J. Nagaraju, K. Ravi Kumar / International Journal of Engineering Research and Applications (IJERA) ISSN: 2248-9622 www.ijera.com
Vol. 3, Issue 2, March -April 2013, pp.611-616

An Ontology for Describing Web Services Through WSDL and RDF

J. Nagaraju*, K. Ravi Kumar**
*(Department of Computer Science,Anurag Engineering College, Kodad,Andhra Pradesh,India)
**((Department of Computer Science,Anurag Engineering College ,Kodad,Andhra Pradesh,India)

ABSTRACT

Now a days web has become an important resource for knowledge.To provide many services to the users to acquire data easily and efficiently web services are being described using ontologies which provide a set of rules to be used easily and understood preferably.The mostly used languages used for describing web services is WSDL.In this paper we have described about WSDL and RDF using graphs to describe ontologies.

Keywords-**ontology, WSDL ,RDF, web services**

I. Introduction

In computer science and information science, an ontology formally represents knowledge as a set of concepts within a domain, and the relationships between pairs of concepts. It can be used to model a domain and support reasoning about entities .In theory, an ontology is a "formal, explicit specification of a shared conceptualisation".[1] An ontology renders shared vocabulary and taxonomy which models a domain with the definition of objects and/or concepts and their properties and relations.[2]Ontologies are the structural frameworks for organizing information and are used in artificial intelligence, the Semantic Web, systems engineering, software engineering, biomedical informatics, library science, enterprise bookmarking, and information architecture as a form of knowledge representation about the world or some part of it.

The creation of domain ontologies is also fundamental to the definition and use of an enterprise architecture framework.Historically, ontologies arise out of the branch of philosophy known as metaphysics, which deals with the nature of reality – of what exists. This fundamental branch is concerned with analyzing various types or modes of existence, often with special attention to the relations between particulars and universals, between intrinsic and extrinsic properties, and between essence and existence. The traditional goal of ontological inquiry in particular is to divide the world "at its joints" to discover those fundamental categories or kinds into which the world's objects naturally fall.[5]During the second half of the 20th century, philosophers extensively debated the possible methods or approaches to building ontologies without actually *building* any very elaborate ontologies themselves.

By contrast, computer scientists were building some large and robust ontologies, such as WordNet and Cyc, with comparatively little debate over *how* they were built.Since the mid-1970s, researchers in the field of artificial intelligence (AI) have recognized that capturing knowledge is the key to building large and powerful AI systems. AI researchers argued that they could create new ontologies as computational models that enable certain kinds of automated reasoning. In the 1980s, the AI community began to use the term *ontology* to refer to both a theory of a modeled world and a component of knowledge systems.

Some researchers, drawing inspiration from philosophical ontologies, viewed computational ontology as a kind of applied philosophy.[6]In the early 1990s, the widely cited Web page and paper "Toward Principles for the Design of Ontologies Used for Knowledge Sharing" by Tom Gruber[7] is credited with a deliberate definition of *ontology* as a technical term in computer science. Gruber introduced the term to mean a specification of a conceptualization:"An ontology is a description (like a formal specification of a program) of the concepts and relationships that can formally exist for an agent or a community of agents.

This definition is consistent with the usage of ontology as set of concept definitions, but more general. And it is a different sense of the word than its use in philosophy.According to Gruber (1993):"Ontologies are often equated with taxonomic hierarchies of classes, class definitions, and the subsumption relation, but ontologies need not be limited to these forms. Ontologies are also not limited to conservative definitions — that is, definitions in the traditional logic sense that only introduce terminology and do not add any knowledge about the world.[9] To specify a

conceptualization, one needs to state axioms that do constrain the possible interpretations for the defined terms.

II. Ontology As Vocabulary

In philosophy, ontology is the study of the kinds of things that exist. It is often said that ontologies "carve the world at its joints." In AI, the term ontology has largely come to mean one of two related things. First of all, ontology is a representation vocabulary, often specialized to some domain or subject matter. More precisely, it is not the vocabulary as such that qualifies as an ontology, but the conceptualizations that the terms in the vocabulary are intended to capture. Thus, translating the terms in an ontology from one language to another, for example from English to French, does not change the ontology conceptually. In engineering design, you might discuss the ontology of an electronic-devices domain, which might include vocabulary that describes conceptual elements—transistors, operational amplifiers, and voltages—and the relations between these elements—operational amplifiers are a *type-of* electronic device, and transistors are a *component-of* operational amplifiers.

