COMMON LANUGAGES FOR SEMANTIC WWW
Beyond RDF and OWL

Seiji Koide
Information Technology, The Graduate University for Advanced Studies Sokendai, Tokyo, Japan

Hideaki Takeda
National Institute of Informatics and SOKENDAI, 2-1-2, Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan

Keywords: Semantic web, RDF, OWL, Object, KIF, Common logic, Reflection, SWCLOS.

Abstract: OWL has established itself as a standard of ontology description language not only in Semantic Web but also in diverse disciplines and engineering fields. However, endeavors to describe ontology in OWL are revealing the extent of ability on the OWL current specification in practical views. In this paper, we see an overview of basic assumptions of knowledge representation languages for Semantic Web, and point out several basic and problematic issues of OWL, which are captured by our own experience of developing a language processor called SWCLOS, the first OWL Full processor developed on top of Common Lisp Object System (CLOS), and we address our approach to solve them. It includes explicit descriptions of role concepts, auto-epistemic local closed world assumption, ternary truth values, and unique name assumption for atomic objects. These settings are implemented into SWCLOS. Finally, we envision the direction of languages for semantic WWW.

1 INTRODUCTION

It seems that Web Ontology Language (OWL) has successfully established itself as a *de facto* standard of ontology description language not only in the Semantic Web community but also in diverse disciplines and engineering fields, e.g., ontology, linguistics, modeling in software engineering, enterprise business patterns, etc. We had also developed an OWL Full language processor called SWCLOS (Koide and Takeda, 2006)[1] on top of Common Lisp Object System (CLOS), and attempted to apply it in several applications. SWCLOS is an amalgamation of Object-Oriented language in Lisp and Web Ontology Language OWL, and then we saw how an ontology description language that is firmly underpinned by formal logic and denotational semantics on Resource Description Framework (RDF) is useful to software engineering so as to assure formal descriptions of system specification.

In the process of developing SWCLOS we, however, encountered a few subtle and basic problems of semantic disparity to be coped with among computational models of RDF, OWL, logics, and objects. For example, ordinary programming languages and logics such as Description Logic (DL) stand on Unique Name Assumption (UNA), but OWL is not regarded to stand on UNA. Ordinary software models and logics are based on Closed World Assumption (CWA), but OWL and DLs are regarded as on Open World Assumption (OWA). In case that we had set such full setting as non-UNA and OWA for Semantic Webs into SWCLOS, it amounted to the result of either very few viable interpretations with less common knowledge or excessive need of common knowledge for models on class disjointness and individual differentiation in several Semantic Web applications. Hence, we refactored SWCLOS with introducing new moderate settings based on *context dependent role and disjointness of substance classes*, *auto-epistemic local closed world assumption*, *ternary truth values* that allow unknown value, and *UNA for atomic objects in non-UNA environment*. Such experience of developing SWCLOS and the subsequent applications brought us to deeper understanding on the theory and relations of RDF(S)/OWL, logics, and Object-Oriented semantics. Table 1 summarize the basic computational assumptions underlying Common Lisp, Description

[1] Source codes are available at http://www-kasm.nii.ac.jp/~koide/SWCLOS2-en.files/Page408.htm

Koide S. and Takeda H. (2010).
COMMON LANUGAGES FOR SEMANTIC WWW - Beyond RDF and OWL.
In *Proceedings of the Fifth International Conference on Evaluation of Novel Approaches to Software Engineering*, pages 86-95
DOI: 10.5220/0002999000860095

Table 1: Basic Computational Features of Languages.

	UNA/nonUNA	CWA/OWA	Truth Value	arity
Common Lisp	UNA	CWA	Ternary	n
DL	UNA	OWA	Binery	2
OWL	nonUNA	OWA	Binery	2
Common Logic	nonUNA	OWA	Ternary?	n

Logic, OWL, and Common Logic.

This paper is structured as follows. In Section 2, we see an overview of RDF, RDF Schema, OWL, Common Lisp, and Common Logic. Section 3 describes problematic non-Unique Name Assumption and OWA, and also discusses the equality of entities in the RDF universe and the OWL universe. In Section 4, we propose a new framework that involves role concepts based on discussion of several top ontologies. The auto-epistemic local closed world assumption and ternary truth value are explained at Section 5. Finally, we envision the direction of languages for semantic WWW at Section 6.

2 SEMANTICS IN RDF, OWL, LISP, AND COMMON LOGIC

2.1 Denotational Semantics in RDF

Every scientific theory is a system of sentences which are accepted as true and which may be called *asserted statements* (Tarski, 1995). To deduce the truth value of asserted sentences, words in sentences are discriminated from things denoted by words (denotations), and then the relations between words and denotations and among denotations are interpreted according to axioms and rules in a given formal way. Such systematic denotational semantics is called "Tarskian" (McDermott, 1978). A set of all entities as denotation is called a *domain* or *universe of discourse* (Tarski, 1995). Axioms and rules for interpretation must enable the structure of the universe of discourse so as to reflect the structure of the real world. Entailment in logic with axioms must follow rules in the real world, see Figure 7.6 in (Russell and Norvig, 2003).

Resource Description Framework (RDF) is an assertional language intended to be used to express propositions using precise formal vocabularies and the syntax which is applicable to WWW with the components such as URI references, literals, XML schema typed literals, or nodes in WWW (Hayes and McBride, 2004). RDF captures WWW as labeled directed graph. The semantics of RDF graph is specified and formalized as follows by set-theoretical denotational semantics based on the Tarskian model theory.

