Date: Tuesday 25th February, 2014

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

Version 1.0

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

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

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: Terms that appear in italics are defined in the glossary. Italic text also represents 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

[1] This International Standard specifies the Distributed Ontology, Modeling and Specification Language (DOL) designed to achieve integration and interoperability of ontologies, specifications and models (OSMs for short). DOL is a language for distributed knowledge representation across multiple OSMs, particularly OSMs that have been formalized in different OSM languages[2]. Note(1) Note(2)

The following features are essential to the design of this International Standard [3] : Note(3)

- DOL is a declarative language with a formal semantics for modular OSMs that consist of structured OSMs that are possibly heterogeneous, i.e. are written within the same or in different OSM languages, and made available at different Web locations.
- DOL provides a superset of the modularization and annotation facilities of a number of commonly used OSM languages, including OWL [W3C/TR REC-owl2-syntax:2009] and Common Logic [ISO/IEC 24707:2007][1].
- DOL is an open, extensible standard that is not restricted to a fixed set of supported OSM language but specifies criteria for any existing or future OSM language to conform with DOL.
- Existing OSMs 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.

The following are within the scope of this International Standard:

1. heterogeneous OSMs that combine parts written in different languages
2. links between distributed and heterogeneous (possibly structured) OSMs
3. annotation and documentation of OSMs, links between OSMs, symbols, and sentences
4. translations between different OSM languages
5. recommendations of vocabularies for annotating and documenting OSMs

[1] NOTE: compare the scope in 6.2 of the RFP
[2] NOTE: maybe: "logical languages" instead of "OSM languages"?
[3] NOTE: [date=D:201110060000+02'00']Proposal for restructuring this (due to Michael Grüninger):

1. heterogeneity
2. modularity
   a) writing structured OSMs
      i. reuse existing OSMs
      ii. write structured OSMs
   b) writing OSMs that have multiple parts
3. mappings (or should we say "links")
4. annotation

[1] See clause 7.1.1 for details.

6. a syntax for embedding the constructs mentioned under (1)–(3) as annotations into existing OSMs
7. a syntax for expressing (1)–(4) as standoff markup that points into existing OSMs
8. a formal semantics of (1)–(4)
9. criteria for existing or future OSM languages to conform with DOL

The following are outside the scope of this International Standard:

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

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

[4] NOTE: TODO: We somehow need to rephrase this (here and elsewhere); it reads quite ugly. Maybe just say “non-logical symbols”? Or, if absolutely necessary, reorder the words into “non-logical ontology symbols”.

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 International Standard describes the conformance with DOL of a number of OSM languages, namely OWL 2, Common Logic, RDF and RDFS, as well as translations among these, in its normative annexes.

It is expected that DOL will be used for more languages than this normative set of DOL-conformant[5] languages. There will be a **registry for DOL-conformant languages and translations** hosted at `http://ontohub.org`. This will ensure that this International Standard remains interoperable with past, present and future OSM languages, even if they do not appear in this standard or do not even have been standardized (yet). 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. Note(5)

There will be Maintenance Authority (MA)[6] 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. Note(6)

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

An OSM language is conformant with DOL if either

- it satisfies the following conditions:
  1. its abstract syntax is given by an EBNF grammar,
  2. at least one concrete syntax is given by a serialisation (see below),
  3. its logical language aspect (for expressing basic OSMs) is conformant, and in particular has a semantics (see below),
  4. its structuring language aspect (for expressing structured OSMs and relations between those) is conformant (see below), and
  5. its annotation language aspect (for expressing comments and annotations) is conformant (see below).
- there exists a translation of it into a conformant language.

For the second condition, cf. the translation of OBO1.4 to OWL, giving a formal semantics to OBO1.4). [7] Note(7)

The *logical language aspect* of an OSM 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 institute. It may additionally be presented as an institution,

[5] NOTE: FYI: coherently rephrased from "compliant" to "conformant"
[6] NOTE: or, depending on advisability, a Registration Authority
[7] NOTE: turn this into a note

leading to the possibility of interpreting additional DOL language constructs. [1] Note that one OSM 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 OSM 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 OSM language is conformant with DOL if its constructs have no impact on the semantics. The annotation language aspect **shall** may be non-empty; it **shall** provide the facility to express comments.
[8]

Note(8)

We define the following levels of conformance of the abstract syntax of a basic OSM 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

A logic translation is conformant with DOL if it is presented either as an institute morphism or as an institute comorphism. If the languages are presented additionally as institutions, the translations **may** also be presented as an institution morphism or 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 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 OSM language with DOL

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

These are the conformance levels, listed from highest to lowest:

[1] Institutes and institutions are necessarily monotonic; conformance criteria for non-monotonic logics are still under development. However, minimization provides non-monotonic reasoning in DOL.

[8] NOTE: say something about "infrastructure theories", i.e. axiomatizations of one logic in another logic. Providers of OSM 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.

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.[9] Note(9)
- The schema shall not forbid attributes from foreign namespaces (here: the DOL namespace) on any elements[10] Note(10)

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.
- The serialization shall specify ways of giving IRIs or URIs to all structural elements of an OSM.[11] Note(11)
- There shall be no additional rules that forbid properties from foreign namespaces (here in particular: the annotation vocabularies recommended by DOL) to be stated about arbitrary subjects[12] Note(12)

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 OSM.
  - The serialization provides line-based comments (ranging from an indicated position to the end of a line) but at the same time allows for flexibly placing line breaks before/after each token that represents a structural element of an OSM.

Standoff markup conformance An OSM 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.

[13] Note(13)

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 that non-logical symbol identifiers have the syntactic form of a CURIE, or any subset of the CURIE grammar that

[9] NOTE: FYI: That means that in a heterogeneous OSM 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).

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

[11] 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.

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

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

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 map the non-logical symbol identifiers of a basic OSM to IRIs in multiple namespaces (cf. clause 8.5.3)

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 OSM to IRIs, which are located in one flat namespace.

[14] Note(14)

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

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

For any conforming OSM language, logic, and serialization of an OSM 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.[15] Note(15)

2.4. Conformance of a serialization with DOL

While clause 2.2 covered the conformance of serializations of *basic* OSM languages with DOL, this clause addresses serializations of DOL itself. This international standard specifies one text-based serialization in annex A, but one can specify further serializations according to the following conformance criteria.[16] Note(16)

It is required that a specification of a conforming serialization be given as a set of mappings from the abstract syntax (cf. clause 8), where each production rule of the abstract syntax is mapped to a production rule of the concrete syntax. Instead of such an explicit mapping, an implicit mapping may be provided via a self-contained EBNF for the concrete syntax, whose nonterminal symbols have to match the nonterminal symbols of the EBNF of the abstract syntax.[17] Note(17)

[14] NOTE: TODO: add the possibility to provide an alternative CURIE separate character in the conformance declaration, as mentioned in clause 8.5.3
[15] NOTE: FYI: that opens the door for, e.g., OMDoc
[16] NOTE: FYI: This section is analogous to Common Logic's "Dialect Conformance".
[17] NOTE: TODO: Say that in a more formal way! – Is it comprehensible? Does it sufficiently state what we have in mind?

2.5. Conformance of a document with DOL

A document conforms with DOL if it contains a well-formed DOL OSM. 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 8), such as:

- the logic of each OSM that is part of the distributed OSM
- 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 OSM 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.

2.6. Conformance of an application with DOL

In practice, DOL-aware *applications* are also allowed to deal with documents that are not conforming with DOL according to the criteria established in clause 2.5. 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 OSM 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 is allowed to process documents that mix OWL and Common Logic ontologies *without* explicitly declaring the respective logics, as the respective syntaxes of OWL and Common Logic allow for telling such ontologies apart by looking at the keywords used. However, for DOL conformance, that application has to be capable of exporting documents with explicit references to the logics used.

[18] Note(18)

[19] Note(19)

[18] NOTE: applications need to strip DOL annotations from embedded fragments in other OSM languages
[19] NOTE: applications need to be able to expand CURIEs into IRIs when requested

3. Normative References

- 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/
- ISO/IEC 14977:1996Information technology – Syntactic metalanguage – Extended BNF
- 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/
- 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/
- ISO/IEC 24707:2007Information technology – Common Logic (CL): a framework for a family of logic-based languages
- 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)
- IETF/RFC 3986Uniform Resource Identifier (URI): Generic Syntax. January 2005. http://tools.ietf.org/html/rfc3986
- IETF/RFC 3987Internationalized Resource Identifiers (IRIs). January 2005. http://tools.ietf.org/html/rfc3987
- IETF/RFC 5147URI Fragment Identifiers for the text/plain Media Type. April 2008. http://tools.ietf.org/html/rfc5147
- 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/
- 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/
- 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/
- ISO/IEC 10646Information technology – Universal Multiple-Octet coded Character Set (UCS)
- 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/
- W3C/TR REC-rdf-mt:2004RDF Semantics. W3C Recommendation, 02 February 2004. http://www.w3.org/TR/2004/REC-rdf-mt-20040210/

[20] Note(20)

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

4. Terms and Definitions

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

4.1. OSMs

OSM (ontology, specification or model)

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

OSM language

language that is used for the formal specification of *OSMs*, equipped with a formal, declarative, logic-based semantics, plus non-logical *annotations*
EXAMPLE OSM languages include OWL, Common Logic, F-logic, UML class diagrams, RDFS, and OBO.

non-logical symbol

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

signature

set of all *non-logical symbols* of an *OSM*

model

semantic interpretation of all *non-logical symbols* of a *signature*
NOTE A model of an OSM is a model of the signature of the OSM that moreover *satisfies* all the *axioms* of the OSM.
NOTE This term is not to be confused with model in the sense of modeling.

term

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

sentence

term that is either true or false in a given *model*, i.e. which is assigned a truth value in this *model*.[21] Note(21)
NOTE In a *model*, on the one hand, a sentence is always true or false. In an *OSM*, 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 OSM language at hand.

axiom

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

theorem

sentence that has been proven from other *axioms* and *theorems*

satisfaction relation

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

4.2. 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

[21] NOTE: FYI: From Common Logic, I changed "unit of logical text" to "term".

