ITEA Office
High Tech Campus 69 - 3
5656 AG Eindhoven
The Netherlands
T + 31 88 003 6136
E info@itea3.org
W www.itea3.org

*ITEA 3 is a EUREKA strategic ICT cluster programme*

# D2.4.1 Specification of the Knowledge Representation Language output by the parser and input to the Generator

ModelWriter

Text & Model-Synchronized Document Engineering Platform

Work Package: WP2

Task: T2.4 – Definition of the target semantic representation language

Edited by: Claire Gardent and Mariem Mahfoudh

Mariem Mahfoudh <mariem.mahfoudh@loria.fr> (CNRS/LORIA)

Claire Gardent <claire.gardent@loria.fr> (CNRS/LORIA)

Date: 05-Feb-2016

Version: 1.0.0

## Document History

| Version | Author(s) | Date | Remarks |
|---|---|---|---|
| 1.0.0 | Mariem Mahfoudh | 05-Feb-2016 | Initial Release |
| 1.0.0 | Claire Gardent | 05-Feb-2016 | Initial Release |

# D.2.4.1 Specification of the Knowledge Representation Language output by the Parser and input to the Generator

Mariem Mahfoudh and Claire Gardent
CNRS/LORIA, Nancy (France)

## 1 Introduction

Representing knowledge is "the area of Artificial Intelligence (AI) concerned with how knowledge can be represented symbolically and manipulated in an automated way by reasoning programs" [BLR92]. It consists in formally representing knowledge in order to simplify their extraction and to automate the reasoning on the facts which they represent.

Several languages and formalisms have been proposed in the literature to represent knowledge: semantic networks [Sow14, PS03], petri nets [CKC90], description logics [Bor96, CDGLN01, BHS05], Frames [Min75, KLW95], Knowledge Interchange Format (KIF) [GF$^{+}$92], conceptual graphs [Sow92], ontologies [Gua95], etc.

In the ModelWriter project, we are interested in synchronising text and formal models. In particular, Work Package 2 (WP2), aims to automatically map text to models using semantic parsing and conversely, to map models to text using natural language generation. For these two processes (semantic parsing and natural language generation) to be compatible[1], it is necessary to choose a language that formalize knowledge and which can function as both the source of natural language generation and the target of semantic parsing. In addition, in the framework of the ModelWriter project, this language must be a plausible pivot language for a wide variety of metamodels. In this deliverable, we therefore focus on languages which have a well defined syntax and semantics and which are widely used for knowledge representation, namely, languages such as Description Logic and the Resource Description Framework (RDF) which are commonly used by the semantic web and by the knowldege representation communities. As these have been less researched in the NLP community (where for a long time, first

[1]By "compatible" here, we mean that it should be possible to use semantic parsing to map the text $T$ to the meaning representation $\phi$ and conversely, to use natural language generation to map the same meaning representation $\phi$ to the same text $T$ or to one of its paraphrases.

order or higher order logic was the preferred language for meaning representation), developing means of going back and forth between text and these knowledge representation languages will not only be useful for ModelWriter applications but also provide an interesting NLP challenge and hopefully result in interesting insights on how to automate the relation between text and the Semantic Web languages.

The rest of the deliverable is organized as follow. Section 2 introduces Description Logics (DL). Section 3 represents the Resource Description Framework language (RDF). Section 4 represents the Resource Description Framework Schema language (RDFS). Section 5 represents the Web Ontology Language (OWL). Section 6 concludes.

## 2 Description Logics (DL)

Description logics are a family of formalisms for knowledge representation. They are based on three entities: 1) the concepts that correspond to classes of individuals, 2) roles that are relations (properties) between these individuals, and 3) individuals that are concrete examples of classes.
Two components are distinguished in description logics:

- the *Terminological box (Tbox)* which defines the ontology structure (the concepts and the roles) and the axioms.
- the *Assertional box (ABox)* which specifies the individuals and their assertions.