In the RDF simple interpretation $\mathcal{I}$ of vocabulary $\mathcal{V}$,

1. A non-empty set $\boldsymbol{R}^{\mathcal{I}}$ of entities, called the domain or universe of $\mathcal{I}$.
2. A set $\boldsymbol{P}^{\mathcal{I}}$, called the set of properties of $\mathcal{I}$.
3. A mapping $EXT^{\mathcal{I}}$ from $\boldsymbol{P}^{\mathcal{I}}$ into the powerset of the set $\boldsymbol{R}^{\mathcal{I}} \times \boldsymbol{R}^{\mathcal{I}}$, i.e., the set of sets of pairs $\langle x,y \rangle$ with x and y in $\boldsymbol{R}^{\mathcal{I}}$.
4. A mapping $S^{\mathcal{I}}$ from URI references in $\mathcal{V}$ into $\boldsymbol{R}^{\mathcal{I}} \cup \boldsymbol{P}^{\mathcal{I}}$.
5. A mapping $L^{\mathcal{I}}$ from typed literals in $\mathcal{V}$ into $\boldsymbol{R}^{\mathcal{I}}$.
6. A distinguished subset LV of $\boldsymbol{R}^{\mathcal{I}}$, called the set of literal values, which contains all the plain literals in $\mathcal{V}$.

Here, $EXT^{\mathcal{I}}(p)$ is called the property extension of p, and it represents a set of sets of pairs $\langle x,y \rangle$, where x and y are entities in the universe of discourse. In other words, a property makes a set of the binary relation between entities in the universe of discourse.

The notion of property is introduced with two words, rdf:Property and rdf:type, in rdf vocabulary as follows.

Axiom 1. If an entity is a member of the set of properties of $\mathcal{I}$, then the entity makes a pair with $rdf\!:\!Property^{\mathcal{I}} = S^{\mathcal{I}}(\mathrm{rdf\!:\!Property}^{2})$ and then the pair is a member of property extension of $rdf\!:\!type^{\mathcal{I}} = S^{\mathcal{I}}(\mathrm{rdf\!:\!type})$, and vice versa:

$$x \in \boldsymbol{P}^{\mathcal{I}} \quad iff \quad \langle x, rdf\!:\!Property^{\mathcal{I}} \rangle \in EXT^{\mathcal{I}}(rdf\!:\!type^{\mathcal{I}})$$

A particular pair of $\langle x,y \rangle$ for property p is also called a triple in infix notation or $x\ p\ y$. In this expression, x is called *subject*, y is called *object*, and p is called *predicate*. A set of triples is called an RDF graph in Semantic Web. An RDF graph may include blank nodes. A blank node has no URI reference and may be designated by a *nodeID* instead of a URI reference. An RDF graph that does not include blank nodes is called a ground graph. The denotation of a ground RDF graph (truth value) in $\mathcal{I}$ is given by recursively applying the above interpretation and axioms for ground triples. The semantics of ungrounded graphs is extended from the ground graphs, see (Hayes and McBride, 2004) for details. Next subsections describe the semantic extensions of interpretation for RDF Schema (RDFS) and OWL.

2.2 Semantics of Class in RDF Schema

The notion of class-instance is introduced as an rdfs-extension of the universe of discourse using the notion of class extension, see below.

[2] rdf:Property is a QName in XML Namespace for URI reference http://www.w3.org/1999/02/22-rdf-syntax-ns#Property.

Axiom 2. If an entity is a member of class extension of another entity, then a pair of both becomes a member of property extension of $rdf:type^I$, and vice versa.

$$x \in CEXT^I(y) \quad iff \quad \langle x,y\rangle \in EXT^I(rdf:type^I)$$

$CEXT^I(y)$ represents the class extension of y, and it denotes a set of instances of y. Here, y is called a class.

It is obvious that every property in the universe turns out to be an instance of $rdf:Property^I$. Here, $\boldsymbol{R}^I$, which is initially defined as the universe itself in the rdf simple interpretation, is redefined with terminology rdfs:Resource as a class extension of $rdfs:Resource^I$. In addition, using the notion of class extension, the set of all classes in the universe is also defined with new terminology rdfs:Class. Datatypes and literals are also defined as follows.

$$\begin{aligned}
\boldsymbol{P}^I &= CEXT^I(rdf:Property^I)\\
\boldsymbol{R}^I &= CEXT^I(rdfs:Resource^I)\\
\boldsymbol{C}^I &= CEXT^I(rdfs:Class^I)\\
\boldsymbol{DC}^I &= CEXT^I(rdfs:Datatype^I)\\
LV &= CEXT^I(rdfs:Literal^I)
\end{aligned}$$

For these rdfs vocabulary, rdfs-interpretation satisfies the extra conditions for RDFS, see (Hayes and McBride, 2004).

The class-superclass relation in the universe is specified with terminology rdfs:subClassOf as the inclusiveness of the class extensions of class/superclass as follows.

Axiom 3. If a pair of two entities is a member of property extensions of $rdfs:subClassOf^I$, then the both entities are instances of $rdfs:Class^I$ and the class extension of the predecessor in the pair is included by the class extension of the successor.

$$\langle x,y\rangle \in EXT^I(rdfs:subClassOf^I) \Rightarrow \\ x,y \in \boldsymbol{C}^I \wedge CEXT^I(x) \subseteq CEXT^I(y)$$

2.3 Semantics in OWL

The OWL Web Ontology Language is a language for defining and instantiating Web ontologies. The OWL language provides three increasingly expressive sublanguages designed for use by specific communities of implementers and users. OWL Lite is the simplest sublanguage and it supports those users primarily needing a classification hierarchy and simple constraint features. OWL DL supports those users who want the maximum expressiveness without losing computational completeness (all entailments are guaranteed to be computed) and decidability (all computations will finish in finite time) of reasoning systems. OWL DL was designed to support the existing Description Logic business segment and has desirable computational properties for reasoning systems (Smith et al., 2004). However, OWL Lite and OWL DL do not completely support RDF semantics. OWL Full is meant for users who want maximum expressiveness and the syntactic freedom of RDF. In OWL Full a class can be treated simultaneously as a collection of individuals[3] and as an individual in its own right (metamodeling).