(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

elementOSM[22]

Note(22)

any *resource* in an *OSM* (e.g. a *non-logical symbol*, a *sentence*, a *correspondence*, the *OSM* 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 BernersLee:LinkedData2006

NOTE The linked data principles (adapted from BernersLee:LinkedData2006 and its paraphrase at Wikipedia:LinkedData2011) 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 W3C:REC-rdf-syntax-grammar-20040210, is the most common format for publishing linked data. However, its usage is not mandatory.

NOTE Using HTTP content negotiation rfc2616 it is possible to serve representations in different formats from the same URL.

4.3. OSM Annotation and Documentation

annotation

additional information without a logical semantics that is attached to an *element* of an *OSM*

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 OSM. The predicate is an RDF property defined in an external OSM and describes in what way the annotation object is related to the annotation subject.

NOTE According to note 4.3 it is possible to interpret annotations under an RDF semantics. "Without a logical semantics" in this definition means that annotations to an OSM are not considered sentences of the OSM.

[22] NOTE: FYI: This term is adopted from OWL 2, where it is used, but in a non-normative context.

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

OSM documentation

set of all *annotations* to an *OSM*, plus any other documents and explanatory comments generated during the entire OSM building process
NOTE Adapted from SuarezFigueroaEtAl:OntologyGlossary2008

4.4. Structured OSMs

basic OSM

set of *non-logical symbols*, *sentences*, *annotations* about them, which is used as a building block for a larger *OSM*

structured OSM

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

subOSM

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

extension

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

consequence-theoretic conservative extension

extension that does not add new *theorems* (in terms of the unextended *signature*)
NOTE An extension O_2 of an OSM 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 *OSM*
NOTE An extension O_2 of an OSM 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 *OSM* 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 *OSM* 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 *OSM* 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 *OSM* 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

subOSM that *conservatively extends* to the whole *OSM*

NOTE The conservative extension can be either model-theoretic or consequence-theoretic; without qualification, the consequence-theoretic version is used.

module extraction

activity of obtaining from an *OSM* concrete *modules* to be used for a particular purpose (e.g. to contain a particular sub-*signature* of the original *OSM*)

NOTE Cited and slightly adapted from SuarezFigueroaEtAl:OntologyGlossary2008

NOTE The goal of module extraction is "decomposing an OSM into smaller, more manageable modules with appropriate dependencies" DAquin:OntologyExtraction2010

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.

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 circ1, circ2.

4.5. Links Between OSMs

correspondence

relationship between an *non-logical symbol* e_1 from an *OSM* O_1 and an *non-logical symbol* e_2 from an OSM 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 from the context ("equivalence" for alignments, and "equality" for logical links)[23] ; when c is omitted, it defaults to 1. Note(23)

NOTE A confidence value of 1 does not imply logical equivalence (cf. KutzEtAl:ChineseWhispers2010 for a worked-out example).

linkOSMs[24]

Note(24)

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

logical link

link that has a formal, logic-based semantics

NOTE Logical links are given as sets of correspondences, which are required to be *signature morphisms*.

NOTE Some specific kinds of logical links will be introduced below.

[23] 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?

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

interpretation
view

logical link that postulates a relation between two *OSMs*

NOTE An interpretation typically leads to proof obligations, i.e. one has to prove that axioms of the source OSM of the link are theorems in the target OSM.

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.
[25]

Note(25)

equivalence

logical link ensuring that two *OSMs* share the same definable concepts

NOTE Two OSMs are equivalent if they have a common definitional extension. The OSMs may be written in different OSM languages.

interface signature

signature mediating between an *OSM* and a *module* of that *OSM* in the sense that it contains those *non-logical symbols* that the *sentences* of the *module* and the *sentences* of the *OSM* have in common

NOTE Adapted from CuencaGrauEtAl:ExtractingModulesOntologies2007

module relation

logical link stating that one *OSM* is a *module* of the other one.

virtual import

logical link between two *OSMs* such that one OSM behaves as if it were imported into the other

NOTE Semantically, a virtual 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.

renaming

logical link assigning new names to some *non-logical symbols* of an *OSM*

reduction

logical link reducing an *OSM* to a smaller *signature*

[25] NOTE: the DOL structuring language should include constructs for claiming that two OSMs are equivalent, see `http://www.informatik.uni-bremen.de/~okutz/hyperontologies.pdf`, p.42ff.

union

aggregation of several *OSMs* to a new *OSM* where (only) identically-named *non-logical symbols* of the involved *OSMs* are identified

combination

aggregation of several *OSMs* along *links* to a new *OSM* where (only) the linked *non-logical symbols* of the involved *OSMs* are identified

alignment

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

matching

algorithmic procedure that generates an *alignment* for two given *OSMs*

4.6. Features of OSM Languages

OSM language translation

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

NOTE An OSM 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 *OSM language* that allows for expressing *non-logical symbols* and *sentences* in a logical language

structuring language aspect

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

annotation language aspect

the (unique) *language aspect* of an *OSM language* that allows for expressing comments and annotations

profile

sublanguage of an *OSM language* 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*.

4.7. OSM Language Serializations

serialization

specific syntactic encoding of a given *OSM language*
NOTE Serializations serve as standard formats for exchanging OSMs 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 serialization of RDF (e.g. RDF/XML, Turtle, ...)
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).

standoff markup

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

4.8. Logic

logic

specification of valid reasoning that comprises *signatures*, *sentences*, *models*, and a *satisfaction relation* between *models* and *sentences*
NOTE Most OSM 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 OSM languages and their logics and serializations.

logical metaframework

framework for the mathematical formalisation of possible *logics*

institute

logical metaframework based on set theory and order theory
NOTE See clause 9 for a formal definition.

institution

logical metaframework based on category theory, also covering the semantics of *renamings* and *combinations*
NOTE See annex I 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 *OSMs*

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 OSMs expressed in the source *logic* with means of the expressivity of the target *logic*
NOTE A unique maximal approximation need not exist.

sublogic

syntactic restriction of a *logic*

heterogeneous OSM

OSM whose parts are supported by different *logics*

4.9. Interoperability

[26] Note(26)

[27] Note(27)

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

[27] NOTE: FYI: Definitions in earlier drafts were not quite helpful:
• OSM integration := "combination of different OSMs into a coherent whole, via alignments"
•OSM interoperability := "relation among OSMs (via OSM 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 OSMs."

[28] Note(28)

logically interoperable

property of *structured OSMs*, which may be written in different *OSM languages* [29]based on different *logics*, of being usable jointly in a coherent way (via suitable *OSM language translations*), such that the notions of their overall consistency and logical entailment have a precise logical semantics Note(29)

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

4.10. Distributed OSMs and the Distributed Ontology, Modeling and Specification Language

distributed OSM hyperontology

collection of *OSMs*, possibly written in different *OSM languages*, linked by *links*

distributed ontology, modeling and specification language DOL

language for formalizing *distributed OSMs*, whose syntax and semantics are specified in this International Standard

NOTE When viewed as an *OSM language*, DOL has *OSMs* as its *non-logical symbols*, and *links* as its *sentences*.

[28] 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
metadata interoperability
interoperability concerning the creation, meaning, computation, use, transfer, and exchange of descriptive data

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

5. Symbols

As listed below, these symbols and abbreviations are generally for the main clauses of the standard. 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 W3C:REC-owl2-rdf-based-semantics-20091027
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

6. Additional Information

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.[30] Note(30)

In complex applications, which involve multiple OSMs with overlapping concept spaces, data mapping is also required on a higher level of abstraction, viz. between different OSMs, and is then called OSM alignment. While OSM alignment is most commonly studied for OSMs formalized[31] in the same OSM language, the different OSMs used by complex applications may also be written in different OSM languages. This international standard faces this diversity not by proposing yet another OSM 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 OSMs that are written in different formalisms. It specifies DOL (Distributed Ontology, Modeling and Specification Language), a formal language for expressing not only OSMs but also links between OSMs formalized in different OSM languages. Note(31)

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

OSM languages are declarative languages for making ontological distinctions formally precise. They are distinguished by the following features:

Logic Most commonly, OSM 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 OSM into reusable parts, reusing parts of other OSMs, mapping imported symbols to those in the importing OSM, and asserting additional properties about imported symbols.

Annotation means of attaching human-readable descriptions to OSM symbols, addressing knowledge engineers and service developers, but also end users of OSM-based services.[32] Note(32)

[30] 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.

[31] NOTE: spell-check everything for British

[32] NOTE: TODO Christoph: reformulate. DOL enables the use of annotations

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 OSM engineering as well as the engineering of OSM-based software.

Within the DOL framework, existing OSMs in conforming established languages such as OWL or Common Logic remain as they are, acknowledging the wide tool support these languages enjoy. 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 OSMs as non-disruptive annotations, or they can be provided as standoff markup, pointing to the OSMs 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 OSM languages.

7. Requirements and design overview

[33] Note(33)

This clause is informative. Its purpose is to briefly describe the purposes of the Distributed Ontology, Modeling and Specification Language (DOL) and the overall guiding principles and constraints on its syntax and semantics.