Identifying such vocabulary—and the underlying conceptualizations—generallyrequires careful analysis of the kinds of objects and relations that can exist in the domain. In its second sense, the term ontology is sometimes used to refer to a body of knowledge describing some domain, typically a commonsense knowledge domain, using a representation vocabulary.

For example, CYC1 often refers to its knowledge representation of some area of knowledge as its ontology. In other words, the representation vocabulary provides a set of terms with which to describe the facts in some domain, while the body of knowledge using that vocabulary is a collection of facts about a domain.

However, this distinction is not as clear as it might first appear. In the electronic-device example, that transistor is a *component-of* operational amplifier or that the latter is a *type-of* electronic device is just as much a fact about its domain as a CYC fact about some aspect of space, time, or numbers. The distinction is that the former emphasizes the use of ontology as a set of terms for representing specific facts in an instance of the domain, while the latter emphasizes the view of ontology as a general set of facts to be shared. There continues to be inconsistencies in the usage of the term ontology. At times, theorists use the singular term to refer to a specific set of terms meant to describe the entity and relation-types in some domain.

Thus,we might speak of *an ontology* for "liquids" or for "parts and wholes." Here, the singular term stands for the entire set of concepts and terms needed to speak about phenomena involving liquids and parts and wholes. When different theorists make different proposals for an ontology or when we speak about ontology proposals for different domains of knowledge,we would then use the plural term *ontologies* to refer to them collectively.

In AI and information-systems literature, however, there seems to be inconsistency: sometimes we see references to "*ontology* of domain" and other times to "*ontologies* of domain," both referring to the set of conceptualizations for the domain. The former is more consistent with the original (and current) usage in philosophy.

III. Ontology As Content Theory

The current interest in ontologies is the latest version of AI's alternation of focus between content theories and mechanism theories. Sometimes, the AI community gets excited by some mechanism such as rule systems, frame languages, neural nets, fuzzy logic, constraint propagation, or unification. The mechanisms are proposed as the secret of making intelligent machines. At other times,we realize that, however wonderful the mechanism, it cannot do much without a good content theory of the domain on which it is to work. Moreover, we often recognize that once a good content theory is available, many different mechanisms might be used equally well to implement effective systems, all using essentially the same content.

AI researchers have made several attempts to characterize the essence of what it means to have a content theory. McCarthy and Hayes'theory (epistemic versus heuristic distinction), 3 Marr's three-level theory (information processing, strategy level, algorithmsand data structures level, and physical mechanisms level),4 and Newell's theory (Knowledge Level versus Symbol Level)5 all grapple in their own ways with characterizing content. Ontologies are quintessentially content theories, because their main contribution is to identify specific classes of objects and relations that exist in some domain. Of course, content theories need a representation language. Thus far, predicate calculuslike formalisms, augmented with *type-of* relations (that can be used to induce class hierarchies), have been most often used to describe the ontologies themselves.

3.1 Use Of Ontology

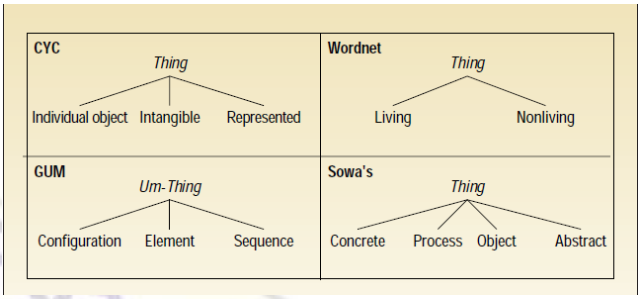
Ontological analysis clarifies the structure of knowledge. Given a domain, its ontology forms the heart of any system of knowledge representation for that domain. Without ontologies, or the

conceptualizations that underlie knowledge, there cannot be a vocabulary for representing knowledge. Thus, the first step in devising an effective knowledgerepresentation system, and vocabulary, is to perform an effective ontological analysis of the field, or domain. Weak analyses lead to incoherent knowledge bases. An example of why performing good analysis is necessary comes from the field of databases.6 Consider a domain having several classes of people (for example, students, professors, employees, females, and males).

