UML diagrams of different types

A typical UML model involves diagrams of different types. Such UML models may have intrinsic errors because diagrams of different types may specify conflicting requirements. Typical questions that arise in this context are, e.g.,

- whether the multiplicities in a class diagram are consistent with each other
- wether the attributes and operations in a state machine are available in a class diagram
- whether the sequential composition of actions in an interaction diagram is justified by an accompanying OCL specification,
- whether cooperating state machines comply with pre-/post-conditions and invariants
- if the behavior prescribed in an interaction diagram is realizable by several state machines cooperating according to a composite structure diagram.

Such questions are currently hard to answer in a systematic manner. One method to answer these questions and find such errors is a check for semantic consistency. Under some restrictions, the proof of semantic consistency can be (at least partially) performed using model-checking tools like Hugo/RT [?]. Once a formal semantics for the different diagram types has been chosen (see, e.g. [?]), it is possible to use DOL to specify in which sense the diagrams need to be consistent, and check this by suitable tools.

7.10. Use case Model-3: Refinements between UML diagrams of different types, and their reuse

A problem is a lack of reusability of refinements: Consider a controller for an elevator, which is specified with a UML protocol state machine, enriched with UML sequence diagrams and OCL constraints. Assume further that this model is not directly implemented, but first refined to a UML behavior state machine (which then can be automatically or semi-automatically transformed into some implementation using standard UML tools). However, there is no standardized language to express, document and maintain the refinement relation itself (UML only allows very simple refinements, namely between state machines). This hinders both the reuse of such refinements in different contexts, as well as the interoperability of tools proving such refinements to be correct. DOL addresses these problems by providing a standardized notation with formal semantics for such refinements. Refinements expressed in this language could, e.g., be parameterized and reused in different contexts.

7.11. Conclusion

In the next sections, we discuss the metalanguage DOL, its features that enable the support of a variety of formalisms, with syntax, well-defined semantics and model theory. DOL distills best practices of modularity and metarelations (such as refinement and alignment) across the three areas of ontology design, formal specification, and model-driven development. It provides the ability to specify the basis for formal interoperability even among heterogeneous OMS. DOL enables the solutions of the problems described in the use cases above. It also enables the development of OMS libraries, tools and workflows that allow a better exchange and reuse of OMS. Eventually, this will also lead to better, easier developed and maintained systems based on these OMS.

8. Design Overview

This clause is informative. Its purpose is to briefly describe the the overall guiding principles and constraints of DOL's syntax and semantics. We give an overview of the most important and innovative language constructs of DOL. Details can be found in clause 9.

DOL gives interoperability a formal grounding and makes heterogeneous OMS and services based on them amenable to checking of coherence (e.g. consistency, conservativity, intended consequences, and compliance). OMS languages are declarative languages for making ontological distinctions formally precise.[43] They are distinguished by the following features: Note(43)

Logic Most commonly, OMS languages are based on a description logic or some other subset of first-order logic, but in some cases, also higher-order, modal, paraconsistent and other logics are used.

Modularity means of structuring an OMS into reusable parts, reusing parts of other OMS, mapping imported symbols to those in the importing OMS, and asserting additional properties about imported symbols.

Annotation means of attaching human-readable descriptions to OMS symbols, addressing knowledge engineers and service developers, but also end users of OMS-based services.[44] Note(44)

Whereas the first feature determines the expressivity of the language and the possibilities for automated reasoning (decidability, tractability, etc.), the latter two intend to facilitate OMS engineering as well as the engineering of OMS-based software.

Acknowledging the wide tool support that conforming established languages such as OWL or Common Logic enjoy, existing OMS in these languages remain as they are within the DOL framework. DOL enhances their modularity and annotation facilities to a superset of the modularity and annotation facilities they provide themselves. DOL's modularity and annotation constructs can either be embedded into existing OMS as non-disruptive annotations, or they can be provided as standoff markup, pointing to the OMS they talk about; DOL specifies a syntax and semantics for both variants. DOL's modularity constructs are semantically well-founded within a library of formal relationships between the logics underlying the different supported OMS languages.

8.1. Overview of DOL

[45] Note(45)

DOL is a language enabling OMS interoperability. DOL is

free DOL is freely available for unrestricted use.

generally applicable DOL is neither be restricted to OMS in a specific domain, nor to foundational OMS, nor to OMS represented in a specific OMS language, nor to OMS stored in any specific repositories.

---

[43] Note: generalize to OMS

[44] Note: TODO Christoph: reformulate. DOL enables the use of annotations

[45] Note: TM: rewrite

open DOL supports mapping, integrating, and annotating OMS across arbitrary internet locations. It makes use of existing open standards wherever suitable. The criteria for extending DOL (see next item) are transparent and explicit.

extensible DOL provides a framework into which any existing, and, desirably, any future OMS language can be plugged.

DOL is applicable to any OMS language that has a formal, logic-based semantics or a semantics defined by translation to another OMS language with such a formal semantics. The annotation framework of DOL is additionally applicable to the non-logical constructs of such languages. This OMG Specification specifies formal criteria for establishing the conformance of an OMS language with DOL. Annexes establish the conformance of a number of relevant OMS languages with DOL; a registry shall offer the possibility to add further (also non-standardized) languages:[46] DOL provides syntactic constructs for structuring OMS regardless of the logic their sentences are formalized in. DOL does provide its own constructs for expressing sentences. Instead, it *inherits* the logical language aspects of conforming OMS languages. It is possible to literally include sentences expressed in such OMS languages in a DOL OMS. DOL provides an initial set of built-in approximation methods and module extraction selectors. Additionally, it provides a means of referring to approximation methods and module extraction selectors defined externally of this OMG Specification.[47] DOL provides an initial vocabulary for expressing relations in correspondences (as part of alignments between OMS). Additionally, it provides a means of reusing relation types defined externally of this OMG Specification. DOL does not provide an annotation vocabulary, i.e. it neither provides annotation properties nor datatypes to be used with literal annotation objects. Instead, an informative annex recommends existing annotation vocabularies for use with DOL. [48] In the interest of wide applicability and tool support, DOL supports multiple alternative serializations. In particular, there is a text serialization targeting human readers and writers, as well as serializations optimized for machine processability. The **text serialization** in particular offers a syntax for abbreviating identifiers of resources within OMS in a way that does not require authors to write down their full global identifiers. An OMS implemented in DOL can comprise parts formalized in any OMS language; any serialization of DOL can literally include such parts, regardless of the OMS language serialization they have been written in. [49] Additionally, an OMS implemented in DOL can refer to any external OMS formalized in any OMS language, as long as they can be identified in a globally unique way. Existing OMS in existing XML serializations (e.g. XCL) or text serializations (e.g. OWL Manchester Syntax) validate as DOL OMS with a minimum amount of syntactic adaptation. Existing OMS files/documents are usable in a DOL context without the need for modification.

Note(46)

Note(47)

Note(48)

Note(49)

DOL does not provide a new elementary OMS language, but provides a layer to be used on top of existing elementary OMS languages which enables OMS engineers to formally express mappings between OMS written in different languages and stored at different Web locations. The purpose of such distributed OMS is enabling a greater extent of interoperability between data and services in complex application settings.

The following features are essential to the design of this OMG Specification:

- DOL is a language covering OMS modularity, OMS heterogeneity, and OMS mapping.

---

[46] Note: John Sowa: Make it modular with a simple core that can run efficiently on small systems, but can grow indefinitely to support as much as anyone could desire.

[47] Note: FYI: In practice we will use IRIs for that purpose.

[48] Note: Q-ALL: We need to revise this following the agreement to drop the XML and RDF serializations.

[49] Note: FYI: advanced namespacing is the solution that addresses this requirement

In particular, it enables writing structured OMS (thereby reusing existing OMS), OMS involving different languages, as well as complex mappings and relations between OMS.

- DOL is a declarative language with a formal semantics.
- DOL provides a superset of the modularization, Web awareness and annotation facilities of a number of commonly used OMS languages, including OWL [?], RDF [?], Common Logic [?] and UML [?].[1]
- DOL is an open, extensible standard that is not restricted to a fixed set of supported OMS language but specifies criteria for any existing or future OMS language to conform with DOL.
- Existing OMS in languages conforming with DOL remain as they are; they can be enriched with DOL's modularity and annotation constructs in a non-disruptive way.

[50] Note(50)

8.2. DOL enables expression of logically heterogeneous OMS and literal reuse of existing OMS.

DOL is a mechanism for expressing logically heterogeneous OMS. It can be used to combine sentences and structured OMS expressed in different conforming OMS languages and logics into single documents or modules. With DOL, sentences or structured OMS of previously existing OMS in conforming languages can be reused by literally including them into a DOL OMS. A minimum of wrapping constructs and other annotations (e.g., for identifying the language of a sentence) are provided. [51] See the abstract syntax category `OMS` in clause 9. Note(51)

8.3. DOL includes provisions for expressing mappings between OMS.

DOL provides a syntax for expressing mappings between OMS – logical OMS Mappings as well as alignments. One use case illustrating both is sketched in Figure 8.1. This OMG Specification specifies a set of logical OMS mapping types and a set of non-logical OMS mapping types.

Logical OMS mappings supported by DOL include:

- imports (particularly including imports that lead to conservative extensions), see the abstract syntax categories `OMSRef` and `ExtensionOMS` in clause 9.
- interpretations, see the abstract syntax category `IntprDefn` in clause 9.
- mappings between OMS and their modules, see the abstract syntax category `ModuleRelDefn` in clause 9.

---

[1] See clause ?? for details.

[50] Note: reformulate this, see RFP

[51] Note: TODO: Figure out what this feedback item from Michael Grüninger (?) means: say that there should be a syntax for relationships btw. OMS as well as a syntax for heterogeneous OMS. (If you write down an OMS, it might involve constructs that only exist in OWL)

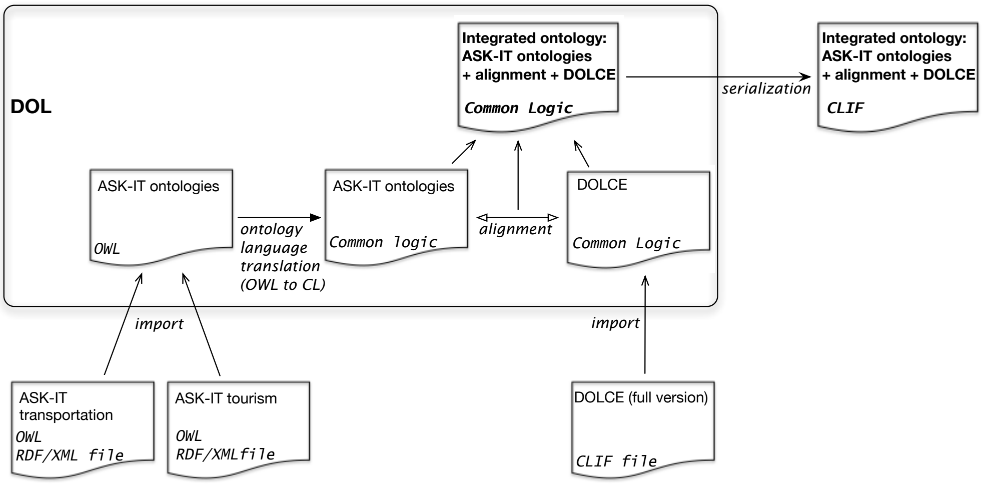

Figure 8.1.: Mapping between two OMS formulated in different OMS languages

DOL allows such links to express signature translations in such OMS mappings; see the abstract syntax category `SymbolMapItems` in clause 9.

DOL need not be able to fully represent logical translations but is capable of referring to them.

[52] Note(52)

DOL can also be used to combine or merge OMS along such OMS mappings, see the rule for `combination` for the abstract syntax category `OMS` in clause 9.

8.4. DOL enables the representation of OMS and OMS mappings at different levels of detail

OMS and OMS mappings expressed in DOL can be based on a number of implicit assumptions about which OMS language translation or which ontology matcher has been employed. Depending on the OMS engineering workflow or application setting, it can be useful to keep these assumptions implicit or to make them explicit. DOL permits to leave such assumptions implicit if desired. However, it also enables the user to capture these assumptions explicitly as annotations to the OMS. This OMG Specification specifies a translation that expands any DOL OMS with implicit assumptions into its explicit counterpart.

The following list covers the possible cases where DOL provides a choice between representing information implicitly or explicitly:

default OMS language translations A heterogeneous OMS can import several (structured) OMS expressed in different conforming logics, for which suitable translations have been defined in the logic graph provided in annex G or in an extension to it that has been provided when establishing the conformance of some other logic with DOL. Determining

---

[52] NOTE: Q-AUT: We had this comment here; what does it mean? “DOL only maps symbols to expressions”

the semantics of the heterogeneous OMS requires a translation into a common target language to be applied (cf. clause 10). This translation is determined via a lookup in the transitive closure of the logic graph. Depending on the reasoners available in the given application setting, it can, however, be necessary to employ a different translation. Authors can express which one to employ. In a multi-step translation, it is possible to implicitly apply as many default translations as possible, and to concentrate on making explicit only those translations that deviate from the default.[53] Note(53)

different matching algorithms OMS alignments, which DOL is able to express, may have been obtained by running different OMS matching algorithms. If, in a given OMS engineering workflow, the information on which algorithm has been applied is clear from the context, it is possible to omit it in the alignment expressed in DOL. Otherwise, e.g. if the next person working on the OMS requires that information, it is possible to make it explicit.[54] Note(54)

[55] Note(55)

8.5. DOL provides a mechanism for rich annotation and documentation of OMS.

[56] DOL supports annotations in the full generality specified in clause 4.4. The DOL serializations further support the fine-grained embedding of annotations into OMS. Note(56)

The DOL serializations also supports the annotation of existing OMS via non-intrusive standoff markup, which points to the annotation subjects from external documentation files or from special embedded comments, extending the comment syntax of the respective OMS language; for XML serializations of OMS languages, RDFa extensions are specified, so that DOL RDF can be embedded.

A list of RDF vocabularies for annotating OMS is recommended as an annex to this OMG Specification.

---

[53] Note: Q-AUT: Will this situation be the same for default approximations, or to we need to add an extra item to the list?

[54] Note: TM: the alignment itself should be there explicitly. right? But then the information about the matching algorithm that has produced it is a mere annotation without semantics, isn't it?
CL: I agree with you. OK, so we will replace this with approximation algorithms.

[55] Note: TODO: ask Michael Grüninger for his mereology example in CL

[56] Note: Q-ALL: I think that now that we have agreed on dumping the RDF and XML serializations of DOL, this requirement can no longer be satisfied. Or maybe it can be satisfied in a trivial way (nonetheless requiring this section to be shortened): DOL provides the mechanism for identifying anything of relevance in a distributed OMS, it will do so by IRIs, and with RDF there is an established mechanism for annotating things identified by IRIs. Still I believe this requirement is an important selling point.

9. DOL abstract syntax

9.1. Abstract syntax categories

DOL provides abstract syntax categories for

- heterogeneous OMS (which can be basic OMS in some OMS language, or unions, translations, minimizations, combinations, approximations of OMS, among others)
- distributed OMS (items in distributed OMS are: OMS definitions, OMS mapping definitions, and qualifications choosing the logic, OMS language and/or serialization)
- identifiers
- annotations

Additionally, the categories of the abstract syntaxes of any conforming OMS languages (cf. clause 2.1) are also DOL abstract syntax categories.

The following subclauses, one per abstract syntax category, specify the abstract syntax of DOL in EBNF ISO/IEC 14977:1996. Note that ISO EBNF lacks an operator for "at least one repetition". This OMG Specification therefore adopts the following convention: Whenever some sequence `S` is repeated at least once, we give it a non-terminal identifier of its own (`RepeatedS = S { S } ;`), or group it as in `LongerExpression = Foo Bar ( S { S } ) ;`.

9.2. Distributed OMS

A distributed OMS consists of named (possibly heterogeneous) OMS, and mappings between its participating (heterogeneous) OMS. More specifically, a distributed OMS consists of a name, followed by a list of `DistOMSItem`s. A `DistOMSItem` is either an OMS definition (`OMSDefn`), or a mapping between OMS (`MappingDefn`), or a `Qualification` selecting a specific OMS language, logic and/or syntax that is used to interpret the subsequent `DistOMSItem`s. Alternatively, a distributed OMS can also be the verbatim inclusion of an OMS written in an OMS language that conforms with DOL (`OMSInConformingLanguage`; cf. 2.1).

```
DistOMS                   = [ PrefixMap ] , DistOMSDefn
                          | OMSInConformingLanguage ;
DistOMSDefn               = 'dist-oms-defn' , DistOMSName , { DistOMSItem } ;
OMSInConformingLanguage   = <language and serialization specific> ;
DistOMSItem               = OMSDefn | MappingDefn | Qualification ;
Qualification             = LanguageQual | LogicQual | SyntaxQual ;
LanguageQual              = 'lang-select' , LanguageRef ;
LogicQual                 = 'logic-select' , LogicRef ;
SyntaxQual                = 'syntax-select' , SyntaxRef ;
DistOMSName               = IRI ;
```

[57] Note(57)

At the beginning of a distributed OMS, one can declare a `PrefixMap` for abbreviating long IRIs; see clause 9.5 for details.