[34] Note(34)

[35] Note(35)

7.1. DOL requirements

DOL has been designed and developed with several requirements in mind, all arising from its intended role of enabling OSM interoperability. The use of "**should**" in the rest of clause 5 indicates a desired goal but is not required of DOL (in accordance with Annex H of ISO/IEC Directives – Part 2).

[36] Note(36)

7.1.1. DOL is free, generally applicable, open, and extensible.

DOL **should** be

free This international standard **should** be freely available for unrestricted use.

generally applicable It **should** neither be restricted to OSMs in a specific domain, nor to foundational OSMs, nor to OSMs represented in a specific OSM language, nor to OSMs stored in any specific repositories.

open It **should** support mapping, integrating, and annotating OSMs across arbitrary internet locations. It **should** make use of existing open standards wherever suitable. The criteria for extending DOL (see next item) **should** be transparent and explicit.

extensible It **should** provide a framework into which any existing, and, desirably, any future OSM language can be plugged.

---

[33] NOTE: TODO: Get rid of formal **should**/**shall** language (e.g. in clause headers: **should**/**shall** applies to conforming implementations anyway, rather than to this standard itself!) – not necessary in an informative clause. TM: I did this in the clause headings, and for the design part also in the clause texts.

[34] NOTE: add somewhere: DOL is a meta language and can be used with OSM languages of any expressiveness. As a Meta-language, DOL provides a framework for combining and relating OSMs written in specific OSM languages. However, DOL cannot be used for writing new basic OSMs.

[35] NOTE: add ref to annex K

[36] NOTE: TODO: give quick overview here. Create clause 5.2 for requirements, and 5.3 for design overview. TM: done

DOL **shall** be applicable to any OSM language that has a formal, logic-based semantics or a semantics defined by translation to another OSM language with such a formal semantics. The annotation framework of DOL **should** additionally be applicable to the non-logical constructs of such languages. This international standard [37] **shall** specify formal criteria for establishing the conformance of an OSM language with DOL. Annexes **shall** establish the conformance of a number of relevant OSM languages with DOL; a registry shall offer the possibility to add further (also non-standardized) languages:[38] Note(37) Note(38)

normative OWL, Common Logic, RDFS[39] Note(39)

informative F-logic, UML class diagrams, OBO (see appendix H for a longer list)

7.1.2. DOL is a logic-agnostic metalanguage, in the sense that its constructs can be used for many different logics.

[40] Note(40)

DOL **shall** provide syntactic constructs for structuring OSMs regardless of the logic their sentences are formalized in. DOL **should** provide syntactic constructs for

- basic and structured OSMs (and facilities to identify them in a globally unique way),
- explicit extraction of modules from existing OSMs, [41]such that, e.g., changes in the OSM can be propagated to the extracted module. Note(41)
- links between OSMs (cf. clause 7.2.2), including interpretations, relations between OSMs and their modules, as well as alignments.

DOL **shall not** provide its own constructs for expressing sentences. Instead, it **shall** *inherit* the logical language aspects of conforming OSM languages. It **should** be possible to literally include sentences expressed in such OSM languages in a DOL OSM.

DOL **shall** provide an initial set of built-in approximation methods and module extraction selectors. Additionally, it **shall** provide a means of referring to approximation methods and module extraction selectors defined externally of this international standard.[42] Note(42)

DOL **shall** provide an initial vocabulary for expressing relations in correspondences (as part of alignments between OSMs). Additionally, it **shall** provide a means of reusing relation types defined externally of this international standard.

DOL **shall not** provide an annotation vocabulary, i.e. it **shall** neither provide annotation properties nor datatypes to be used with literal annotation objects. Instead, an informative annex **shall** recommend existing annotation vocabularies for use with DOL.

7.1.3. DOL has user- and machine-readable serializations.

[43] In the interest of wide applicability and tool support, DOL **should** support multiple Note(43)

[37] NOTE: FYI: We can afford to say "shall" here, as these criteria are really something that we can fully provide

[38] 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.

[39] NOTE: RIF as well? See `http://trac.informatik.uni-bremen.de:8080/OntoIOp/ticket/16`

[40] NOTE: here and elsewhere: remove "shall" from section headers

[41] NOTE: Q-AUT: This rather sounds like a use case description to me than like a requirement. Move it somewhere else? Where?

[42] NOTE: FYI: In practice we will use IRIs for that purpose.

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

alternative serializations. In particular, there **should** be a text serialization targeting human readers and writers, as well as serializations optimized for machine processability.

This international standard **shall** specify criteria for a serialization to conform with DOL, and it **shall** specify the following conforming serializations:

- a human-readable **text serialization**
- a machine-processable **interchange format**, to be implemented as

  an XML schema (DOL XML) particularly targeting document or form based authoring, validation, as well as translation from and to serializations of existing OSM languages[44], and Note(44)

  an RDF vocabulary (DOL RDF) particularly targeting interlinking and annotation.

The **text serialization** in particular **shall** offer a syntax for abbreviating identifiers of resources within OSMs in a way that does not require authors to write down their full global identifiers.

An OSM implemented in DOL **should** be able to comprise parts formalized in any OSM language; any serialization of DOL **should** be able to literally include such parts, regardless of the OSM language serialization they have been written in. [45] Additionally, an OSM Note(45) implemented in DOL **should** be able to refer to any external OSMs formalized in any OSM language, as long as they can be identified in a globally unique way.

Existing OSMs in existing XML serializations (e.g. XCL) or text serializations (e.g. OWL Manchester Syntax) **should** validate as DOL OSMs with a minimum amount of syntactic adaptation. Existing OSM files/documents **should** be usable in a DOL context without the need for modification.

7.1.4. DOL has a well-defined formal, logic-based semantics.

The structural elements and structural links of DOL **should** have a formal, logic-based semantics.

This international standard specifies OSM language translations between conforming languages:[46] Note(46)

- OSM language translations between their logical language aspects. For any such OSM language translation its properties **should** be determined, e.g. whether it is a sublogic, a theoroidal translation, etc.
  [47] Note(47)
- OSM language translations between their structuring language aspects and the structuring language aspect of DOL.

DOL can express the application $T(O)$ of an OSM language translation $T\colon L_1 \to L_2$ to an OSM O written in langauge L_1[48], see the abstract syntax category `Translation` in clause 8. Note(48) DOL need not be capable of expressing OSM language translations.

[44] NOTE: Q-ALL: I think it's reasonable to call this "DOL XML" instead of "DIF XML", as to emphasize the "brand" DOL

[45] NOTE: FYI: advanced namespacing is the solution that addresses this requirement

[46] NOTE: FYI: we shall establish the conformance of an initial set of languages with DOL. As a part of that work we deliver the "onto-logical translation graph" between these languages. Anyone, who wants to establish the conformance of another language with DOL, has to add a node to the graph, and at least one edge from/to an existing node.

[47] NOTE: TODO: meet the requirements of people who combine OWL reasoners with Prolog. Some

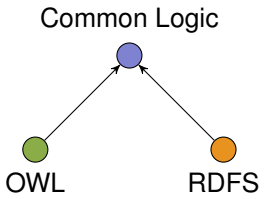

Figure 7.1.: Translating two OSM languages into a third one

For each pair L_1 and L_2 of OSM languages, OSM language translations T_1 and T_2 into a common target OSM language L_T **should** be specified. (If L_T does not exist, the only way to express a heterogeneous OSM involving L_1 and L_2 may be to keep the DOL expression and the individual OSMs in L_1 and L_2.) These **should** be translations into an OSM language that is more expressive than both L_1 and L_2, such that the union of the images of the translations is a subset of the target OSM language ($T_1(L_1) \cup T_2(L_2) \subseteq L_T$). Figure 7.1 outlines such an example, where Common Logic serves as the common target for OWL and RDFS, as it is more expressive than either of them.[49] If such a target OSM language or suitable translations do not yet exist, translations into a less expressive language may be specified as an alternative, such that the intersection of the images of the translations forms a subset of the target language ($T_1(L_1) \cap T_2(L_2) \subseteq L_T$), which **should** be as large as possible. For example, an OSM language that is more expressive than both Common Logic and F-Logic does not yet exist; therefore, it would be possible to specify translations into the first-order logic subset of either OSM language. Note(49)

Reductions of DOL to conforming OSM languages, as well as approximations of DOL in conforming OSM languages, are specified. This is to ensure that OSMs that have originally been written in DOL can be reused and extended in the respective target OSM languages. While approximations are desirable that preserve as much information from the DOL OSM as the logic underlying the target OSM language is capable of expressing (possibly after a suitable OSM language translation), there **should** at least be a trivial reduction that throws away all syntactic constructs of the DOL OSM that are not syntactic constructs in the target OSM language. However, those constructs are optionally preserved as annotations in the output (cf. clause 7.2.4 for annotations).
[50] Note(50)

7.2. DOL design

We give an overview of the most important and innovative language constructs of DOL. Details can be found in clause 8.

---

additional research needed on combining logics that have a model theory with those that don't

[48] NOTE: FYI: T shall be identified by a IRI. There might be multiple different possible translations between two languages, e.g. two ways of expressing OWL roles in CL (binary predicate vs. boolean function). But in order to free the user from always writing down such IRIs, we shall specify some defaults in our translation graph.

[49] NOTE: FYI: In the context of that, specify when a document/an OSM conforms with DOL.

