Web-Based Ontology Languages and its Based Description Logics

Mohammad Mustafa Taye
Department of Computer Information Systems
Faculty of Information Technology
Philadelphia University, Jordan
mtaye@philadelphia.edu.jo

Abstract
The main object of semantic web languages is to add semantics to the existing information on the Web. These web languages have been developed to represent or express ontologies. Therefore, these description languages provide richer constructors for forming complex class expressions and axioms. An ontology is expressed in a knowledge representation language, which provides a formal frame of semantics. Since description logic is the basis of most ontology language, it is appropriate to explain briefly the base of the description to understand the ontology language clearly. The aim of this paper is to provide a brief survey of state-of-the-art ontology languages which are used to express ontology over the Web is provided. Also the goal of this paper is to provide a basic understanding of ontologies and description logics, which are the basis of ontology languages.

Keywords
Ontology, Semantic Web, web languages, description languages, knowledge representation language, description logic.

1. Introduction
The goal of an ontology is to achieve a common and shared knowledge that can be transmitted between people and between application systems [19]. Thus, ontologies [8, 13] play an important role in achieving interoperability across organizations and on the Semantic Web [24], because they aim to capture domain knowledge and their role is to create semantics explicitly in a generic way, providing the basis for agreement within a domain. Ontology is used to enable interoperability between Web applications from different areas or from different views on one area.

The main object of semantic web languages is to add semantics to the existing information on the Web. RDF/RDFS [6], OIL [9], DAML+OIL [11] and OWL [1] are modelling web languages that have been developed to represent or express ontologies. In general, most of them are based on XML [5] syntax, but they have different terminologies and expressions. Indeed, some of these languages have the ability to represent certain logical relations which others do not. Because some languages have greater expressive power than others, their selection for representing ontologies is based mainly on what the ontology represents or what it will be used for [12]. In other words, different kinds of ontological knowledge-based applications need different language facilitators to enable reasoning on ontology data. These description languages provide richer constructors for forming complex class expressions and axioms [3, 23].

Before OWL, much research had been conducted into creating a powerful ontology modelling language. This research stream began with the XML-based RDF and RDF/S, progressed to the Ontology Inference Layer (OIL) and continued with the creation of DAML+OIL, the result of joining the American proposal DAML-ONT5 with the European language OIL. All these languages are based on XML or RDF syntax and are consequently compatible with web standards. Indeed, RDF and OWL make searching for and reusing information both easier and more reliable, because they are considered as standards that enable the Web to be a global infrastructure for sharing documents and data equally [3].

As mentioned in [1, 7], some important requirements for quality support should be taken into account when developing languages for encoding ontologies. These include giving the user explicit written format, ease of use, expressive power, compatibility, sharing and versioning, internationalisation, formal conceptualisations of domain models, well-defined syntax and semantics, efficient reasoning support, sufficient expressive power and convenience of expression [2,12].

Syntax is one of the most important features of any language, so it should be well-defined; it is also the most significant condition required for the processing of information by machine [2, 23].

The semantics of knowledge should be well defined, because it represents the meaning of that knowledge [2, 3]. Formal semantics should be established in the domain of mathematical logic in a clearly defined way that will lead to unambiguous meaning, since well defined semantics will lead to correct reasoning. Semantics can be considered a prerequisite to support reasoning [12]. On the other hand, reasoning will help to check and discover consistent ontology, to verify unintended relationships between classes and to classify individuals into classes [12].

This paper sets out the definition, structure of ontology. It also has detailed the most common and important languages, RDF, RDF/S, DAML+OIL and OWL, all of which are based on XML. It also describes Description Logic (DL).

2. Ontology
Ontologies [8, 19], which are used in order to support interoperability and common understanding between the different parties, are a key component in solving the problem of semantic heterogeneity, thus enabling semantic interoperability between different web applications and services.