9.3. Heterogeneous OMS

An OMS (`OMS`) can be one of the following:

- a basic OMS `BasicOMS` written inline, in a conforming serialization of a conforming OMS language[1],
- a translation of an OMS into a different signature or OMS language,
- a reduction of an OMS to a smaller signature and/or less expressive logic (that is, some non-logical symbols are hidden, but the semantic effect of sentences involving these is kept),
- an approximation of an OMS, normally in a sublogic, using a given approximation method (with the effect that sentences not expressible in the sublogic are weakened or removed),
- a union of OMS,
- an extension of an OMS by other ones, it can be optionally named and/or marked as conservative, monomorphic, definitional or implied,
- a module extracted from an OMS, using a restriction signature,
- a reference to an OMS existing on the Web,
- an OMS qualified with the OMS language that is used to express it,
- a combination of OMS (technically, this is a colimit, see [41]),
- a minimization of an OMS, forcing the subsequently declared non-logical symbols to be interpreted in a minimal way, while the non-logical symbols declared so far are fixed (alternatively, the non-logical symbols to be minimized and to be varied can be explicitly declared).

```
BasicOMS             = OMSInConformingLanguage ;
MinimizableOMS       = BasicOMS
                     | 'oms-ref' , OMSRef , [ ImportName ] ;
ExtendingOMS         = MinimizableOMS
                     | 'minimize' , MinimizableOMS ;
OMS                  = ExtendingOMS
                     | 'minimize-symbols' , OMS , CircMin , CircVars
```

---

[57] NOTE: FYI: Things changed from HetCASL:
- logic-select now mandatory (no default logic) and tree-scoped
- download-items (encourage linked data best practices instead)
- item-name-map (to be replaced by namespaces??)
- lib-version (to be replaced by metadata annotations, e.g. OMV)
- indirect-mapping (will always use full IRIs, and abbreviate them by syntactic namespaces)

[1] In this place, any OMS in a conforming serialization of a conforming OMS language is permitted. However, DOL's module sublanguage should be given preference over the module sublanguage of the respective conforming OMS language; e.g. DOL's extension construct should be preferred over OWL's import construct.

```
                     | 'translation' , OMS , Translation
                     | 'reduction' , OMS , Reduction
                     | 'module-extract' , OMS , Extraction
                     | 'approximation' , OMS , Approximation
                     | 'union' , OMS , [ ConsStrength ] , OMS
                     | 'extension' , OMS , ExtensionOMS
                     | 'qual-oms' , { Qualification } , OMS
                     | 'bridge' , OMS, { Translation } , OMS
                     | 'combination' , Graph ;

CircMin              = Symbol , { Symbol } ;
CircVars             = { Symbol } ;

Translation          = 'renaming' , { LogicTranslation } , [ SymbolMapItems ] ;
LogicTranslation     = 'logic-translation' , OMSLangTrans ;

Reduction            = 'hidden' , { LogicReduction } , [ SymbolItems ]
                     | 'revealed' , [ SymbolMapItems ] ;
LogicReduction       = 'logic-reduction' , OMSLangTrans ;

SymbolItems          = 'symbol-items' , ( Symbol , { Symbol } ) ;
SymbolMapItems       = 'symbol-map-items' , ( SymbolOrMap , { SymbolOrMap } ) ;  Note (58) 58

Extraction           = 'extraction', ModuleProperties, [ InterfaceSignature ] ;

ModuleProperties     = Conservative | 'minimal' | 'safe' | 'depleting';

Approximation        = 'approximation' , InterfaceSignature , [ LogicRef ] ;

ExtensionOMS         = [ ConsStrength ] , [ ExtensionName ] , ExtendingOMS ;

ConsStrength         = Conservative | 'monomorphic'
                     | 'weak-definitional' | 'definitional' | 'implied' ;

Conservative         = 'consequence-conservative' | 'model-conservative' ;

InterfaceSignature   = 'interface-signature' , SymbolItems ;

ImportName           = IRI ;
ExtensionName        = IRI ;
```

An OMS definition `OMSDefn` names an OMS. It can be optionally marked as consistent, using `ConsStrength`.[2]. An `SymbolItems`, used in an OMS `Reduction`, is a list of non-logical symbols that are to be hidden. A `LogicReduction` denotes a logic reduction to a less expressive OMS language. A `SymbolMapItems`, used in OMS `Translations`, maps

---

[58] Note: TODO: say that this default may be overridden by specific logics, such as CASL

[2] More precisely, `'consequence-conservative'` here requires the OMS to have a non-trivial set of logical consequences, while `'model-conservative'` requires its satisfiability.

symbols to symbols[59] , or a logic translation. An OMS language translation `OMSLangTrans` or `ApproxMethod` can be either specified by its name (optionally qualified with source and target OMS language), or be inferred as the default translation or approximation method between a given source and target (where even the source may be omitted; it is then inferred as the OMS language of the current OMS). Note(59)

```
OMSDefn          = 'oms-defn' , OMSName , [ ConsStrength ] , OMS ;

Symbol           = IRI ;
SymbolMap        = 'symbol-map' , Symbol , Symbol ;
SymbolOrMap      = Symbol | SymbolMap ;
Term             = <an expression specific to a basic OMS language> ;

OMSName          = IRI ;

OMSRef           = IRI ;
OMSOrMappingorGraphRef  = IRI ;
ExtensionRef     = IRI ;

LoLaRef          = LanguageRef | LogicRef ;

LanguageRef      = IRI ;
LogicRef         = IRI ;
SyntaxRef        = IRI ;

OMSLangTrans     = 'named-trans' , OMSLangTransRef
                 | 'qual-trans' , OMSLangTransRef , LoLaRef , LoLaRef
                 | 'anonymous-trans' , LoLaRef , LoLaRef
                 | 'default-trans' , LoLaRef[60]  ;                    Note(60)

OMSLangTransRef  = IRI ;

ApproxMethod     = 'named-approx', ApproxMethodRef
                 | 'qual-approx' , ApproxMethodRef , LoLaRef
                 | 'default-approx' , LoLaRef[61]  ;                   Note(61)
ApproxMethodRef  = IRI ;

ExtractionMethod = IRI ;
```

---

[59] Note: FYI: On 2012-07-18 we decided not to specify lambda-style symbol-to-term mappings for now. Would be convenient, but specifying its semantics in an OMS language independent way would require additional institution infrastructure – and the same effect can be achieved by auxiliary definitional extensions, cf. Colore (so promote this, informatively, as a "best practice"?)

[60] Note: TODO: need to figure out which of these we actually want to keep. named-trans and default-trans are sufficient, because the other ones contain redundant information that is only stated once more for clarity. Source and target logic of qual-trans are clear from inspecting the translation, and the source logic of anonymous-trans is clear from the OMS that is translated.

[61] Note: TODO: These alternatives are coherent with what we discussed about the approximation syntax with defaults, but they are different from OMSLangTrans. But see the comment for OMSLangTrans above.

9.4. OMS Mappings

A OMS mapping provides a connection between two OMS. A OMS mapping definition is the definition of either a named interpretation (`IntprDefn`), a named declaration of the relation between a module of an OMS and the whole OMS (`ModuleRelDefn`), or a named alignment (`AlignDefn`). The `SymbolMapItems` in an interpretation always must lead to a signature morphism; a proof obligation expressing that the (translated) source OMS logically follows from the target OMS is generated. In contrast to this, an alignment just provides a connection between two OMS without logical semantics, using a set of `Correspondences`. Each correspondence may map some OMS non-logical symbol to another one (possibly given by a term) and an optional confidence value. Moreover, the relation between the two non-logical symbols can be explicitly specified (like being equal, or only being subsumed). A `ModuleRelDefn` declares that a certain OMS actually is a module of some other OMS with respect to the `InterfaceSignature`.

```
MappingDefn          = IntprDefn | EquivDefn | GraphDefn | ModuleRelDefn | AlignDefn;

IntprDefn            = 'intpr-defn' , IntprName , [ Conservative ] , IntprType ,
                                      { LogicTranslation } , [ SymbolMapItems ] ;
IntprName            = IRI ;
IntprType            = 'intpr-type' , OMS , OMS ;

EquivDefn            = 'equiv-defn' , EquivName , EquivType , OMS ;
EquivName            = IRI ;
EquivType            = 'equiv-type' , OMS , OMS ;

GraphDefn            = 'graph-defn', GraphName, Graph ;
GraphName            = IRI ;
Graph                = 'graph', GraphElements , ExcludeExtensions
GraphElements        = 'graph-elements', { OMSOrMappingorGraphRef } ;
ExcludeExtensions    = 'exclude-imports' , { ExtensionRef } ;

ModuleRelDefn        = 'module-defn' , ModuleName , [ Conservative ] , ModuleType ,
                                       InterfaceSignature ;
ModuleName           = IRI ;
ModuleType           = 'module-type' , OMS , OMS ;

AlignDefn            = 'align-defn' , AlignName , [ AlignCard ] , AlignType[3]
                                      { Correspondence } ;
AlignName            = IRI ;
AlignCards           = AlignCardForward , AlignCardBackward[62]  ;
AlignCardForward     = 'align-card-forward' , AlignCard ;
AlignCardBackward    = 'align-card-backward' , AlignCard ;
AlignCard            = 'injective-and-total'
                     | 'injective'
```

Note(62)

---

[3] Note that this grammar uses “type” as in “the type of a function”, whereas the Alignment API uses “type” for the totality/injectivity of the relation/function. For the latter, this grammar uses “cardinality”.

[62] Note: TODO: mention that the default is twice “injective and total”

```
                     | 'total'
                     | 'neither-injective-nor-total' ;
AlignType            = 'align-type' , OMS , OMS ;

Correspondence       = CorrespondenceBlock
                     | SingleCorrespondence
                     | 'default-correspondence'[63] ;                    Note(63)
CorrespondenceBlock  = 'correspondence-block' , [ RelationRef ] , [ Confidence ][64]   Note(64)
                                                  { Correspondence } ;
SingleCorrespondence = 'correspondence' , SymbolRef , [ RelationRef ] , [ Confidence ] ,
                                          TermOrSymbolRef , [ CorrespondenceID ][65]   Note(65)
 ;
CorrespondenceID     = IRI ;
SymbolRef            = IRI ;
TermOrSymbolRef      = Term | SymbolRef ;
RelationRef          = 'subsumes' | 'is-subsumed' | 'equivalent' | 'incompatible'
                     | 'has-instance' | 'instance-of' | 'default-relation'[66]   Note(66)
 | IRI ;
Confidence           = Double[67] ;                                      Note(67)
Double               = ? a number ∈ [0,1] ? ;
```

[68]   Note(68)

A symbol map in an interpretation is **required** to cover all non-logical symbols of the source OMS; the semantics specification in clause 10 makes this assumption[4]. Applications **shall** implicitly map those non-logical symbols of the source OMS, for which an explicit mapping is not given, to non-logical symbols of the same (local) name in the target OMS, wherever this is uniquely defined – in detail:

```
Require: $O_s, O_t$ are OMS
Require: $M \subseteq \Sigma(O_s) \times \Sigma(O_t)$ maps non-logical symbols (i.e. elements of the signature) of
  $O_s$ to non-logical symbols of $O_t$
  for all $e_s \in \Sigma(O_s)$ not covered by $M$ do
    $n_s \leftarrow \text{localname}(e_s)$
    $N_t \leftarrow \{\text{localname}(e) | e \in \Sigma(O_t)\}$
    if $N_t = \{e_t\}$ then {i.e. if there is a unique target}
      $M \leftarrow M \cup \{(e_s, e_t)\}$
    end if
```

---

[63] Note: TODO: add concrete syntax, plus explanation: applies current default correspondence to all non-logical symbols with the same local names, using the "same local name" algorithm presented elsewhere

[64] Note: TODO: How do we say that at least one of these should be given?

[65] Note: TODO: concrete syntax, e.g., a = x, b my:similarTo y %(correspond-b-to-y)%, c my:similarTo 0.75 z

[66] Note: TODO: say that, unless a different default is specified in a surrounding `CorrespondenceBlock`, the default is `'equivalent'`

[67] Note: TODO: check if Double really makes sense for *implementations*, maybe we'd like to compare confidence values for equality

[68] Note: TODO: cite Alignment API for `RelationRef`; recommend linked data for `RelationRef = IRI`, or recommend registry?

[4] Mapping a non-logical symbol twice is an error. Mapping two source non-logical symbols to the same target non-logical symbol is legal, this then is a non-injective OMS mapping.

```
  end for
Ensure: $M$ completely covers $\Sigma(O_s)$
```

The local name of a non-logical symbol is determined as follows[5]:

```
Require: $e$ is a non-logical symbol (identified by an IRI; cf. clause 9.5)
  if $e$ has a fragment $f$ then {production ifragment in IETF/RFC 3987:2005}
    return $f$
  else
    $n \leftarrow$ the longest suffix of $e$ that matches the Nmtoken production of XML W3C/TR REC-xml:2008
    return $n$
  end if
```

[69]

Note(69)

9.5. Identifiers

This section specifies the abstract syntax of identifiers of DOL OMS and their elements.

9.5.1. IRIs

In accordance with best practices for publishing OMS on the Web, identifiers of OMS and their elements **should** not just serve as *names*, but also as *locators*, which, when dereferenced, give access to a concrete representation of an OMS or one of its elements. (For the specific case of RDFS and OWL OMS, these best practices are documented in [20]. The latter is a specialization of the linked data principles, which apply to any machine-processable data published on the Web [28].) It is recommended that publicly accessible DOL OMS be published as linked data.

[70]Therefore, in order to impose fewer conformance requirements on applications, DOL commits to using IRIs for identification IETF/RFC 3987:2005. It is **recommended** that distributed OMS use IRIs that translate to URLs when applying the algorithm for mapping IRIs to URIs specified in IETF/RFC 3987:2005, Section 3.1. DOL descriptions of any element of a distributed OMS that is identified by a certain IRI **should** be *located* at the corresponding URL, so that agents can locate them. As IRIs are specified with a concrete syntax in IETF/RFC 3987:2005, DOL adopts the latter into its abstract syntax as well as all of its concrete syntaxes (serializations)[71] .

Note(70)

Note(71)

In accordance with semantic web best practices such as the OWL Manchester Syntax [17], this OMG Specification does not allow relative IRIs, and does not offer a mechanism for defining a base IRI, against which relative IRIs could be resolved.

Concerning these languages, note that they allow arbitrary IRIs in principle, but in practice they strongly recommend using IRIs consisting of two components [20]:

---

[5] In practice, this can often have the effect of undoing an IRI abbreviation mechanism that was used when writing the respective OMS (cf. clause 9.5). In general, however, functions that turn abbreviations into IRIs are not invertible. For this reason, the implicit mapping of non-logical symbols is specified independently from IRI abbreviation mechanisms possibly employed in the OMS.

[69] NOTE: some text that was left over here, but I don't recall what we meant by it: recommendations for dealing with OMS language dialects

[70] NOTE: Q-AUT: Does this motivation/justification sound reasonable to you?

[71] NOTE: Q-ALL: I meant to say: for IRIs, the abstract syntax is the same as the concrete syntax.

namespace an IRI that identifies the complete OMS (a *basic OMS* in DOL terminology), usually ending with # or /

local name a name that identifies a non-logical symbol within an OMS

```
IRI     = 'full-iri' , FullIRI | 'curie' , CURIE[6] ;
FullIRI = ? as defined by the IRI production in IETF/RFC 3987:2005 ? ;
```

9.5.2. Abbreviating IRIs using CURIEs

As IRIs tend to be long, and as syntactic mechanisms for abbreviating them have been standardized, it is **recommended** that applications employ such mechanisms and support expanding abbreviative notations into full IRIs. For specifying the *semantics* of DOL, this OMG Specification assumes full IRIs everywhere, but the DOL abstract *syntax* adopts CURIEs (compact URI expressions) as an abbreviation mechanism, as it is the most flexible one that has been standardized to date.

The CURIE abbreviation mechanism works by binding prefixes to IRIs. A CURIE consists of a *prefix*, which may be empty, and a *reference*. If there is an in-scope binding for the prefix, the CURIE is valid and expands into a full IRI, which is created by concatenating the IRI bound to the prefix and the reference.

DOL adopts the CURIE specification of RDFa Core 1.1 W3C/TR REC-rdfa-core-20120607, Section 6 with the following changes:

- DOL does not support the declaration of a “default prefix” mapping [72] (covering CURIEs such as `:name`). Note(72)
- DOL does support the declaration of a “no prefix” mapping (covering CURIEs such as `name`).
- DOL does not make use of the `safe_curie` production.
- DOL does not allow binding a relative IRI to a prefix.
- Concrete syntaxes of DOL are encouraged but **not required** to support CURIEs.[7]

CURIEs can occur in any place where IRIs are allowed, as stated in clause 9.5.1. Informatively, we can restate the CURIE grammar supported by DOL as follows:

```
CURIE     = [ Prefix ] , Reference ;
Prefix    = NCName , ':' (* see “NCName” in W3C/TR REC-xml-names:2009, Sec-
tion 3 *) ;
Reference = Path , [ Query ] , [ Fragment ] ;
Path      = ipath-absolute | ipath-rootless | ipath-empty
            (* as defined in IETF/RFC 3987 *) ;
Query     = '?' , iquery (* as defined in IETF/RFC 3987 *) ;
Fragment  = '#' , ifragment (* as defined in IETF/RFC 3987 *) ;
```

---

[6] specified below in clause 9.5.2

[72] Note: Q-AUT: Are such explanatory notes OK here?

[7] This is a concession to having an RDF-based concrete syntax among the normative concrete syntaxes. RDFa is the only standardized RDF serialization to support CURIEs so far. Other serializations, such as RDF/XML or Turtle, support a subset of the CURIE syntax, whereas some machine-oriented serializations, including N-Triples, only support full IRIs.

Prefix mappings can be defined at the beginning of a distributed OMS (specified in clause 9.2; these apply to all parts of the distributed OMS, including basic OMS as clarified in clause 9.5.3). Their syntax is:

```
PrefixMap         = 'prefix-map' , { PrefixBinding } ;
PrefixBinding     = 'prefix-binding' , BoundPrefix , IRIBoundToPrefix ;
BoundPrefix       = 'bound-prefix' , [ Prefix ] ;
IRIBoundToPrefix = 'full-iri' , FullIRI ;
```

Bindings in a prefix map are evaluated from left to right. Authors **should not** bind the same prefix twice, but if they do, the later binding wins.

9.5.3. Mapping identifiers in basic OMS to IRIs

While DOL uses IRIs as identifiers throughout, basic OMS languages do not necessarily do; for example:

- OWL W3C/TR REC-owl2-syntax:2009, Section 5.5 does use IRIs.
- Common Logic ISO/IEC 24707:2007 supports them but does not enforce their use.
- F-logic [26] does not use them at all.

However, DOL OMS mappings as well as [73] certain operations on OMS require making unambiguous references to non-logical symbols of basic OMS (`SymbolRef`). Therefore, DOL provides a function that maps global identifiers used within basic OMS to IRIs. This mapping affects all non-logical symbol identifiers (such as class names in an OWL ontology), but not locally-scoped identifiers such as bound variables in Common Logic ontologies. DOL reuses the CURIE mechanism for abbreviating IRIs for this purpose (cf. clause 9.5.2). Note(73)

CURIEs that have a prefix may not be acceptable identifiers in every serialization of a basic OMS language, as the standard CURIE separator character, the colon (`:`), may not be allowed in identifiers. [74] Therefore, the declaration of DOL-conformance of the respective serialization (cf. clause 2.2) **may** define an *alternative CURIE separator character*, or it **may** forbid the use of prefixed CURIEs altogether. Note(74)

The IRI of a non-logical symbol identifier in a basic OMS O is determined by the following function:

```
Require: $D$ is a distributed OMS
Require: $O$ is a basic OMS in serialization $S$
Require: $id$ is the identifier in question, identifying a symbol in $O$ according to the specification of $S$
Ensure: $i$ is an IRI
  if $id$ represents a full IRI according to the specification of $S$ then
    $i \leftarrow id$
  else
    {first construct a pattern $cp$ for CURIEs in $S$, then match $id$ against that pattern}
    if $S$ defines an alternative CURIE separator character $cs$ then
      $sep \leftarrow cs$
    else if $S$ forbids prefixed CURIEs then
```

---

[73] Note: TODO: maybe clarify which ones, by checking the grammar for all occurrences of SymbolRef

[74] Note: Q-ALL: I recall that in the 2012-04-18 teleconference we agreed on this – but does it really make sense? Are there any relevant OMS language serializations that do not allow : in identifiers (or that do allow it theoretically but discourage it in practice) but allow some other non-letter character?

```
    $sep$ ← undefined
  else
    $sep$ ← : {the standard CURIE separator character}
  end if
  {The following statements construct a modified EBNF grammar of CURIEs; see ISO/IEC
  14977:1996 for EBNF, and clause 9.5.2 for the original grammar of CURIEs.}
  if $sep$ is defined then
    $cp$ ← $[NCName, sep], Reference$
  else
    $cp$ ← $Reference$
  end if
  if $id$ matches the pattern $cp$, where $ref$ matches $Reference$ then
    if the match succeeded with a non-empty $NCName$ $pn$ then
      $p$ ← $concat(pn, :)$
    else
      $p$ ← no prefix
    end if
    if $O$ binds $p$ to an IRI $pi$ according to the specification of $S$ then
      $nsi$ ← $pi$
    else
      $P$ ← the innermost prefix map in $D$, starting from the place of $O$ inside $D$, and
      going up the abstract syntax tree towards the root of $D$
      while $P$ is defined do
        if $P$ binds $p$ to an IRI $pi$ then
          $nsi$ ← $pi$
          break out of the while loop
        end if
        $P$ ← the next prefix map in $D$, starting from the place of the current $P$ inside
        $D$, and going up the abstract syntax tree towards the root of $D$
      end while
      return an error
    end if
    $i$ ← $concat(nsi, ref)$
  else
    return an error
  end if
end if
return $i$
```

This mechanism applies to basic OMS given inline in a distributed OMS document (`BasicOMS`), not to OMS in external documents (`OMSInConformingLanguage`); the latter **shall** be self-contained.

While CURIEs used for identifying parts of a distributed OMS (cf. clause 9.5.2) are merely syntactic sugar, the prefix map for a basic OMS is essential to determining the semantics of the basic OMS within the distributed OMS. Therefore, any DOL serialization **shall** provide constructs for expressing such prefix maps, even if the serialization does not support prefix maps otherwise.