[50] NOTE: TODO: provide example of integrating two OSMs in a single-sorted logic by translating into many-sorted logic, where only many-sorted logic would guarantee consistency

7.2.1. DOL allows for expressing logically heterogeneous OSMs and literal reuse of existing OSMs.

DOL allows for expressing logically heterogeneous OSMs, i.e. to combine, in the same DOL document, sentences and structured OSMs expressed in different conforming OSM languages and logics. It is possible to reuse sentences or structured OSMs of previously existing OSMs in conforming languages by literally including them into a DOL OSM. A minimum of wrapping constructs and other annotations (e.g. for disambiguating in what language a sentence has been expressed) are provided.[51] See the abstract syntax category `Onto` in clause 8. Note(51)

7.2.2. DOL allows for expressing links between OSMs.

DOL provides a syntax for expressing links between OSMs – logical links as well as alignments. One use case illustrating both is sketched in Figure 7.2. This international standard specifies a set of logical link types and a set of non-logical link types.

Logical links supported by DOL include:

- imports (particularly including imports that lead to conservative extensions), see the abstract syntax categories `OntoRef` and `ExtensionOnto` in clause 8.
- interpretations, see the abstract syntax category `IntprDefn` in clause 8.
- links between OSMs and their modules, see the abstract syntax category `ModuleRelDefn` in clause 8.

DOL allows for expressing signature translations in such links, see the abstract syntax category `SymbolMapItems` in clause 8.

DOL need not be able to fully represent logical translations but is capable of referring to them.
[52] Note(52)

DOL also allows for combining/merging OSMs along such links, see the rule for `combination` for the abstract syntax category `Onto` in clause 8.

7.2.3. DOL allows for expressing OSMs and links at different levels of detail

OSMs and OSM links expressed in DOL can be based on a number of implicit assumptions about which OSM language translation or which ontoloy matcher has been employed. Depending on the OSM engineering workflow or application setting, it can be useful to keep these assumptions implicit, or to make them explicit. DOL allows for keeping such assumptions implicit if desired. It also allows for explicitly writing them down as annotations to the OSM. This international standard specifies a translation that expands any DOL OSM with implicit assumptions into its explicit counterpart.

The following list covers the possible cases where DOL allows for making information implicit or explicit:

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

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

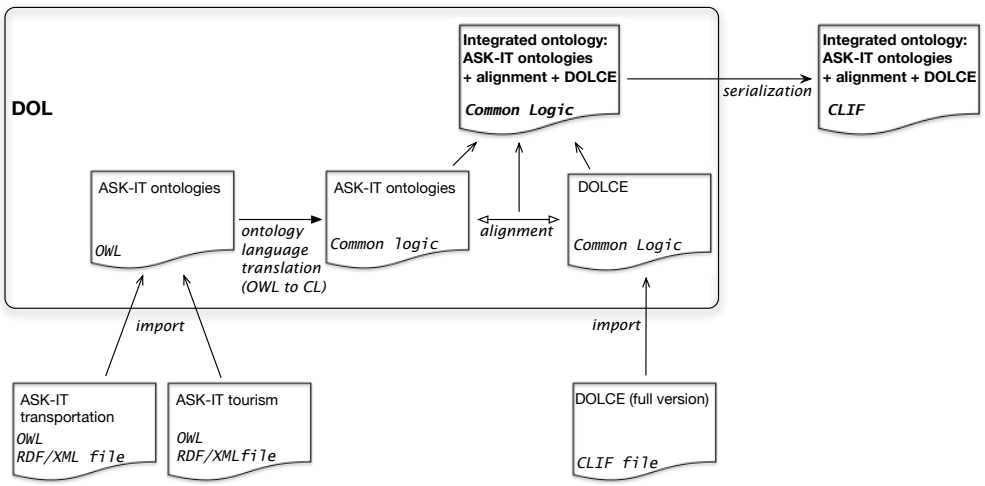

Figure 7.2.: Linking two OSMs formulated in different OSM languages

default OSM language translations A heterogeneous OSM can import several (structured) OSMs 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 the semantics of the heterogeneous OSM requires a translation into a common target language to be applied (cf. clause 7.1.4). 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 OSM alignments, which DOL is able to express, may have been obtained by running different OSM matching algorithms. If, in a given OSM 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 OSM requires that information, it is possible to make it explicit.[54] Note(54)

[55] Note(55)

[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

7.2.4. DOL allows for rich annotation and documentation of OSMs.

[56] DOL allows for annotations in the full generality specified in clause 4.3. The DOL serializations allow for fine-grained embedding of annotations into OSMs. Note(56)

The DOL serializations also allow for annotating existing OSMs 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 OSM language; for XML serializations of OSM languages, RDFa extensions are specified, so that DOL RDF can be embedded.

A list of RDF vocabularies for annotating OSMs is recommended as an annex to this international standard.

[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 will allow for identifying anything of relevance in a distributed OSM, 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.

8. DOL abstract syntax

[57] Note(57)

8.1. Abstract syntax categories

DOL provides abstract syntax categories for[58] Note(58)

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

Additionally, the categories of the abstract syntaxes of any conforming OSM 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 standard 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 } ) ;`.

8.2. Distributed OSMs

A distributed OSM consists of at least one (possibly heterogeneous) OSM, plus, optionally, links between its participating (heterogeneous) OSMs. More specifically, a distributed OSM consists of a name, followed by a list of `DistOntoItems`. A `DistOntoItem` is either an OSM definition (`OntoDefn`), or a link between OSMs (`LinkDefn`), or a `Qualification` selecting a specific OSM language, logic and/or syntax that is used to interpret the subsequent `DistOntoItems`. Alternatively, a distributed OSM can also be the verbatim inclusion of an OSM written in an OSM language that conforms with DOL (`OntoInConformingLanguage`; cf. 2.1).

```
DistOnto                = [ PrefixMap ] , DistOntoDefn
                        | OntoInConformingLanguage ;
DistOntoDefn            = 'dist-onto-defn' , DistOntoName , { DistOntoItem } ;
```

[57] NOTE: TODO: split into two clauses: one for abstract syntax, one for semantics. TM: done

[58] NOTE: Q-AUT: In the previous draft we had more fine-grained categories: OSM languages, OSM language translation, links between OSMs, OSM combination (are these the colimits that we now call "combinations"?)

```
OntoInConformingLanguage = ? language and serialization specific ? ;
DistOntoItem             = OntoDefn | LinkDefn | Qualification ;
Qualification            = LanguageQual | LogicQual | SyntaxQual ;
LanguageQual             = 'lang-select' , LanguageRef ;
LogicQual                = 'logic-select' , LogicRef ;
SyntaxQual               = 'syntax-select' , SyntaxRef ;
DistOntoName             = IRI ;
```

[59] Note(59)

At the beginning of a distributed OSM, one can declare a `PrefixMap` for abbreviating long IRIs; see clause 8.5 for details.

8.3. Heterogeneous OSMs

An OSM (`Onto`) can be one of the following:

- a basic OSM `BasicOnto` written inline, in a conforming serialization of a conforming OSM language[1],
- a translation of an OSM into a different signature or OSM language,
- a reduction of an OSM 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 OSM, 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 OSMs,
- an extension of an OSM by other ones, it can be optionally named and/or marked as conservative, monomorphic, definitional or implied,
- a module extracted from an OSM, using a restriction signature,
- a reference to an OSM existing on the Web,
- an OSM qualified with the OSM language that is used to express it,
- a combination of OSMs (technically, this is a colimit, see ZimmermanEtAl06),
- a minimization of an OSM, 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).

[59] 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-link (will always use full IRIs, and abbreviate them by syntactic namespaces)

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

```
BasicOnto            = OntoInConformingLanguage ;
MinimizableOnto      = BasicOnto
                     | 'onto-ref' , OntoRef , [ ImportName ] ;
ExtendingOnto        = MinimizableOnto
                     | 'minimize' , MinimizableOnto ;
Onto                 = ExtendingOnto
                     | 'minimize-symbols' , Onto , CircMin , CircVars
                     | 'translation' , Onto , Translation
                     | 'reduction' , Onto , Reduction
                     | 'module-extract' , Onto , Extraction
                     | 'approximation' , Onto , Approximation
                     | 'union' , Onto , [ ConsStrength ] , Onto
                     | 'extension' , Onto , ExtensionOnto
                     | 'qual-onto' , { Qualification } , Onto
                     | 'bridge' , Onto, { Translation } , Onto
                     | 'combination' , CombinedElements , ExcludeExtensions ;

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

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

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

SymbolItems          = 'symbol-items' , ( Symbol , { Symbol } ) ;
SymbolMapItems       = 'symbol-map-items' , ( SymbolOrMap , { SymbolOrMap } ) ;[60]

Extraction           = 'extraction', Conservative, InterfaceSignature, ExtractionMethod ;

Approximation        = 'approximation' , ApproxMethod ;[61]

ExtensionOnto        = [ ConsStrength ] , [ ExtensionName ] , ExtendingOnto ;

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

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

InterfaceSignature   = 'interface-signature' , SymbolItems ;
```

Note(60)

Note(61)

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

[61] NOTE: TODO: can we identify Approximation with ApproxMethod?
CL: At least in our concrete syntax it's easier to keep them separate; please have a look at the corresponding concrete syntax (and revise *that* if necessary).

```
CombinedElements     = OntoOrLink [ Id ] [62] Ref { OntoOrLinkRef } ;      Note(62)
ExcludeExtensions    = 'exclude-imports' , { ExtensionRef } ;

ImportName           = IRI ;
ExtensionName        = IRI ;
```

An OSM definition `OntoDefn` names an OSM. It can be optionally marked as consistent, using `ConsStrength`.[2]. An `SymbolItems`, used in an OSM `Reduction`, is a list of non-logical symbols that are to be hidden. A `LogicReduction` denotes a logic reduction to a less expressive OSM language. A `SymbolMapItems`, used in OSM `Translations`, maps symbols to symbols[63] , or a logic translation. An OSM language translation `OntoLangTrans` Note(63)
or `ApproxMethod` can be either specified by its name (optionally qualified with source and target OSM 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 OSM language of the current OSM).

```
OntoDefn          = 'onto-defn' , OntoName , [ ConsStrength ] , Onto ;

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

OntoName          = IRI ;

OntoRef           = IRI ;
OntoOrLinkRef     = IRI ;
ExtensionRef      = IRI ;

LoLaRef           = LanguageRef | LogicRef ;

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

OntoLangTrans     = 'named-trans' , OntoLangTransRef
                  | 'qual-trans' , OntoLangTransRef , LoLaRef , LoLaRef
                  | 'anonymous-trans' , LoLaRef , LoLaRef
                  | 'default-trans' , LoLaRef[64]  ;                      Note(64)

OntoLangTransRef = IRI ;
```

[62] NOTE: or should we leave `OntoOrIntprRef`? Does combination have a semantics for (informal) alignments? TM: indeed yes, at least if we ignore confidence values (or all are equal to 1). Then, an (informal) alignment leads to a span of logical links, and this can be used in the combination. This feature will greatly increase the use of combinations. I will work this out in the semantics.