The description logics have a common base, called *Attributive Language (AL)*, whose the constructors are presented in the Table 1. These constructors represent knowledge with a simple expressivity.

| **Constructors** | **Syntax** | **Semantic** |
|---|---|---|
| universal concept | $\top$ | $\triangle^I$ |
| bottom concept | $\bot$ | $\emptyset$ |
| atomic concept | $A$ | $A \subseteq \triangle^I$ |
| atomic negation | $\neg A$ | $\triangle^I \setminus A^I$ |
| intersection | $C \sqcap D$ | $C^I \cap D^I$ |
| value restriction | $\forall R.C$ | $\{x \in \triangle \vert \forall y.[(x,y) \in R \wedge y \in C]\}$ |
| limited existential quantification | $\exists R.\top$ | $\{x \in \triangle \vert \exists y.[(x,y) \in R \wedge y \in C]\}$ |

Table 1: Attributive Language constructors.

Figure 1 shows an example of a light ontology represented by attributive language. It is an extract from component ontology of Airbus company which describes the system installation component of planes Airbus (more details of this ontology can be founded in [GMCLM15]). The extracted ontology is composed by : 1) four classes (”Component”, ”FlexibleComponent”, ”RigidComponent” and ”ATAChapter”), 2) one role ”relatedATA”(”Component”, ”ATAChapter”), 3) two individuals (”ATA-22-41” and ”ATA-45”) and 4) two subsumption axioms (”FlexibleComponent” is-a ”Component” and ”RigidComponent” is-a ”Component”).

**TBox**

$$FlexibleComponent \sqsubseteq Component$$
$$RigidComponent \sqsubseteq Component$$
$$\exists\, relatedChapter \sqsubseteq Component$$
$$\top \sqsubseteq \forall\, relatedChapter.ATAChapter$$

**ABox**

*ATAChapter (ATA-44)*
*ATAChapter (ATA-22-31)*

Figure 1: Example of an ontology represented with Attributive language.

In order to increase its expressiveness and define more properties and axioms, attributive language has been enriched by several extensions which form the description logics family (see Figure 2).

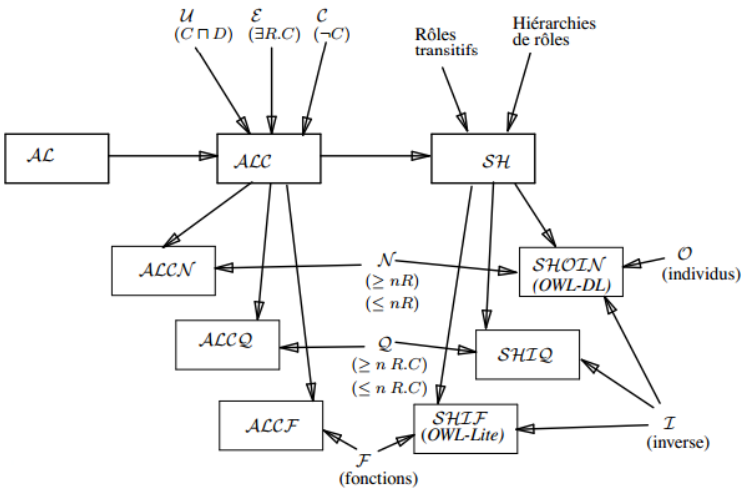
$\mathcal{U}$ $(C \sqcap D)$
$\mathcal{E}$ $(\exists R.C)$
$\mathcal{C}$ $(\neg C)$
Rôles transitifs
Hiérarchies de rôles
$\mathcal{AL}$
$\mathcal{ALC}$
$\mathcal{SH}$
$\mathcal{ALCN}$
$\mathcal{N}$ $(\geq nR)$ $(\leq nR)$
$\mathcal{SHOIN}$ (OWL-DL)
$\mathcal{O}$ (individus)
$\mathcal{ALCQ}$
$\mathcal{Q}$ $(\geq n\, R.C)$ $(\leq n\, R.C)$
$\mathcal{SHIQ}$
$\mathcal{ALCF}$
$\mathcal{SHIF}$ (OWL-Lite)
$\mathcal{I}$ (inverse)
$\mathcal{F}$ (fonctions)

Figure 2: Description logics family [Gag07].

