ISSN: 2278 – 1323
International Journal of Advanced Research in Computer Engineering & Technology
Volume 1, Issue 4, june2012

SHOE: A Platform for Semantic Web Language Usage and Analysis

Chikkala Jayaraju, Pedasanaganti.Divya, Epuru Madhavarao ,ASK Ratnam

Abstract— **The World Wide Web before Web 2.0 is an immense search for expertise has superior in modern years. Although in the sequence resource with near unlimited potential for the reason that there are tranquil many types of search that return inadequate outcome . In this era can be to a immense extent improved if web pages use a semantic markup language to portray their content in semantic web before web 3.0. We have look at the thoughts about SHOE, a language for this intention toting up to specifying the semantics of a set of vocabulary, the ontologies can pull out or revise one another. In this paper we describe a circumstances for how the language could be used by search perpose of the upcoming. A crucial pace to this system is constructing an inquiry tool. In this perpose we current the some of SHOE crucial search tackle that can capture improvement of the power of a knowledge base while tranquil being easy enough for the informal user.**

Index Terms— **WorldWideWeb, SHOE, Semantic web, Ontologies, Semantic markup language.**

I. Introduction

The SemanticWeb offers a hopeful explanation to publishing in sequence and services on the World Wide Web augmented with similes in a form that is easier for machines to process and understand. The main pillar sustaining the development of what we understand by SemanticWeb[1] –"the conceptual structuring of the web in an explicit machine-readable way"(Berners-Lee). Information published in the Semantic Web languages uses terms denoting classes and properties drawn from one or more ontologies. These ontologies are online RDF documents that declare a set of terms with unique URIs and further define them by asserting logical relationships and constraints ontology language for the Web. SHOE is used to develop extensible shared among them. The potential of the Semantic Web is demonstrated using SHOE a archetype ontologies and create assertions that commit to particular ontologies. SHOE can be compact to datalog, allowing it to scale to the extent allowed by the optimized algorithms residential for deductive databases. To demonstrate the feasibility of the SHOE approach, we describe a basic architecture for a SHOE system and a suite of general purpose tools that allow SHOE to be created, discovered and queried. SHOE which stands for Simple HTML Ontology Extensions, was originally developed by the PLUS cluster at the University of Maryland. Since then the PLUS cluster SHOE combines facial appearance of markup languages, knowledge representation, datalog, and ontologies in an endeavor to address the unique problems of semantics on the Web. It supports knowledge gaining by augmenting the Web with tags that provide semantic significance. The basic structure consists of ontologies, which define rules that guide what kinds of assertions may be made and what kinds of conclusions may be drawn from these assertions, and instances that make assertions based on those rules. As a knowledge representation language, SHOE borrows characteristics from both predicate logics and framework systems.

II.A BASIC ARCHITECTURE

Figure 1 depicts a various requirements and choices for a SHOE system, and general architecture. The foundation of any SHOE architecture depends on the existence of web pages that contain SHOE instances,where each instance commits to one or more SHOE ontologies[2] that are also available on the Web. A number of architectures can be built around this concept, with different sets of tools for producing and consuming this data. This idea is influenced by the design of the Web, where HTML is a lingua[3]franca that is produced by text editors, web page editors, and databases; and processed by web browsers, search engines, and other programs.We describe a basic architecture that was investigated extensively in this thesis. In this architecture, a number of tools can be used to create SHOE web pages. These tools include text editors, the Knowledge Annotator, Running SHOE and possibly other domain specific tools. For efficiency reasons, the assertions are gathered from the web pages by a web crawler called Expos´e and stored in a knowledge base. The specific knowledge base system used can vary depending on the needs of the application, and multiple knowledge base systems can be used simultaneously. Finally, a number of front-ends, including SHOE Search domain specific tools, or KB specific tools can be used to query the data. The generic SHOE tools are discussed extensively in this paper.