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

[63] 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 OSM language independent way would require additional institution infrastructure – and the same effect can be achieved by auxiliary definitional

```
ApproxMethod     = 'named-approx', ApproxMethodRef
                 | 'qual-approx' , ApproxMethodRef , LoLaRef
                 | 'default-approx' , LoLaRef[65]  ;                    Note(65)
ApproxMethodRef  = IRI ;

ExtractionMethod = IRI ;
```

8.4. Links

A link provides a connection between two OSMs. A link definition is the definition of either a named interpretation (`IntprDefn`), a named declaration of the relation between a module of an OSM and the whole OSM (`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 OSM logically follows from the target OSM is generated. In contrast to this, an alignment just provides a connection between two OSMs without logical semantics, using a set of `Correspondences`. Each correspondence may map some OSM 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 OSM actually is a module of some other OSM with respect to the `InterfaceSignature`.

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

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

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


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

extensions, cf. Colore (so promote this, informatively, as a "best practice"?)

[64] 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 OSM that is translated.)

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

```
AlignDefn            = 'align-defn' , AlignName , [ AlignCard ] , AlignType[3]
                                    { Correspondence } ;
AlignName            = IRI ;
AlignCards           = AlignCardForward , AlignCardBackward[66]  ;          Note(66)
AlignCardForward     = 'align-card-forward' , AlignCard ;
AlignCardBackward    = 'align-card-backward' , AlignCard ;
AlignCard            = 'injective-and-total'
                     | 'injective'
                     | 'total'
                     | 'neither-injective-nor-total' ;
AlignType            = 'align-type' , Onto , Onto ;

Correspondence       = CorrespondenceBlock
                     | SingleCorrespondence
                     | 'default-correspondence'[67]  ;                      Note(67)
CorrespondenceBlock  = 'correspondence-block' , [ RelationRef ] , [ Confidence ][68]   Note(68)
                                                 { Correspondence } ;
SingleCorrespondence = 'correspondence' , SymbolRef , [ RelationRef ] , [ Confidence ] ,
                                          TermOrSymbolRef , [ Correspondence ][69]   Note(69)
 ;
CorrespondenceID     = IRI ;
SymbolRef            = IRI ;
TermOrSymbolRef      = Term | SymbolRef ;
RelationRef          = 'subsumes' | 'is-subsumed' | 'equivalent' | 'incompatible'
                     | 'has-instance' | 'instance-of' | 'default-relation'[70]   Note(70)
 | IRI ;
Confidence           = Double[71]  ;                                        Note(71)
Double               = ? a number ∈ [0, 1] ? ;
```

[72] Note(72)

A symbol map in an interpretation is **required** to cover all non-logical symbols of the source OSM; the semantics specification in clause 9 makes this assumption[4]. Applications

[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".

[66] NOTE: TODO: mention that the default is twice "injective and total"

[67] 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

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

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

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

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

[72] 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 link.

shall implicitly map those non-logical symbols of the source OSM, for which an explicit mapping is not given, to non-logical symbols of the same (local) name in the target OSM, wherever this is uniquely defined – in detail:

Require: O_s, O_t are OSMs
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
  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 8.5)
  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

[73] Note(73)

8.5. Identifiers

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

8.5.1. IRIs

In accordance with best practices for publishing OSMs on the Web, identifiers of OSMs and their elements **should** not just serve as *names*, but also as *locators*, which, when dereferenced, give access to a concrete representation of an OSM or one of its elements. (For the specific case of RDFS and OWL OSMs, these best practices are documented in W3C:NOTE-swbp-vocab-pub-20080828. The latter is a specialization of the linked data principles, which apply to any machine-processable data published on the Web BernersLee:LinkedData2006.) It is recommended that publicly accessible DOL OSMs be published as linked data.

[74] Note(74)

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

[5] In practice, this can often have the effect of undoing an IRI abbreviation mechanism that was used when writing the respective OSMs (cf. clause 8.5). 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 OSMs.

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

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

distributed OSMs 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 OSM 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)[75] . Note(75)

In accordance with semantic web best practices such as the OWL Manchester Syntax W3C:NOTE-owl2-manchester-syntax-20091027, this International Standard 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 W3C:NOTE-swbp-vocab-pub-20080828:

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

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

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

8.5.2. 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 International Standard 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 allow for declaring a "default prefix" mapping [76] (covering CURIEs such as `:name`). Note(76)
- DOL does allow for declaring 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]

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

[6] specified below in clause 8.5.2

[76] 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.