We present here some languages of this family:

- $ALC = (AL+\neg C)$ constitutes the base of all the expressive description languages. It adds the definition of the following constructors:

1. the union between concepts ($C_1 \sqcup C_2$) ;
2. the complement ($\neg C$) which defines the negation of a class. Note that attributive language expresses only the negation of the atomic concept (e.g $\neg humain$). Although, ALC could express the negation of not atomic concept (e.g. $\neg(animal \cup raisonnable)$) ;
3. exists restriction ($\exists R.C$) which specifies that an entity should have at least one relation with an object while specifying the class of this object (e.g. the specification $\exists relatedChapter.ATAChapter$ defines the class of all the components which are related at least to a "ATAChapter") ;

- $SH = (ALC + H + Tr(R))$ aims to define more axioms on the roles (relations). It specifies for examples the transitivity of a role ($Tr(R)$) and also the hierarchical relation between two roles ($H$).
- $SHIF = (SH+I+F)$ includes the constructors of SH and adds others constructors such as the inverse of a role ($I$) and functions ($F$) which specify that a role is a function (i.e. an entity can be related at most with one entity by this role).
- $SHOIN = (SH + O + I + N)$ adds the specification of the cardinality restriction $N$ ($\leq n.R$ and $\geq n.R$) and the enumeration ($O$) which defines a class from a set of individuals (e.g. *Season* $\equiv$ {*Autumn, Winter, Springer, Summer*}.

## 3 Resource Description Framework (RDF)

RDF (Resource Description Framework) is a semantic web language which represents web resources and the relations between them [Mil98]. It has a simplified semantic which represents knowledge with triples < *subject, predicate, object* >, where:

- the *subject* denotes a resource ;
- the *predicate* indicates the relationship between the subject and object. It called also property ;
- the *object* denotes a resource.

The subject, the predicate and often the object are identified by URI (Uniform Resource Identifier) or also by IRI (Internationalized Resource Identifier) [Dür01] to ensure the accessibility and the share of the described resources.
An RDF triple is represented by an oriented and labelled graph. The subject and the object are represented by nodes. The predicate is represented

by an edge which links the subject to the object. Figure 3 presents an example of an RDF graph which describes the ModelWriter website. The resource "https://itea3.org/project/modelwriter.html" represents the subject. "is-created-by" corresponds to the predicate and "itea3" corresponds to the object.

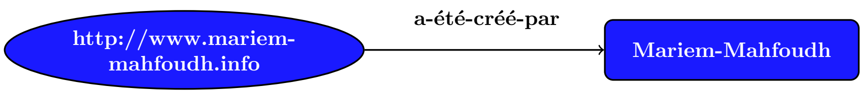

Figure 3: Exemple d'un graphe RDF.

## 4 Resource Description Framework Schema (RDFS)

RDFS [McB04] is a formal language which was principally proposed to extend the semantic of RDF language. It adds the specification of classes (`rdfs:class`) and the hierarchical relations between them (`rdfs:subClassOf`). It also enables defining properties (`rdfs:property`), their members (`rdfs:domain` and `rdfs:range`) and hierarchical relations between properties (`rdfs:subPropertyOf`).

These features allow RDFS to represent taxonomies or ontologies light but not heavy-weight ontologies. Note that a *taxonomy* is a "a subject-based classification that arranges the terms in the controlled vocabulary into a hierarchy" [Gar04]. *Light-weight ontologies* represents concepts, taxonomy of concepts and properties that describe these concepts [GZ09]. *Heavy-weight ontologies* are ontologies that integrate axioms and can express all the semantic of the modelled domain [FT06] ;

Although its rich semantic, RDFS suffers from some limits. For example, it can not express the features of properties (transitivity, symmetric, etc.) or also the cardinality restriction. Thus, to overcome these limitations, the W3C proposed the OWL language.

## 5 Web Ontology Language (OWL)

OWL is the standard currently proposed by W3C to represent ontologies [MVH$^+$04]. It is a semantic web language, based on RDFS and DL. OWL is an expressive language which defines many constructors and axioms (see tables 2 and 3). It defines the disjunction between classes (`owl:disjointWith`), the equivalent (`owl:equivalentOf`), restrictions (`owl:Restriction`), cardinalities (`owl:cardinality, owl:minCardinality, owl:maxCardinality`), enumerated classes (`owl:oneOf`), etc.
OWL language has two versions OWL1 et OWL2. OWL1 has been recom-

mended since 2004 and offers three sub-languages with increasing expressivity:

- *OWL Lite* defines reduced functionalities (e.g. it cannot specify the negation, and the defined cardinalities are limited to 0 or 1). It corresponds to SHIF (D) language from description logics, where $D$ corresponds to Dataproperty (a constructor that relies an individual to literal).
- *OWL DL* is an expressive language whose inference procedures are complete, i.e, all inferences are calculated. It corresponds to SHOIN(D) language and often used to represents heavy-weight ontologies ;
- *OWL Full* is the most complex variant of OWL which offers maximum expressiveness, but don't propose any guarantee of completeness and on the termination of inference procedures.

| **OWL constructors** | **DL syntax** |
|---|---|
| intersectionOf ($C_1$, $C_2, ...$) | $C_1 \sqcap C_2$ |
| unionOf ($C_1$, $C_2, ...$) | $C_1 \sqcup C_2$ |
| complementOf(C) | $\neg C$ |
| oneOf($I_1, I_2, ...$) | $\{I_1, I_2, ...\}$ |
| Restriction(P allValuesFrom(C)) | $\forall P.C$ |
| Restriction(P someValues-From(C)) | $\exists P.C$ |
| Restriction(P hasValue(I)) | $P : I$ |
| Restriction(P cardinality(n)) | $= nP$ |
| Restriction(P minCardinality(n)) | $\leqslant nP$ |
| Restriction(P maxcardinality(n)) | $\geqslant nP$ |

Table 2: OWL constructors.

| **OWL constructors** | **DL syntax** |
|---|---|
| Axioms defined on the classes | |
| subClassOf ($C_1$, $C_2$) | $C_1 \sqsubseteq C_2$ |
| equivalentClasses ($C_1$, ..., $C_n$) | $C_1 \equiv ... \equiv C_n$ |
| disjointWith ($C_1$, ..., $C_2$) | $C_1 \sqsubseteq \neg C_2$ |
| Axioms defined on the properties | |
| subPropertyOf ($P_1, P_2$) | $P_1 \sqsubseteq P_2$ |
| equivalentProperties ($P_1, ..P_n$ | $P_1 \equiv ... \equiv P_n$) |
| inverseOf ($P_1, P_2$) | $P_1 \equiv {P_2}^-$ |
| symetricProperty($P$) | $P \equiv P^-$ |
| functionalProperty ($P$) | $\top \sqsubseteq \leqslant 1P$ |
| inverseFunctionalProperty ($P$) | $\top \sqsubseteq \leqslant 1P^-$ |

| | |
|---|---|
| transitiveProperty($P$) | $Tr(P)$ |
| Axioms defined on the individuals | |
| sameAs($I_1, ..., I_n$) | $I_1 = ... = I_n$ |
| differentFrom($I_1, ..., I_n$) | $I_1 \neq ... \neq I_n$ |

Table 3: OWL axioms.

## 6 Conclusion

In this deliverable, we presented the main languages (DL, RDF, RDFS and OWL) used in the knowledge representation and the Semantic Web community. These will be the basis for the NLP components to be developed in WP2. Specifically, we will use OWL as the meaning representation language in which the ModelWriter parser will convert text and from which, its natural language generator will generate text. The motivations for this choice are the following.

OWL has a well-defined syntax and semantics – this will make it possible to specify systematic translations from the metamodels that might be used by the various ModelWriter usecases into this single NLP target representation language.

The UC-FR4 (Synchronisation of regulation documentation with a design rule repository) usecase involves a model (ontology) written in OWL and some associated text (SIDP rules). We can therefore directly make use of this usecase to develop, train and evaluate an OWL-based semantic parser and generator and to test the usefulness of these NLP tools for synchronising text and models.

OWL comes equipped with highly optimised automated reasoners such as HermiT which supports not only the standard reasoning task of entailment checking but also several specialised reasoning services such as class and property classification, as well as a range of features outside the OWL 2 standard such as DL-safe rules, SPARQL queries, and description graphs [GHM$^+$14]. Furthermore, [MLH$^+$] identify 35 OWL reasoners that are, at least to some degree, actively maintained. More generally, using OWL gives access to powerful reasoning and querying techniques which may well be relevant for the synchronisation issues addressed by the ModelWriter project.