[75] Note(75)

9.6. DOL Serializations

Say how existing OMS in existing serializations have to be adapted/wrapped (or ideally: not adapted at all!) in order to become valid OMS in some DOL serialization.[76][77] Note(76) Note(77)

9.7. Annotations

[78] [79] Annotations always have a subject, which is identified by an IRI. Where the given OMS language does not provide a way of assigning IRIs to a desired subject of an annotation (e.g. if one wants to annotate an import in OWL), a distributed OMS may employ RDF annotations that use XPointer or IETF/RFC 5147 as means of non-destructively referencing pieces of XML or text by URI.[8] Note(78) Note(79)

---

[75] NOTE: TODO: somewhere we need to mention semantic annotations to embedded fragments in conforming OMS languages, e.g. %implied

[76] NOTE: TODO: Essential points are:– need to be able to say: "the file at URL U is in OWL 2 Manchester syntax"– maybe use packaging/wrapping format– compare MIME types, HTTP content negotiation (but don't go too deep into communication protocols)

[77] NOTE: Reply: Maybe we can implement something like the Linux command "file"?

[78] NOTE: this subclause will be moved to annex M

[79] NOTE: TODO: Properly integrate this text from our LaRC 2011 paper

[8] We intend to utilise the extensibility of the XPointer framework by developing additional XPointer schemes, e.g. for pointing to subterms of Common Logic sentences.

10. DOL semantics

We pursue a threefold approach of assigning a semantics to the DOL abstract syntax:

Direct Model-Theoretic Semantics On the level of basic OMS, this semantics reuses the existing semantics of the involved logics, as well as translations between these logics. The semantics of structured DOL OMS and OMS mappings is specified on top of this.

Translational Semantics The semantics of Common Logic is employed for all basic OMS languages, taking advantage of the fact that Common Logic is a common translation target for many OMS languages. In detail, the translational semantics first translates the DOL abstract syntax of into the abstract syntax of $DOL(\mathsf{CL})$, where $DOL(\mathsf{CL})$ is the homogeneous restriction of DOL to distributed OMS with all parts written in Common Logic only. The latter is interpreted as in the case of the direct semantics, with basic OMS interpreted in terms of the existing Common Logic semantics.

Collapsed Semantics The collapsed semantics extends the translational semantics to a semantics that is fully given specified in Common Logic. It further translates the abstract syntax $DOL(\mathsf{CL})$ to Common Logic, and then reuses the semantics of Common Logic, without employing a separate semantics for the DOL language. Here, the meta and object levels are collapsed into Common Logic, but may still be distinguished by a closer look into the Common Logic theory.

The model-theoretic nature of the semantics ensures a better representation of the model theory than a theory-level semantics would do. In particular, Theorem 13 of [36] ensures that models classes of logical theories represented in Common Logic can be recovered through a model translation. This is of particular importance when studying model-theoretic properties like finite model or tree model properties.

We now specify the theoretical foundations of the semantics of DOL.[80] Since DOL involves heterogeneous OMS, the semantics is parameterised over an arbitrary but fixed *heterogeneous logic environment*. This notion is defined below, it corresponds to a graph of OMS languages and OMS language translations. Below, also notions of *institute* and *institute comorphism* are defined, which provide formalisations of the terms "logic", resp. "logic translation". Note(80)

[81] The notion of institute deliberately avoids the use of category theory in order to keep the mathematical background simple. Most of the abstract syntax can be interpreted using institutes, but not all of it. More specifically, the notion of institute needs to be replaced by that of institutional logic [[34]], and analogously for comorphisms [[13]]. Note(81)

Details of the mapping of the abstract syntax into the semantic domains given by the heterogeneous logic environment will be provided later.

We recall the notion of satisfaction system [7], called 'rooms' in the terminology of [11]. They capture the Tarskian notion of satisfaction of a sentence in a model. For the semantics of minimization, we assume a pre-order on models.

---

[80] NOTE: TODO: later on we also need to say something about the semantics of the syntax. TM: what is this?

[81] NOTE: use institutions here

Definition 1 *A triple* $\mathcal{R} = (Sen, \mathcal{M}, \models)$ *is called a* ***satisfaction system****, or* ***room****, if* $\mathcal{R}$ *consists of*

- *a set* Sen *of* ***sentences****,*
- *a pre-ordered class* $\mathcal{M}$ *of* ***models****, and*
- *a binary relation* $\models \subseteq \mathcal{M} \times Sen$*, called the* ***satisfaction relation****.*

While this signature-free treatment enjoys simplicity and is wide-spread in the literature, many concepts and definitions found in logics, e.g. the notion of a conservative extension, involve the *vocabulary* or *signature* Σ used in sentences. Signatures can be extended with new non-logical symbols; abstractly, this leads to an ordering relation on signatures.

Definition 2 *An* ***institute*** $\mathcal{I} = (Sig, \leq, Sen, \mathcal{M}, \models)$ *is a signature-indexed room, i.e. consists of*

- *a preorder* $(Sig, \leq)$ *of signatures;*
- *a room* $(Sen, \mathcal{M}, \models)$*;*
- *a function* $sig : Sen \to Sig$*, giving the (minimal) signature of a sentence;*
- *a function* $sig : Mod \to Sig$*, giving the signature of a model,*
- *for any* Σ_2*-model* M*, a* Σ_1*-model* $M|_{\Sigma_1}$ *(called the* ***reduct****), provided that* $\Sigma_1 \leq \Sigma_2$*,*

such that the following properties hold:

- *given* $\Sigma_1 \leq \Sigma_2$*, for any* Σ_2*-model* M *and any* Σ_1*-sentence* φ
  $$M \models \varphi \text{ iff } M|_{\Sigma_1} \models \varphi$$
  (satisfaction is ***invariant under reduct****),*
- *for any* Σ*-model,* $M|_\Sigma = M$*, and given* $\Sigma_1 \leq \Sigma_2 \leq \Sigma$*,*
  $$(M|_{\Sigma_2})|_{\Sigma_1} = M|_{\Sigma_1}$$
  (reducts are compositional*), and*
- *for any model* M *and sentence* φ*,*
  $$M \models \varphi \text{ implies } sig(M) \geq sig(\varphi)$$
  (signature coherence*).*

Here, the class of models over a signature Σ (short: Σ-models) is defined as

$$Mod(\Sigma) := \{M \in \mathcal{M} | sig(M) = \Sigma\}$$

Note that we here require equality of signature, unlike we did for sentences. The reason is that a model always needs to interpret all of the non-logical symbols of a signature (and not more), while a sentence might use only part of the non-logical symbols of the signature.

EXAMPLE **Propositional Logic** is an institute as follows: Signatures in **Prop** are just sets Σ (of propositional non-logical symbols) as signatures, and signature inclusion is just set inclusion. A Σ-model M is a mapping from Σ to $\{true, false\}$. Σ-sentences are built from

Σ with the usual propositional connectives. Finally, satisfaction of a sentence in a model is defined by the standard truth-table semantics.

Further examples of institutes are: $\mathcal{SROIQ}(D)$, Common Logic, unsorted first-order logic, many-sorted first-order logic, and many others. Note that reduct is generally given by forgetting parts of the model, and the pre-order on models is given as follows: $M_1 \leq M_2$ if M_1 and M_2 only differ in the interpretation of propositional non-logical symbols and predicates, and moreover each propositional (and predicate) symbol true in M_1 is also true in M_2 (for a given tuple of arguments).

Assume an arbitrary institute.

A **theory** is a set $\Delta \subseteq \mathit{Sen}$ of sentences It is **consistent** iff it has at least one model. A theory $\Delta \subseteq \mathit{Sen}$ is **satisfiable**, if it has a model M (i.e., a model $M \in \mathcal{M}$ such that $M \models \varphi$ for $\varphi \in \Delta$). **Semantic entailment** is defined as usual: for a theory $\Delta \subseteq \mathit{Sen}$ and $\varphi \in \mathit{Sen}$, we write $\Delta \models \varphi$, if all models satisfying all sentences in Δ also satisfy φ.

Lemma 3 (Coincidence Lemma) *Let Δ be a theory with $sig(\Delta) = \Sigma$, and φ a sentence. For determining whether the semantic entailment $\Delta \models \varphi$ holds, it suffices to consider Σ-models only.*

Corridors are the mappings between rooms. A corridor maps both sentences and models (syntax and semantics). Models are mapped in reverse direction. The rationale behind this is as follows: usually, the target room is either logically more expressive or well-suited for logical coding. Sentences of the source room are represented, or coded in the target room. Models of the target room are usually richer, so that from a model in the target room, a model in the source room can be extracted.

Definition 4 *A **corridor** $(\alpha, \beta)\colon (\mathit{Sen}_1, \mathcal{M}_1, \models_1) \longrightarrow (\mathit{Sen}_2, \mathcal{M}_2, \models_2)$ consists of*

- *a sentence translation function $\alpha\colon \mathit{Sen}_1 \longrightarrow \mathit{Sen}_2$, and*
- *a model reduction function $\beta\colon \mathcal{M}_2 \longrightarrow \mathcal{M}_1$, such that*

$$M_2 \models_2 \alpha(\varphi_1) \text{ if and only if } \beta(M_2) \models_1 \varphi_1$$

*holds for each $M_2 \in \mathcal{M}_2$ and each $\varphi_1 \in \mathit{Sen}_1$ **(satisfaction condition)**.*

*A **partial corridor** is one where β is partial, and the satisfaction condition is only required for those M_2 such that $\beta(M_2)$ is defined.*

A corridor is called **model-expansive**, if β is a surjection.

Definition 5 (Relative Interpretation) *Given Δ_i a theory in $\mathcal{R}_i$ $(i = 1, 2)$, a corridor $(\alpha, \beta) : \mathcal{R}_1 \to \mathcal{R}_2$ is a **relative interpretation**, if*

$$\beta(\mathsf{Mod}(\Delta_2)) \subseteq \mathsf{Mod}(\Delta_1)$$

Institute comorphisms capture the intuition of translating a logic into another one. They extend corridors by mapping also signatures.

Definition 6 *Given institutes $\mathcal{I}_1 = (\mathit{Sig}_1, \leq_1, \mathit{Sen}_1, \mathcal{M}_1, \models_1)$ and $\mathcal{I}_2 = (\mathit{Sig}_2, \leq_2, \mathit{Sen}_2, \mathcal{M}_2, \models_2)$, an **institute comorphism** $\rho = (\Phi, \alpha, \beta) : \mathcal{I}_1 \longrightarrow \mathcal{I}_2$ consists of*

- *a monotone map $\Phi : (\mathit{Sig}^1, \leq^1) \to (\mathit{Sig}^2, \leq^2)$, and*
- *a partial corridor $(\alpha, \beta) : (\mathit{Sen}_1, \mathcal{M}_1, \models_1) \to (\mathit{Sen}_2, \mathcal{M}_2, \models_2)$*

such that

- $sig^2(\alpha(\varphi_1)) \leq \Phi(sig^1(\varphi_1))$ *for any sentence* $\varphi_1 \in Sen^1$*;*
- *for each* $\mathcal{I}_1$*-signature* Σ*,* β *restricts to a total function* $\beta_\Sigma : Mod_2(\Phi(\Sigma)) \to Mod_1(\Sigma)$*;*
- *model translation commutes with reduct, that is, given* $\Sigma_1 \leq \Sigma_2$ *in* $\mathcal{I}_1$ *and a* $\Phi(\Sigma_2)$*-model* M *in* $\mathcal{I}_2$*,*

$$\beta_{\Sigma_2}(M)|_{\Sigma_1} = \beta_{\Sigma_1}(M|_{\Phi(\Sigma_1)}).$$

An institute comorphism is called **model-expansive**, if all β_Σ are surjective.

A **subinstitute comorphism** is a institute comorphism $(\Phi, \alpha, \beta) \colon \mathcal{I}_1 \longrightarrow \mathcal{I}_2$ with Φ injective and preorder-reflecting, α injective and β_Σ bijective for each Σ. In this case, $\mathcal{I}_1$ is said to be a **subinstitute** of $\mathcal{I}_2$.

A **simple theoroidal comorphism** is like a comorphism, except that the signature translation functor Φ maps signatures to *theories* over the target institute.

Institute morphisms capture the intuition of reducing a logic into another one, and are used for logic reductions.

Definition 7 *Given institutes* $\mathcal{I}_1 = (Sig_1, \leq_1, Sen_1, \mathcal{M}_1, \models_1)$ *and* $\mathcal{I}_2 = (Sig_2, \leq_2, Sen_2, \mathcal{M}_2, \models_2)$*, an* ***institute morphism*** $\mu = (\Phi, \alpha, \beta) : \mathcal{I}_1 \longrightarrow \mathcal{I}_2$ *consists of*

- *a monotone map* $\Phi : (Sig^1, \leq^1) \to (Sig^2, \leq^2)$*, and*
- *a partial corridor* $(\alpha, \beta) : (Sen_2, \mathcal{M}_2, \models_2) \to (Sen_1, \mathcal{M}_1, \models_1)$

such that

- $\Phi(sig^1(\alpha(\varphi_2))) \leq sig^2(\varphi_2)$ *for any sentence* $\varphi_2 \in Sen^2$*;*
- *for each* $\mathcal{I}_1$*-signature* Σ*,* β *restricts to a total function* $\beta_\Sigma : Mod_1(\Phi(\Sigma)) \to Mod_2(\Sigma)$*;*
- *model translation commutes with reduct, that is, given* $\Sigma_1 \leq \Sigma_2$ *in* $\mathcal{I}_1$ *and a* Σ_2*-model* M*,*

$$\beta_{\Sigma_2}(M)|_{\Phi(\Sigma_1)} = \beta_{\Sigma_1}(M|_{\Sigma_1}).$$

[82] Note(82)

10.1. Direct semantics of DOL language constructs

The semantics of DOL is based on a fixed (but in principle arbitrary) heterogeneous logical environment is assumed. The semantic domains are based on this heterogeneous logical environement. A specific heterogeneous logical environment is given in the annexes.

A heterogeneous logical enviroment is given by a collection of OMS languages and OMS language translations[1], a collection of institutes, institute morphisms and institute comorphisms (serving as logics, logic reductions and logic translations), and a collection of serializations. Moreover, there is a binary supports relation between OMS languages and institutes, and a binary supports relation between OMS languages and serializations. Some of the comorphisms are marked as default translations.

For pairs of institutes $\mathcal{I}_1$ and $\mathcal{I}_2$, we assume a pair of default union institute comorphisms $(\Phi_i, \alpha_i, \beta_i) \colon \mathcal{I}_i \longrightarrow \mathcal{I}$ into a common target institute. The default union may also be undefined.

We also assume a language-specific semantics of basic OMS, depending on a triple $L = (lang, logic, ser)$ comprising of an OMS language, a logic (institute) and a serialization as follows:

---

[82] Note: Introduce exactness

[1] The terms *OMS language* and *serialization* are not defined formally. For this semantics, it suffices to know that there is a language-specific semantics of basic OMS as defined below.

$$\boxed{\mathit{sem}_L(\Sigma, \texttt{BasicOMS}) = (\Sigma', \Delta') \text{ where } \Sigma' \geq \Sigma}$$

This is given by semantics of `BasicOMS` in L. The signature Σ is the *local environment* of non-logical symbols that have been declared previously to `BasicOMS`. $\Sigma' \geq \Sigma$ is an extension of Σ with the non-logical symbols declared in `BasicOMS`. Δ' is a set of sentences over Σ'.

We further assume a language-specific semantics of complete (possibly structured) OMS $\mathit{sem}(L, \texttt{OMSInSpecificLanguage}) = (\Sigma, \mathcal{M})$, where Σ is a signature and $\mathcal{M}$ a class of models over Σ.

We assume that in each institute there is a trivial signature $\emptyset$ with model class $\mathcal{M}_\emptyset$. Moreover, we assume that for each signature Σ, there is a set of non-logical symbols $\mathit{ent}(\Sigma)$, such that $\Sigma \leq \Sigma'$ implies $\mathit{ent}(\Sigma) \subseteq \mathit{ent}(\Sigma')$. This concludes the definition of heterogeneous logical enviroment.

The semantics of OMS generally depends on a global environment Γ mapping `IRIs` to semantics of OMS (given below), and a current triple L consisting of the current language, logic and serialization.[2]

[83] Note(83)

$$\boxed{\mathit{sem}(\Gamma, L, \texttt{DistOMSDefn}) = \Gamma'}$$

$\mathit{sem}(\Gamma, L, \texttt{'dist-oms-defn'}\ \ ,\ \texttt{DistOMSName}\ DI_1, \ldots DI_n) = \Gamma'$
where $\mathit{sem}(\ldots \mathit{sem}(\mathit{sem}(\Gamma, L, DI_1), DI_2), \ldots, DI_n) = (\Gamma', L')$ [84] Note(84)

$\mathit{sem}(\Gamma, L, \texttt{OMSInSpecificLanguage}) = \Gamma'$
where $\Gamma' = \Gamma[\texttt{IRI} \mapsto (L, \Sigma, \mathcal{M})]$,
$(\Sigma, \mathcal{M}) = \mathit{sem}(L, \texttt{OMSInSpecificLanguage})$
and `IRI` is the IRI of `OMSInSpecificLanguage`.

$$\boxed{\mathit{sem}(L, \texttt{Qualification}) = L'}$$

$\mathit{sem}((\mathit{lang}, \mathit{logic}, \mathit{ser}), \texttt{'lang-select'}\ \ ,\ \texttt{LanguageRef}) = (\texttt{LanguageRef}, \mathit{logic}', \mathit{ser}')$

where $\mathit{logic}' = \begin{cases} \mathit{logic}, & \text{if } \texttt{LanguageRef} \text{ supports } \mathit{logic} \\ \text{default logic for } \texttt{LanguageRef}, & \text{otherwise} \end{cases}$

$\mathit{ser}' = \begin{cases} \mathit{ser}, & \text{if } \texttt{LanguageRef} \text{ supports } \mathit{ser} \\ \text{default serialization for } \texttt{LanguageRef}, & \text{otherwise} \end{cases}$

$\mathit{sem}((\mathit{lang}, \mathit{logic}, \mathit{ser}), \texttt{'logic-select'}\ \ ,\ \texttt{LogicRef}) = (\mathit{lang}', \texttt{LogicRef}, \mathit{ser})$

where $\mathit{lang}' = \begin{cases} \mathit{lang}, & \text{if } \mathit{lang} \text{ supports } \texttt{LogicRef} \\ \text{the unique language supporting } \texttt{LogicRef}, & \text{otherwise} \end{cases}$

Note that "the unique language supporting `LogicRef`" may be undefined; in this case, the semantics of the whole `'logic-select'  , LogicRef` construct is undefined.

---

[2] The initial L is obtained from the file name extension of the file containing a particular distributed OMS, while Γ is obtained by looking up IRIs in the internet and applying the semantics to thus obtained OMS.

[83] Note: Q-AUT: @TM: Please decide if you like the stuff from 'dist-oms-defn'. I have now used literal ISO-conforming EBNF syntax here, which means that keywords are enclosed in single quotes, and all tokens separated by commas.

[84] Note: `DistOMSName` is not used. How could we use it? It seems that the individual OMS are directly named with IRIs, and the `DistOMSName` is not relevant for that? Answer from telco: The `DistOMSName` is an IRI that should (as a good practice, but not enforced) agree with the IRI of the document. Indeed, this applies to any usage of IRI in the standard. This should be stated in the standard (Christoph). (This is known as "linked data compliance", a good practice to be encouraged but not to be enforced, as it would break a lot of old OMS)

$sem((lang, logic, ser), \texttt{'logic-select'} \ , \ \texttt{SyntaxRef}) = (lang, logic, \texttt{SyntaxRef})$
The semantics is defined only if *lang* supports `SyntaxRef`.

$$\boxed{sem(L, \texttt{Qualification}\star) = L'}$$

$sem(L, Q_1 \ldots Q_n) = sem(\ldots sem(sem(L, Q_1), Q_2), \ldots, Q_n)$

$$\boxed{sem(\Gamma, L, \texttt{DistOMSItem}) = (\Gamma', L')}$$

$sem(\Gamma, L, \texttt{Qualification}) = (\Gamma, L')$ where $L' = sem(L, \texttt{Qualification})$.