Manuscript received May, 2012*.*

Chikkala jayarajue*, Pursuing M.Tech(CSE), Vignan's Lara Institute of Technology and Science, Vadlamudi,, (e-mail: glorious.ch@gmail.com). Guntur, India, Mobile No:9493552307.*

Pedasanaganti.Divya,*, Pursuing M.Tech(CSE),Vignan's Lara Institute of Technology and Science, Vadlamudi,,(e-mail: divya.lukky522 @gmail.com). Guntur, India, Mobile No:7799468179.*

Epuru Madhavarao, *Pursuing M.Tech(CSE),Vignan's Lara Institute of TechnologyandScience,Vadlamudi,(e-mail:madhavarao.epurru916@gmail. com),Gunture,India,Mobile o:9848510397.*

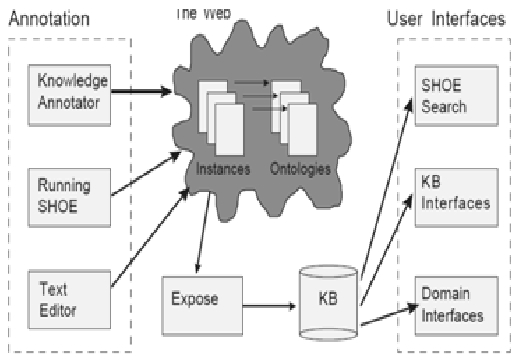

Figure 1 A basic Archetecture of SHOE

III.THE SHOE LANGUAGE

we are preseant SHOE, an actual Web language based on these concepts. SHOE which stands for Simple HTML Ontology Extensions was originally developed by the PLUS Group at the University of Maryland. SHOE combines features of markuplanguages[4], knowledgerepresentation, we are preseant SHOE, an actual Web language based on these concepts. SHOE which stands for Simple HTML Ontology Extensions was originally developed by the PLUS Group at the University of Maryland. SHOE combines features of markuplanguages[4], knowledgerepresentation, datalog, and ontologies in an attempt to address the unique problems of semantics on the Web. It supports knowledge acquisition by augmenting the Web with tags that provide semantic meaning. The basic structure consists of ontologies, which define rules that guide what kinds of assertions may be made and what kinds of conclusions may be drawn from these assertions, and instances that make assertions based on those rules. As a knowledge representation language, SHOE borrows characteristics from both predicate logics and frame systems. We chose to keep SHOE easy to understand and implement, and have carefully designed the language to eliminate the possibility of contradictions between agent assertions.
SHOE does this in four ways:

- SHOE only permits assertions, not retractions.
- SHOE does not permit logical negation.
- SHOE does not allow relations to specify a cardinality, and thus limit how many relation assertions of a particular kind can be made for any single insta
- SHOE does not permit the specification of disjoint classes.

LanguageDescription: which provides a way to incorporate machine-readable semantic knowledge in World Wide Web documents. We describe the syntax and semantics of ontologies.
Syntax of SHOE: SHOE has two syntactical variations. The first syntax is an SGML application that extends the HTML syntax with additional semantic tags. This syntax can be used to embed SHOE in ordinary web documents. To indicate conformance with SHOE, HTML documents must include the following text in the HEAD section of the document:

```
<METAHTTP-EQUIV="SHOE"
CONTENT="VERSION=1.0">
```

The syntactic descriptions in these sections use a sans serif font to indicate key words of the language and *italics* to indicate that the author must supply a parameter or expression. The second SHOE syntax is an XML application. While the SGML syntax allows SHOE to be easily embedded in the numerous existing HTML web pages, the XML syntax allows SHOE to leverage emerging web standards and technologies. Since XML is basically a subset of SGML, the XML syntax for SHOE is very similar to the SGML[5] one.The XML version of SHOE can either stand alone, or be included in another XML document. A stand-alone SHOE XML document must begin with the appropriate XML prolog:
<?xml version="1.0"?>
<!DOCTYPE shoe SYSTEM "http://www.cs.umd.edu/projects/plus/SHOE/shoe_xml.dtd" >
The root element of this document must be shoe, and it must contain a version attribute with value “1.0”, as shown below: <shoe version="1.0"> All SHOE elements must be between this tag and a closing </shoe> tag. the elements of SHOE using the SGML syntax, most of it is still applicable to the XML variation. However, since XML is more restrictive, the following additional rules must be
applied:

- All empty elements, i.e., elements which have no content and no end tag, must end with a ’/>’ instead of a ’>’. Specifically, this applies to the USE-ONTOLOGY, DEF-CATEGORY, DEF-ARG, DEF-RENAME,DEF-CONSTANT, DEF-TYPE, CATEGORY, and ARG elements.
- No attribute minimization is allowed. In SGML attribute minimization allows the names of some attributes to be omitted and only their values to be specified. In the SGML syntax, this is usually used to specify VAR within the subclauses of an inference rule. In the XML syntax, the attribute name USAGE must be explicitly provided, e.g. USAGE=”VAR” instead of VAR.
- Since XML is case-sensitive, all element and attribute names must be in lowercase.
- All attribute values must always be quoted, including those which are numeric as well as the FROM and TO keywords.

A. *The Base Ontology*

The base ontology is the ultimate ancestor of all other ontologies. There is a one-to-one correspondence between versions of the SHOE language and versions of the base ontology, thus the version number of the META tag or the version attribute of the shoe element indicates which version of the base ontology is applicable. The base ontology provides some fundamental categories, relations, and data types. It defines the categories Entity and SHOEEntity, where the latter is a subcategory of the former. SHOEEntity is the superclass of all classes defined in other SHOE ontologies. The relations name and description can be used to provide names and definitions for any instance.

B.Ontology Definitions:

SHOE uses ontologies to define the valid elements that may be used in describing instances. An ontology is stored in an HTML or XML file and is made available to document authors and SHOE agents by placing it on a web server. The ontology can include tags that state which ontologies are extended, and define the various elements of the ontology, such as categories, relations, and inference rules. Figure II shows basic form of a SHOE ontology.Each ontology has an identifier and a version number that uniquely defines it. Accidental reuse of ontology identifiers can be avoided by including the domain name of its author in this identifier. Ontologies with the same identifier but different versionnumbers are considered to be different versions of the same ontology. An ontology is a revision of another ontology if it has the same identifier but a later version number.A SHOE document may contain any number of ontology definitions. Many of the
definitions within an ontology have an associated name, and these are collectively called named components. The named components are categories, relations,constants, and types. The names of these components are all subject to the same restrictions: they must begin with a letter, may only contain letters, digits, and hyphens; cannot contain whitespace, and are case-sensitive. There is a singlenamespace for all named components, and thus it is invalid for an ontology to define two components that have the same name.
In HTML documents, the ONTOLOGY element must be a subelement of the BODY element; in XML documents, it must be a subelement of the shoe element. Only the SHOE elements described in the rest of this section may be nested in the ONTOLOGY element. An ONTOLOGY element has the following form:

< ONTOLOGY ID="*id*"
VERSION="*version*"
[BACKWARD-COMPATIBLE-WITH="*bcw1 bcw2 bcw3……bcwn*"]
[DESCRIPTION="*text"*]
[DECLARATORS="*dec1 dec2 dec3…..decm*"] >
content
</ONTOLOGY>

Figure 2 A basic form of SHOE ontology

Id **:** Specifies the ontology's identifier. This must begin with a letter,contain only letters, digits, and hyphens; and may not contain whitespace.Identifiers are case-sensitive.

version:Specifies the ontology's version number. Version numbers may only contain digits and dots.

Backward-Compatible-with: Specifies a whitespace-delimited list of previous versions that this ontology subsumes. Each bcwi must be a valid ontology version number.

Description**:** A short, human-readable description of the purpose of the ontology.

Declarators**:** Specifies a whitespace-delimited list of URLs for content resources the ontology has associated with itself. Ordinarily, an ontology cannot assert relationships or categorizations, only define the rules thatgovern such assertions. This mechanism allows an ontology to state that one or more resources contain important standard assertions associated with the ontology.

In this paper we also express the some of very important tools for improve the SemanticWeb[6] Language Usage and Analysis.These tools are mentioned below.