The primary use of the word “ontology” is in the discipline of philosophy, where it means “the study or theory of the explanation of being”; it thus defines an entity or being and its relationship with and activity in its environment. In other disciplines, such as software engineering and AI, it is defined as “a formal explicit specification of a shared conceptualization” [19]. The foundations of this definition are:

- All knowledge (e.g. the type of concepts used and the constraints on their use) in ontology must have an explicit specification.
- An ontology is a conceptualisation, which means it has a universally comprehensible concept.
- “Shared” indicates an agreement about the meaning in such domains. In other words, an ontology should capture consensual knowledge accepted by the communities.
resources, which is why it has become a common method of describing the properties, time, information and content of web resources, so that it

XML, which is designed for syntax, while RDF is intended for semantics. As has been mentioned, it is a framework for describing web

RDF, which is recommended by the W3C, uses URIs to identify resources or things (the root of an ontology is called a thing). It is based upon

On 10 February, 2004, the World Wide Web Consortium (W3C) announced its support for two Semantic Web technology standards, RDF and OWL; that is, the information resources described in semantic language specification. OWL is a standard ontology description language, built on the RDF, which is based on the XML-authoring tools, used mainly to express the needs of computer applications to deal with knowledge and information in the document.

2.1 Ontology Representation
Ontology is comprised of four main components: concepts, instances, relations and axioms. The present research adopts the following definitions of these ontological components [13]:

- A Concept (also known as a class or a term) is an abstract group, set or collection of objects. It is the fundamental element of the domain and usually represents a group or class whose members share common properties. This component is represented in hierarchical graphs, such that it looks similar to object-oriented systems. The concept is represented by a “super-class”, representing the higher class or so-called “parent class”, and a “subclass” which represents the subordinate or so-called “child class”. For instance, a university could be represented as a class with many subclasses, such as faculties, libraries and employees.

- An Instance (also known as an individual) is the “ground-level” component of an ontology which represents a specific object or element of a concept or class. For example, “Jordan” could be an instance of the class “Arab countries” or simply “countries”.

- A Relation (also known as a slot) is used to express relationships between two concepts in a given domain. More specifically, it describes the relationship between the first concept, represented in the domain, and the second, represented in the range. For example, “study” could be represented as a relationship between the concept “person” (which is a concept in the domain) and “university” or “college” (which is a concept in the range).

- An Axiom is used to impose constraints on the values of classes or instances, so axioms are generally expressed using logic-based languages such as first-order logic; they are used to verify the consistency of the ontology.

3. Ontology Description Languages
Ontology language is the basis of ontological knowledge systems, the definition of a system of knowledge representation language specification; it not only has a rich and intuitive ability to express and use it, but the body should be easily understood by the computer, processing and applications.

On 10 February, 2004, the World Wide Web Consortium (W3C) announced its support for two Semantic Web technology standards, RDF and OWL; that is, the information resources described in semantic language specification. OWL is a standard ontology description language, built on the RDF, which is based on the XML-authoring tools, used mainly to express the needs of computer applications to deal with knowledge and information in the document.

3.1 Resource Description Framework (RDF)
RDF [4, 6, 16], a language used to provide a standard for metadata about the resources on the Web, is capable of representing data on and exchanging knowledge over the Web. It was developed to be understood by computers, facilitating interoperability between applications. In other words, it is a framework for using and representing metadata and describing the semantics of information about web resources in a way that is accessible to machines.

RDF, which is recommended by the W3C, uses URIs to identify resources or things (the root of an ontology is called a thing). It is based upon XML, which is designed for syntax, while RDF is intended for semantics. As has been mentioned, it is a framework for describing web resources, which is why it has become a common method of describing the properties, time, information and content of web resources, so that it can be read and understood by computer applications.

RDF can be used in several applications, one of the most important being resource discovery, used to enhance search engine capabilities. It is also used to facilitate knowledge sharing and exchange in intelligent software agents and, as previously mentioned, to describe the content and content relationships available with any resource, such as a page.

The RDF model has three elements: a resource (the subject), the object and the predicate. It is possible to say that <subject> has a property <predicate> valued by <object>.

For example, a triplet could be “H. Zedan is the Head of the STRL Group”. In an RDF graph, all triplets “nodes and arcs” should be labelled with qualified URIs. In this example, it could be said that the subject (resource) is the STRL Group, the predicate (property) is Head of, and the object (literally) is H. Zedan.