Equations for `OMSDefn` and `MappingDefn` are given below.

$$\boxed{sem(\Gamma, L, (\Sigma, \mathcal{M}), \texttt{MinimizableOMS}) = (I, \Sigma', \mathcal{M}')}$$

In the context of a global environment Γ, the current language, logic and serialization L, and a local environment $(\Sigma, \mathcal{M})$ (of previously declared non-logical symbols), an OMS (`MinimizableOMS`) O is intepreted as an institute $\mathcal{I} = logic(\Gamma, L, O)$, a signature $\Sigma = sig(\Gamma, L, O)$ in institute $\mathcal{I}$ and a class of models $\mathcal{M} = Mod(\Gamma, L, O)$ over signature Σ. We combine this into $sem(\Gamma, L, O) = (logic(\Gamma, L, O), sig(\Gamma, L, O), Mod(\Gamma, L, O))$.

$sem(\Gamma, L, (\Sigma, \mathcal{M}), \texttt{BasicOMS}) = (L, \Sigma', \{M' \in Mod(\Sigma') \mid M \models \Delta', M'|_\Sigma \in \mathcal{M}\})$,
where $sem_L(\Sigma, \texttt{BasicOMS}) = (\Sigma', \Delta')$

$sem(\Gamma, L, (\Sigma, \mathcal{M}), \texttt{'oms-ref'} \ , \ \texttt{OMSRef}) = \Gamma(\texttt{OMSRef})$

Note that $\Gamma(\texttt{OMSRef})$ may be undefined. That is, if a reference (IRI) to an OMS is not defined, the semantics of the enclosing DOL construct is undefined.

$$\boxed{sem(\Gamma, L, (\Sigma, \mathcal{M}), \texttt{ExtendingOMS}) = (\mathcal{I}, \Sigma', \mathcal{M}')}$$

Semantics for `MinimizableOMS` is inherited.

The semantics for minimization selects the models that are minimal in the class of all models with the same interpretation for the local environment (= fixed non-logical symbols, in the terminology of circumscription).

$sem(\Gamma, L, (\Sigma, \mathcal{M}), \texttt{'minimize'} \ , \ \texttt{MinimizableOMS}) = (\mathcal{I}, \Sigma', \mathcal{M}'')$,
where $(\Sigma', \mathcal{M}') = sem(\Gamma, L, (\Sigma, \mathcal{M}), \texttt{MinimizableOMS})$
and $\mathcal{M}'' = \{M \in \mathcal{M}' \mid M \text{ is minimal in } \{M' \in \mathcal{M}' \mid M'|_\Sigma = M|_\Sigma\}\}$

$$\boxed{sem(\Gamma, L, \texttt{OMS}) = (\mathcal{I}, \Sigma, \mathcal{M})}$$

`OMS` is interpreted in a context similar to that for `MinimizableOMS`; the difference being that there is no local environment.
[85] [86] [87]

Note(85)

Note(86)

Note(87)

---

[85] Note: TODO: specify semantics of module extraction

[86] Note: TODO: specify semantics of approximation

[87] Note: TODO: specify semantics of implicit translations using default translations

O	$sem(\Gamma, L, O) = \ldots$
`ExtendingOMS`	$sem(\Gamma, L, (\emptyset, \mathcal{M}_\emptyset), \texttt{ExtendingOMS})$
`'minimize-symbol' , OMS , CircMin , CircVars`	$(I, \Sigma, \mathcal{M}')$ where $sem(\Gamma, L, \texttt{OMS}) = (I, \Sigma, \mathcal{M})$, $\Sigma_{min} = sem(\texttt{CircMin}, \Sigma)$, $\Sigma_{var} = sem(\texttt{CircVars}, \Sigma)$, $\Sigma_{fixed} = \Sigma \setminus (\Sigma_{min} \cup \Sigma_{var})$ and $\mathcal{M}' = \{ \quad M \in \mathcal{M} \mid M\|_{\Sigma_{min} \cup \Sigma_{fixed}}$ is minimal in $\{M' \in \mathcal{M}\|_{\Sigma_{min} \cup \Sigma_{fixed}} \mid M'\|_{\Sigma_{fixed}} = M\|_{\Sigma_{fixed}}\} \quad \}$
`'translation' , OMS , Translation`	$(J, \Phi(\Sigma), \{M \mid \beta(M) \in \mathcal{M}\})$, where $(I, \Sigma, \mathcal{M}) = sem(\Gamma, L, \texttt{OMS})$ and $sem(L, \Sigma, \texttt{Translation}) = (\Phi, \alpha, \beta) : I \to J$
`'reduction' , OMS , Reduction`	$(J, \Sigma', \{\beta(M)\|_{\Sigma'} \mid M \in \mathcal{M}\})$, where $(I, \Sigma, \mathcal{M}) = sem(\Gamma, L, \texttt{OMS})$ and $sem(L, \Sigma, \texttt{Reduction}) = ((\Phi, \alpha, \beta) : I \to J, \Sigma')$
`'approximation', OMS , Approximation`	TODO
`'union' , OMS , [ ConsStrength ] , OMS`	$(\mathcal{I}, \Sigma, \mathcal{M})$ where $\Sigma_i = sig(\Gamma, L, O_i)$, $\mathcal{I}_i = logic(\Gamma, L, O_i)$ $(i = 1, 2)$ $(\Phi_i, \alpha_i, \beta_i) \colon \mathcal{I}_i \longrightarrow \mathcal{I}$ are the default union comorphisms for $\mathcal{I}_1$ and $\mathcal{I}_2$ (if existing) $\Sigma = \Phi_1(\Sigma_1) \vee \Phi_2(\Sigma_2)$ (if the supremum is defined) $\mathcal{M} = \{M \in Mod(\Sigma) \mid \beta_i(M)\|_{\Sigma_i} \in Mod(\Gamma, L, O_i)\}$
`'extension' , OMS , ExtensionOMS`	$sem(\Gamma, L, (\Sigma, \mathcal{M}), \texttt{ExtensionOMS})$
`'module-extract' , OMSRef , Conservative, ExtractionMethod` Σ	TODO
`'qual-oms' , { Qualification } , OMS`	$sem(\Gamma, sem(L, \texttt{\{ Qualification \}}), \texttt{OMS})$

$$\boxed{sem(L, \Sigma, \texttt{Reduction}) = (\mu = (\Phi, \alpha, \beta), \Sigma') \text{ where } \Sigma' \leq \Phi(\Sigma)}$$

$sem(L, \Sigma, \texttt{'hidden'}\ \ LR_1 \ldots LR_n\ (\texttt{'symbol-items'}\ \ EI_1 \ldots EI_n)) = (\mu, \Sigma')$
where $\mu = (\Phi, \alpha, \beta) = sem(LR_n) \circ \cdots \circ sem(LR_1)$
and Σ' is the maximal subsignature of $\Phi(\Sigma)$ with $ent(\Sigma')$ disjoint from $EI_1 \ldots EI_n$. (The semantics is undefined, if such a subsignature does not exist.)

$sem(L, \Sigma, \texttt{'revealed'}\ \ (\texttt{'symbol-items'}\ \ EI_1 \ldots EI_n)) = (id, \Sigma')$
where id is the identity institute morphism, and and Σ' is the minimal subsignature of Σ with $ent(\Sigma')$ containing $EI_1 \ldots EI_n$. (The semantics is undefined, if such a subsignature does not exist.)

$$\boxed{sem(L, \Sigma, \texttt{SymbolItems}) = \Sigma' \text{ where } \Sigma' \leq \Sigma}$$

$sem(L, \Sigma, \texttt{'symbol-items'}\ \ EI_1 \ldots EI_n) = \bigvee \{\Sigma' \leq \Sigma \text{ in } L \mid$ the non-logical symbols in $EI_1 \ldots EI_n$ do not occur in $ent(\Sigma')\}$

$$\boxed{sem(L, \Sigma, \texttt{Translation}) = \rho}$$

$sem(L, \Sigma, \texttt{'renaming'}\ \ LT_1 \ldots LT_n(\texttt{'symbol-map-items'}\ \ E_1 \ldots E_m)) = \rho = (\Phi, \alpha, \beta)$
where $rho = sem(LT_n) \circ \cdots \circ sem(LT_1)$

The semantics is defined only if $E_1 \ldots E_m$ occur in $\Phi(\Sigma)$.

$$\boxed{\mathit{sem}(L, \Sigma, \texttt{SymbolMapItems}) = \Sigma' \text{ where } \Sigma' \geq \Sigma}$$

True renamings are not possible without institutional logics, only the presence of non-logical symbols in a signature can be checked.

$$\mathit{sem}(L, \Sigma, \texttt{'symbol-map-items'}\ E_1 \ldots E_n) = \begin{cases} \Sigma & \text{if } E_1, \ldots, E_n \text{ are contained in } \Sigma \\ \text{undefined} & \text{otherwise} \end{cases}$$

$$\boxed{\mathit{sem}(\Gamma, L, (\Sigma, \mathcal{M}), \texttt{ExtensionOMS}) = (\Sigma', \mathcal{M}')}$$

$\mathit{sem}(\Gamma, L, (\Sigma, \mathcal{M}), \texttt{[ConsStrength] , [ExtensionName] , ExtendingOMS}) = (\Sigma', \mathcal{M}')$ where $(\Sigma', \mathcal{M}') = \mathit{sem}(\Gamma, L, (\Sigma, \mathcal{M}), \texttt{ExtendingOMS})$

If `ConsStrength` is `'model-conservative'` or `'implied'`, the semantics is only defined if each model in $\mathcal{M}$ is the Σ-reduct of some model in $\mathcal{M}'$. In case that `ConsStrength` is `'implied'`, it is furthermore required that $\Sigma = \Sigma'$. If `ConsStrength` is `'consequence-conservative'`, the semantics is only defined if for each Σ-sentence φ, $\mathcal{M}' \models \varphi$ implies $\mathcal{M} \models \varphi$. If `ConsStrength` is `'definitional'`, the semantics is only defined if each model in $\mathcal{M}$ is the Σ-reduct of a unique model in $\mathcal{M}'$.

$$\boxed{\mathit{sem}(\Gamma, L, \texttt{OMSDefn}) = (\Gamma', L)}$$

An OMS definition extends the global environment:

$\mathit{sem}(\Gamma, L, \texttt{'oms-defn' , OMSName , [ConsStrength] , OMS})$ [88] Note(88)

$= (\Gamma[\texttt{OMSName} \mapsto \mathit{sem}(\Gamma, L, \texttt{OMS})], L)$

If `ConsStrength` is `'conservative'`, the semantics is only defined if $\mathit{sem}(\Gamma, L, \texttt{OMS}) \neq \emptyset$. If `ConsStrength` is `'conservative'`, the semantics is only defined if $\mathit{sem}(\Gamma, L, \texttt{OMS})$ is a singleton.

$$\boxed{\mathit{sem}(\texttt{LogicRef}) = L}$$

L is the institute from the heterogeneous logical environment named by `LogicRef`.

$$\boxed{\mathit{sem}(L, \texttt{OMSLangTrans}) = \rho}$$

$\mathit{sem}(L, \texttt{'named-trans' , OMSLangTransRef}) = \rho$ where ρ is the institute comorphism from the heterogeneous logical environment named by `OMSLangTransRef`. This is defined only if the domain of ρ is L.

$\mathit{sem}(L, \texttt{'qual-trans' , OMSLangTransRef}\ \mathit{LR}_1\ \mathit{LR}_2) = \rho$ where ρ is the institute comorphism from the heterogeneous logical environment named by `OMSLangTransRef`. This is defined only if $\rho : \mathit{sem}(\mathit{LR}_1) \to \mathit{sem}(\mathit{LR}_2)$ and $L = \mathit{sem}(\mathit{LR}_1)$.

[89] Note(89)

$\mathit{sem}(L, \texttt{'anonymous-trans'}\ \mathit{LR}_1\ \mathit{LR}_2) = \rho$ where ρ is the unique institute comorphism from the heterogeneous logical environment running from $\mathit{sem}(\mathit{LR}_1)$ to $\mathit{sem}(\mathit{LR}_2)$. This is defined only if $L = \mathit{sem}(\mathit{LR}_1)$.

$\mathit{sem}(L, \texttt{'default-trans' , LolaRef}) = \rho$ where ρ is the unique institute comorphism from the heterogeneous logical environment running from L to $\mathit{sem}(\texttt{LolaRef})$.

---

[88] NOTE: Should we allow for overriding existing OMS definitions? Or should `OMSName` be new?

[89] NOTE: We need some "algorithm" for handling `LolaRef`s that are actually `LanguageRef`s, not `LogicRef`s. Suppose a translation lang1→lang2 is referenced, let e(lang) be the logic that exactly captures the expressivity of lang. For lang1→lang2 there might be a "language-side" default translation, which does not have a corresponding "logic-side" mapping at all, or whose exactly corresponding "logic-side" mapping is not the default for e(lang1)→e(lang2).

$$\boxed{sem(\Gamma, L, \texttt{MappingDefn}) = (\Gamma', L)}$$

See equations for `IntprDefn`, `EquivDefn` and `AlignDefn`.

$$\boxed{sem(\Gamma, L, \texttt{IntprDefn}) = (\Gamma', L)}$$

[90] $sem(\Gamma, L, \texttt{'intpr-defn' , IntprName , ('intpr-type'}\ O_1\ O_2\ \texttt{))} =$ Note(90)
$(\Gamma[\texttt{IntprName} \mapsto (\Sigma_1, \Sigma_2)], L)$ where

- $(\Sigma_1, \mathcal{M}_1) = sem(\Gamma, L, O_1)$;
- $(\Sigma_2, \mathcal{M}_2) = sem(\Gamma, L, O_2)$;
- the semantics is defined only if $\mathcal{M}_2|_{\Sigma_1} \subseteq \mathcal{M}_1$.

$$\boxed{sem(\Gamma, L, \texttt{EquivDefn}) = (\Gamma', L)}$$

$sem(\Gamma, L, \texttt{'equiv-defn' , EquivName , ('equiv-type'}\ O_1\ O_2\ \texttt{)}\ O_3) =$
$(\Gamma[\texttt{EquivName} \mapsto (\Sigma_1, \Sigma_2, \Sigma_3)], L)$ where

- $(\Sigma_1, \mathcal{M}_1) = sem(\Gamma, L, O_1)$;
- $(\Sigma_2, \mathcal{M}_2) = sem(\Gamma, L, O_2)$;
- $(\Sigma_3, \mathcal{M}_3) = sem(\Gamma, L, O_3)$;
- the semantics is defined only if for $i = 1, 2$, $\Sigma_i \leq \Sigma_3$ and each model in $\mathcal{M}_i$ can be uniquely expanded to a model in $\mathcal{M}_3$.

$$\boxed{sem(\Gamma, L, \texttt{AlignDefn}) = (\Gamma', L)}$$

Alignments are interpreted only syntactically:
[91] $sem(\Gamma, L, \texttt{'align-defn' , AlignName , [AlignCard] , AlignType , \{ Correspondence \}}) =$ Note(91)

$(\Gamma[\texttt{AlignName} \mapsto \texttt{\{ Correspondence \}}], L)$

10.2. Translational semantics of DOL language constructs

The translational semantics uses Common Logic as a foundational framework for the distributed OMS, modeling and specification language DOL, similar to what set theory provides for general mathematical theories. This semantics assumes that each involved OMS language is mapped to CL by a weakly exact translation. The semantics is defined by first translating a heterogeneous OMS to CL, and then using the direct semantics for the result.

Note that since the result of translating a DOL OMS entirely to CL is homogeneous, the clause for logic translation of the direct semantics will not be used. Using default logic translations and compositions of these, many logics can be mapped to Common Logic, while the DOL constructs like interpretations stay the same.[3]

---

[90] Note: Q-AUT: Optional `[ Conservative ]` argument is missing.

[91] Note: Q-AUT: Semantics does not yet cover optional `[ AlignCard ]`.

[3] The translational semantics is not applicable for logics without a default translation of Common Logic.

We define the syntactic translation CL_ρ of DOL OMS, depending on a logic translation $\rho : L \to \mathsf{CL}$, to Common Logic below. (The translations of the other syntactic categories are straightforward.)

$CL_\rho() = \langle \Phi(\Sigma), \alpha(\Delta) \rangle$, where $\rho = (\Phi, \alpha, \beta)$
$CL_\rho(O \texttt{ with logic } \rho') = CL_{\rho \circ \rho'}(O)$
92 $CL_\rho(O \texttt{ then } CS\ ) = CL_\rho(O) \texttt{ then } CS\ CL_\rho()$ Note(92)
$CL_\rho(\texttt{OMSRef}) = \texttt{OMSRef}$
$CL_\rho(\texttt{logic LogicRef } O) = CL_{default(\texttt{LogicRef},\mathsf{CL})}(O)$

10.3. Collapsed Semantics of DOL language constructs

The collapsed semantics requires the representation of the meta level within CL. For this purpose, the model-level semantics introduced in the previous section should be complemented by a theory-level semantics: a distributed OMS then denotes a basic theory in some logic (which amounts to flattening out all structure), plus some conditions for conservativity and relative interpretations. For each logic, one needs to axiomatise a specific partial order of signatures in CL, plus a set of sentences equipped with a logical consequence relation. In order to avoid the formalisation of models and the satisfaction relation (which would require the inclusion of a set theory like ZFC), a sound and complete calculus is axiomatised for each logic. For each logic translation, the signature and sentence translations need to be axiomatised. We require that this axiomatisation is done in such a way that the resulting semantics is compatible with the translational semantics. Although this formalisation is doable in principle, we refrain form providing the (massive) details.

93 Note(93)

10.4. OMS language translations

The concept of OMS language translation has been formalized as institute comorphism.

Provide some examples

special cases to be described

---

[92] Note: Extend this to all DOL constructs

[93] Note: Q-ALL: The collapsed semantics is still very vague, and is more a research plan than a definite proposal. Any ideas how to make this more precise?

11. Keyword index

/to be supplemented in the final version/

Annex

A. Annex (normative): DOL text serialization

A.1. Document type

MIME type *application/dol+text*[94] Note(94)

Filename extension .dol[95] Note(95)

A.2. Concrete Syntax

A.2.1. Distributed OMS

```
DistOMS                  = [ PrefixMap ] , DistOMSDefn
                          | OMSInConformingLanguage ;
DistOMSDefn              = 'distributed␣OMS' , DistOMSName , { DistOMSItem } ;
OMSInConformingLanguage = ? language-specific ? ;
DistOMSItem              = OMSDefn | MappingDefn | Qualification ;
Qualification             = LanguageQual | LogicQual | SyntaxQual ;
LanguageQual              = 'language' , LanguageRef ;
LogicQual                 = 'logic' , LogicRef ;
SyntaxQual                = 'serialization' , SyntaxRef ;
DistOMSName              = IRI ;

PrefixMap                 = '%prefix(' , { PrefixBinding } , ')%' ;
PrefixBinding             = BoundPrefix , IRIBoundToPrefix ;
BoundPrefix               = ':' | Prefix (* see definition in clause 9.5.2  *)[96]
 ;
IRIBoundToPrefix          = '<' , FullIRI , '>' ;
```

Note(96)

Note that we denote the empty prefix (called "no prefix" in W3C/TR REC-rdfa-core-20120607, Section 6) by a colon inside the prefix map, but completely omit it in CURIEs. This is the style of the OWL Manchester syntax [17] but differs from the RDFa Core 1.1 syntax.

A.2.2. Heterogeneous OMS

[97] Note(97)

---

[94] Note: FYI: just a placeholder so far, needs discussion

[95] Note: the most intuitive one, but gives the text serialization a privileged role over the others

[96] Note: Q-AUT: I think that, in contrast to OWL Manchester, we can allow prefix names that match keywords of the DOL syntax, as we are enclosing the whole prefix map into an annotation construct – right?