The OWL specifications include many features and capabilities that are useful to describe Web ontologies, and OWL Lite and DL specifications and its semantics are described in the bunch of specifications (Smith et al., 2004; McGuinness and van Harmelen, 2004; Patel-Schneider et al., 2004a). OWL Full specification is, however, removed from them, and not yet developed as well.

2.3.1 OWL compatibility to RDF

The compatibility between OWL and RDF is discussed in (Patel-Schneider et al., 2004b). Although OWL is regarded to be constructed on top of RDF, it is not so easy actually by historical reasons and others. The specification (Patel-Schneider et al., 2004b) that specifies the universe of OWL is included in the universe of RDF on one hand, see below.

$$\begin{aligned}
\boldsymbol{OC}^I = CEXT^I(owl:Class^I) &\subseteq \boldsymbol{C}^I\\
\boldsymbol{OT}^I = CEXT^I(owl:Thing^I) &\subseteq \boldsymbol{R}^I
\end{aligned}$$

On the other hand, it simultaneously states that $\boldsymbol{OC}^I = \boldsymbol{C}^I$ and $\boldsymbol{OT}^I = \boldsymbol{R}^I$ for OWL Full. As a matter of fact, the OWL definition file[4] contains the definition of the first inclusiveness above, and the second one can be entailed in the RDF universe by the rdfs entailment rule **rdfs4a**[5]. Thus, SWCLOS realized the OWL universe in the RDF universe naturally by loading the file in RDF subsystem of SWCLOS.

In addition, we introduced the following proposition as axiom in order to include OWL classes in the OWL universe, and to make the OWL universe OWL Full. See details in (Koide and Takeda, 2006) and SWCLOS documents[6].

Proposition 1. The extension of the denotation of URI reference of owl:Class is included by the extension of the denotation of URI reference of owl:Thing.

$$\boldsymbol{OC}^I \subseteq \boldsymbol{OT}^I \tag{1}$$

[3]"individual" is a term in Description Logic and synonymous with "instance".

[4]http://www.w3.org/2002/07/owl.rdf

[5]http://www.w3.org/TR/rdf-mt/#rulerdfs4

[6]SWCLOS documents are available at http://www-kasm.nii.ac.jp/~koide/SWCLOS2.files/Page408.htm

This axiom enables every class in the OWL universe to have roles (properties) for individuals such as owl:sameAs, owl:differentFrom, etc.

2.3.2 Metamodeling capability in OWL Full

SWCLOS is the first full-fledged language as OWL Full processor, in which the capability of metamodeling objects is borrowed from the power of the dynamic and reflective features of Lisp and metaclassing capability of CLOS. We had implemented many OWL axioms into CLOS using Meta-Object Protocol (MOP) of CLOS. Whereas the complete freedom of metamodeling certainly results in undecidability, most examples demonstrated by the OWL DL party as OWL Full undecidability are unreasonably extreme and make no sense from the view of engineering. We had showed several metamodeling examples of SWCLOS in (Koide and Takeda, 2006) within the understandable rationale of engineering from our practical experience, and in (Koide and Takeda, 2009), we addressed a set of metamodeling criteria that enables SWCLOS to perform ontology metamodeling.

2.4 Semantics in Common Lisp

Common Lisp as a dialect of lisp is produced by the activity of ANSI standardization on lisps during 1981 through 1997 in U.S., and many systems are running on Common Lisp today in academic fields and industrial applications. However, the semantics embraces some ambiguities specifically from the viewpoint of denotational and extensional semantics like RDF semantics. The problem of subclassing and metaclassing in Common Lisp is discussed in (Koide and Takeda, 2009). In this section, we attempt to categorize computer languages with emphasizing the specialty of lisp as computer language, and discuss the relation among them according to the semantics that is addressed by (Smith, 1984) on the reflective language 3-Lisp.

In a lisp system like early lisp system Lisp 1.5, which is equipped with symbol, function, and list, and without any other structural devices like object in Object Oriented programming, a syntactic lisp expression (S-expression) is reduced to a nominal form that is equivalent to the original one in the sense of λ-calculus. For example, an expression "(+ 1 2)" is reduced to "3". However, we can quote an expression to inhibit the reduction, and then we can change expressions and construct another form, such as "(cons (quote *) (cdr (quote (+ 1 2))))" turns out "(* 1 2)". This specific feature of lisp family languages is recently called *homoiconic*. In this first computational model, lexical expressions are not discriminated from the denotations.

Second computational model in this section discriminates symbols from the denotations. A lexical expression "3" denotes number 3, and a lexical expression "t" and "nil", which are reduced to "t" and "nil", respectively, denote true and falsity, respectively, in the universe of discourse. However, no symbol but "t" and "nil" denotes anything until it is defined so. This mapping from a lexical expression to the denotation is analogous to that in RDF semantics.

In the third computational model, a symbol refers a complex structure as internal structured device in a modern computer language as well as the external list structure in the first lisp model. In this case, a symbol can be used to refer a referent that denotes an entity in the universe. Figure 1 shows the framework in this third computational semantic model (Smith, 1984).