3.2 Resource Description Framework Schema (RDFS)
The Resource Description Framework Schema (RDFS) [6] has been built upon the XML and RDF models and upon syntax. RDFS offers extra facilities to encourage evolution in both the individual RDF vocabularies and the core RDF Schema vocabulary.

It provides a machine-understandable system for defining the vocabularies needed for such applications or descriptive vocabularies. In other words, it is a group of RDF resources that can be used to define or express the properties of other RDF resources which define application-specific RDF vocabularies. At the same time, RDFS helps developers to describe classes and properties in a specific way and to specify relationships between those properties and classes, allowing combinations between classes, properties or values. In other words, RDFS is used to define RDF vocabularies.

In general, RDFS is defined in a namespace informally called ‘rdfs’ and identified by the URI reference http://www.w3.org/2000/01/rdf-schema#. On the other hand, RDF is defined in a namespace informally called ‘rdf’ and identified by the URI reference http://www.w3.org/1999/02/22-rdf-syntax-ns#.
formalise the meaning of a language. To this end, it uses URIs to define the resources as RDF. DAML+OIL was actually developed to describe the structure of web resources. In fact, it was built on RDF and RDFS, which is to say that it has an RDF/XML syntax based on the frame paradigm; so DAML+OIL could be considered a specific kind of RDF extended with richer modelling primitives to cope with the weaknesses of RDF/RDFS. To this end, it uses URIs to define the resources as RDF. DAML+OIL was actually developed to describe the structure of a domain, as most web-based languages describe structure in terms of hierarchies: subclass and sub-property relationships, domain and range restrictions, and instances of classes. However, some features, such as the specialised or defined properties of local scope and the specialisation of their domains and ranges, are still missing. It is impossible to separate some classes from each other. For example, it is wrong to say male and female are disjoint, but RDF can cater for subclass relationships only; e.g. male is a subclass of human being. On other hand, it is impossible to combine or create classes using Boolean relations. The expression of many restrictions is limited. The need consequently arose for a new language to overcome all these deficiencies.

There are also many limitations to RDFS, among which are its inability to express equality and inequality, and its limited ability to define the enumeration of property values. Regarding the latter, it cannot describe some relations among entities, such as union, intersection, unique, symmetric, transitive and inverse, and, as far as complement constraints go, it cannot apply cardinality and existence. Domain and range can only be specified globally. As a result, several languages, such as OWL and DAML+OIL, have been developed to meet these limitations.

### 3.3 Annotated DAML+OIL Ontology Markup

DARPA Agent Markup Language (DAML) + Ontology Inference Layer (OIL), or DAML+OIL [11, 18] is a semantic markup language designed for use in Web resources. In fact, it was built on RDF and RDFS, which is to say that it has an RDF/XML syntax based on the frame paradigm; so DAML+OIL could be considered a specific kind of RDF extended with richer modelling primitives to cope with the weaknesses of RDF/RDFS. To this end, it uses URIs to define the resources as RDF. DAML+OIL was actually developed to describe the structure of a domain, as most web-based languages describe structure in terms of classes and properties. DAML+OIL uses a description logic (DL)-style model theory to formalise the meaning of a domain. [16]
Researchers first created OIL and a further effort produced DAML+OIL, an amalgamation of an American proposal and a European language.

3.4 Ontology Interchange Language (OIL)
A semantic markup language for Web semantics has been built on RDF and RDF/S, this language providing modelling primitives used in frame-based and DL-oriented ontologies [9].

DAML+OIL has many limitations [11]: it lacks property constructors, it has no composition or transitive closure, its only property types are transitive and symmetrical, sets are the only collection type (there are no bags or lists), there is no comparison in data value, it allows only unary and binary relations, and there are neither default values nor variables.

3.5 Web Ontology Language (OWL)
OWL [1, 7, 17, 20, 21], which is a language for processing web information, became a W3C recommendation in February 2004 and was built using RDF to remedy the weaknesses in RDF/S and DAML+OIL. It provides more affluent integration and interoperability of data between communities and domains.