[97] Note: TODO: merge ALIGN-TYPE with INTPR-TYPE

```
BasicOMS             = OMSInConformingLanguage ;
MinimizableOMS       = BasicOMS
                      | OMSRef , [ ImportName ] ;
ExtendingOMS         = MinimizableOMS
                      | MinimizeKeyword , '{' , MinimizableOMS , '}'
                      | OMS , Extraction ;
MinimizeKeyword       = 'minimize' | 'closed-world' ;
OMS                  = ExtendingOMS
                      | OMS , MinimizeKeyword , CircMin , [ CircVars ]
                      | OMS , Translation
                      | OMS , Reduction
                      | OMS , Approximation
                      | OMS , 'and' , [ ConsStrength ] , OMS
                      | OMS , 'then' , ExtensionOMS
                      | { Qualification } , ':' , GroupOMS
                      | OMS, 'bridge', { Translation } , OMS
                      | 'combine' , GraphElements , [ ExcludeExtensions ] ;
                        98

CircMin = Symbol , {  Symbol } ;
CircVars  = 'vars' ,  ( Symbol , { Symbol } ) ;

GroupOMS             = '{' , OMS , '}'
                      | OMSRef ;

Translation           = 'with' , { LogicTranslation } , [ SymbolMapItems ] ;
LogicTranslation      = 'translation' , OMSLangTrans ;

Reduction             = 'hide' , { LogicReduction } , [ SymbolItems ]
                      | 'reveal' , [ SymbolMapItems ] ;
LogicReduction        = 'along' , OMSLangTrans ;

SymbolItems           = Symbol { ',' , Symbol } ;
SymbolMapItems        = SymbolOrMap { ',' , SymbolOrMap } ;

Extraction            = 'extract' , ModuleProperties , InterfaceSignature;
ModuleProperties =  Conservative | '%min' | '%depliting' | '%safe'

Approximation         = 'approximate' , ApproxMethod ;

ExtensionOMS         = [ ConsStrength ] , [ ExtensionName ] , ExtendingOMS ;

ConsStrength          = Conservative
                      | '%mono'
                      | '%wdef'
                      | '%def'
                      | '%implied' ;
```

Note(98)

---

[98] Note: combine O1 O2 takes all views coming into O1 and O2 into consideration

```
Conservative          = '%ccons'
                      | '%mcons'[99]  ;                                  Note(99)

InterfaceSignature    = SymbolItems ;

ImportName            = '%(' , IRI , ')%' ;
ExtensionName         = '%(' , IRI , ')%' ;

OMSOrMappingorGraphRef         = IRI ;

GraphElements         = CombinedElement \{ ',' , CombinedElement \} ;
CombinedElement       = [ Id , ':' ] , OMSOrMappingorGraphRef ;
ExcludeExtensions     = 'excluding' , ExtensionRef , \{ ',' , ExtensionRef \} ;

OMSkeyword           = 'ontology' | 'onto' | 'specification' | 'spec' | 'model'

OMSDefn              = OMSkeyword , OMSName , '=' , [ ConsStrength ] , OMS , [ 'end' ][100]
 ;                                                                       Note(100)

Symbol                = IRI ;
SymbolMap             = Symbol , '    ' , Symbol ;
SymbolOrMap           = Symbol
                      | SymbolMap ;

OMSName              = IRI ;
IntprName             = IRI ;

OMSRef               = IRI ;
IntprRef              = IRI ;
ExtensionRef          = IRI ;

LanguageRef           = IRI ;
LogicRef              = IRI ;
SyntaxRef             = IRI ;

LoLaRef               = LanguageRef
                      | LogicRef ;

OMSLangTrans         = OMSLangTransRef
                      | OMSLangTransRef , ':' , LoLaRef , '→' , LoLaRef
                      | LoLaRef , '→' , LoLaRef
                      | '→' , LoLaRef ;
```

---

[99] NOTE: Q-AUT: Do we want the CASL-style "cons" as a synonym for "mcons" in the standard? Or just in Hets, as a "hidden feature"? TM: I would say: the latter. CL: OK, I wanted to file this as a Hets ticket, but Trac was down. Let's do it some other time and then remove this comment.

[100] NOTE: TODO: Here and in other, similar contexts, we might need to say more, as "end" is not really always optional. This is the "underlined end" from the CASL specification, but there is no such construct in ISO EBNF.

```
OMSLangTransRef     = IRI ;

ApproxMethod         = 'with' , ApproxMethodRef
                     | 'in' , LoLaRef , 'with' , ApproxMethodRef
                     | 'in' , LoLaRef ;

ApproxMethodRef      = IRI ;

ExtractionMethod     = IRI ;
```

A.2.3. OMS Mappings

```
MappingDefn              = IntprDefn | EquivDefn | ModuleRelDefn | AlignDefn ;

IntprDefn            = IntprKeyword , IntprName , [ Conservative ] , ':' , IntprType , [ '
                     | IntprKeyword , IntprName , [ Conservative ] , ':' , IntprType , '='
                       { LogicTranslation } , [ SymbolMapItems ] , [ 'end' ] ;

IntprKeyword         = 'interpretation' | 'view' ;
IntprName            = IRI ;
IntprType            = GroupOMS , 'to' , GroupOMS ;

EquivDefn            = EquivKeyword , EquivName , ':' , EquivType , '=' , OMS , [ 'end' ]
EquivKeyword         = 'equivalence' ;
EquivName            = IRI ;
EquivType            = GroupOMS , '<->' , GroupOMS ;

ModuleRelDefn        = 'module' , ModuleName , [ Conservative ] , ':' , ModuleType ,
                       'for' , InterfaceSignature ;
ModuleName           = IRI ;
ModuleType           = OMS , 'of' , OMS ;

AlignDefn            = 'alignment' , AlignName , [ AlignCards ] , ':' , AlignType , [ 'end
                     | 'alignment' , AlignName , [ AlignCards ] , ':' , AlignType , '=' ,
                       Correspondence , { ',' , Correspondence } , [ 'end' ] ;

AlignName            = IRI ;
AlignCards           = AlignCardForward , AlignCardBackward ;
AlignCardForward     = AlignCard ;
AlignCardBackward    = AlignCard ;
AlignCard            = '1' | '?' | '+' | '*' ;
AlignType            = GroupOMS , 'to' , GroupOMS[101]  ;

Correspondence       = CorrespondenceBlock
                     | SingleCorrespondence
                     | '*' ;
```

Note(101)

---

[101] NOTE: Q-AUT: would it make sense to merge this with IntprType?

```
CorrespondenceBlock   = 'relation' , [ RelationRef ] , [ Confidence ] ,
                        '{' , Correspondence , { ',' , Correspondence } , '}' ;
SingleCorrespondence = SymbolRef , [ RelationRef ] ,
                        [ Confidence ] , TermOrSymbolRef , [ CorrespondenceId ] ;
CorrespondenceId      = '%(' , IRI , ')%' ;
SymbolRef             = IRI ;
TermOrSymbolRef       = Term | SymbolRef (* Term is logic-specific *) ;[102]
RelationRef           = '>' | '<' | '=' | '%' | [103] '$\ni$' | '$\in$' | '$\mapsto$' | IRI ;
Confidence            = Double ? where Double ∈ [0,1] ? ;
```

Note(102)

Note(103)

A.3. Identifiers

```
IRI     = '<' , FullIRI , '>' | CURIE ;
FullIRI = ? an IRI as defined in \nisref{IETF/RFC 3987:2005} ? ;
CURIE   = ? see \cref{c:curies} ? ;
```

In a CURIE without a prefix, the `reference` part is **not allowed** to match any of the keywords of the DOL syntax (cf. clause).

A.4. Lexical Symbols

The character set for the DOL text serialization is the UTF-8 encoding of Unicode ISO/IEC 10646. However, OMS can always be input in the Basic Latin subset, also known as ASCII.[104] For enhanced readability of OMS, the DOL text serialization particularly supports the native Unicode glyphs that represent common mathematical operators.

Note(104)

A.4.1. Key Words and Signs

The lexical symbols of the DOL text serialization include various key words and signs that occur as terminal symbols in the context-free grammar in annex A.2. Key words and signs that represent mathematical signs are displayed as such, when possible, and those signs that are available in the Unicode character set may also be used for input.

Key Words

Key words are always written lowercase. The following key words are reserved, and are not available for use as complete identifiers[105], although they can be used as parts of tokens:

Note(105)

---

[102] Note: Q-AUT: In interpretations we did away with symbol-to-term mappings, as parsing for them will be hard to implement, and as Michael convinced us with the COLORE example where auxiliary theories using equality take care of the mapping. Do we want to keep them for alignments? (In writing this I have not yet looked into the Alignment API.)

[103] Note: Q-AUT: For `'has-instance'` and `'instance-of'`, the Alignment API does not quite have a symbolic notation, but simply "HasInstance" and "InstanceOf", which, in our syntax, conflicts with abbreviated IRIs. I'd suggest either referring to these relations using normal DOL IRIs (abbreviated or not), or to come up with some symbolic notation. The one I gave here works for Unicode, but I don't really know how to write it in ASCII.

[104] Note: TODO: maybe we need to say something about encoding IRIs as URIs in the latter case

[105] Note: TODO: figure out what that actually means. If we use OWL Manchester's style of abbreviating IRIs, it probably means that in the worst case some IRIs can't be abbreviated but must be given as complete global IRIs

Table A.1.: Key Signs

Sign	Unicode Code Point	Basic Latin substitute
{	U+007B LEFT CURLY BRACKET	
}	U+007D RIGHT CURLY BRACKET	
:	U+003A COLON	
=	U+003D EQUALS SIGN	
,	U+002C COMMA	
↦	U+21A6 RIGHTWARDS ARROW FROM BAR	|->
→	U+2192 RIGHTWARDS ARROW	->

```
  and distributed end hide interpretation library logic minimize
model onto ontology spec specification reveal then to vars view with
```

Key Signs

Table A.1 following key signs are reserved, and are not available for use as complete identifiers. Key signs that are outside of the Basic Latin subset of Unicode may alternatively be encoded as a sequence of Basic Latin characters.

B. Annex (normative): DOL RDF vocabulary

[106] Note(106)

B.1. Document type

DOL RDF does not have one specific document type; instead, it may be represented in any RDF serialization, for example RDF/XML, whose MIME type is *application/rdf+xml*.

RDF namespace `http://purl.net/dol/1.0/rdf#`

For reasons of practical applicability, the RDF vocabulary is given as an OWL ontology[1]. The RDFS subset of this OWL ontology is normative; all features beyond that are informative but intended to be useful for applications supporting DOL.

About mapping identifiers in basic OMS to IRIs (clause 9.5.3), note that prefix maps are not part of the RDF abstract syntax. Therefore, to prevent loss of this semantically essential information, the DOL RDF serialization provides a dedicated vocabulary for expressing prefix maps. A DOL OMS in an RDF serialization that supports prefix maps **may** state them redundantly as syntactic RDF prefixes as well as using the DOL RDF vocabulary.

---

[106] NOTE: Given the agreement to drop the RDF serialization, this is obsolete. Still I need to revise it, as part of this may be relevant w.r.t. the registry vocabulary.

[1] The implementation is available for download as RDF/XML from the namespace URL given above, or as a source file in OWL Manchester Syntax from `http://interop.cim3.net/file/pub/OntoIOp/Working_Draft/syntax/dol-rdf.omn`.

C. Annex (normative): RDF vocabulary for describing OMS languages conforming with DOL

This annex specifies an RDF vocabulary, formalized in RDFS W3C/TR REC-rdf-schema:2004, for describing OMS languages that conform with DOL, and their features, including logics and serializations. This vocabulary shares its namespace (`http://purl.net/dol/1.0/rdf#`) with the DOL RDF vocabulary for serializing DOL OMS (cf. annex B).[107] Note(107)

The tables in this annex list the classes and properties of the RDF vocabulary for describing OMS languages. All class and properties are assumed to be in the DOL RDF namespace unless stated otherwise.

Table C.1 lists the classes of the RDF vocabulary for describing OMS languages. Each row of the table translates into the following RDF triples (given in Turtle serialization):[108] Note(108)

```
_:class rdf:type     rdfs:Class ;
        rdfs:comment "documentation" .
```

Table C.1.: Classes of the RDF vocabulary for describing OMS languages

Class	*documentation*
OMSLanguage	*an OMS language*
Logic	*a logic that defines the semantics of an OMS language*
Serialization	*a serialization of an OMS language*

Table C.2 lists the properties of the RDF vocabulary for describing OMS languages. Each row of the table translates into the following RDF triples (given in Turtle serialization):

```
_:property rdf:type     rdf:Property ;
           rdfs:domain  _:domain ;
           rdfs:range   _:range ;
           rdfs:comment "documentation" .
```

[109] Note(109)

---

[107] Note: FYI: given its light weight I think that makes sense. It doesn't rule out extensions to OWL (or even DOL) anyway.

[108] Note: TODO: also cover rdfs:subClassOf (once we have such cases)

[109] Note: Q-AUT: we need to define "sublogic" as a term – how? I guess that would include the notion of an "OWL profile"

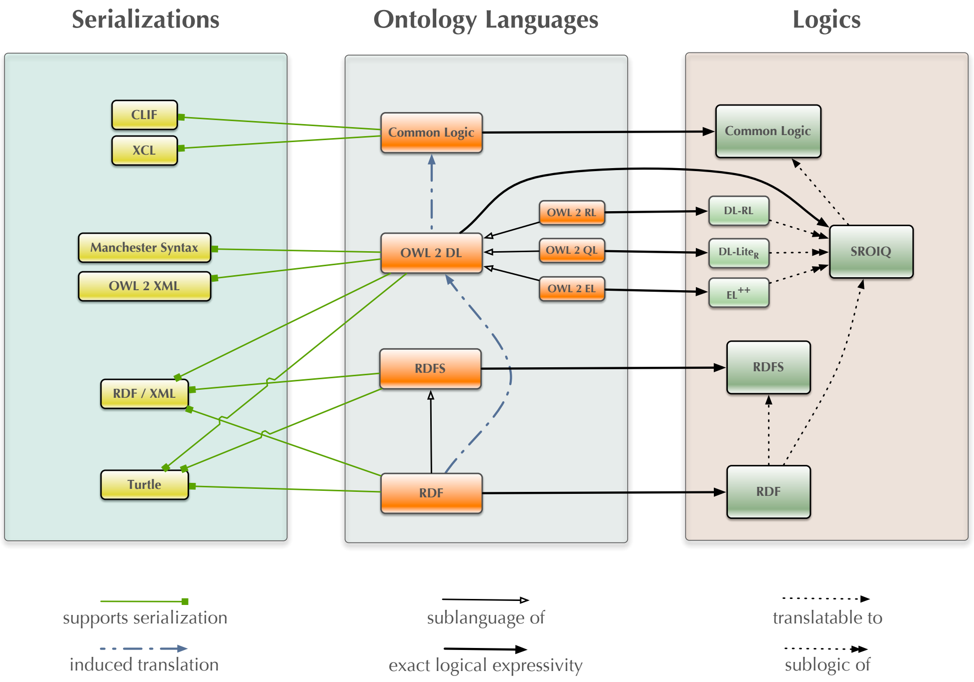

Figure C.1.: Subset of the OntoIOp registry, shown as an RDF graph

Table C.2.: Properties of the RDF vocabulary for describing OMS languages

Property	domain	range	*documentation*
subLogicOf	Logic	Logic	*The subject is a sublogic of the object*
supportsLogic	OMSLanguage	Logic	*The subject OMS language has a semantics specified in terms of the object logic.*
specifiesSemanticsOf	Logic	OMSLanguage	*The subject logic is used to specify the semantics of the object OMS language; inverse of supportsLogic.*
supportsSerialization	OMSLanguage	Serialization	*OMS in the subject OMS language can be serialized in the object serialization. Note that the serialization should be as specific as possible, i.e. one should not say that "OWL can be serialized in XML" and "Common Logic can be serialized in XML", but instead "OWL can be serialized in OWL/XML" and "Common Logic can be serialized in XCL", taking into account that OWL/XML and XCL are two different XML languages.*
serializes	Serialization	OMSLanguage	*The subject logic is used to specify the semantics of the object OMS language; inverse of supportsSerialization.*

D. Annex (normative): Conformance of OWL 2 with DOL

The semantic conformance of OWL 2 (as specified in W3C/TR REC-owl2-syntax:2009) with DOL is established in [35].

The OWL/XML serialization satisfies the criteria for XML conformance. The mapping of OWL 2 to RDF graphs satisfies the criteria for RDF conformance[110] . The OWL 2 Manchester syntax satisfies the criteria for text conformance. [111] Note(110) Note(111)

OWL can be formalised as an institute as follows:

Definition 8 OWL ***2 DL.*** OWL *2 DL is the description logic (DL) based fragment of the web ontology language* OWL *2. We start with the simple description logic $\mathcal{ALC}$, and then proceed to the more complex description logic $\mathcal{SROIQ}$ which is underlying* OWL *2 DL. Signatures of the description logic $\mathcal{ALC}$ consist of a set $\mathcal{A}$ of atomic concepts, a set $\mathcal{R}$ of roles and a set $\mathcal{I}$ of individual constants. The partial order on signatures is defined as component wise inclusion. Models are first-order structures $I = (\Delta^I, .^I)$ with universe Δ^I that interpret concepts as unary and roles as binary predicates (using $.^I$). $I_1 \leq I_2$ if $\Delta^{I_1} = \Delta^{I_2}$ and all concepts and roles of I_1 are subconcepts and subroles of those in I_2. Sentences are subsumption relations $C_1 \sqsubseteq C_2$ between concepts, where concepts follow the grammar*[112] Note(112)

$$C ::= \mathcal{A} \,|\, \top \,|\, \bot \,|\, C_1 \sqcup C_2 \,|\, C_1 \sqcap C_2 \,|\, \neg C \,|\, \forall R.C \,|\, \exists R.C$$

These kind of sentences are also called TBox sentences. Sentences can also be ABox sentences, which are membership assertions of individuals in concepts (written $a : C$ for $a \in \mathcal{I}$) or pairs of individuals in roles (written $R(a, b)$ for $a, b \in \mathcal{I}, R \in \mathcal{R}$). Satisfaction is the standard satisfaction of description logics.

The logic $\mathcal{SROIQ}$ [23], which is the logical core of the Web Ontology Language OWL *2 DL*[1]*, extends $\mathcal{ALC}$ with the following constructs: (i) complex role inclusions such as $R \circ S \sqsubseteq S$ as well as simple role hierarchies such as $R \sqsubseteq S$, assertions for symmetric, transitive, reflexive, asymmetric and disjoint roles (called RBox sentences, denoted by $\mathcal{SR}$), as well as the construct $\exists R.$*Self *(collecting the set of 'R-reflexive points'); (ii) nominals, i.e. concepts of the form $\{a\}$, where $a \in \mathcal{I}$ (denoted by $\mathcal{O}$); (iii) inverse roles (denoted by $\mathcal{I}$); qualified and unqualified number restrictions ($\mathcal{Q}$). For details on the rather complex grammatical restrictions for $\mathcal{SROIQ}$ (e.g. regular role inclusions, simple roles) compare [23].*

OWL profiles *are syntactic restrictions of* OWL *2 DL that support specific modelling and reasoning tasks, and which are accordingly based on DLs with appropriate computational properties. Specifically,* OWL *2* EL *is designed for ontologies containing large numbers of concepts or relations,* OWL *2* QL *to support query answering over large amounts of data, and* OWL *2* RL *to support scalable reasoning using rule languages (*EL, QL, *and* RL *for short) .*

---

[110] NOTE: This is not exactly true, as some things, e.g. imports, can't be identified.

[111] NOTE: also need conformance propositional logic; use PL "profile" of the CASL "IFIP standard"

[112] NOTE: Q-AUT: This grammar should also be adapted to ISO EBNF.

[1] See also `http://www.w3.org/TR/owl2-overview/`

We sketch the logic $\mathcal{EL}$ which is underlying the EL *profile.*[2] *$\mathcal{EL}$ is a syntactic restriction of $\mathcal{ALC}$ to existential restriction, concept intersection, and the top concept:*

$$C ::= \mathcal{A} \,|\, \top \,|\, C_1 \sqcap C_2 \,|\, \exists R.C$$

Note that $\mathcal{EL}$ does not have disjunction or negation, and is therefore a sub-Boolean logic.

---

[2] To be exact, EL adds various 'harmless' expressive means and syntactic sugar to $\mathcal{EL}$ resulting in the DL $\mathcal{EL}$ ++.

E. Annex (normative): Conformance of Common Logic with DOL

The semantic conformance of Common Logic (as specified in ISO/IEC 24707:2007) with DOL is established in [35].

The XCF dialect of Common Logic has a serialization that satisfies the criteria for XML conformance. The CLIF dialect of Common Logic has a serialization that satisfies the criteria for text conformance.