CURIEs can occur in any place where IRIs are allowed, as stated in clause 8.5.1. 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, Section 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 *) ;
```

Prefix mappings can be defined at the beginning of a distributed OSM (specified in clause 8.2; these apply to all parts of the distributed OSM, including basic OSMs as clarified in clause 8.5.3). 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.

8.5.3. Mapping identifiers in basic OSMs to IRIs

While DOL uses IRIs as identifiers throughout, basic OSM 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 flogic does not use them at all.

However, DOL links as well as [77] certain operations on OSMs require making unambiguous references to non-logical symbols of basic OSMs (`SymbolRef`). Therefore, DOL provides a function that maps global identifiers used within basic OSMs 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 8.5.2). Note(77)

CURIEs that have a prefix may not be acceptable identifiers in every serialization of a basic OSM language, as the standard CURIE separator character, the colon (`:`), may not be allowed in identifiers. [78] 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(78)

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

Require: D is a distributed OSM

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

[78] 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 OSM 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?

Require: O is a basic OSM 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**
      $sep \leftarrow$ undefined
    else
      $sep \leftarrow$: {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 8.5.2 for the original grammar of CURIEs.}
    if sep is defined **then**
      $cp \leftarrow [NCName, sep], Reference$
    else
      $cp \leftarrow Reference$
    end if
    if id matches the pattern cp, where ref matches $Reference$ **then**
      if the match succeeded with a non-empty $NCName$ pn **then**
        $p \leftarrow concat(pn, :)$
      else
        $p \leftarrow$ no prefix
      end if
      if O binds p to an IRI pi according to the specification of S **then**
        $nsi \leftarrow pi$
      else
        $P \leftarrow$ 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**
            $nsi \leftarrow pi$
            break out of the **while** loop
          end if
          $P \leftarrow$ 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
      $i \leftarrow concat(nsi, ref)$
    else
      return an error
    end if
  end if
  return i

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

While CURIEs used for identifying parts of a distributed OSM (cf. clause 8.5.2) are merely syntactic sugar, the prefix map for a basic OSM is essential to determining the semantics of the basic OSM within the distributed OSM. Therefore, any DOL serialization **shall** provide constructs for expressing such prefix maps, even if the serialization does not support prefix maps otherwise.

[79] Note(79)

8.6. DOL Serializations

Say how existing OSMs in existing serializations have to be adapted/wrapped (or ideally: not adapted at all!) in order to become valid OSMs in some DOL serialization.[80][81] Note(80) Note(81)

8.7. Annotations

[82] [83] Annotations always have a subject, which is identified by an IRI. Where the given OSM 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 OSM 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(82) Note(83)

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

[80] 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)

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

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

[83] 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.

9. 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 OSMs, this semantics reuses the existing semantics of the involved logics, as well as translations between these logics. The semantics of structured DOL OSMs and links is specified on top of this.

Translational Semantics The semantics of Common Logic is employed for all basic OSM languages, taking advantage of the fact that Common Logic is a common translation target for many OSM 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 OSMs with all parts written in Common Logic only. The latter is interpreted as in the case of the direct semantics, with basic OSMs 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 ThreeSemantics 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.[84] Since DOL involves heterogeneous OSMs, the semantics is parameterised over an arbitrary but fixed *heterogeneous logic environment*. This notion is defined below, it corresponds to a graph of OSM languages and OSM language translations. Below, also notions of *institute* and *institute comorphism* are defined, which provide formalisations of the terms "logic", resp. "logic translation". Note(84)

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. Some parts (namely symbol maps, `combinations` and the construct `monomorphic`; these are marked in ***bold italics***) of the abstract syntax need a more sophisticated and more general category-theoretic foundation in terms of institutions [Institutions]. More specifically, the notion of institute needs to be replaced by that of institutional logic [Meseguer89], and analogously for comorphisms [Comorphisms].

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

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

We recall the notion of satisfaction system carnielli2008analysis, called 'rooms' in the terminology of CharPar. 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.

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 $\Delta \subseteq Sen$ of sentences It is **consistent** iff it has at least one model. A theory $\Delta \subseteq Sen$ 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 $\Delta \subseteq Sen$ and $\varphi \in Sen$, 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 links 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** (α, β): $(Sen_1, \mathcal{M}_1, \models_1) \longrightarrow (Sen_2, \mathcal{M}_2, \models_2)$ consists of*

- *a sentence translation function α: $Sen_1 \longrightarrow Sen_2$, and*
- *a model reduction function β: $\mathcal{M}_2 \longrightarrow \mathcal{M}_1$, 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 $\varphi_1 \in Sen_1$ (**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 $\mathcal{I}_1 = (Sig_1, \leq_1, Sen_1, \mathcal{M}_1, \models_1)$ and $\mathcal{I}_2 = (Sig_2, \leq_2, Sen_2, \mathcal{M}_2, \models_2)$, an **institute comorphism** $\rho = (\Phi, \alpha, \beta) : \mathcal{I}_1 \longrightarrow \mathcal{I}_2$ consists of*

- *a monotone map $\Phi : (Sig^1, \leq^1) \to (Sig^2, \leq^2)$, and*

- *a partial corridor* $(\alpha, \beta) : (Sen_1, \mathcal{M}_1, \models_1) \to (Sen_2, \mathcal{M}_2, \models_2)$

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*,*

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

[85] Note(85)

9.1. 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 OSM languages and OSM 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 OSM languages and institutes, and a binary supports relation between OSM languages and serializations. Some of the comorphisms are marked as default translations.

[85] NOTE: Introduce exactness

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

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 OSMs, depending on a triple $L = (lang, logic, ser)$ comprising of an OSM language, a logic (institute) and a serialization as follows:

$$\boxed{sem_L(\Sigma, \texttt{BasicOnto}) = (\Sigma', \Delta') \text{ where } \Sigma' \geq \Sigma}$$

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

We further assume a language-specific semantics of complete (possibly structured) OSMs $sem(L, \texttt{OntoInSpecificLanguage}) = (\Sigma, \mathcal{M})$, 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 $ent(\Sigma)$, such that $\Sigma \leq \Sigma'$ implies $ent(\Sigma) \subseteq ent(\Sigma')$. This concludes the definition of heterogeneous logical enviroment.

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

[86] Note(86)

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

$sem(\Gamma, L, \texttt{'dist-onto-defn' , DistOntoName } DI_1, \ldots DI_n) = \Gamma'$
where $sem(\ldots sem(sem(\Gamma, L, DI_1), DI_2), \ldots, DI_n) = (\Gamma', L')$ [87] Note(87)

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

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

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

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

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

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

[86] NOTE: Q-AUT: @TM: Please decide if you like the stuff from 'dist-onto-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.

[87] NOTE: `DistOntoName` is not used. How could we use it? It seems that the individual OSMs are directly named with IRIs, and the `DistOntoName` is not relevant for that? Answer from telco: The `DistOntoName` 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 OSMs)

$sem((lang, logic, ser), \texttt{'logic-select' , LogicRef}) = (lang', \texttt{LogicRef}, ser)$
where $lang' = \begin{cases} lang, & \text{if } lang \text{ supports } \texttt{LogicRef} \\ \text{the unique language supporting } \texttt{LogicRef}, & \text{otherwise} \end{cases}$
Note that "the unique language supporting LogicRef" may be undefined; in this case, the semantics of the whole 'logic-select' , LogicRef construct is undefined.

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

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

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

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

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

Equations for OntoDefn and LinkDefn are given below.

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

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 OSM (MinimizableOnto) 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 $sem(\Gamma, L, O) = (logic(\Gamma, L, O), sig(\Gamma, L, O), Mod(\Gamma, L, O))$.

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

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

Note that Γ(OntoRef) may be undefined. That is, if a reference (IRI) to an OSM is not defined, the semantics of the enclosing DOL construct is undefined.

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

Semantics for MinimizableOnto 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).

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

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

Onto is interpreted in a context similar to that for MinimizableOnto; the difference being that there is no local environment.

[88] [89] [90]

[88] NOTE: TODO: specify semantics of module extraction Note(88)

[89] NOTE: TODO: specify semantics of approximation Note(89)

[90] NOTE: TODO: specify semantics of implicit translations using default translations Note(90)

O	$sem(\Gamma, L, O) = \ldots$
`ExtendingOnto`	$sem(\Gamma, L, (\emptyset, \mathcal{M}_\emptyset), \texttt{ExtendingOnto})$
`'minimize-symbol' , Onto , CircMin , CircVars`	$(I, \Sigma, \mathcal{M}')$ where $sem(\Gamma, L, \texttt{Onto}) = (I, \Sigma, \mathcal{M})$, $\Sigma_{min} = sem(\texttt{CircMin}, \Sigma)$, $\Sigma_{var} = sem(\texttt{CircVars}, \Sigma)$, $\Sigma_{fixed} = \Sigma \setminus (\Sigma_{min} \cup \Sigma_{var})$ and $\mathcal{M}' = \{ \quad M \in \mathcal{M} \mid M|_{\Sigma_{min} \cup \Sigma_{fixed}}$ is minimal in $\{M' \in \mathcal{M}|_{\Sigma_{min} \cup \Sigma_{fixed}} \mid M'|_{\Sigma_{fixed}} = M|_{\Sigma_{fixed}}\} \ \}$
`'translation' , Onto , Translation`	$(J, \Phi(\Sigma), \{M \mid \beta(M) \in \mathcal{M}\})$, where $(I, \Sigma, \mathcal{M}) = sem(\Gamma, L, \texttt{Onto})$ and $sem(L, \Sigma, \texttt{Translation}) = (\Phi, \alpha, \beta) : I \to J$
`'reduction' , Onto , Reduction`	$(J, \Sigma', \{\beta(M)|_{\Sigma'} \mid M \in \mathcal{M}\})$, where $(I, \Sigma, \mathcal{M}) = sem(\Gamma, L, \texttt{Onto})$ and $sem(L, \Sigma, \texttt{Reduction}) = ((\Phi, \alpha, \beta) : I \to J, \Sigma')$
`'approximation', Onto , Approximation`	TODO
`'union' , Onto , [ ConsStrength ] , Onto`	$(\mathcal{I}, \Sigma, \mathcal{M})$ where $\Sigma_i = sig(\Gamma, L, O_i)$, $\mathcal{I}_i = logic(\Gamma, L, O_i)$ $(i = 1, 2)$ $(\Phi_i, \alpha_i, \beta_i)$: $\mathcal{I}_i \longrightarrow \mathcal{I}$ 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' , Onto , ExtensionOnto`	$sem(\Gamma, L, (\Sigma, \mathcal{M}), \texttt{ExtensionOnto})$
`'module-extract' , OntoRef , Conservative, ExtractionMethod` Σ	TODO
`'qual-onto' , { Qualification } , Onto`	$sem(\Gamma, sem(L, \texttt{\{ Qualification \}}), \texttt{Onto})$

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

$sem(L, \Sigma, \texttt{'hidden'}\ \ LR_1 \ldots LR_n\ (\texttt{'symbol-items'}\ \ EI_1 \ldots EI_n)) = (\mu, \Sigma')$
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 $\Phi(\Sigma)$.

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

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

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

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

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

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 $\mathcal{M}'$.

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

An OSM definition extends the global environment:

$sem(\Gamma, L, \texttt{'onto-defn'}\ \texttt{,}\ \texttt{OntoName}\ \texttt{,}\ \texttt{[ConsStrength] , Onto})$ [91]
$= (\Gamma[\texttt{OntoName} \mapsto sem(\Gamma, L, \texttt{Onto})], L)$ Note(91)

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

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

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

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

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

$sem(L, \texttt{'qual-trans'}\ \texttt{,}\ \texttt{OntoLangTransRef}\ LR_1\ LR_2) = \rho$ where ρ is the institute comorphism from the heterogeneous logical environment named by `OntoLangTransRef`. This is defined only if $\rho : sem(LR_1) \to sem(LR_2)$ and $L = sem(LR_1)$.

[92] Note(92)

$sem(L, \texttt{'anonymous-trans'}\ LR_1\ LR_2) = \rho$ where ρ is the unique institute comorphism from the heterogeneous logical environment running from $sem(LR_1)$ to $sem(LR_2)$. This is defined only if $L = sem(LR_1)$.

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

[91] NOTE: Should we allow for overriding existing OSM definitions? Or should `OntoName` be new?

[92] 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).

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

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

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

[93] $sem(\Gamma, L, \texttt{'intpr-defn' , IntprName , ('intpr-type' } O_1\ O_2\ \texttt{)}) =$ Note(93)
$(\Gamma[\texttt{IntprName} \mapsto (\Sigma_1, \Sigma_2)], L)$ 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:

[94] $sem(\Gamma, L, \texttt{'align-defn' , AlignName , [AlignCard] , AlignType , \{ Correspondence \}}) =$ Note(94)

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

9.2. Translational semantics of DOL language constructs

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

Note that since the result of translating a DOL OSM 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]

[93] NOTE: Q-AUT: Optional `[ Conservative ]` argument is missing.
[94] 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 OSMs, depending on a logic translation $\rho : L \to \mathsf{CL}$, to Common Logic below. (The translations of the other syntactic categories are straightforward.)

$CL_\rho(\langle \Sigma, \Delta \rangle) = \langle \Phi(\Sigma), \alpha(\Delta) \rangle$, where $\rho = (\Phi, \alpha, \beta)$

$CL_\rho(O \texttt{ with logic } \rho') = CL_{\rho \circ \rho'}(O)$

[95] $CL_\rho(O \texttt{ then } CS\ \langle \Sigma, \Delta \rangle) = CL_\rho(O) \texttt{ then } CS\ \ CL_\rho(\langle \Sigma, \Delta \rangle)$ Note(95)

$CL_\rho(\texttt{OntoRef}) = \texttt{OntoRef}$

$CL_\rho(\texttt{logic LogicRef } O) = CL_{default(\texttt{LogicRef},\mathsf{CL})}(O)$

9.3. 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 OSM 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.

[96] Note(96)

9.4. OSM language translations

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

Provide some examples

special cases to be described

[95] NOTE: Extend this to all DOL constructs

[96] 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?

10. 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*[97] Note(97)

Filename extension .dol[98] Note(98)

A.2. Concrete Syntax

A.2.1. Distributed OSMs