It can be said that there is a similarity between OWL and RDF, but the former has a stronger syntax with more machine interpretability and vocabulary language than the latter. Obviously, RDF is commonly limited to binary ground predicates, and RDFS also has the limitation that it represents a subclass hierarchy and a property hierarchy, with the domain and range definitions of these properties. In other words, the language of OWL is more expressive than that of RDF(S).

To cope with the limitations of RDF, RDFS and DAML+OIL, W3C defined OWL. Indeed, OWL is an extension of RDFS; in other words, it builds on RDF and RDFS, using XML syntax; overall, OWL uses the RDF meaning of classes and properties. W3C classifies OWL into three sublanguages, each of which is intended to supply different aspects of these incompatibilities. These are OWL Lite, OWL DL and OWL Full [76].

OWL Lite is the simplest version of OWL and provides a classification hierarchy and simple constraints; it permits only the expression of relationships with maximum cardinality equal to 0 or 1, thus being designed for easy implementation. In this sublanguage, there is some restriction of OWL DL to a subset of language constructors, with some limitations such as an absence of explicit negation or union. The disadvantage of this sublanguage is restricted expressiveness.

OWL DL is so called because it uses Description Logic to represent the relations between objects and their properties. Indeed, it provides maximum expressiveness while preserving the completeness of computational properties. OWL Lite is a sublanguage of OWL DL.

The sublanguage OWL Full provides highest expressiveness and the syntactic freedom of RDF but without preserving guarantees on computational complexity. OWL Lite and OWL DL are sublanguages of OWL Full.

Table 1 shows a comparison between the ontology languages discussed above.

Table 1: Comparison between ontology languages
The Research Bulletin of Jordan ACM, ISSN: 2078-7952, Volume II (II)  Page 15

The following table shows the limitations and differences between RDF, DAML+OIL and OWL. The table shows many limitations to RDFS, among which are its inability to express equality and inequality, and its limited ability to define enumeration of property values. DAML+OIL has many limitations, such as property constructors; it has no composition or transitive closure, in property types contain transitive and symmetrical.

<table>
<thead>
<tr>
<th>The expression</th>
<th>RDF/RDFS</th>
<th>DAML+OIL</th>
<th>OWL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>rdf:Property</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>rdfs:subClassOf</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>rdfs:subPropertyOf</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>rdfs:domain</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>rdfs:range</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Individual</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>sameClassAs</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>samePropertyAs</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>sameIndividualAs</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>differentIndividualFrom</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>inverseOf</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TransitiveProperty</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>SymmetricProperty</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>FunctionalProperty</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>InverseFunctionalProperty</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>allValuesFrom</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>someValuesFrom</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>minCardinality</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>maxCardinality</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>cardinality</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>oneOf</td>
<td>✓</td>
<td>×</td>
<td>✓</td>
</tr>
<tr>
<td>disjointWith</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>complementOf</td>
<td>×</td>
<td>✓</td>
<td>✓</td>
</tr>
</tbody>
</table>

4. Representation Knowledge Structures

There are many types of representation knowledge structures, including DL (ontology languages), network-based forms (semantic networks), frame-based and object-oriented representations.

4.1 Network-based Representation Structures

The term “network-based” [2, 23] refers to the structure of this type, so the present proposal will deal with a general view of networks containing the two important elements of nodes and links. In general, nodes are used to illustrate concepts such as classes or individual objects. Concepts could have attributes which are linked to the corresponding nodes; links are used to describe relationships between concepts.

4.2 Frame-based Structures

Frame-based structures [2, 23] provide modelling support to the ontology developer and user, but have limitations in that they lack a well-defined semantic. There is, however, a frame-based language called Frame Logic (F-Logic), grounded in first-order logic, which supports a well-defined and well-understood semantic. The lack of well-defined semantics is a universal problem when using a frame-based structure as an ontology language. Well-defined semantics are essential to enable applications to “understand” the ontology, or at least process it according to well-defined rules.

4.3 Description Logics

Description logics [2, 3, 12, 23] are a family of decidable logics for formal knowledge representation. Moreover, they are formal languages for representing knowledge used to express the conceptualisation of such domains in an organised and formally well-understood manner, leading to the provision of decidable formal semantics. They can represent concepts and their relationships (roles), also giving them formal semantics. Actually, the core descriptive tools of a DL are represented, combining concepts in a suitable form. Thus, it is possible to describe the concepts and their relations.