Common Logic can be defined as an institute as follows:

Definition 9 ***Common Logic.*** *A common logic signature Σ (called vocabulary in Common Logic terminology) consists of a set of names, with a subset called the set of discourse names, and a set of sequence markers. An inclusion of signatures needs to fulfil the requirement that a name is a discourse name in the smaller signature if and only if it is one in the larger signature. A Σ-model $I = (UR, UD, rel, fun, int)$ consists of a set UR, the universe of reference, with a non-empty subset $UD \subseteq UR$, the universe of discourse, and four mappings:*

- *rel from UR to subsets of $UD^* = \{< x_1, \dots, x_n > | x_1, \dots, x_n \in UD\}$ (i.e., the set of finite sequences of elements of UD);*
- *fun from UR to total functions from UD^* into UD;*
- *int from names in Σ to UR, such that $int(v)$ is in UD if and only if v is a discourse name;*
- *seq from sequence markers in Σ to UD^*.*

A Σ-sentence is a first-order sentence, where predications and function applications are written in a higher-order like syntax: $t(s)$. Here, t is an arbitrary term, and s is a sequence term, which can be a sequence of terms $t_1 \dots t_n$, or a sequence marker. A predication $t(s)$ is interpreted by evaluating the term t, mapping it to a relation using rel, and then asking whether the sequence given by the interpretation s is in this relation. Similarly, a function application $t(s)$ is interpreted using fun. Otherwise, interpretation of terms and formulae is as in first-order logic. A further difference is the presence of sequence terms (namely sequence markers and juxtapositions of terms), which denote sequences in UD^*, with term juxtaposition interpreted by sequence concatenation. Note that sequences are essentially a non-first-order feature that can be expressed in second-order logic.

Model reducts are defined in the following way: Given a signature inclusion $\Sigma' \leq \Sigma$ and a Σ-model $I = (UR, UD, rel, fun, int)$, $I|_{\Sigma'} = (UR', UD, rel', fun', int')$ is defined by

- *UR' is the restriction of UR to those elements satisfying the following conditions:*
    1. *they are not in the universe of discourse UD;*
    2. *they are the interpretation (according to int) of a non-discourse name in Σ;*
    3. *they are not the interpretation (according to int) of a non-discourse name in Σ'.*
- *rel' is rel restricted to UR';*

- *fun' is fun restricted to UR';*
- *int' is int restricted to Σ'.*

Note that with this notion of reduct, extensions commonly understood as definitions in segregated dialects of Common Logic are indeed both definitional and conservative extensions. [113] Note(113)

We call the restriction of CL *to sentence without sequence markers* CL^{-}.

---

[113] NOTE: Ordering on models! Universes agree, $fun_1(x) = fun_2(x)$, $rel_1(x) \subseteq rel_2(x)$, $int_1(n) = int_2(n)$

F. Annex (normative): Conformance of RDF and RDFS with DOL

The semantic conformance of RDFS (as specified in W3C/TR REC-rdf-schema:2004) with DOL is established in [35].

The way of representing RDFS ontologies as RDF graphs satisfies the criteria for RDF conformance.

Definition 10 (RDF **and** RDFS) *Following [30], we define the institutions for the Resource Description Framework (*RDF*) and* RDF-*Schema (*RDFS*), respectively. These are based on a logic called* bare RDF *(*SimpleRDF*), which consists of triples only (without any predefined resources).*

A signature $\mathbf{R_s}$ *in* SimpleRDF *is a set of resource references. For* $sub, pred, obj \in \mathbf{R_s}$*, a triple of the form* $(sub, pred, obj)$ *is a sentence in* SimpleRDF*, where sub, pred, obj represent subject name, predicate name, object name, respectively. An* $\mathbf{R_s}$*-model* $M = \langle R_m, P_m, S_m, EXT_m\rangle$ *consists of a set* R_m *of resources, a set* $P_m \subseteq R_m$ *of predicates, a mapping function* $S_m : \mathbf{R_s} \to R_m$*, and an extension function* $EXT_m : P_m \to \mathcal{P}(R_m \times R_m)$ *mapping every predicate to a set of pairs of resources. Satisfaction is defined as follows:*

$$\mathfrak{M} \models_{\mathbf{R_s}} (sub, pred, obj) \Leftrightarrow (S_m(sub), (S_m(obj)) \in EXT_m(S_m(pred)).$$

Both RDF *and* RDFS *are built on top of* SimpleRDF *by fixing a certain standard vocabulary both as part of each signature and in the models.*[114] *Actually, the standard vocabulary is given by a certain theory. In case of* RDF*, it contains e.g. resources* `rdf:type` *and* `rdf:Property` *and* `rdf:subject`*, and sentences like, e.g.* (`rdf:type`, `rdf:type`, `rdf:Property`)*, and* (`rdf:subject`, `rdf:type`, `rdf:Property`)*.* Note(114)

In the models, the standard vocabulary is interpreted with a fixed model. Moreover, for each RDF*-model* $M = \langle R_m, P_m, S_m, EXT_m\rangle$*, if* $p \in P_m$*, then it must hold* $(p, S_m(\texttt{rdf:Property})) \in EXT_m(\texttt{rdf:type})$*. For* RDFS*, similar conditions are formulated (here, for example also the subclass relation is fixed).*

In the case of RDFS*, the standard vocabulary contains more elements, like rdf:domain, rdf:range, rdf:Resource, rdf:Literal, rdf:Datatype, rdf:Class, rdf:subClassOf, rdf:subPropertyOf, rdf:member, rdf:Container, rdf:ContainerMembershipProperty.*

There is also OWL *full, an extension of* RDFS *with resources like owl:Thing and owl:oneOf, tailored towards the representation of* OWL.

---

[114] Note: Refer to the RDF standard here.

G. Annex (normative): A Core Logic Graph

This annex provides a core graph of logics and translations, covering those OMS languages whose conformance with DOL is established in the preceding, normative annexes (OWL 2 in annex D, Common Logic in annex E, and RDFS in annex F). The graph is shown in Figure G.1. Its nodes refer to the following OMS languages and profiles:

- RDF W3C/TR REC-rdf-concepts:2004
- RDFS W3C/TR REC-rdf-schema:2004
- EL, QL, RL (all being profiles of OWL) W3C/TR REC-owl2-profiles:2009
- OWL W3C/TR REC-owl2-syntax:2009
- CL (Common Logic) ISO/IEC 24707:2007

The translations are specified in [35]. [115] Note(115)

[116] Note(116)

G.1. EL → OWL and $\mathcal{EL}++ \to \mathcal{SROIQ}(D)$

EL → OWL is the sublanguage inclusion obtained by the syntactic restriction according to the definition of EL, see W3C/TR REC-owl2-profiles:2009. Since by definition, $\mathcal{EL}++$ is a syntactic restrition of $\mathcal{SROIQ}(D)$, $\mathcal{EL}++ \to \mathcal{SROIQ}(D)$ is the corresponding sublogic inclusion.

G.2. QL → OWL and DL-Lite$_R \to \mathcal{SROIQ}(D)$

QL → OWL is the sublanguage inclusion obtained by the syntactic restriction according to the definition of QL, see W3C/TR REC-owl2-profiles:2009. Since by definition, DL-Lite$_R$ is a syntactic restrition of $\mathcal{SROIQ}(D)$, DL-Lite$_R \to \mathcal{SROIQ}(D)$ is the corresponding sublogic inclusion.

G.3. RL → OWL and RL $\to \mathcal{SROIQ}(D)$

RL → OWL is the sublanguage inclusion obtained by the syntactic restriction according to the definition of RL, see W3C/TR REC-owl2-profiles:2009. Since by definition, RL is a syntactic restrition of $\mathcal{SROIQ}(D)$, RL $\to \mathcal{SROIQ}(D)$ is the corresponding sublogic inclusion.

---

[115] NOTE: TODO: Provide linear syntax here (as in the paper)

[116] NOTE: FYI: We need this in order to be able to say something about default translations, and about establishing conformance by translation to a language that already conforms.

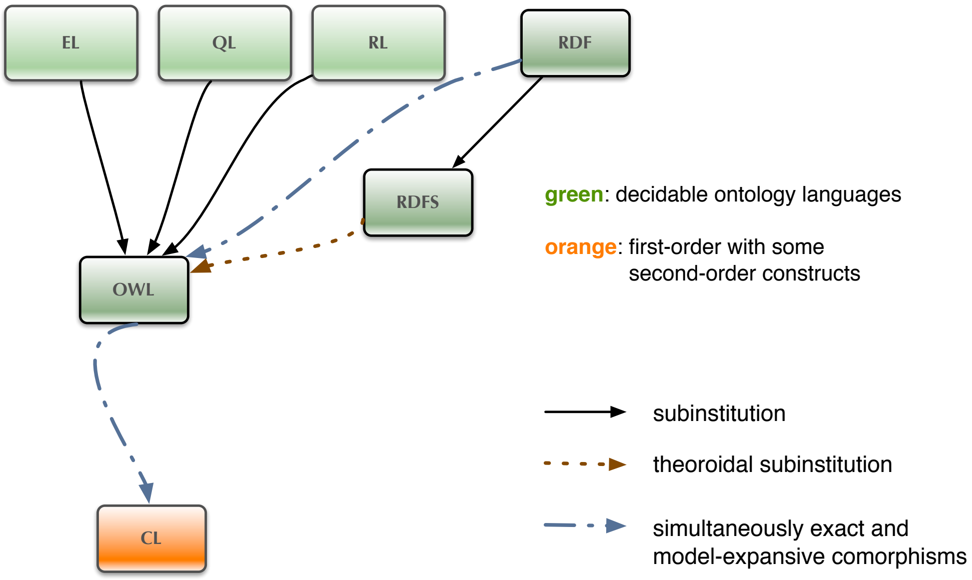

Figure G.1.: Translations between conforming OMS languages (normative)

G.4. SimpleRDF → RDF

SimpleRDF → RDF is an obvious inclusion, except that SimpleRDF resources need to be renamed if they happen to have a predefined meaning in RDF. The model translation needs to forget the fixed parts of RDF models, since this part can always reconstructed in a unique way, we get an isomorphic model translation.

G.5. RDF → RDFS

This is entierly analogous to SimpleRDF → RDF.

G.6. SimpleRDF → $\mathcal{SROIQ}(D)$

[117] Note(117)

A SimpleRDF signature is translated to $\mathcal{SROIQ}(D)$ by providing a class P and three roles sub, $pred$ and obj (these reify the extension relation), and one individual per SimpleRDF resource. A SimpleRDF triple (s, p, o) is translated to the $\mathcal{SROIQ}$ (D) sentence

$$\top \sqsubseteq \exists U.(\exists sub.\{s\} \sqcap \exists pred.\{p\} \sqcap \exists obj.\{o\}).$$

From an $\mathcal{SROIQ}$ (D) model $\mathcal{I}$, obtain a SimpleRDF model by inheriting the universe and the interpretation of individuals (then turned into resources). The interpretation $P^{\mathcal{I}}$ of P gives

---

[117] Note: This translation is not really useful. Consider the RDF-OWL-reduct construction instead.

P_m, and EXT_m is obtained by de-reifying, i.e.

$$EXT_m(x) := \{(y,z) | \exists u.(u,x) \in pred^{\mathcal{I}}, (u,y) \in sub^{\mathcal{I}}, (u,z,) \in obj^{\mathcal{I}}\}.$$

RDF → $\mathcal{SROIQ}(D)$ is defined similarly. The theory of RDF built-ins is (after translation to $\mathcal{SROIQ}$ (D)) added to any signature translation. This ensures that the model translation can add the built-ins.

G.7. OWL → FOL

G.7.1. Translation of Signatures

$\Phi((\mathbf{C}, \mathbf{R}, \mathbf{I})) = (F, P)$ with

- function symbols: $F = \{a^{(1)} | a \in \mathbf{I}\}$
- predicate symbols $P = \{A^{(1)} | A \in \mathbf{C}\} \cup \{R^{(2)} | R \in \mathbf{R}\}$

G.7.2. Translation of Sentences

Concepts are translated as follows:

- $\alpha_x(A) = A(x)$
- $\alpha_x(\neg C) = \neg \alpha_x(C)$
- $\alpha_x(C \sqcap D) = \alpha_x(C) \wedge \alpha_x(D)$
- $\alpha_x(C \sqcup D) = \alpha_x(C) \vee \alpha_x(D)$
- $\alpha_x(\exists R.C) = \exists y.(R(x,y) \wedge \alpha_y(C))$
- $\alpha_x(\exists U.C) = \exists y.\alpha_y(C)$
- $\alpha_x(\forall R.C) = \forall y.(R(x,y) \rightarrow \alpha_y(C))$
- $\alpha_x(\forall U.C) = \forall y.\alpha_y(C)$
- $\alpha_x(\exists R.\mathrm{Self}) = R(x,x)$
- $\alpha_x(\leq nR.C) = \forall y_1, \ldots, y_{n+1}. \bigwedge_{i=1,\ldots,n+1}(R(x,y_i) \wedge \alpha_{y_i}(C)) \rightarrow \bigvee_{1 \leq i < j \leq n+1} y_i = y_j$
- $\alpha_x(\geq nR.C) = \exists y_1, \ldots, y_n. \bigwedge_{i=1,\ldots,n}(R(x,y_i) \wedge \alpha_{y_i}(C)) \wedge \bigwedge_{1 \leq i < j \leq n} y_i \neq y_j$
- $\alpha_x(\{a_1, \ldots a_n\}) = (x = a_1 \vee \ldots \vee x = a_n)$

For inverse roles R^-, $R^-(x,y)$ has to be replaced by $R(y,x)$, e.g.

$$\alpha_x(\exists R^-.C) = \exists y.(R(y,x) \wedge \alpha_y(C))$$

This rule also applies below.

Sentences are translated as follows:

- $\alpha_\Sigma(C \sqsubseteq D) = \forall x. (\alpha_x(C) \rightarrow \alpha_x(D))$
- $\alpha_\Sigma(a : C) = \alpha_x(C)[a/x]$[1]
- $\alpha_\Sigma(R(a,b)) = R(a,b)$
- $\alpha_\Sigma(R \sqsubseteq S) = \forall x,y.R(x,y) \rightarrow S(x,y)$

---

[1] Replace x by a.

- $\alpha_\Sigma(R_1; \ldots ; R_n \sqsubseteq R) =$ $\forall x, y.(\exists z_1, \ldots z_{n-1}.R_1(x, z_1) \wedge R_2(z_1, z_2) \wedge \ldots \wedge R_n(z_{n-1}, y)) \rightarrow R(x, y)$
- $\alpha_\Sigma(\mathrm{Dis}(R_1, R_2)) = \neg\exists x, y.R_1(x, y) \wedge R_2(x, y)$
- $\alpha_\Sigma(\mathrm{Ref}(R)) = \forall x.R(x, x)$
- $\alpha_\Sigma(\mathrm{Irr}(R)) = \forall x.\neg R(x, x)$
- $\alpha_\Sigma(\mathrm{Asy}(R)) = \forall x, y.R(x, y) \rightarrow \neg R(y, x)$
- $\alpha_\Sigma(\mathrm{Tra}(R)) = \forall x, y, z.R(x, y) \wedge R(y, z) \rightarrow R(x, z)$

G.7.3. Translation of Models

- For $M' \in \mathrm{Mod}^{FOL}(\Phi\Sigma)$ define $\beta_\Sigma(M') := (\Delta, \cdot^I)$ with $\Delta = |M'|$ and $A^I = M'_A, a^I = M'_a, R^I = M'_R$.

Proposition 11 $C^\mathcal{I} = \{m \in M'_{Thing} | M' + \{x \mapsto m\} \models \alpha_x(C)\}$

Proof. By Induction over the structure of C.

- $A^\mathcal{I} = M'_A = \{m \in M'_{Thing} | M' + \{x \mapsto m\} \models A(x)\}$
- $(\neg C)^\mathcal{I} = \Delta \backslash C^\mathcal{I} =^{I.H.} \Delta \backslash \{m \in M'_{Thing} | M' + \{x \mapsto m\} \models \alpha_x(C)\} = \{m \in M'_{Thing} | M' + \{x \mapsto m\} \models \neg\alpha_x(C)\}$

□

The satisfaction condition holds as well.

G.8. OWL → CL

H. Annex (informative): Extended Logic Graph

This annex extends the graph of logics and translations given in annex G by a list of OMS language whose conformance with DOL will be established through the registry. The graph is shown in Figure H.1. Its nodes are included in the following list of OMS languages and profiles (in addition to those mentioned in annex G):

- PL (propositional logic)
- SimpleRDF (RDF triples without a reserved vocabulary)
- $\mathrm{OBO}^{\mathrm{OWL}}$ and OBO1.4
- RIF (Rule Interchange Format)
- EER (Enhanced Entity-Relationship Diagrams)
- Datalog
- ORM (object role modeling)
- the meta model of schema.org
- UML (Unified Modelling Language), with possibly different logics according to different UML semantics
- SKOS (Simple Knowledge Organization System)
- $\mathrm{FOL}^{=}$ (untyped first-order logic, as used for the TPTP format)
- F-logic
- CASL (Common Algebraic Specification Language)

The actual translations are specified in [35].

[118] Note(118)

---

[118] Note: TODO: Provide linear syntax here (as in the paper). TM: what do you mean by this?

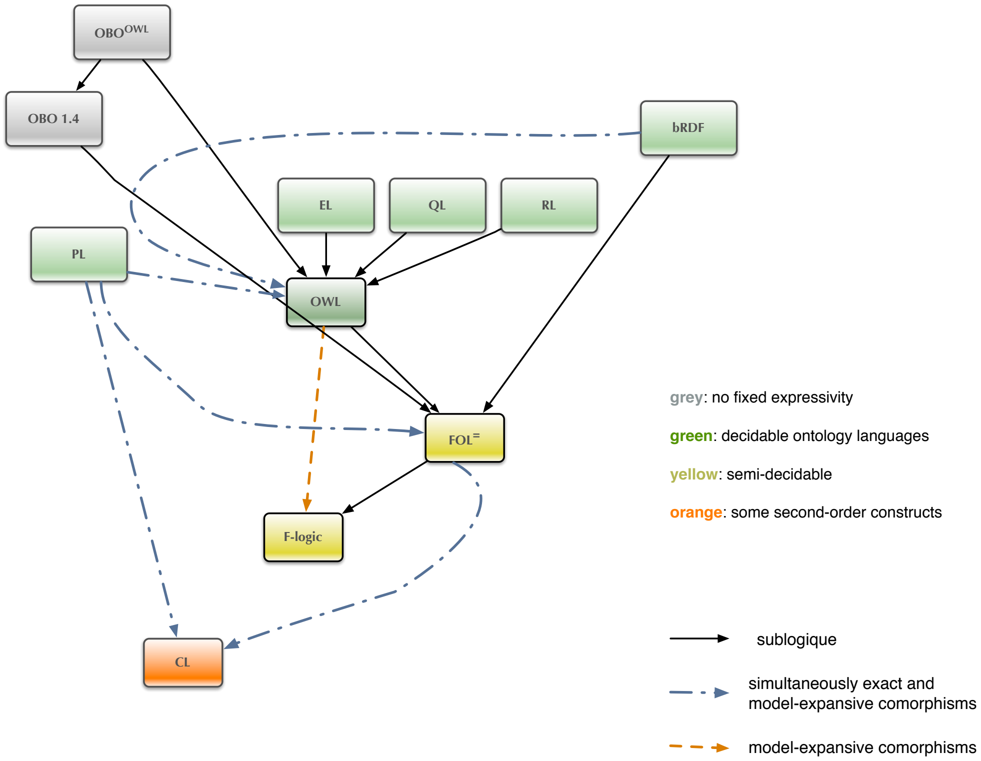

Figure H.1.: Translations between conforming OMS languages (extended)

I. Annex (informative): Institutional semantics

Note that the institute-based semantics for DOL does not cover `SYMBOL-MAPs`, `combinations` and the construct `monomorphic`. The institutional semantics will provide a mechanism for giving a semantics to the full distributed ontology, modeling and specification language (DOL).