```
DistOnto                 = [ PrefixMap ] , DistOntoDefn
                         | OntoInConformingLanguage ;
DistOntoDefn             = 'distributed␣OSM' , DistOntoName , { DistOntoItem } ;
OntoInConformingLanguage = ? language-specific ? ;
DistOntoItem             = OntoDefn | LinkDefn | Qualification ;
Qualification            = LanguageQual | LogicQual | SyntaxQual ;
LanguageQual             = 'language' , LanguageRef ;
LogicQual                = 'logic' , LogicRef ;
SyntaxQual               = 'serialization' , SyntaxRef ;
DistOntoName             = IRI ;

PrefixMap                = '%prefix(' , { PrefixBinding } , ')%' ;
PrefixBinding            = BoundPrefix , IRIBoundToPrefix ;
BoundPrefix              = ':' | Prefix (* see definition in clause 8.5.2 *)[99] Note(99)
 ;
IRIBoundToPrefix         = '<' , FullIRI , '>' ;
```

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 W3C:NOTE-owl2-manchester-syntax-20091027 but differs from the RDFa Core 1.1 syntax.

A.2.2. Heterogeneous OSMs

[100] Note(100)

[97] NOTE: FYI: just a placeholder so far, needs discussion

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

[99] 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?

[100] NOTE: TODO: merge ALIGN-TYPE with INTPR-TYPE

```
BasicOnto            = OntoInConformingLanguage ;
MinimizableOnto      = BasicOnto
                     | OntoRef , [ ImportName ] ;
ExtendingOnto        = MinimizableOnto
                     | MinimizeKeyword , '{' , MinimizableOnto , '}' ;
MinimizeKeyword      = 'minimize' | 'closed-world' ;
Onto                 = ExtendingOnto
                     | Onto , MinimizeKeyword , CircMin , [ CircVars ]
                     | Onto , Translation
                     | Onto , Reduction
                     | Onto , Extraction
                     | Onto , Approximation
                     | Onto , 'and' , [ ConsStrength ] , Onto
                     | Onto , 'then' , ExtensionOnto
                     | { Qualification } , ':' , GroupOnto
                     | Onto, 'bridge', { Translation } , Onto
                     | 'combine' , CombinedElements , [ ExcludeExtensions
]  ;
                       [101]                                          Note(101)

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

GroupOnto            = '{' , Onto , '}'
                     | OntoRef ;

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

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

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

Extraction           = 'extract' , Conservative , InterfaceSignature, 'with', ExtractionMe

Approximation        = 'approximate' , ApproxMethod ;

ExtensionOnto        = [ ConsStrength ] , [ ExtensionName ] , ExtendingOnto ;

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

[101] NOTE: combine O1 O2 takes all views coming into O1 and O2 into consideration

```
Conservative          = '%ccons'
                      | '%mcons'[102]  ;                                  Note(102)

InterfaceSignature    = SymbolItems ;

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

OntoOrLinkRef         = IRI ;

CombinedElements      = CombinedElement { ',' , CombinedElement } ;
CombinedElement       = [ Id , ':' ] , OntoOrLinkRef ;
ExcludeExtensions     = 'excluding' , ExtensionRef , { ',' , ExtensionRef
} ;

OntoDefn              = 'ontology' , OntoName , '=' , [ ConsStrength ] , Note(103)[ 'end' ][103]
 ;

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

OntoName              = IRI ;
IntprName             = IRI ;

OntoRef               = IRI ;
IntprRef              = IRI ;
ExtensionRef          = IRI ;

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

LoLaRef               = LanguageRef
                      | LogicRef ;

OntoLangTrans         = OntoLangTransRef
                      | OntoLangTransRef , ':' , LoLaRef , '→' , LoLaRef
                      | LoLaRef , '→' , LoLaRef
                      | '→' , LoLaRef ;

OntoLangTransRef      = IRI ;
```

[102] 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.

[103] 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.

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

ApproxMethodRef       = IRI ;

ExtractionMethod      = IRI ;
```

A.2.3. Links

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

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

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

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

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

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             = GroupOnto , 'to' , GroupOnto[104]  ;                Note(104)

Correspondence        = CorrespondenceBlock
                      | SingleCorrespondence
                      | '*' ;
CorrespondenceBlock   = 'relation' , [ RelationRef ] , [ Confidence ] ,
```

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

```
                        '{' , Correspondence , { ',' , Correspondence } , '}' ;
SingleCorrespondence = SymbolRef , [ RelationRef ] ,
                        [ Confidence ] , TermOrSymbolRef , [ CorrespondenceId ] ;
CorrespondenceId     = '%(' , IRI , ')%' ;
SymbolRef            = IRI ;
TermOrSymbolRef      = Term | SymbolRef (* Term is logic-specific *) ;[105] Note(105)
RelationRef          = '>' | '<' | '=' | '%' | [106] '$\ni$' | '$\in$' | '$\mapsto$' | IRI ;
Confidence           = Double ? where Double ∈ [0,1] ? ;                    Note(106)
```

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, OSMs can always be input in the Basic Latin subset, also known as ASCII.[107] Note(107)
For enhanced readability of OSMs, the DOL text serialization particularly allows for using the native Unicode glyphs that represent common mathematical operators.

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[108], although they can be used as parts of tokens: Note(108)

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

[105] 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.)

[106] 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.

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

[108] 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	->

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

[109] Note(109)

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 OSMs to IRIs (clause 8.5.3), 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 OSM in an RDF serialization that supports prefix maps **may** state them redundantly as syntactic RDF prefixes as well as using the DOL RDF vocabulary.

[109] 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 OSM languages conforming with DOL

This annex specifies an RDF vocabulary, formalized in RDFS W3C/TR REC-rdf-schema:2004, for describing OSM 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 OSMs (cf. annex B).[110] Note(110)

The tables in this annex list the classes and properties of the RDF vocabulary for describing OSM 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 OSM languages. Each row of the table translates into the following RDF triples (given in Turtle serialization):[111] Note(111)

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

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

Class	*documentation*
OSMLanguage	*an OSM language*
Logic	*a logic that defines the semantics of an OSM language*
Serialization	*a serialization of an OSM language*