This study first examined the way this database would be commonly organized: students, employees, professors, males, and female would be represented as *types-of* the class *humans*. However, some of the problems that exist with this ontology are that students can also be employees at times and can also stop being students. Further analysis showed that the terms *students* and *employee* do not describe categories of humans, but are roles that humans can play, while terms such as *females* and *males* more appropriately represent subcategories of humans. Therefore, clarifying the terminology enables the ontology to work for coherent and cohesive reasoning purposes. Second, ontologies enable knowledge sharing. Suppose we perform an analysis and arrive at a satisfactory set of conceptualizations, and their representative terms, for some area of knowledge—for example, the electronic-devices domain. The resulting ontology would likely include domain-specific terms such as *transistors* and *diodes*; general terms such as *functions*, *causal processes*, and *modes*; and terms that describe behavior such as *voltage*.

The ontology captures the intrinsic conceptual structure of the domain. In order to build a knowledge representation language based on the analysis, we need to associate terms with the concepts and relations in the ontology and devise a syntax for encoding knowledge in terms of the concepts and relations. We can share this knowledge representation language with others who have similar needs for knowledge representation in that domain, thereby eliminating the need for replicating the knowledge-analysis process. Shared ontologies can thus form the basis for domain-specific knowledge-representation languages.

In contrast to the previous generation of knowledge-representation languages (such as KL-One), these languages are *content-rich*; they have a large number of terms that embody a complex content theory of the domain. Shared ontologies let us build specific knowledge bases that describe specific situations. For example, different electronicdevices manufacturers can use a common vocabulary and syntax to build catalogs that describe their products. Then the manufacturers could share the catalogs and use them in automated design systems. This kind of sharing vastly increases the potential for knowledge reuse.



In AI, knowledge in computer systems is thought of as something that is explicitly represented and operated on by inference processes. However, that is an overly narrow view. All information systems traffic in knowledge. Any software that does anything useful cannot be written without a commitment to a model of the relevant world—to entities, properties, and relations in that world. Data structures and procedures implicitly or explicitly make commitments to a domain ontology. It is common to ask whether a payroll system"knows" about the new tax law, or whether a database system "knows" about employee salaries. Information-retrieval systems, digital libraries, integration of heterogeneous information sources, and Internet search engines need domain ontologies to organize information and direct the search processes. For example, a search engine has categories and subcategories that help organize the search.

The search-engine community commonly refers to these categories and subcategories as ontologies. Object-oriented design of software systems similarly depends on an appropriate domain ontology. Objects, their attributes, and their procedures more or less mirror aspects of the domain that are relevant to the application. Object systems representing a useful analysis of a domain can often be reused for a different application program.Object systems and ontologies emphasize different aspects, but we anticipate that over time convergence between these technologies will increase. As information systems model large knowledge domains, domain ontologies will become as important in general software systems as in many areas of AI.

In AI,while knowledge representation pervades the entire field, two application areas in particular have depended on a rich body of knowledge. One of them is natural-language understanding. Ontologies are useful in NLU in two ways. First, domain knowledge often plays a crucial role in disambiguation.

A welldesigned domain ontology provides the basis for domain knowledge representation. In addition, ontology of a domain helps identify the semantic categories that are involved in understanding discourse in that domain. For this use, the ontology plays the role of a concept
dictionary.

IV. RDF

The Resource Description Framework is an extensible infrastructure to express, exchange and re-use structured metadata [Mil98]: "Everything is URI" Information resources are commonly identified by Uniform Resource Identifiers (URIs). By generalizing the concept of "resource", whatever is identifiable by an URI can be described in RDF. In this way, URIs can be assigned to anything, even physical objects, living beings,abstract concepts, etc. It is important to note that the identifiability does
not imply retrievability of the resource. The principal advantages of this approach are that URIs are a globally unambiguous way to reference resources, and that no centralized authority is necessary to provide them. A common way to abbreviate it is the XML qualified name (or QName) syntax of the form prefix:suffix. For example, an URI such as http://www.w3.org/TR/rdf-primer/ would be written as w3:rdf-primer/ if it has been agreed that w3 stands for http://www.w3.org/TR/. RDF Statements The atomic structure for RDF specifications is the statement, which is a <subject predicate object>-triple. The information re- 1 For a more thorough introduction see the "RDF Primer" [MM04] or other references given in subsection A.1.3 38 3. Preliminaries source being described is the subject of the statement and is denoted by an URI. The predicate of a statement is an URI reference representing a property, whose property value appears as the statement object. The property value can be a resource as well as a literal value. A literal is a string (e.g., a personal name) of a certain datatype and may only occur as the object of a statement. RDF triples can be visualized as a directed labeled graph, __ __ __subject__ predicate /__ ____object__in which subjects and objects are represented as nodes, and predicates as arcs.
In [KC04] a drawing convention is given—which will be neglected in this document. According to this convention nodes representing literals are drawn as rectangles and nodes representing URIs as ovals. In the drawings of this study, however, we need not to make these distinctions as we equally treat them as nodes. For an example of a graph drawn following that convention.