This kind of representation language refers to two elements: concept description, used to describe a domain, and logic-based semantics. It is considered to define a subset of First Order Logic (FOL).

The main structures of this type are:
• Syntax (formula), which concerns the definition of how to write correct sentences in the language. In this type, a set of symbols called “concept” and a set of binary relations called “role” are defined.

• Semantics, which concerns the definition of the meaning of sentences by interpreting concepts and roles as sets of individuals. This refers to truth value and interpretation.

• Logical inference (reasoning), which derives results logically, should be decidable and efficient.

DL is considered an extension of those frame-based and semantic networks which are not capable of logic-based semantics; it is therefore also called “terminological logics”.

The structure of knowledge in DL is represented in a hierarchical organisation of classes. These concepts are defined by using descriptions which identify the properties that objects must satisfy in order to belong to the concept.

Actually, a description logic theory consists of statements about concepts, individuals and their relations. Thus, DL systems allow the representation of ontologies in three components:

• Concepts in DL are as in the frame paradigm: they represent classes of objects. In other words, they are a set of individuals of the application domain that have some common characteristics, as can be the case for people or cars.

• Roles are the logical representations of relationships between concepts; for example, the role hasFather or the role madeOf. In other words, they describe binary relations between concepts; consequently they also allow the description of properties of concepts.

• Individuals represent instances of classes.

In DL, there are two special concepts, named Top and Bottom; Top (⊤) is a concept that contains all the individuals of the domain, while Bottom (⊥) is the empty concept, which also represents the contradiction.

4.3.1 Description Logic Families

Section describes the constructors of the DLs and, as a consequence, identifies the most common DL families. For every DL, the concepts Top and Bottom are interpreted as follows:

\[ ⊤ ≡ A \cup \neg A \Rightarrow \mathcal{I}^I = \Delta^I \]

\[ ⊥ ≡ A \cap \neg A \Rightarrow \mathcal{I}^I = \emptyset \]

Where: \( \Delta^I \) is a interpretation domain and \( \mathcal{I} \) is a interpretation function

Table 2 shows the main construct and syntax of each DL language. For example, the language SHIQ contains all constructs and syntax of the S, H, I and Q languages [2].

<table>
<thead>
<tr>
<th>Construct</th>
<th>Syntax</th>
<th>Languages</th>
</tr>
</thead>
<tbody>
<tr>
<td>Concept</td>
<td>( A )</td>
<td>FL(_0)</td>
</tr>
<tr>
<td>Role name</td>
<td>( R )</td>
<td>FL(_-)</td>
</tr>
<tr>
<td>Intersection</td>
<td>( C \cap D )</td>
<td>AL</td>
</tr>
<tr>
<td>Value Restriction</td>
<td>( \forall R.C )</td>
<td>S</td>
</tr>
<tr>
<td>Limited existential Quantification</td>
<td>( \exists R )</td>
<td></td>
</tr>
<tr>
<td>Top Or Universal</td>
<td>( \top )</td>
<td></td>
</tr>
<tr>
<td>Bottom</td>
<td>( \bot )</td>
<td></td>
</tr>
<tr>
<td>Atomic Negation</td>
<td>( \neg A )</td>
<td></td>
</tr>
<tr>
<td>Negation</td>
<td>( \neg C )</td>
<td>C</td>
</tr>
<tr>
<td>Union</td>
<td>( C \cup D )</td>
<td>U</td>
</tr>
<tr>
<td>Existential Restriction</td>
<td>( \exists R.C )</td>
<td>E</td>
</tr>
<tr>
<td>Number Restrictions</td>
<td>( (\geq n R) (\leq n R) )</td>
<td>N</td>
</tr>
<tr>
<td>Nominals</td>
<td>{ a_1 ... a_n }</td>
<td>O</td>
</tr>
<tr>
<td>Role Hierarchy</td>
<td>( R \subseteq S )</td>
<td>H</td>
</tr>
<tr>
<td>Inverse Role</td>
<td>( R^- )</td>
<td>I</td>
</tr>
<tr>
<td>Qualified Number Restriction</td>
<td>( (\geq n R.C) (\leq n R.C) )</td>
<td>Q</td>
</tr>
</tbody>
</table>