## References

[BHS05] Franz Baader, Ian Horrocks, and Ulrike Sattler. *Mechanizing Mathematical Reasoning*, volume 2605, chapter Description logics as ontology languages for the semantic web, pages 228–248. Springer, 2005.

[BLR92] Ronald J Brachman, Hector J Levesque, and Raymond Reiter. *Knowledge representation*. MIT press, 1992.

[Bor96] Alex Borgida. On the relative expressiveness of description logics and predicate logics. *Artificial intelligence*, 82(1):353–367, 1996.

[CDGLN01] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Daniele Nardi. Reasoning in expressive description logics. *Handbook of Automated Reasoning*, 2:1581–1634, 2001.

[CKC90] Shyi-Ming Chen, Jyh-Sheng Ke, and Jin-Fu Chang. Knowledge representation using fuzzy petri nets. *Knowledge and Data Engineering, IEEE Transactions on*, 2(3):311–319, 1990.

[Dür01] Martin J Dürst. Internationalized resource identifiers: From specification to testing. In *19th International Unicode Conference, IUC 2001*, San Jose, CA, 2001.

[FT06] Frédéric Fürst and Francky Trichet. Raisonner sur des ontologies lourdes à l'aide de graphes conceptuels. In *24 ème Congrès INFORSID (INFormatique des ORganisations et Systèmes d'Information et de Décision*, pages 879–894, Hammamet, Tunisie, 2006.

[Gag07] Michel Gagnon. Logique descriptive et owl. *Cours dispensé à l'école polytechnique de Montréal, Canada*, 2007.

[Gar04] Lars Marius Garshol. Metadata? thesauri? taxonomies? topic maps! making sense of it all. *Journal of information science*, 30(4):378–391, 2004.

[GF$^+$92] Michael R Genesereth, Richard E Fikes, et al. Knowledge interchange format-version 3.0: Reference manual. 1992.

[GHM$^+$14] Birte Glimm, Ian Horrocks, Boris Motik, Giorgos Stoilos, and Zhe Wang. Hermit: An owl 2 reasoner. *Journal of Automated Reasoning*, 53(3):245–269, 2014.

[GMCLM15] Claire Gardent, Mariem Mahafoudh, Samuel Cruz-Lara, and Anne Morceaux. D2.1.2 documentation of the corpora, modelwriter, text and model-synchronized document engineering platform. Technical report, CNRS-LORIA, 2015.

[Gua95] Nicola Guarino. Formal ontology, conceptual analysis and knowledge representation. *International journal of human-computer studies*, 43(5):625–640, 1995.

[GZ09] Fausto Giunchiglia and Ilya Zaihrayeu. Lightweight ontologies. In *Encyclopedia of Database Systems*, pages 1613–1619. Springer, 2009.

[KLW95] Michael Kifer, Georg Lausen, and James Wu. Logical foundations of object-oriented and frame-based languages. *Journal of the ACM (JACM)*, 42(4):741–843, 1995.

[McB04] Brian McBride. The resource description framework (rdf) and its vocabulary description language rdfs. In *Handbook on ontologies*, pages 51–65. Springer, 2004.

[Mil98] Eric Miller. An introduction to the resource description framework. *Bulletin of the American Society for Information Science and Technology*, 25(1):15–19, 1998.

[Min75] Marvin Minksy. A framework for representing knowledge. *The psychology of computer vision*, pages 211–277, 1975.

[MLH$^+$] Nicolas Matentzoglu, Jared Leo, Valentino Hudhra, Uli Sattler, and Bijan Parsia. A survey of current, stand-alone owl reasoners. In *Informal Proceedings of the 4th International Workshop on OWL Reasoner Evaluation*, volume 1387.

[MVH$^+$04] Deborah L McGuinness, Frank Van Harmelen, et al. Owl web ontology language overview. *W3C recommendation*, 10(10):2004, 2004.

[PS03] Stephen Peters and Howard E Shrobe. Using semantic networks for knowledge representation in an intelligent environment. page 323. IEEE, 2003.

[Sow92] John F Sowa. Conceptual graph summary. *Conceptual Structures: Current Research and Practice. Ellis Horwood, New York London Toronto*, pages 3–66, 1992.

[Sow14] John F Sowa. *Principles of Semantic Networks: Explorations in the representation of knowledge.* Morgan Kaufmann, 2014.