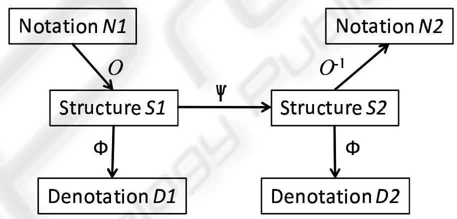

Figure 1: The Framework for Computational Semantics.

Smith called the mapping O *internalization*, and the inverse operation O^{-1} *externalization*. He also noted that O (and O^{-1}) is usually ignored in logic. The ϕ is the interpretation function, which is analogous to the interpretation in denotational semantics, and the reduction ψ, which is, Smith says, the relationship among symbols, corresponds entailment rules and rule application in logic. Smith pointed out that the lisp evaluator crosses semantical levels, and therefore obscures the difference between the simplification ψ and the interpretation ϕ. Smith called this lisp specific nature *de-reference* ($\phi = \psi$). It has become the theoretic base of his work on the reflective language 3-Lisp.

Assumptions and axioms in domain knowledge can be syntactically represented by a set of symbols and structures expressed among them. Those expressions of assumptions are reduced to entailed assertions by a prover ψ.

2.5 Semantics in Common Logic

Common Logic[7] is an abstract language of ISO standard of a logic framework intended for information exchange and transmission. The framework allows a

[7] http://common-logic.org/

variety of different syntactic forms, called dialects, all expressible within a common XML-based syntax and all sharing a single semantics. The dialects, which have different syntax but interchangeable from one to another, include Common Logic Interchange Format (CLIF)[8], Conceptual Graph Interchange Format (CGIF)[9], XML Common Logic (XCL)[10], and Common Logic Controlled English (CLCE)[11]. CLIF may be conceived to be a modernized version of Knowledge Interchange Format (KIF)[12](Hayes and Menzel, 2001).

Common Logic has some novel features syntactically and semantically. It allows a syntax which is signature-free. The abstract syntax model is analogous to polymorphism in Object-Oriented programming and no fixed arity like Common Lisp (*polyadic*). As shown in Table 1, the arity of RDF and OWL is strictly constrained to 2. Not only Common Logic allows *n*-ary, but also the arity is not fixed for a predicate or property. Guha and Hayes initially proposed such features as the RDF syntax for a common base language of Semantic Web languages (Guha and Hayes, 2003). In their proposal for the candidate of RDF, they expected a base language for Semantic Web languages, and claimed the basic language, called L_{base}, that supports basic inference and semantics, and then allows RDF and extending different semantics at the upper layers in the Semantic Web stack. They imagined that L_{base} provides L_i language in i-th layer of Semantic Web language stack.

Common Logic also permits 'higher-order' constructions such as quantification over classes or relations while preserving a first-order model theory. The semantics allows theories to describe intensional entities such as classes and properties. The first solution of this 'higher-order' constructions will be *metamodeling* in Common Logic.

It seems that the modernized features of Common Logic is a reflection of the progress of modern computer languages. For example, the semantics of Common Logic introduced a new term, *universe of reference*, in addition to the universe of discourse in denotational semantics. A dialect is called *segregated* in which some names are *non-discourse names*, namely the denotations of the non-discourse names are in the universe of reference, but not in the universe of discourse in an interpretation. "Segregated dialects are commonly described to have a universe of discourse, without mentioning the universe of reference; and for non-segregated dialects the universes of discourse and of reference are identical. The distinction makes it possible to provide a single semantics which can cover both styles of dialect". The motivation of introducing the universe of reference and non-discourse names is likely to be for the provision against people who do not want to concern some terminologies out of concerning ontologies.[13] However, this notion is very akin to Smith's reference and dereference framework in the previous section. Therefore, we might be able to rephrase that the second language model of Common Lisp is non-segregated, and the third is segregated, in which symbols and internal structures are segregated. This language model will support to develop logic systems using objects in imperative computer languages which include symbols or variable names, structures, objects, whereas the meaning of "segregated" may be misunderstood from its introductory usage in Common Logic.

[8]http://www.ihmc.us/users/phayes/CLIF.html

[9]http://conceptualgraphs.org/

[10]http://www.altheim.com/specs/xcl/1.0/

[11]http://www.jfsowa.com/clce/specs.htm

[12]http://www-ksl.stanford.edu/knowledge-sharing/kif/

3 NON-UNIQUE NAME ASSUMPTION AND EQUALITY

3.1 Equality of Individuals

Unique Name Assumption, that is, different names always denote different entities, which is usually adopted into computer languages, is not adopted in Semantic Web. In OWL language, owl:sameAs property may be applied to different URIs to indicate that two different URI references denote the same entity as individual in the OWL universe. Oppositely, the owl:differentFrom property (and the combination of owl:AllDifferent and owl:distinctMembers, too) may be used to indicate two different URI references denote different entities. Thus, in case of no information on the equality in OWL, the equality of two entities is not determined[14], then, the decision of the equality of entity must be performed in the RDF universe. To discuss the equality of entities in RDF semantics, it is appropriate to discuss the equality of two subgraphs that the two entities are in position of *subject*.

The algorithm for the equality computation in the RDF universe is explained as follows.[15].

Two RDF graphs G and G' are equivalent if there

[13]from the discussion on (Neuhaus, 2010) at the conference.

[14]Instance properties of owl:FunctionalProperty and owl:InverseFunctionalProperty also affect the equality as individual.