RDF Graph A set of RDF statements is an RDF Graph. For example,

<wos:texbook dc:Creator wos:knuth>
<wos:texbook dc:Title "The TEXbook">
<wos:knuth foaf:name "Donald Knuth">
form an RDF Graph of three statements2. The TEXbook by Knuth is represented by the URI wos:texbook (wos is the namespace prefix of an—imaginary—"Web of Scientists" vocabulary, which shall be presented later in this chapter) and described in two statements. The object of the first statement, wos:knuth, is an URI representing the

4.1 The Resource Description Framework

The meaning of this RDF Graph is "an information resource, identified by wos:texbook, has the title "The TEXbook" and was created by something which is identified by wos:knuth and whose name is "Donald Knuth". Anonymous Resources There are situations in which we wish to describe information using more complex structures of data than using a literal string or an URI pointer. For this, "anonymous" resources are used: the object of
a statement can be an anonymous resource—or a blank node—which itself is the subject of other statements. Such a resource is represented by a blank node identifier, which is usually denoted as :n, with n being an integer. For example, a more sophisticated version of the above example about Knuth's authorship of the TEXbook would be
<wos:texbook dc:Creator :1>
<wos:texbook dc:Title "The TEXbook">
< :1 foaf:name "Donald Knuth">
< :1 rdf:type xy:Person>
< :1 wos:described wos:knuth>
which states more clearly that the author of the TEXbook is a human, which has a personal name and is further described in another resource (wos:knuth).

It is important to note that the blank node identifiers carry no meaning; they are used merely for the purpose of serialization (e.g., file storage). RDF Concepts We can now state more formally the triples which are syntactically correct: let "uris" be the set of URIs, "blanks" the set of blank node
identifiers, and "lits" the set of possible literal values of whatever datatype (we consider all these sets as infinite). Then (s, p, o) 2 (uris[blanks) × (uris) × (uris[blanks [lits) is an RDF statement. Observe that there is no restriction to what URIs may appear as statement property. We say that x is a resource if x 2 uris[blanks, and everything occurringin an RDF statement is a value (x 2 uris[blanks [lits). In this document, most of the time it will be referred to values because the type—URI, blank, literal—is not of interest. To recall, a RDF Graph T is a set of RDF statements (T abbreviates triples). A subgraph of T is a subset of T. A ground RDF Graph is an RDF Graph without blank nodes

With univ(T) we denote the set of all values occurring in all triples of T and call it the universe of T; and vocab(T), the vocabulary of T, is the set of all values of the universe that are not blank nodes. The size of T is the number of statements it contains and is denoted by |T|. With subj(T) (respectively pred(T), obj(T)) we designate all values which occur as subject
(respectively predicate, object) of T.
Let V be a set of URIs and literal values. We define RDFG(V) := { T : T is RDF Graph and vocab(T) _ V } i.e. the set of all RDF Graphs with a vocabulary included in V .Let M be a map from a set of blank nodes to some set of literals, blanknodes and URI references; then any RDF Graph T0 obtained from the RDF Graph T by replacing some or all of the blank nodes N in T by M(N) is an instance of T.
Consider an RDF Graph T1, and a bijective map M : B1 ! B2 which replaces blank node identifiers of T1 with other blank node identifiers. Then T2 = M(T1) is an instance of T1, and T1 is an instance of T2 (by the inverse of M which is trivially defined). Two such RDF Graphs are considered as equivalent.
Equivalent RDF Graphs are treated as identical RDF Graphs, which is in conformance with the notion of blank nodes as "anonymous resources" Reasons for Graph Representation of RDF Graphs are mathematical objects which enjoy wide-spread usage for many tasks, which include the visualization and analysis of data for humans, mathematical reasoning, and the implementation as a data structure for developing software. These tasks are relevant in the context of RDF data as well, as this section shall present.