Table 2: Main constructs and syntax of DL languages [2]
4.3.2 Types of Description Logic

4.3.2.1 SHOIN (D) Description Logic

SHOIN (D) [12] is a type of description logic that provides a high level of expressivity and offers full negation, disjunction within verse roles and a restricted form of the universal form of existential quantification; it is therefore called “concept description”. SHOIN (D) additionally supports reasoning with concrete data-types. At present, OWL DL is correspondent to SHOIN (D). It clear that OWL DL adds very little in expressiveness to OWL Lite.

4.3.2.2 SHIQ (D) Description Logic

A associated logic, SHIQ(D) [12] is distinguished from SHOIN(D) essentially by not supporting nominal concepts (or named objects), allowing a qualified number of restrictions of the concept and simple roles. There is a mapping or translation from DAML+OIL to the SHIQ (D) language.

4.3.2.3 SHIF (D) Description Logic

SHIF (D) [12] is just SHOIN (D) with the exclusion of the oneOf constructor and the inclusion of the (at-least) and (at-most) constructors limited to 0 and 1. In fact, OWL Lite can be translated to SHIF (D) to allow for reasoning. On the other hand, OWL Full is not computationally decidable.

4.4 Base of Ontology Language

Indeed, since saying that description logic is the basis of most ontology language, it is appropriate to explain briefly the base of the description to understand the ontology language clearly. OWL have been chosen because it is almost the last version of these languages, and our research is concerned with this type of RDF.

In particular, description logic named SHOIN (Dn) is used because it is the underlying logic of OWL.

The expressiveness of DLs is limited by the lack of variables. However, this limitation does the following: it ensures decidability, improves tractability and allows for efficient reasoning.

On the other hand, there are some advantages in using DL languages to build ontology. These include the fact that they are well-understood declarative semantics and that there are tested and tried algorithms to verify the completeness and decidability of a number of properties. Many relationships exist between concepts in Description Logics, including subsumption; that is, when a concept X subsumes another concept Y, meaning that X is the more general concept.

Description logics are seen as tools that support the Semantic Web and help to realise its vision. The Semantic Web uses DLs to define, maintain and integrate ontologies. One of the significant uses of ontology is to provide a common understanding of terms between different agents, which is to say to establish a joint terminology between the agents.

DLs have intuitive semantics and syntax which help to communicate the intended meaning, which is why they play a major role in several application areas such as natural language processing and database management. For DL languages to be able to model several application domains, some important features, such as effectiveness and ease of comprehensibility, should appear in all of them.

Table 3 is in three parts, listing the main component of ontology (i.e. class, property and individual) in OWL and DL syntaxes. These tables show how these syntaxes are built on each other in order to describe the feature of the ontology in an accepted way [2].

5. Knowledge Bases and KRS

A Knowledge Base (KB) [3] is a particular and evolved form of information system, which can contain many different conceptualizations of different domains, named ontologies, and which is managed through a Knowledge Representation System (KRS) that gives the facilities for managing and querying the KB, defining new concepts and roles and inferring new knowledge. If a KRS can only query, the KB is called a knowledge inference system or simply a reasoner. A KB is usually constituted by two elements: KB = ⟨Tbox, Abox⟩.

A DL knowledge base is generally composed of an intentional component (TBox), which defines the concepts and roles, and an extensional one (ABox), which defines the membership of individuals and couples to concepts of individual relationships.

TBox [3]: The TBox (terminological box) contains all the concept and role definitions, as well as the axioms of our logical theory (e.g. “A father is a man with a child”). The axioms of a TBox can be divided into definitions (C ≡ D) and subsumptions (C ⊑ D); the former is used to say that a concept C is equivalent to another concept D (atomic or complex), while the latter is used to say that a concept C is a subclass of the concept D.