Institutions generalise institute to arbitary signature mappings (called signature morphisms) between signatures.

Definition 12 *An* ***institution*** *[12] is a quadruple* $I = (\mathbb{S}ign, \mathbf{Sen}, \mathbf{Mod}, \models)$ *consisting of the following:*

- *a category* $\mathbb{S}ign$ *of* signatures *and* signature morphisms,
- *a functor* $\mathbf{Sen}\colon \mathbb{S}ign \longrightarrow \mathbb{S}et$[1] *giving, for each signature* Σ, *the set of* sentences $\mathbf{Sen}(\Sigma)$, *and for each signature morphism* $\sigma : \Sigma \to \Sigma'$, *the* sentence translation map $\mathbf{Sen}(\sigma) : \mathbf{Sen}(\Sigma) \to \mathbf{Sen}(\Sigma')$, *where often* $\mathbf{Sen}(\sigma)(\varphi)$ *is written as* $\sigma(\varphi)$,
- *a functor* $\mathbf{Mod} : \mathbb{S}ign^{op} \to \mathbb{C}at$[2] *giving, for each signature* Σ, *the category of* models $\mathbf{Mod}(\Sigma)$, *and for each signature morphism* $\sigma\colon \Sigma \longrightarrow \Sigma'$, *the* reduct functor $\mathbf{Mod}(\sigma) : \mathbf{Mod}(\Sigma') \to \mathbf{Mod}(\Sigma)$, *where often* $\mathbf{Mod}(\sigma)(M')$ *is written as* $M'\restriction_\sigma$, *and* $M'\restriction_\sigma$ *is called the* σ-reduct *of* M', *while* M' *is called a* σ-expansion *of* $M'\restriction_\sigma$,
- *a satisfaction relation* $\models_\Sigma \subseteq |\mathbf{Mod}(\Sigma)| \times \mathbf{Sen}(\Sigma)$ *for each* $\Sigma \in |\mathbb{S}ign|$,

such that for each $\sigma\colon \Sigma \longrightarrow \Sigma'$ *in* $\mathbb{S}ign$ *the following* ***satisfaction condition*** *holds:*

$$(\star) \qquad M' \models_{\Sigma'} \sigma(\varphi) \text{ iff } M'\restriction_\sigma \models_\Sigma \varphi$$

for each $M' \in |\mathbf{Mod}(\Sigma')|$ *and* $\varphi \in \mathbf{Sen}(\Sigma)$, *expressing that truth is invariant under change of notation and context.* □

Definition 13 (Propositional Logic) *The institution* **Prop** *is like the institute* **Prop**. *Signature morphisms are functions* $\sigma : \Sigma_1 \to \Sigma_2$. *The reduct of a* Σ_2*-model* M_2 *along* $\sigma : \Sigma_1 \to \Sigma_2$ *is the* Σ_1*-model given by the composition* $M_2 \circ \sigma$.

Definition 14 (Common Logic - CL) *The institution of Common Logic (CL) is like the institute. A CL signature morphism consists of two maps between the sets of names and of sequence markers, such that the property of being a discourse name is preserved and reflected.*[3] *Model reducts leave UR, UD, rel and fun untouched, while int and seq are composed with the appropriate signature morphism component.*

---

[1] $\mathbb{S}et$ is the category having all small sets as objects and functions as arrows.

[2] $\mathbb{C}at$ is the category of categories and functors. Strictly speaking, $\mathbb{C}at$ is not a category but only a so-called quasicategory, which is a category that lives in a higher set-theoretic universe.

[3] That is, a name is a discourse name if and only if its image under the signature morphism is.

Institute comorphisms can be generalised to institution comorphisms, see [14].

Definition 15 (Institution Comorphism) *Given two institutions I and J with $I = (\mathbb{S}ign^I, \mathsf{Mod}^I, \mathsf{Sen}^I, \models^I)$ and $J = (\mathbb{S}ign^J, \mathsf{Mod}^J, \mathsf{Sen}^J, \models^J)$, an* ***institution comorphism*** *from I to J consists of a functor $\Phi : \mathbb{S}ign^I \longrightarrow \mathbb{S}ign^J$, and natural transformations $\beta : \mathsf{Mod}^J \circ \Phi \Longrightarrow \mathsf{Mod}^I$ and $\alpha : \mathsf{Sen}^I \Longrightarrow \mathsf{Sen}^J \circ \Phi$, such that*

$$M' \models^J_{\Phi(\Sigma)} \alpha_\Sigma(\varphi) \Leftrightarrow \beta_\Sigma(M') \models^I_\Sigma \varphi.$$

holds, called the ***satisfaction condition****.*

Here, $\Phi(\Sigma)$ is the translation of signature Σ from institution I to institution J, $\alpha_\Sigma(\varphi)$ is the translation of the Σ-sentence φ to a $\Phi(\Sigma)$-sentence, and $\beta_\Sigma(M')$ is the translation (or perhaps better: reduction) of the $\Phi(\Sigma)$-model M' to a Σ-model.

Institute morphisms can be generalised to institution morphisms.

Definition 16 (Institution Morphism) *Given two institutions I and J with $I = (\mathbb{S}ign^I, \mathsf{Mod}^I, \mathsf{Sen}^I, \models^I)$ and $J = (\mathbb{S}ign^J, \mathsf{Mod}^J, \mathsf{Sen}^J, \models^J)$, an* ***institution morphism*** *from I to J consists of a functor $\Phi : \mathbb{S}ign^I \longrightarrow \mathbb{S}ign^J$, and natural transformations $\beta : \mathsf{Mod}^I \Longrightarrow \mathsf{Mod}^J \circ \Phi$ and $\alpha : \mathsf{Sen}^J \circ \Phi \Longrightarrow \mathsf{Sen}^I$, such that*

$$M \models^I_\Sigma \alpha_\Sigma(\varphi) \Leftrightarrow \beta_{\Phi(\Sigma)}(M) \models^J_{\Phi(\Sigma)} \varphi.$$

holds, called the ***satisfaction condition****.*

An **institution-based heterogeneous logical environment** is like an institute-based one, execpt that institutions (institution morphisms, institution comorphisms) are used in place of institutes (institute morphisms, institute comorphisms).

The full DOL language can be interpreted over an arbitrary institution-based heterogeneous logical environment. [Details to be given.]

We will give (as normative annexes) one such environment. These will define the “default translations” that we assume.

J. Annex (informative): Example Uses of all DOL Constructs

[119] Note(119)

Top-level declarations in distributed OMS	
Top-level declaration	**Examples**
language IRI	Alignments, Publications
logic IRI	Alignments, Mereology
serialization IRI	Alignments, Mereology
PrefixMap	Mereology
ontology IRI = OMS end	Alignments, Mereology
ontology IRI = %mcons OMS end	Mereology
interpretation IRI : OMS to OMS = Symbol |-> Symbol ...	Mereology
interpretation IRI : OMS to OMS = %cons Symbol |-> Symbol ...	
interpretation IRI : OMS to OMS = translation IRI	Mereology
equivalence IRI : OMS <-> OMS = OMS end	Algebra
module IRI : OMS of OMS for Symbols	
module IRI %ccons : OMS of OMS for Symbols	
alignment IRI : OMS to OMS end	
alignment IRI 1 : OMS to OMS end	
alignment IRI ? : OMS to OMS end	
alignment IRI + : OMS to OMS end	
alignment IRI * : OMS to OMS end	
alignment IRI : OMS to OMS = Correspondences	Alignments

[120] Note(120)

---

[119] Note: the uses cases in the RFP should be reused and worked into DOL examples

[120] Note: Q-AUT: Should we have another column here that refers to the *abstract* syntax?

OMS	
OMS notation	**Examples**
BasicOMS	Alignments, Mereology
IRI	Alignments, Mereology
IRI %(IRI)%	
minimize { OMS }	BlocksWithCircumscription
OMS minimize Symbols var Symbols	BlocksWithCircumscription
OMS with Symbol |-> Symbol ...	Alignments
OMS with translation IRI	Mereology
OMS with translation IRI : IRI → IRI	
OMS with translation IRI → IRI	
OMS with translation → IRI	
OMS hide SymbolItems	Algebra
OMS reveal Symbols	
OMS reveal Symbol |-> Symbol ...	
OMS hide along IRI	
OMS hide along IRI : IRI → IRI	
OMS hide along IRI → IRI	
OMS hide along → IRI	
OMS approximate with IRI	
OMS approximate in IRI with IRI	
OMS approximate in IRI	
OMS and OMS	
OMS then OMS	Mereology
OMS then %ccons OMS	
OMS then %ccons %(IRI)% OMS	
OMS then %mcons OMS	
OMS then %mono OMS	
OMS then %wdef OMS	
OMS then %def OMS	
OMS then %implied OMS	BlocksWithCircumscription
logic IRI : OMS	
language IRI : OMS	
serialization IRI : OMS	
OMS bridge Translation OMS	Publications
combine GraphElements	Alignments, Publications
combine GraphElements excluding IRIs	

J.1. Mereology: Distributed and Heterogeneous Ontologies

[121] Note(121)

---

[121] Note: Q-AUT: In the TKE paper we made the name of the propositional logic ontology syntax explicit. The propositional logic listing now leaves us with a problem: neither is propositional logic specified as DOL-conformant, nor is Hets' CASL-like syntax, nor is anything of this intended to ever be normative. TM: hence either leave it out, or make propositional logic normative. What about the examples in OWL+CL develop during the Ontology Summit Hackathon?

```
%prefix( :      <http://www.example.org/mereology#>
         owl:   <http://www.w3.org/2002/07/owl#>
         log:   <http://purl.net/dol/logic/>    %% descriptions of logics ...
         trans: <http://purl.net/dol/translations/> )%  %% ... and translations

distributed OMS Mereology

logic log:Propositional syntax ser:Prop/Hets       %% non-standard serialization built int
ontology Taxonomy = %mcons      %% basic taxonomic information about mereology reused from
  props PT, T, S, AR, PD
  . S ∨ T ∨ AR ∨ PD ⟶ PT
%% PT is the top concept
  . S ∧ T  ⟶ ⊥                                              %% PD, S, T, AR are pairwise di
  . T ∧ AR ⟶ ⊥
%% and so on
end

language lang:OWL2 logic log:SROIQ syntax ser:OWL2/Manchester
%% OWL Manchester syntax
ontology BasicParthood =                             %% Parthood in SROIQ, as far as easil
  Class: ParticularCategory SubClassOf: Particular
                                            %% omitted similar declarations of the other c
    DisjointUnionOf: SpaceRegion, TimeInterval, AbstractRegion, Perdurant
                                 %% pairwise disjointness more compact thanks to an OWL bu
  ObjectProperty: isPartOf        Characteristics: Transitive
  ObjectProperty: isProperPartOf  Characteristics: Asymmetric  SubPropertyOf: isPartOf
  Class: Atom EquivalentTo: inverse isProperPartOf only owl:Nothing
end                                          %% an atom has no proper parts

interpretation TaxonomyToParthood : Taxonomy to BasicParthood =
  translation trans:PropositionalToSROIQ,    %% translate the logic, then rename the entit
  PT ↦ Particular, S ↦ SpaceRegion, T ↦ TimeInterval, A ↦ AbstractRegion, %[ and so on

logic log:CommonLogic syntax ser:CommonLogic/CLIF
                      %% syntax: the Lisp-like CLIF dialect of Common Logic
ontology ClassicalExtensionalParthood =
  BasicParthood with translation trans:SROIQtoCL
      %% import the OWL ontology from above, translate it to Common Logic, then extend it
then
  . (forall (X) (if (or (= X S) (= X T) (= X AR) (= X PD))
                    (forall (x y z) (if (and (X x) (X y) (X z))
                                        (and
%% now list all the axioms
      (if (and (isPartOf x y) (isPartOf y x)) (= x y))
%% antisymmetry
      (if (and (isProperPartOf x y) (isProperPartOf y z)) (isProperPartOf x z))
                              %% transitivity; can't be expressed in OWL together with asy
      (iff (overlaps x y) (exists (pt) (and (isPartOf pt x) (isPartOf pt y))))
      (iff (isAtomicPartOf x y) (and (isPartOf x y) (Atom x)))
```

```
      (iff (sum z x y)
           (forall (w) (iff (overlaps w z) (and (overlaps w x) (overlaps w y)))))
      (exists (s) (sum s x y))
%% existence of the sum
      )))))
  . (forall (Set a) (iff (fusion Set a)
%% definition of fusion
            (forall (b) (iff (overlaps b a)
                             (exists (c) (and (Set c) (overlaps c a)))))))
  }
```

J.2. Blocks World: Minimization

[122] Note(122)

```
distributed OMS BlocksWithCircumscription
logic log:OWL

ontology Blocks =
  %% FIXED PART
  Class: Block
  Individual: B1 Types: Block
  Individual: B2 Types: Block DifferentFrom: B1
              %% B1 and B2 are different blocks
then
  %% CIRCUMSCRIBED PART
  minimize {
    Class: Abnormal
    Individual: B1 Types: Abnormal
       %% B1 is abnormal
  }
then
  %% VARYING PART
  Class: Ontable
  Class: BlockNotAbnormal EquivalentTo: Block and not Abnormal SubClassOf: Ontable
        %% Normally, a block is on the table
then %implied
  Individual: B2 Types: Ontable
     %% B2 is on the table
end
```

ToDo

To Do: Instead of Blocks World, perhaps we could specify an ontology that uses inheritance networks with exceptions, and then use circumscription to axiomatize that ontology.

```
ontology Blocks_Alternative =
```

---

[122] NOTE: Q-AUT: Here we need the prefixes for registry entries (e.g. logics) once more; they should be reused across examples. Or we need to specify a mechanism that gets rid of *these* prefixes altogether. @TM, could you please comment on my specification enhancement request `http://trac.informatik.uni-bremen.de:8080/hets/ticket/1020#comment:33`?

```
  Class: Block
  Class: Abnormal
  Individual: B1 Types: Block, Abnormal
  Individual: B2 Types: Block DifferentFrom: B1
              %% B1 and B2 are different blocks
              %% B1 is abnormal
  Class: Ontable
  Class: BlockNotAbnormal EquivalentTo: Block and not Abnormal SubClassOf: Ontable
        %% Normally, a block is on the table
  minimize Abnormal var Ontable, BlockNotAbnormal
then %implied
  Individual: B2 Types: Ontable
     %% B2 is on the table
end
```

J.2.1. Alignments

```
%prefix( :     <http://www.example.org/alignment#>
         owl   <http://www.w3.org/2002/07/owl#>
         log   <http://purl.net/dol/logic/> %% descriptions of logics ...
         trans <http://purl.net/dol/translations/> )% %% ... and translations

distributed OMS Alignments

language lang:OWL2 logic log:SROIQ syntax ser:OWL2/Manchester

alignment Alignment1 : { Class: Woman } to { Class: Person } =
  Woman < Person
end

ontology AlignedOntology1 =
  combine Alignment1
end


ontology Onto1 =
  Class: Person
  Class: Woman SubClassOf: Person
  Class: Bank
end

ontology Onto2 =
  Class: HumanBeing
  Class: Woman SubClassOf: HumanBeing
  Class: Bank
end
```

```
alignment VAlignment : Onto1 to Onto2 =
  Person = HumanBeing,
  Woman = Woman
end

ontology VAlignedOntology =
  combine 1 : Onto1, 2 : Onto2, VAlignment
  %% 1:Person is identified with 2:HumanBeing
  %% 1:Woman is identified with 2:Woman
  %% 1:Bank and 2:Bank are kept distinct
end

ontology VAlignedOntologyRenamed =
  VAlignedOntology with 1:Bank |-> RiverBank, 2:Bank |-> FinancialBank
end
```

J.3. Distributed Description Logics

```
%prefix( :     <http://www.example.org/mereology#>
         owl   <http://www.w3.org/2002/07/owl#>
         log   <http://purl.net/dol/logic/> %% descriptions of logics ...
         trans <http://purl.net/dol/translations/> )% %% ... and translations

distributed OMS Publications

language lang:OWL2 logic log:SROIQ syntax ser:OWL2/Manchester

ontology Publications1 =
  Class: Publication
  Class: Article SubClassOf: Publication
  Class: InBook SubClassOf: Publication
  Class: Thesis  SubClassOf: Publication
  Class: MasterThesis  SubClassOf: Thesis
  Class: PhDThesis SubClassOf: Thesis
end

ontology Publications2 =
  Class: Thing
  Class: Article SubClassOf: Thing
  Class: BookArticle SubClassOf: Thing
  Class: Publication SubClassOf: Thing
  Class: Thesis  SubClassOf: Thing
end

ontology Publications_Combined =
combine
  1 : Publications1 with translation OWL2MS-OWL,
  2 : Publications2 with translation OWL2MS-OWL
```

```
  %% implicitly: Article ↦ 1:Article ...
  %%             Article ↦ 2:Article ...
bridge with translation MS-OWL2DDL
  %% implicitly added my translation MS-OWL2DDL: binary relation providing the bridge
  1:Publication $\xrightarrow{\sqsubseteq}$ 2:Publication
  1:PhdThesis $\xrightarrow{\sqsubseteq}$ 2:Thesis
  1:InBook $\xrightarrow{\sqsubseteq}$ 2:BookArticle
  1:Article $\xrightarrow{\sqsubseteq}$ 2:Article
  1:Article $\xrightarrow{\sqsupseteq}$ 2:Article
end


ontology Publications_Extended =
Publications
then
bridge with translation DDL2-ECO
  %% turns implicit domain-relation into default relation 'D'
  %% add E-connection style bridge rules on top
end

%% Note: unfinished...
%% add second spec following example from AI journal paper on E-connections,
%% page 22: three different bridge relations between two ontologies; first DDL
%% modelling, translation to ECO with default relation, renaming and extension
%% in ECO style.

distributed OMS Market


language lang:OWL2 logic log:SROIQ syntax ser:OWL2/Manchester
ontology Purchases =
combine
  1 : { Class: PurchaseOrder },
  2 : { ObjectProperty: Buyer
       ObjectProperty: Good
       ObjectProperty: BoughtBy }
bridge with translation OWL2DDLwithRoles
  1:PurchaseOrder -into-> 2:BoughtBy
%% means in FOL: forall x 1PurchaseOrder(x) -> forall yz CR12(x,y,z) -> 2BoughtBy(y,z)
end
```

J.3.1. Algebra

[123]

Note(123)

```
%prefix( :     <http://www.example.org/alignment#>
```

---

[123] Note: use “spec” here???

```
         owl   <http://www.w3.org/2002/07/owl#>
         log   <http://purl.net/dol/logic/> %% descriptions of logics ...
         trans <http://purl.net/dol/translations/> )% %% ... and translations

distributed OMS Algebra

logic log:CommonLogic syntax ser:CommonLogic/CLIF

ontology implicit_group =
(forall (x y z)
        (= (op x (op y z)) (op (op x y) z)))
(exists (e)
        (forall (x)
                (and    (= x (op e x))
                        (= x (op x e)))))
(forall (x)
        (exists (y)
                (and    (= x (op x (op x y)))
                        (= x (op x (op y x))))))
end

ontology explicit_group =
(forall (x y z)
        (= (op x (op y z)) (op (op x y) z)))
(forall (x)     (and    (= x (op e x))
                        (= x (op x e)))))
(forall (x)
                (and    (= x (op x (op x (inv x))))
                        (= x (op x (op (inv x) x))))))
end

equivalence groups_equiv : implicit_group <-> { explicit_group hide e, inv }
end
```

K. Annex (informative): Use cases

This annex sketches scenarios that outline how DOL is intended to be applied. For each scenario, we list its status of implementation, the DOL features it makes use of, and provide a brief description.

K.1. Generating multilingual labels for menus in a user interface

Status exists (but not yet DOL-based)

Features Aligning (multiple OWL ontologies), Annotation

DO-ROAM (**D**ata and **O**ntology driven **R**oute-finding **O**f **A**ctivity-oriented **M**obility[1]) is a web service with an interactive frontend that extends OpenStreetMap by an ontology-based search for located activities and opening hours [8]. The service is driven by a set of different OWL ontologies that have been aligned to each other using the Falcon matching tool [24]. The user interface of the DO-ROAM web frontend offers multilingual labels, which are maintained in close connection to the underlying ontologies.