4.2 Motivation

4.2.1 Fixing the Specification

The first specification of RDF in the status of a WWW Consortium Recommendationappeared in 1999 [LS99]. Since then, it has taken five years to revise the original specification and to replace it by a suite of six documents which gained recommendation status just recently, in February 2004 [MM04, KC04, Hay04, GB04, Bec04, BG04]. The success of RDF appears to take place at a rather modest pace, and one is tempted to conclude that the arduously advancing process of specification is one reason for this. The fact that the 2004WWWConsortium Recommendation still contains ambiguities as described above gives motivation to supply a constructive critique and a proposition for refinement, with the hope to contribute to future revisions of the specification.
The issue—an incomplete definition of a graph representation, and a representation with certain limitations—might appear trivial. However, it has considerable impact: Numerous publications, including tutorials, exist which claim that RDF "is" a directed labeled graph. The immediate result is the—artificial—distinction between resources and properties which many people make.

This prevents users from recognizing the actual simplicity of the RDF model. The results of the understanding of RDF bounded by the directed labeled graph model becomes especially evident in the limitations of current RDF query languages as studied in [AGH04].

4.3 Graphs as a Concept of Human Understanding

Graphs are a successful method to visualize and understand complex data. RDF, as a language developed to annotate and describe information resources and their relations among each other, allows the expression of potentially highly interconnected collections of metadata assertions.
For the visualization of RDF data directed labeled graphs may be employed successfully for not too complex RDF Graphs. Also, to explain the RDF model it is natural to use graphs. While the examples provided, e.g., in the RDF Primer [MM04] are simple and therefore above-mentioned limitations of directed labeled graphs are not as relevant, care should be taken that the (abstract) graph nature of RDF, the well-defined concept of RDF Graph and the representation of RDF as directed labeled graph are not confused

V. Conclusion

Here by we can conclude that RDF and WSDL help us to describe ontologies. Moreover RDF using graphs creates a better understanding of ontology to humans and provides good visualization of web services

References

[1] Adobe Systems Incorporated. PDF Reference, Version World Wide Web, http://partners.adobe.com/asn/tech/pdf/specifications.jsp, 2003. 2
[2] [Ado04] Adobe Developer Technologies. XMP - Extensible Metadata Platform. World Wide Web,http://www.adobe.com/products/xmp/pdfs/xmpspec.pdf, 2004. 2
[3] ttp://www.openrdf.org/doc/users/userguide.html, 2004. 10.2.3,
[4] Renzo Angles, Claudio Gutierrez, and Jonathan Hayes. RDF Query Languages Need Support for Graph Properties. Technical Report TR/DCC-2004-3, Universidad de Chile,

http://www.dcc.uchile.cl/_cgutierr/ftp/graphproperties.pdf, 2004. 4.3.1, 4.3.5, 10.3.1, 3

[5] Rakesh Agrawal. Alpha: An Extension of Relational Algebra to Express a Class of Recursive Queries. IEEE Trans. Softw. Eng.,14(7):879–885, 1988. 9.3.3

[6] Joshua Allen. Making a Semantic Web. World Wide Web, www.netcrucible.com/semantic.html, 2001. 2

[7] Boanerges Aleman-Meza, Chris Halaschek, Ismailcem Budak Arpinar, and Amit P. Sheth. Context-Aware Semantic Association Ranking. In I. F. Cruz, V. Kashyap, S. Decker, and R. Eckstein, editors, Proceedings of SWDB'03, 2003.

[8] D.B. Lenat and R.V. Guha, Building Large Knowledge-Based Systems: Representation and Inference in the CYC Project, Addison-Wesley, Reading, Mass., 1990.

[9] B. Chandrasekaran, "AI, Knowledge, and the Quest for Smart Systems," IEEE Expert,Vol. 9, No. 6, Dec. 1994, pp. 2–6

.[10] J. McCarthy and P.J. Hayes, "Some Philosophical Problems from the Standpoint of Artificial Intelligence," Machine IntelligenceVol. 4, B. Meltzer and D. Michie, eds., Edinburgh University Press, Edinburgh, 1969, pp.463–502.