The TBox contains intentional (terminological) knowledge in the form of a terminology; in other words, it contains the definitions of concepts and roles. Moreover, it is built through declarations that describe general properties of concepts.

ABox [3]: The ABox contains all the assertions (also known as facts) of the logic theory, and an assertion is used to express a property of an individual of the domain (for example “Tom is a father” is represented as Father [Tom]). An assertion is also R (a, b) where R is a role (e.g.
hasFather (James, Tom)). The ABox contains extensional (assertional) knowledge, which is definite and specific to the individuals (instances) of the discourse domain [3].

**Table 3: Main Components of OWL and DL syntax [2]**

<table>
<thead>
<tr>
<th>OWL Syntax (Axioms of Class)</th>
<th>DL Syntax</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class(A partial C₁ ... Cₙ)</td>
<td>A ⊑ C₁ ∩ ... ∩ Cₙ</td>
</tr>
<tr>
<td>Or Class (A)</td>
<td>A</td>
</tr>
<tr>
<td>Class (owl: Thing)</td>
<td>T</td>
</tr>
<tr>
<td>Class (owl: Nothing)</td>
<td>⊥</td>
</tr>
<tr>
<td>Class(A complete C₁ ... Cₙ)</td>
<td>A ≡ C₁ ∩ ... ∩ Cₙ</td>
</tr>
<tr>
<td>Or IntersectionOf(C₁ C₂ ...)</td>
<td>C₁ ∩ C₂</td>
</tr>
<tr>
<td>EnumeratedClass (A a₁ ... aₙ)</td>
<td>A ≡ {a₁} ∪ ... ∪ {aₙ}</td>
</tr>
<tr>
<td>Or OneOf (a₁ a₂ ...)</td>
<td>{a₁} ∪ {aₙ}</td>
</tr>
<tr>
<td>UnionOf(C₁ C₂ ...)</td>
<td>C₁ ∪ C₂</td>
</tr>
<tr>
<td>ComplementOf(C)</td>
<td>¬C</td>
</tr>
<tr>
<td>SubClassOf(C D)</td>
<td>C ⊑ D</td>
</tr>
<tr>
<td>EquivalentClasses(C₁ ... Cₙ)</td>
<td>C₁ = ... = Cₙ</td>
</tr>
<tr>
<td>DisjointClasses(C₁ ... Cₙ)</td>
<td>Cᵢ ≡ ¬Cⱼ, (1 ≤ i &lt; j ≤ n)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Reasoning</th>
</tr>
</thead>
</table>
| An ontology language needs to be based on a logical form to enable inferencing and reasoning. DLs consider a decidable subset of first-order logic with well-understood inference rules, but FOL is not decidable; this decidability is very convenient for reasoning about ontologies. Logics-based language is in fact required to facilitate inferencing and reasoning. Reasoning [2, 19, 23] is used in several development phases to ensure the quality of an ontology. It could be used to check whether concepts are consistent and to obtain implied relations during ontology design. Inference engines such as FaCT and RACER have been successfully implemented in practical reasoning systems to provide reasoning services for DL.
6.1 Reasoning Services

Reasoning services are the tasks of the KRS [2], which can be differentiated into services for the TBox and services for the ABox [3, 23].

The services that involve only the TBox are:

- Subsumption: This task verifies whether a concept C is subsumed by another concept D(TBox $\models C \sqsubseteq D$); this is true if and only if for all the interpretations I there is $(C_I \sqsubseteq D_I)$.

- Consistency: This task verifies that there exists at least one interpretation I for a given TBox (TBox $\not\models \bot$).

The services for the ABox are:

- Local Satisfiability: This task verifies for a given concept C that there exists at least one interpretation in which CI $\not\models \emptyset$.

- Instance Checking: This task verifies whether a given individual x belongs to a particular concept C(ABox $\models C(x)$).

- Instance Retrieval: This task returns the extension of a given concept C, that is, the set of individuals belonging to C.

7. Conclusion

Ontology language is the basis of ontological knowledge systems, the definition of a system of knowledge representation language specification; it not only has a rich and intuitive ability to express and use it, but the body should be easily understood by the computer, processing and applications.

8. References