[15]http://www.w3.org/TR/rdf-concepts/#section-graph-equality

is a bijection $\mathcal{M}$ between the sets of triples for the two graphs, such that:

1. $\mathcal{M}$ maps blank nodes to blank nodes.
2. $\mathcal{M}(lit) = lit$ for all RDF literals lit which denotes nodes of G.
3. $\mathcal{M}(uri) = uri$ for all RDF URI references uri which denotes nodes of G.
4. s^I, p^I, and o^I for triple $s/p/o$, where I means the interpretation of lexical token s, p, and o, are in G if and only if $\mathcal{M}(s)/p/\mathcal{M}(o)$ also denotes a triple in G'.

For the discussion of equality under the non-UNA condition, we superimpose owl:sameAs and owl:differentFrom properties onto the above algorithm. In case that s in G and s' in G' are blank nodes in a bijection $\mathcal{M}$, $s/p/o$ is equivalent to $\mathcal{M}(s')/p/\mathcal{M}(o')$, if $o = \mathcal{M}(o')$ in OWL semantics, that is, two blank nodes s and s' are equivalent. In case that s and s' are named with different names and we cannot determine the equality by the names, our approach determines the equality between s and s' through the structures of edges in two graphs. Namely, we check p and the equality of o and o'. This algorithm traverses two graphs, until the decision is obtained. Note that RDF graph is a directed graph. In graph equality checking, if two nodes have sub-trees, the corresponding sub-trees on both graphs are recursively checked for the equality. Thus, if we reach at terminal nodes (atomic nodes that do not have edges any more) but no information is obtained, we fall into a troublesome situation. For example, in comparison of ex:Y/ex:p/ex:A and ex:Z/ex:p/ex:B, if ex:A and ex:B are both atomic, the non-UNA computation cannot conclude whether or not ex:Y is equivalent to ex:Z as well as ex:A and ex:B. In such condition, in order to derive useful computational results, we must define the equality or difference among every atomic individuals. It is very laborious work to describe common knowledge such as Bill is different from George, Barack, Al, and so on.

Therefore, we devised a flag for non-UNA and set up falsity to the flag as default. Note that the equality of two blank nodes is checked both in UNA and in non-UNA. In the default condition, we stand in UNA as well as for ordinary computer languages, then two nodes that have different URI references are different, and then two blank node trees are distinct if we cannot find the corresponding edges of graphs or we find the lexically different URI references at the corresponding position in the trees. In non-UNA condition with the flag setting, the graph equality checking is performed even though two URIs at the corresponding position are different, until we find either the difference of graph structures or the difference of nodes that are explicitly stated in OWL statements. In our approach, two atomic nodes with different names are regarded as different in the equality checking, even though the flag indicates non-UNA. Thus, this algorithm is paraphrased *UNA for atomic objects in the non-UNA condition*.

3.2 Equivalency and Disjointness of Classes

3.2.1 Complete Relation for Class Equivalency

In OWL, owl:equivalentClass is applicable to indicate the equivalency of two objects as class. For example, food:Wine in Food Ontology[16] is equivalent to vin:Wine in Wine Ontology with the statement of owl:equivalentClass. In addition, the other three complete relations,[17] i.e., owl:intersectionOf, owl:unionOf, and owl:oneOf also decide the equivalency of classes. If two concepts (classes) have equivalent values for these complete relational properties, the two concepts must be regarded as equivalent class. For example, vin:DryWine and vin:TableWine in Wine Ontology are equivalent as class in extensional semantics (they share the same class extensions), because the both have the same value for owl:intersectionOf property.

3.2.2 Explicit and Implicit Disjointness of Classes

On the other hand, owl:disjointWith and owl:complementOf can be applied to classes to state disjoint classes. These properties explicitly state that two concepts are definitely different as class.

Thus, in case of no declaration of equivalency and disjointness of classes, we cannot conclude the equality as classes immediately. However, the complete relations except owl:equivalentClass decide not only the equality but also the difference of classes. For example, even though we have no direct statement of disjointness for vin:RedWine and vin:WhiteWine, the disjointness is deduced through property owl:intersectionOf and owl:hasValue restriction vin:Red of vin:RedWine and vin:White of vin:WhiteWine, because it is explicitly stated that vin:Red is different from vin:White. Furthermore, we can also conclude some useful results by resorting to the rdf graph checking mentioned above. For example, we can find that vin:CaliforniaWine

[16] http://www.w3.org/TR/2004/REC-owl-guide-20040210/food.rdf

[17] http://www.w3.org/TR/owl-ref/#DescriptionAxiom

is not equal to vin:ItalianWine in spite of no explicit information of disjointness, because the graph equality checking deduces that vin:CaliforniaRegion, in which vin:CaliforniaWine is located, is different from vin:ItalianRegion, in which vin:ItalianWine is located, even if we are in non-UNA.

However, for atomic concepts, we cannot conclude that Man is disjoint to Woman, if those concepts are atomic in non-UNA. Thus, we are forced to do very laborious work to describe common knowledge such as Man and Woman are disjoint, Plant and Animal are disjoint, Ape and Monkey are disjoint, Virus and Bacteria are disjoint, and so on.

ANSI Common Lisp specifies that CLOS classes are pairwise disjoint if they have no common subclass and one class is not a subclass of the other. Namely, each class is disjoint to the others as default until we connect them in superclass relation or set a common subclass. This agreement is supported by the premise that an object in CLOS is typed to only one class. In the RDF universe, an entity can be typed to multiple classes. So, the nature of disjointness in CLOS is not applicable in the RDF universe in theory. However, in SWCLOS, the pseudo multiple-classing machinery is implemented, using the CLOS class and multiple-inheritance mechanism. Therefore, from the viewpoint of CLOS, the algorithm of disjointness for CLOS is still valid in the RDF universe in virtue of CLOS. In the next section, we introduce an idea of *role concept* that is divided from *substantial concept* with the premise of pairwise disjointness.