Table C.2 lists the properties of the RDF vocabulary for describing OSM 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" .
```

[112] Note(112)

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

[111] NOTE: TODO: also cover rdfs:subClassOf (once we have such cases)

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

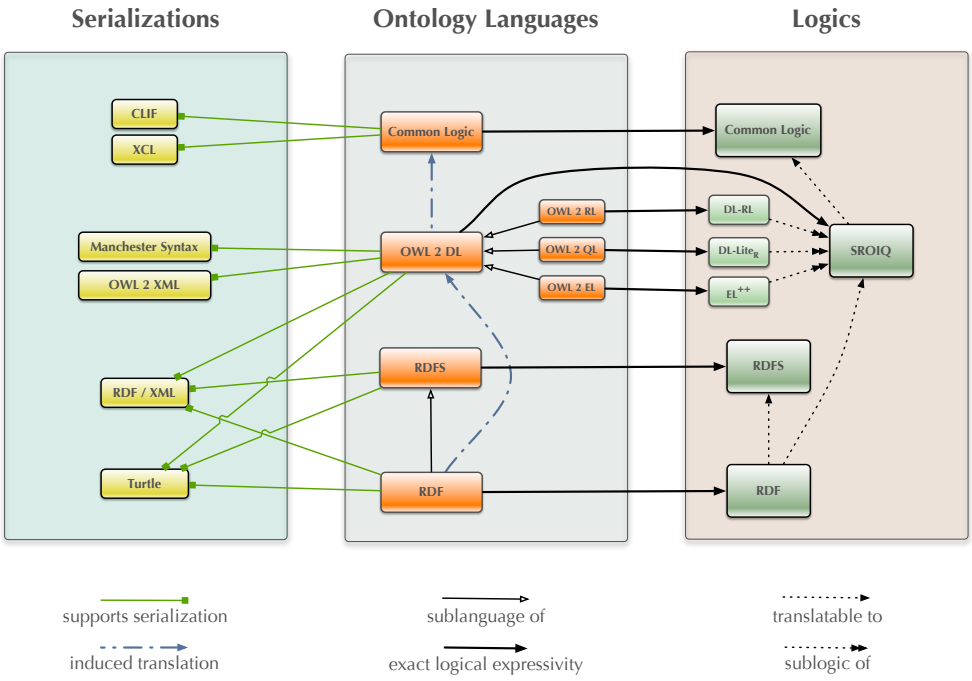

Figure C.1.: Subset of the OntoIOp registry, shown as an RDF graph

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

Property	domain	range
documentation		
subLogicOf	Logic	Logic
The subject is a sublogic of the object		
supportsLogic	OSMLanguage	Logic
The subject OSM language has a semantics specified in terms of the object logic.		
specifiesSemanticsOf	Logic	OSMLanguage
The subject logic is used to specify the semantics of the object OSM language; inverse of supportsLogic.		
supportsSerialization	OSMLanguage	Serialization
OSMs in the subject OSM 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	OSMLanguage
The subject logic is used to specify the semantics of the object OSM 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 OntoGraph.

The OWL XML serialization satisfies the criteria for XML conformance. The mapping of OWL 2 to RDF graphs satisfies the criteria for RDF conformance[113] . The OWL 2 Manchester syntax satisfies the criteria for text conformance. [114] Note(113) Note(114)

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*[115] Note(115)

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

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 $\mathcal{SROIQ}$ SROIQ, 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 SROIQ.*

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) .*

[113] NOTE: This is not exactly true, as some things, e.g. imports, can't be identified.
[114] NOTE: also need conformance propositional logic; use PL "profile" of the CASL "IFIP standard"
[115] NOTE: Q-AUT: This grammar should also be adapted to ISO EBNF.
[1] See also `http://www.w3.org/TR/owl2-overview/`

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

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:*

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

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 OntoGraph.

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 $UD^* = \{< x_1, \ldots, x_n > | x_1, \ldots, x_n \in UD\}$ (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 $t_1 \ldots t_n$, 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. [116] Note(116)

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

[116] 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 OntoGraph.

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

Definition 10 (RDF **and** RDFS) *Following Lucanu, 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.*[117] *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(117)

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*.*

[117] 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 OSM 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 OSM 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 OntoGraph. [118] Note(118)
[119] Note(119)

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

EL $\rightarrow$ 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 $\mathcal{SROIQ}(D)$, $\mathcal{EL}++ \rightarrow \mathcal{SROIQ}(D)$ is the corresponding sublogic inclusion.

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

QL $\rightarrow$ 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$_R$ $\rightarrow \mathcal{SROIQ}(D)$ is the corresponding sublogic inclusion.

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

RL $\rightarrow$ 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 $\rightarrow \mathcal{SROIQ}(D)$ is the corresponding sublogic inclusion.

---

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

[119] 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.

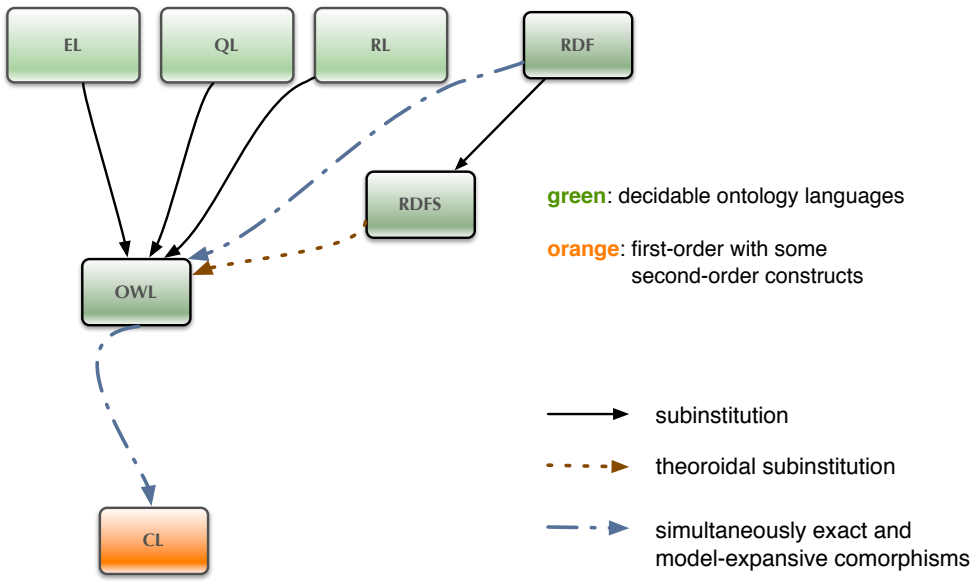

Figure G.1.: Translations between conforming OSM languages (normative)

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 → $\mathcal{SROIQ}(D)$

[120] Note(120)

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

[120] 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 $\rightarrow \mathcal{SROIQ}(D)$ 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 $\rightarrow FOL$

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:

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

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

Proof. By Induction over the structure of C.

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

□

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 OSM 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 OSM languages and profiles (in addition to those mentioned in annex G):

- PL (propositional logic)
- SimpleRDF (RDF triples without a reserved vocabulary)
- OBO^{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)
- $\text{FOL}^{=}$ (untyped first-order logic, as used for the TPTP format)
- F-logic
- CASL (Common Algebraic Specification Language)

The actual translations are specified in OntoGraph.
[121] Note(121)

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

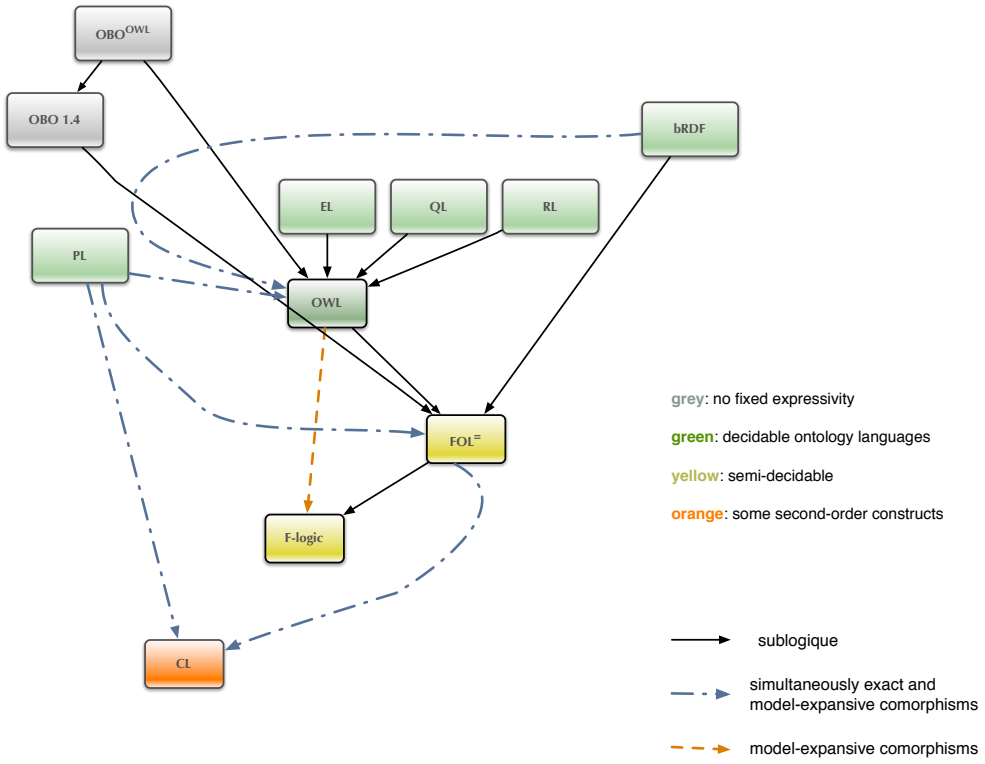

Figure H.1.: Translations between conforming OSM languages (extended)

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** Institutions 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 inst-mor.

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

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

[122] Note 122)

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

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

J.1. Mereology: Distributed and Heterogeneous Ontologies

[123] Note(123)

[123] 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 OSM 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

[124] Note(124)

```
distributed OSM 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 =
```

[124] 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 OSM 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 OSM 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 ⊑→ 2:Publication
  1:PhdThesis ⊑→ 2:Thesis
  1:InBook ⊑→ 2:BookArticle
  1:Article ⊑→ 2:Article
  1:Article ⊒→ 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 OSM 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

```
%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 OSM 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 do-roam. The service is driven by a set of different OWL ontologies that have been aligned to each other using the Falcon matching tool HuQu-08. 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 allow for coherently representing the aligned ontologies as one distributed OSM, and it would allow for maintaining 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 links 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

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

DOL would be capable of expressing all that within one distributed OSM of heterogeneous ontologies arranged around an OWL core (here: the OpenAAL ontology), including logical links 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 links

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") dolce. 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

[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/

ontology development. DOLCE has been used also for semantic middleware, and in OWL-formalised 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 allow for formalizing all of these views and, given suitable consistency checking tools, allow for analyzing 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 links

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 [OWLTime,OWLSTime]. DOL would instead allow for modeling OWL Time as one coherent heterogeneous ontology, using OWL and, e.g., Common Logic.[125] For the temporal description logic $\mathcal{DLR}_{\mathcal{US}}$ for knowledge bases and logic-based temporal conceptual data modelling [Artale02,Artale07a]; $\mathcal{DLR}_{\mathcal{US}}$ 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 [Artale07time], 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

Note(125)

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

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

constraints' in the ontology (evolution constraints constrain membership of an object or an individual relation to a concept 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 [AGK08].

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 allows for implementing the remainder of the COLORE metadata vocabulary as an ontology, reusing suitable existing metadata vocabularies such as OMV (cf. annex L), and it allows for implementing one or multiple Common Logic ontologies plus their annotations as one coherent distributed OSM.

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

[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 OSM* in this international standard

[8] Note that this may cover any *sentencs* in the sense of this international standard

- CASL can specify syntax (but not quite in the same way as XML Schema) *and* semantics of datatypes

126 127

Note(126)

Note(127)

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

[127] 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

[128] Note(128)

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

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

[129] Note(129)

[128] NOTE: Q-ALL: Or should this rather be normative?
[129] NOTE: TODO: maybe mention: How do we use the ISO 12620 DCR for our extension of the OMV?

M. Annex (informative): Bibliography

The bibliography will be added later