Porting DO-ROAM to DOL would enable the coherent representation of the aligned ontologies as one distributed OMS, and it would enable the maintenance of the user interface labels as annotations inside the ontology.

K.2. Connecting devices of differing complexity in an Ambient Assisted Living setting

Status core ontology (not DOL-based) and service environment exists – the DOL-based extensions not yet

Features Logical OMS mappings across different logics, connection to linked open datasets

Consider the following ambient assisted living (AAL) scenario:

> Clara instructs her **wheelchair** to get her to the **kitchen** (**next door** to the **living room**. For **dinner**, she would like to take a *pizza* from the **freezer** and bake it in the **oven**. (Her diet is *vegetarian*.) **Afterwards** she needs to rest in **bed**.

Existing ontologies for ambient assisted living (e.g. the OpenAAL[2] OWL ontology) cover the *core* of these concepts; they provide at least classes (or generic superclasses) corresponding to the concepts highlighted in **bold**. However, that does not cover the scenario completely:

---

[1] `http://www.do-roam.org`

[2] `http://openaal.org`

- Some concepts (here: food and its properties, *italicized*) are not covered. There are separate ontologies for that (such as the Pizza ontology[3]), whereas information about concrete products (here: information about the concrete pizza in Clara's oven) would rather come from Linked Open Datasets than from formal ontologies.
- Not all concepts (here: space and time, underlined) are covered at the required level of complexity. OpenAAL says that appointments have a date and that rooms can be connected to each other, but not what exactly that means. Foundational ontologies and spatial calculi, often formalized in first-order logic, cover space and time at the level of complexity required by a central controller of an apartment and by an autonomously navigating wheelchair.
- Thirdly, even description logic might be too complex for very simple devices involved into the scenario, such as the kitchen light switch, for which propositional logic may be sufficient.

Thus, an adequate formalization of this scenario has to be heterogeneous. For example, one could imagine the following axioms:

light switch "light is switched on if and only if someone is in the room and it is dark outside" – this could be formalized in propositional logic as $\mathrm{light\_on} \equiv \mathrm{person\_in\_room} \wedge \mathrm{dark\_outside}$.

freezer "a vegetarian pizza is a pizza whose toppings are all vegetarian" – this could be formalized in description logic as $\mathrm{VegetarianPizza} \equiv \mathrm{Pizza} \sqcap \forall \mathrm{hasTopping}.\mathrm{Vegetarian}$

wheelchair "two areas in a house (e.g. a working area in a room) are either the same, or intersecting, or bordering, or separated, or one is part of the other" – this could be formalized as an RCC-style spatial calculus in first-order logic as

$$\begin{aligned}\forall a_1, a_2.\quad & \mathrm{equal}(a_1, a_2) \veebar \mathrm{overlapping}(a_1, a_2) \veebar \mathrm{bordering}(a_1, a_2) \veebar \mathrm{disconnected}(a_1, a_2) \\ & \veebar \mathrm{part\_of}(a_1, a_2) \veebar \mathrm{part\_of}(a_2, a_1).\end{aligned}$$

DOL would be capable of expressing all that within one distributed OMS of heterogeneous ontologies arranged around an OWL core (here: the OpenAAL ontology), including logical OMS mappings from OpenAAL to the other ontologies, as well as a re-declaration of a concrete pizza product from a product dataset as an instance of the Pizza OWL class.

K.3. Interpreting the OWL formalization of the DOLCE foundational ontology in First-order logic

Status potential use case

Features Logical OMS mappings

DOLCE is a foundational ontology that has primarily been formalized in the first-order logic ontology language KIF (a predecessor of Common Logic), but also in OWL ("DOLCE Lite") [32]. This 'OWLized' version was targeting use in semantic web services and domain ontology interoperability, and to provide the generic categories and relationships to aid domain ontology development. DOLCE has been used also for semantic middleware, and in OWL-formalised

---

[3] This is not a fully comprehensive food ontology, but rather a well-known sample OWL ontology; cf. `http://owl.cs.manchester.ac.uk/tutorials/protegeowltutorial/`

ontologies of neuroimaging, computing, ecology, and data mining and optimization. Given the differences in expressivity, DOLCE Lite had to simplify certain notions. For example, the DOLCE Lite formalization of "temporary parthood" (something is part of something else at a certain point or interval in time) omits any information about the time, as OWL only supports binary predicates (a.k.a. "properties"). That leaves ambiguities for modeling a view from DOLCE Lite to the first-order DOLCE, as such a view would have to reintroduce the third (temporal) component of such predicates:

- Should a relation asserted in terms of DOLCE Lite be assumed to hold for *all* possible points/intervals in time, i.e. should it be universally quantified?
- Or should such a relation be assumed to hold for *some* points/intervals in time, i.e. should it be existentially quantified?
- Or should a concrete value for the temporal component be assumed, e.g. "0" or "now"?

DOL would support the formalization of all of these views and, given suitable consistency checking tools, the analyzation of whether any such view would satisfy all further axioms that the first-order DOLCE states about temporal parthood.

K.4. Extending the OWL Time ontology to a more comprehensive coverage of time

Status potential use case

Features Logical OMS mappings

The OWL Time ontology[4] covers temporal concepts such as instants and intervals and has been designed for describing the temporal content of Web pages and the temporal properties of Web services. While OWL is suitable for these intended applications, only a first-order axiomatization is capable of faithfully capturing all relevant notions, such as the trichotomy of the "before" relation: One instant is either before another one, or at the same time, or after. Moreover, a relationship between facts expressed in terms of instants and facts expressed in terms of intervals (both of which is, independently, possible in OWL), can only be established via first-order logic, e.g. by declaring an interval of length zero equivalent to an instant.

A separate first-order axiomatization of OWL Time exists [[22],[37]]. DOL would instead provide the mechanism of modeling OWL Time as one coherent heterogeneous ontology, using OWL and, e.g., Common Logic.[124] For the temporal description logic $\mathcal{DLR}_{\mathcal{US}}$ for knowledge bases and logic-based temporal conceptual data modelling [[1],[2]]; $\mathcal{DLR}_{\mathcal{US}}$ combines the propositional temporal logic with the *Since* and *Until* operators and the (non-temporal) description logic $\mathcal{DLR}$ and can be regarded as an expressive fragment of the first-order temporal logic $L^{since,until}$. Within DOL, this would enable one to have 'lightweight' time aspects with OWL Time, which are then properly formalised with $\mathcal{DLR}_{\mathcal{US}}$ or a leaner variant TDL-Lite [[4]], where notions such as (some time) "before" are given a formal semantics of the intended meaning that the plain OWL Times human-readable object property does not have. The latter, then, would enable the modeller to represent the meaning—hence, restrict the possible models—and check the consistency of the temporal constraints and so-called 'evolution constraints' in the ontology (evolution constraints constrain membership of an object or an individual relation to a concept Note(124)

---

[4] `http://www.w3.org/TR/2006/WD-owl-time-20060927/`

[124] NOTE: This is also a use case for multiple namespaces: OWL supports namespaces, CL doesn't.

or relationship over time). For instance, that each divorcee must have been a participant in a marriage before, that boarding only may occur after checking in, and that any employee must obtain a salary increase after two years of employment. It also can be used to differentiate between essential and immutable parthood, therewith being precise in the ontology about, e.g., the distinction how a human brain is part of a human (humans cannot live without it), versus how a hand is part of a human (humans can live without it), versus how the hand is part of, say, a boxer, which is essential to the boxer but only for has long as he is a boxer [[3]].

K.5. Metadata in COLORE (Common Logic Repository)

Status exists (but not yet DOL-based)

Features Annotation, Metadata vocabularies

COLORE, the Common Logic Repository[5] is an open repository of more than 150 ontologies as of December 2011, all formalized in Common Logic. COLORE stores metadata about its ontologies, which are represented using a custom XML schema that covers the following aspects[6], without specifying a formal semantics for them:

module provenance author, date, version, description, keyword, parent ontology[7]

axiom source provenance name, author, year[8]

direct relations maps (signature morphisms), definitional extension, conservative extension, inconsistency between ontologies, imports, relative interpretation, faithful interpretation, definable equivalence

DOL provides built-in support for a subset of the "direct relations" and specifies a formal semantics for them. In addition, it supports the implementation of the remainder of the COLORE metadata vocabulary as an ontology, reusing suitable existing metadata vocabularies such as OMV (cf. annex L), and it supports the implementation of one or multiple Common Logic ontologies plus their annotations as one coherent distributed OMS.

K.6. Extending OWL with datatypes defined in CASL

Status potential use case

Features ...

- OWL datatypes are in practice restricted to the XML Schema datatypes
- XML Schema can only specify the *syntax* of datatypes
- CASL can specify syntax (but not quite in the same way as XML Schema) *and* semantics of datatypes

125 126

Note(125)

---

[5] `http://stl.mie.utoronto.ca/colore/`

[6] `http://stl.mie.utoronto.ca/colore/metadata.html`

[7] Note that this use of the term "module" in COLORE corresponds to the term *structured OMS* in this OMG Specification

[8] Note that this may cover any *sentencs* in the sense of this OMG Specification

[125] NOTE: TODO: ModuleRelDefn combined with approximation and RDF-based querying of annotation/metadata dimensions

Note(126)

---

[126] NOTE: TODO: Maybe have an(other?) appendix that refers to the usage of DOL within ontology engineering methodologies, or at least to some good practices of using DOL

L. Annex (informative): Annotation Vocabularies

[127] Note(127)

Table L.1.: Vocabularies recommended for annotating DOL OMS

Vocabulary name	Purpose	ref.
DCMI Metadata Terms	general-purpose and biographical metadata	[6]
Ontology Metadata Vocabulary (OMV)	ontology engineering metadata	[21]

[128] Note(128)

---

[127] Note: Q-ALL: Or should this rather be normative?

[128] Note: TODO: maybe mention: How do we use the ISO 12620 DCR for our extension of the OMV?

M. Annex (informative): Bibliography

Bibliography

[1] A. Artale, E. Franconi, F. Wolter, and M. Zakharyaschev. A temporal description logic for reasoning about conceptual schemas and queries. *In: S. Flesca and S. Greco and N. Leone and G. Ianni, Proceedings of the 8th Joint European Conference on Logics in Artificial Intelligence (JELIA-02), Springer LNAI 2424, 98-110, 2002*, 2002.

[2] A. Artale, C. Parent, and S. Spaccapietra. Evolving objects in temporal information systems. *Annals of Mathematics and Artificial Intelligence 50:5-38, 2007*, 2002.

[3] Alessandro Artale, Nicola Guarino, and C. Maria Keet. Formalising temporal constraints on part-whole relations. *In: Gerhard Brewka and Jerome Lang, 11th International Conference on Principles of Knowledge Representation and Reasoning (KR'08), 673-683, AAAI Press 2008*, 2008.

[4] Alessandro Artale, Roman Kontchakov, Carsten Lutz, Frank Wolter, and Michael Zakharyaschev. Temporalising tractable description logic. *Proc. of the 14th International Symposium on Temporal Representation and Reasoning (TIME-07), 2007*, 2007.

[5] C. Bizer and T. Heath. Linked data—the story so far. *International journal on semantic web and information systems*, 5(3):1–22, 2009.

[6] DCMI USAGE BOARD. Dcmi metadata terms. *DCMI recommendation, Dublin Core Metadata Initiative, 2003.*, 2003.

[7] W.A. CARNIELLI, M. CONIGLIO, D.M. GABBAY, P. GOUVEIA, and C. SERNADAS. Analysis and synthesis of logics: how to cut and paste reasoning systems. *Applied logic series, Springer, 2008.*, 2008.

[8] M. CODESCU, G. HORSINKA, O. KUTZ, T. MOSSAKOWSKI, and R. RAU. Do-roam: Activity-oriented search and navigation with openstreetmap. *In: Fourth International Conference on GeoSpatial Semantics, C. Claramunt, S. Levashkin, M. Bertolotto (eds.), pp. 88–107, no. 6631 in LNCS, Springer, 2011.*, 2011.

[9] Jérôme Euzenat and Pavel Shvaiko. *Ontology Matching, Second Edition*. Springer, 2013.

[10] J. A. Goguen and R. M. Burstall. Institutions: Abstract model theory for specification and programming. *Journal of the Association for Computing Machinery*, 39:95–146, 1992.

[11] J. A. GOGUEN and R.M. BURSTALL. A study in the foundations of programming methodology: Specifications, institutions, charters and parchments. *In: Category Theory and Computer Programming, D. Pitt et al. (eds.), pp. 313–333, no. 240 in LNCS, Springer, 1985.*, 1985.

[12] J. A. GOGUEN and R.M. BURSTALL. Institutions: Abstract model theory for specification and programming. *Journal of the Association for Computing Machinery 39, pp. 95–146, , 1992*, 1992.

[13] J. A. GOGUEN and G. ROŞU. Institution morphisms. *Formal aspects of computing 13, pp. 274-307, 2002*, 2002.

[14] J. A. GOGUEN and G. ROSU. Institution morphisms. *In: Formal aspects of computing 13, pp. 274–307, 2002*, 2002.

[15] B. CUENCA GRAU, I. HORROCKS, Y. KAZAKOV, , and U. SATTLER. Just the right amount: Extracting modules from ontologies. *In: Proc. of the 16th International World Wide Web Conference (WWW), pp. 717–726, 8–12 May 2007, Banff, Alberta, Canada.* `https://www.cs.ox.ac.uk/isg/people/yevgeny.kazakov/publications/confs/CueHorKazSat07Modularity.pdf`, 2007.

[16] IETF NETWORK WORKING GROUP. Hypertext transfer protocol – HTTP/1.1. *RFC 2616, 1999*, 1999.

[17] W3C OWL WORKING GROUP. Owl 2 web ontology language manchester syntax. *W3C Working Group Note, 27 October 2009.* `http://www.w3.org/TR/2009/NOTE-owl2-manchester-syntax-20091027/`, 2009.

[18] W3C OWL WORKING GROUP. Owl 2 web ontology language: Rdf-based semantics. *W3C Recommendation, 27 October 2009.* `http://www.w3.org/TR/2009/REC-owl2-rdf-based-semantics-20091027/`, 2009.

[19] W3C RDF CORE WORKING GROUP. RDF/XML syntax specification (revised). *W3C Recommendation, 10 February 2004.* `http://www.w3.org/TR/2004/REC-rdf-syntax-grammar-20040210/`, 2004.

[20] W3C SEMANTIC WEB DEPLOYMENT WORKING GROUP. Best practice recipes for publishing rdf vocabularies. *W3C Working Group Note, 28 August 2008.* `http://www.w3.org/TR/2008/NOTE-swbp-vocab-pub-20080828/`, 2008.

[21] J. HARTMANN, R. PALMA, Y. SURE, M. C. SUÁREZ-FIGUEROA, P. HAASE, A. GÓMEZ-PÉREZ, and R. STUDER. Ontology metadata vocabulary and applications. *In: On the Move to Meaningful Internet Systems 2005: OTM Workshops, R. Meersman, Z. Tari, P. Herrero, et al. (eds.), pp. 906–915, no. 3762 in LNCS, Springer, Oct. 2005.*, 2005.

[22] J.R. HOBBS and F. PAN. An ontology of time for the semantic web. *ACM Transactions on Asian Language Processing (TALIP): Special issue on Temporal Information Processing, Vol. 3, No. 1, March 2004, pp. 66–85.*, 2004.

[23] I. HORROCKS, O. KUTZ, and U. SATTLER. The even more irresistible sroiq. *In Knowledge representation 2006, p. 57–67*, 2006.

[24] W. HU and Y. QU. Falcon-ao: A practical ontology matching system. *In: Proceedings of WWW-07, pp. 237–239, 2008.*, 2008.

[25] Z. Khan and C.M. Keet. Addressing issues in foundational ontology mediation. In J. Filipe and J. Dietz, editors, *5th International Conference on Knowledge Engineering and Ontology Development (KEOD'13), Vilamoura, Portugal, 19-22 September*, pages 5–16. SCITEPRESS, 2013.

[26] M. KIFER, G. LAUSEN, and J. WU. Logical foundations of object-oriented and frame-based languages,. *Journal of the ACM*, 42:741–843, 1995.

[27] O. KUTZ, I. NORMANN, T. MOSSAKOWSKI, , and D. WALTHER. Chinese whispers and connected alignments. *In: Proc. of the 5th International Workshop on Ontology Matching, OM-2010, co-located with the 9th International Semantic Web Conference (ISWC), 7 November 2010, Shanghai, China.* `http://disi.unitn.it/~p2p/OM-2010/om2010_Tpaper3.pdf`, 2010.

[28] BERNERS-T. LEE. Design issues: Linked data. *27 July 2006.* `http://www.w3.org/DesignIssues/LinkedData.html`, 2006.

[29] V. LIFSCHITZ. Circumscription. *In Handbook of Logic in Artificial Intelligence and Logic Programming, Volume 3, pages 297–352. Oxford University Press 1994.*, 1994.

[30] D. LUCANU, Y.-F. LI, and J. SONG DONG. Semantic web languages — towards an institutional perspective. *In: Essays Dedicated to Joseph A. Goguen, Lecture Notes in Computer Science 4060 Springer 2006, p. 99–123*, 2006.

[31] Carsten Lutz and Frank Wolter. Foundations for uniform interpolation and forgetting in expressive description logics. In Toby Walsh, editor, *IJCAI*, pages 989–995. IJCAI/AAAI, 2011.

[32] C. MASOLO, S. BORGO, A. GANGEMI, N. GUARINO, and A. OLTRAMARI. Ontology library. *WonderWeb Deliverable, Report No. 18 of the Laboratory for Applied Ontology – ISTC-CNR, Dec. 2003.* `http://www.loa-cnr.it/Papers/D18.pdf`, 2003.

[33] J. McCARTHY. Circumscription — a form of non-monotonic reasoning. *Artificial Intelligence, 13:27–39, 1980*, 1980.

[34] J. MESEGUER. General logics. *Logic Colloquium 87, pp. 275–329, 1989*, 1989.

[35] T. MOSSAKOWSKI and O. KUTZ. The onto-logical translation graph. *In: 5th International Workshop on Modular Ontologies (WoMO) at the 23rd European Summer School in Logic, Language and Information (ESSLLI) (Ljubljana, Slovenia, Aug. 8–12, 2011), O. Kutz, T. Schneider (eds.), IOS Press 2011.*, 2011.

[36] T. MOSSAKOWSKI, C. LANGE, and O. KUTZ. Three semantics for the core of the distributed ontology language. *In Michael Gruninger (Ed.), FOIS 2012: 7th International Conference on Formal Ontology in Information Systems*, 2012.

[37] F. PAN and J.R. HOBBS. Time in owl-s. *In Proceedings of the AAAI Spring Symposium on Semantic Web Services, Stanford University, CA, pp. 29–36.*

[38] Heiner Stuckenschmidt, Christine Parent, and Stefano Spaccapietra, editors. *Modular Ontologies: Concepts, Theories and Techniques for Knowledge Modularization*, volume 5445 of *Lecture Notes in Computer Science*. Springer, 2009.

[39] MC. SUÁREZ-FIGUEROA and A. GÓMEZ-PÉREZ. First attempt towards a standard glossary of ontology engineering terminology. *In: The 8th International Conference on Terminology and Knowledge Engineering, 18–21 August 2008, Copenhagen, Denmark.*, 2008.

[40] ed. WIKIMEDIA FOUNDATION. Linked data. *from Wikipedia, the free encyclopedia, 9 November 2011.* `http://en.wikipedia.org/w/index.php?title=Linked_Data&oldid=459835053`, 2011.

[41] A. ZIMMERMANN, M. KRÖTZSCH, J. EUZENAT, and P. HITZLER. Formalizing ontology alignment and its operations with category theory. *In: B. Bennett and C. Fellbaum: Proceedings of the Fourth International Conference on Formal Ontology in Information Systems (FOIS 2006), pp. 277–288,Frontiers in Artificial Intelligence and Applications 150, IOS Press 2006*, 2006.