4 SUBSTANTIAL CONCEPTS AND ROLE CONCEPTS

4.1 Pairwise Disjoint Datatype

In SWCLOS, we defined XML datatype wrappers as mapping XML Schema descriptions to lisp datatypes in lisp value space. We defined xsd datatypes in lisp space and the same datatypes are also defined as CLOS classes in the RDF universe in order to treat them in the RDF universe. Therefore, independent XML Schema datatypes in SWCLOS are pairwise disjoint by the implicit disjointness of CLOS classes, e.g., xsd:float is disjoint with xsd:integer, xsd:URI, xsd:string, xsd:boolean, etc.

4.2 Ontological Categories and Disjointness

OWL provided the description of class disjointness and forced us labor-intensive work as described above. W3C new recommendation for OWL, namely OWL 2 specification (Carroll et al., 2009), attempts to solve the disjointness problems without ontological consideration in depth. In OWL 2, Person may be described as owl:disjointUnionOf Man and Woman. However, we are still forced to describe explicitly disjointness for all disjoint classes, or basic atomic concepts. We strongly claim that the approach to describe disjointness must be well-founded on ontological consideration.

Sowa (Sowa, 1995; Sowa, 1999) showed a lattice of the top-level ontological categories of things. Each of the twelve elemental concepts in the top ontology has different characteristics and those combinations, i.e., *independent*, *physical*, *relative*, *abstract*, and *mediating*. The concepts of the independent exist itself and they show the firstness. The concepts of the relative or *role* only live with the firstness and they show the secondness. The mediating describes concepts that mediate the firstness and the secondness.

Guarino (Guarino, 1998) parted ontology into two catagories, i.e., *particular* that represents substantial entities and *universal* that is the category of entities required to describe the particulars. Physical objects, abstract processes, phenomena, quality, and materials fall into the particular, and attributes, relations are categorized into the universal.

Mizoguchi, et al., developed an ontology building tool called Hozo (Mizoguchi et al., 2007; Kozaki et al., 2007) based on ontological deep discussion and have utilized Hozo for many application field of ontology building. Using Hozo, ontology builders can easily construct complex concepts that are composed of substantial sorts and non-substantial roles. For example, Wife is a part of Family and composed of Woman and Wife-role. The concept Woman is a substantial and may have slots of gender, age, etc. The role concept Wife-role is not a substantial, in other words, it always requires substantial concepts to work, but may have its own slots such as married-year, partner, etc. In a sense, it is regarded that the concept Family represents the context in which the concept Wife is activated from Woman with Wife-role.

We also proposed *Aspect Theory* of ontology in the study of *Knowledgeable Community* (Takeda et al., 1995), which is a framework of knowledge sharing and reuse based on a multi-agent architecture. In this framework, while ontologies are the minimum requirement for each agent to join the community, each of heterogeneous ontologies describes an aspect of an entity and knowledge. A mediator agent that embodied knowledge for mediation helps other agents to communicate each other. In this theory, the aspect may be rephrased as a context on which an agent focused for discourse. For example, a concept Temple is an aggregation of concepts in aspect of religion, cultural asset, building architecture, corporate body, and so on. In most case without communication, we usually focus on one aspect of entity and do not need to take care the other aspects in a particular context. However, for agents in a particular discourse, the mediator translates heterogeneous ontologies from one to another and mediates agent's speech acts that are broad-casted in the community.

4.3 Introduction of Role Concepts

In this paper, in order to solve the labor-intensive disjointness problem, we propose two ontological categories according to (Kozaki et al., 2007), i.e., substantial concepts and role concepts, and realize them on top of RDF semantics. Neither owl:disjointUnionOf nor owl:AllDisjointClasses in OWL 2 are introduced. The substantial concepts are described in OWL syntax, but we adopt the assumption of implicit disjointness for substantial concepts in the same way of CLOS as described above. On the other hand, the discussion of disjointness on role concept is meaningless, because the role concept cannot have any instance by itself. A complex concept (role holder) is composed of a substantial concept and role concepts. Two complex concepts such as Husband and Teacher can share individuals, but those individuals should be interpreted as instances of Man in Husband and Person in Teacher. The syntax and semantics of role concepts are described at Appendix A.

5 OPEN WORLD ASSUMPTION AND TERNARY TRUTH VALUE

5.1 Auto-epistemic Local Closed World Assumption

Negation as Failure (NaF) is a well-known convention for inference in Closed World Assumption (CWA). This convention is, however, not applicable in World Wide Web. Therefore, two queries are usually issued as "*P*?" and "not *P*?" for query-answer systems. In case that we cannot obtained any results with two queries, it may be called unknown. As shown so far, rigorous non-UNA and full Open World Assumption in Semantic Web often produce shortcomings in practical ontology building. The implicit disjointness principle adopted in SWCLOS is very useful all over the life-cycle of ontology engineering. This principle can be rephrased such that we assume axiomatized classes are disjoint each other until the disjointness are explicitly axiomatized. In some sense, it is a kind of convenient and arbitrarily default reasoning. A reasoner replies that the concept *A* is disjoint with the concept *B* because of no evidence that supports it. After the statement that the concept *A* and *B* has a subclass *C*, the same agent replies the concept *A* and *B* are not disjoint. However, even so, note that SWCLOS signals an alarm if a user attempts to make an instance of class *A* and *B* that is explicitly stated as disjoint, and also note that SWCLOS implicitly makes a *shadowed*-class as a subclass in CLOS of class *A* and *B*, if *A* and *B* are not explicitly disjoint and it is required by users (Koide and Takeda, 2006).

Concerned with the existential restriction of property or owl:someValuesFrom, the full OWA is also meaningless from the viewpoint of ontology building, since the existential restriction under the OWA means the possibility that a satisfiable value may be defined somewhere in WWW or someone in the team members may add a proper constraint tomorrow or after. The full OWA implies that ontology builders cannot know all for target ontologies. However, this assumption is not enjoyable in actual fact in personal and collaborative ontology building process. It is natural to distinguish the local world for target ontologies and the given general WWW. Hence, we have introduced the notion of *auto-epistemic local closed world assumption*. In this idea, agents can introspectively check their knowledge within their extent of capabilities. An agent sits in locally closed world as environments around it. The flag for auto-epistemic local closed world assumption is set true as default in SWCLOS, and the satisfiability for slot value is aggressively checked even in case of the existential restriction. Namely, if an existential restriction is not satisfied, then the interpretation is not satisfied. Setting the flag false means the completely full OWA. In this case, no alarm is signaled for the existential restrictions.

5.2 cl:subtypep

`cl:subtypep` in ANSI Common Lisp returns two values, say, *value1* and *value2*. Table 2 is taken from ANSI Common Lisp specs[18].

[18] http://www.franz.com/support/documentation/8.1/ansicl/dictentr/subtypep.htm

Table 2: Two Return Values on cl:subtypep(type1, type2).

*value*1	*value*2	meaning
true	true	*type*1 is definitely a subtype of *type*2.
false	true	*type*1 is definitely not a subtype of *type*2.
false	false	subtypep could not determine the relation ship, so *type*1 might or might not be a subtype of *type*2.

If *value1* is true, then *value2* is definitely true. So, a return value of pair ⟨t, nil⟩ never happens in ANSI Common Lisp.

We extended this semantics and applied it for subtype (subclass) predicates in RDFS and OWL. Namely, we see that ⟨t, t⟩ is true value, ⟨nil, t⟩ is false value, and ⟨nil, nil⟩ is unknown value in RDF(S) and OWL semantics. The ternary truth table are used for the subsumption computation and elsewhere in SWCLOS, see the details in (Koide and Takeda, 2006).

6 CONCLUSIONS AND FUTURE WORK

In this paper, we described an overview of several knowledge representation languages around World Wide Web, focusing on semantics of languages. We claimed that today's OWL, including OWL 2, embraces some drawbacks for the practical usage. It seems to lead people into a blind alley without thinking what ontology is and how it should be represented. Common Logic intends to be a common framework of concrete knowledge representation languages that are compatible with World Wide Web. Although actual dialect implementation of Common Logic is not emerging and no one can foresee the future of Common Logic, our experience for SWCLOS suggests the future of something else than OWL.

REFERENCES

Carroll, J., Herman, I., and Patel-Schneider, P. F. (2009). Owl 2 web ontology language rdf-based semantics. W3C Candidate Recommendation. http://www.w3.org/TR/owl2-rdf-based-semantics/.

Guarino, N. (1998). Some ontological principles for designing upper level lexical resources. In Rubio, A., Gallardo, N., Castro, R., and Tejada, A., editors, *the First International Conference on Lexical Resources and Evaluation*, pages 527–534, Granada, Spain. ELRA - European Language Resources Association.

Guha, R. V. and Hayes, P. (2003). Lbase: Semantics for languages of the semantic web. Note, W3C.

Hayes, P. and McBride, B. (2004). RDF Semantics. W3C Recommendation. http://www.w3.org/TR/rdf-mt/.

Hayes, P. and Menzel, C. (2001). A semantics for the knowledge interchange format. In *In IJCAI 2001 Workshop on the IEEE Standard Upper Ontology*.

Koide, S. and Takeda, H. (2006). OWL-Full reasoning from an object oriented perspective. In *Asian Semantic Web Conf., ASWC2006*, pages 263–277. Springer.

Koide, S. and Takeda, H. (2009). Meta-circularity and mop in common lisp for owl full. In *ELW '09: Proceedings of the 6th European Lisp Workshop*, pages 28–34, New York, NY, USA. ACM.

Kozaki, K., Sunagawa, E., Kitamura, Y., and Mizoguchi, R. (2007). Role representation model using owl and swrl. In *Proc. of 2nd Workshop on Roles and Relationships in Object Oriented Programming, Multiagent Systems, and Ontologies*, Berlin.

McDermott, D. (1978). Tarskian semantics, or no notation without denotation! *Cognitive Science*, 2:277–282.

McGuinness, D. L. and van Harmelen, F. (2004). OWL Web Ontology Language Overview. W3C Recommendation. http://www.w3.org/TR/owl-features/.

Mizoguchi, R., Sunagawa, E., Kozaki, K., and Kitamura, Y. (2007). The model of roles within an ontology development tool: Hozo. *Appl. Ontol.*, 2(2):159–179.

Neuhaus, F. (2010). The semantics of modules in common logic. In Smith, B., Mizoguchi, R., and Nakagawa, S., editors, *Interdisciplinary Ontology*, volume 3, pages 107–117. Open Research Centre for Logic and Formal Ontology, Keio University.

Patel-Schneider, P. F., Hayes, P., and Horrocks, I. (2004a). Owl web ontology language semantics and abstract syntax. W3C Recommendation.

Patel-Schneider, P. F., Hayes, P., and Horrocks, I. (2004b). OWL Web Ontology Language Semantics and Abstract Syntax section 5. rdf-compatible model-theoretic semantics. W3C Recommendation, http://www.w3.org/TR/owl-semantics/rdfs.html.

Russell, S. and Norvig, P. (2003). *Artificial Intelligence: A Modern Approach Second Edition*. Prentice Hall.

Smith, B. C. (1984). Reflection and semantics in Lisp. In *POPL '84: Proceedings of the 11th ACM SIGACT-SIGPLAN symposium on Principles of programming languages*, pages 23–35, New York, NY, USA. ACM Press.

Smith, M. K., Welty, C., and McGuinness, D. L. (2004). OWL Web Ontology Language Guide. W3C Recommendation. http://www.w3.org/TR/owl-guide/.

Sowa, J. F. (1995). Top-level ontological categories. *Int. J. Hum.-Comput. Stud.*, 43(5-6):669–685.

Sowa, J. F. (1999). *Knowledge Representation: Logical, Philosophical, and Computational Foundations*. Brooks Cole Publishing Co., Pacific Grove, CA.

Takeda, H., Iino, K., and Nishida, T. (1995). Agent organization and communication with multiple ontologies. *Int. J. Cooperative Inf. Syst.*, 4(4):321–338.

Tarski, A. (1946/1995). *Introduction to Logic*. Dover. First publishing in 1936 in Polish.

APPENDIX

Syntax and Semantics of Role Concept Description

Syntax in S-expression of SWCLOS.

substantial-concept-def ::= (`defConcept` *concept-name slot-form**)
slot-form ::= (*role* [*lang*] *form+*)
form ::= (*typetag* [*name*] [*lang-form*] *slot-form**)
| (*datatype* *data*) | (*lang* *form*) | *cl:string* | *cl:number* | *uri*
lang-form ::= (`xml:lang` *lang*)

role-concept-def ::= (`defRole` *role-name* [*holder-name*] `:partOf` *whole-name*
from-concept *slot-form**)
| (`defRole` *name* [*holder-name*] `:in` *context-name* *from-concept* *slot-form**)
from-concept ::= (`from` *name* *slot-form**)

concept-name, *role-name*, *holder-name*, *whole-name*, *context-name*, and *name* are QNames in S-expression in SWCLOS, or lisp symbols.

Example

```
(defConcept Woman
  (owl:intersectionOf Person
                      (owl:Restriction (owl:onProperty hasGender)
                                       (owl:hasValue Femail)))
  (rdfs:subClassOf (owl:Restriction (owl:onProperty name)
                                    (owl:someValuesFrom xsd:string))
                   (owl:Restriction (owl:onProperty age)
                                    (owl:allValuesFrom xsd:positiveInteger))))
(defRole WifeRole Wife :partOf Family
  (from Woman (name xsd:string) (age (greaterThan 16)))
  (partner Husband) (marriageYear Year))}
```

Table 3: Role Concept Semantic Conditions (Axioms).

$x \in CEXT^I(y)$ *iff* $\langle x,y\rangle \in EXT^I(\mathrm{rdf{:}type}^I)$ $\boldsymbol{C}^I = CEXT^I(\mathrm{rdfs{:}Class}^I)$, $\boldsymbol{R}^I = CEXT^I(\mathrm{rdfs{:}Resource}^I)$
$x \in \boldsymbol{C}^I \Rightarrow \langle x, \mathrm{rdfs{:}Resource}^I\rangle \in EXT^I(\mathrm{rdfs{:}subClassOf}^I)$
$\langle x,y\rangle \in EXT^I(\mathrm{rdfs{:}subClassOf}^I) \Rightarrow x$ and $y \in \boldsymbol{C}^I \wedge CEXT^I(x) \subseteq CEXT^I(y)$
$EXT^I(\mathrm{rdfs{:}subClassOf}^I)$ is transitive and reflexive on $\boldsymbol{C}^I$
$\mathrm{owl{:}Class}^I \in \boldsymbol{C}^I \wedge \boldsymbol{OC}^I = CEXT^I(\mathrm{owl{:}Class}^I) \subseteq \boldsymbol{C}^I$
$\mathrm{owl{:}Thing}^I \in \boldsymbol{OC}^I \wedge \boldsymbol{OT}^I = CEXT^I(\mathrm{owl{:}Thing}^I) \subseteq \boldsymbol{R}^I$
$\boldsymbol{OC}^I \subseteq \boldsymbol{OT}^I$
$\mathrm{Concept}^I \in \boldsymbol{OC}^I \wedge \boldsymbol{CO}^I = CEXT^I(\mathrm{Concept}^I) \subseteq \boldsymbol{OC}^I$
$\mathrm{Role}^I \in \boldsymbol{C}^I \wedge \boldsymbol{RO}^I = CEXT^I(\mathrm{Role}^I) \subseteq \boldsymbol{C}^I$
$\langle x,y\rangle \in EXT^I(\mathrm{partOf}^I) \Rightarrow x \in \boldsymbol{RO}^I, y \in \boldsymbol{CO}^I$
$\langle x,y\rangle \in EXT^I(\mathrm{in}^I) \Rightarrow x \in \boldsymbol{RO}^I, y \in \boldsymbol{C}^I$
$\langle x,y\rangle \in EXT^I(\mathrm{from}^I) \Rightarrow x \in \boldsymbol{RO}^I, y \in \boldsymbol{CO}^I$
$x \in \boldsymbol{RO}^I \Rightarrow CEXT^I(x) = \{\}$
$x,y \in \boldsymbol{CO}^I : CEXT^I(x) \cap CEXT^I(y) = \{\} \,/\, CEXT^I(x) \cap CEXT^I(y) = \{\}$
$\boldsymbol{RH} \equiv \boldsymbol{CO}^I \times \boldsymbol{RO}^I$

$\boldsymbol{RH}$ expresses the extension of role holders in the universe.
$A : B/B$ means default reasoning such that if A and no evident of not B, then B.