A.Knowledge Annotator
B.Expos´e
C.The SHOE KB Library
D.XSB
E.Parka

A.*Knowledge Annotator:*

The Knowledge Annotator is a tool that makes it easy to add SHOE knowledge[7] to web pages by making selections and filling in forms. The tool has an interface that displays instances, ontologies, and assertions . A variety of methods can be used to view the knowledge in the document. These include a view of the source HTML, a logical notation view, and a view that organizes assertions by subject and describes them using simple English. The Annotator can open documents from the local disk or the Web. These are parsed using the SHOE library, and any SHOE instances contained in the document are displayed in the instances panel. When an instance is selected, the ontologies it commits to and its assertions are displayed in the other panels. Instances can be added, edited and deleted. When adding an instance, the user must specify its key and an optional name. If desired, the name can be extracted from the document's TITLE .

B.*Expos´e:*

After SHOE content has been created it can be accessed by Expos´e, a web-crawler that searches for web pages with SHOE markup. Expos´e stores the knowledge it gathers in a knowledge base, and thus can be used as part of a repository-based system. The web-crawler is initialized by specifying a starting URL, a repository, and a set of constraints on which web sites or directories it may visit. Expos´e can either build a new repository of SHOE information or revisit a set of web pages to refresh an existing repository. A web-crawler essentially performs a graph traversal where the nodes are web pages and the arcs are the hypertext links between them. Expos´e maintains an open list of URLs to visit, and a closed list of URLs that have already been visited. When visiting web pages, it follows standard web robot etiquette by not requesting pages that have been disallowed by a server's robot.txt file and by waiting 30 seconds between page requests, so as not to overload a server . Expos´e's display shows each URL that has been requested, the timestamp of the request, and
the status of the request. The tool also keeps track of the total number of page requests, how many of the pages have SHOE on them, and how many requests resulted in errors. When

Expos´e loads a web page, it parses it using the SHOE library, identifies all of the hypertext links, category instances, and relation arguments within the page, and evaluates each new URL as above. Finally, the agent uses the SHOE KB Library API to store SHOE category and relation assertion in a specified knowledge base. This API, described in the next section, makes it easy to adapt Expos´e for use with different knowledge bases.

C.The SHOE KB Library:

The SHOE KB library is a Java package that provides a generic API for storing SHOE data and accessing a query engine. Applications that use this API can be easily modified to use a different reasoning system, thus allowing them to execute in a different portion of the completeness / execution time tradeoff space. ShoeKb, the main class of the SHOE KB Library API contains methods for storing ontologies, storing SHOE document data, and issuing queries to a repository. It maintains a catalog of all ontologies stored in the KB, and provides a renaming method that distinguishes between components defined in different ontologies. When storing document data, it deletes any assertions that were in a previous version of the
document, since they are no longer reflected on the Web. The class also allows a default ontology to be specified, which is used to disambiguate query predicates and identify the basis of the query's perspective.The ShoeKb[8] class also provides the option to forward-chain a special kind of inference. SHOE's formal semantics state that if an instance appears in an argument of a relation which is of an instance type, then a logical consequence of the assertion is that the instance is a member of the required category. However, this can result in the addition of a large number of rules to the KB. Therefore, the library supports the option to forward-chain these particular consequences and explicitly store them in the knowledge base.

D.XSB:

XSB [9] is an open source, logic programming system that can be used as a deductive database engine. It is more expressive than datalog, and thus can be used to completely implement SHOE. XSB's syntax is similar to that of Prolog, but while Prolog will not terminate for some datalog programs, XSB uses tabling to ensure that all datalog programs terminate. We will now describe how compatible ontology perspectives can be implemented in XSB. In this approach, the SHOE ontologies and instances are translated into an XSB program that can evaluate contextualized queries. Although this discussion is specific to XSB, with minor modifications the approach can be applied to other logic programming systems. First, we must consider a naming convention for predicates in the deductive database. Since two ontologies may use the same term with different meanings, the definitions of such terms could interact in unintended ways, and thus a renaming must be applied . A simple convention is to assign a sequence number to each ontology and append this number to each term defined by the ontology. The sequence numbers can be stored in the deductive database using a ontSeq/2 predicate which associates an ontology with its sequence number.s.

*E.Parka***:**

Parka [10] is a high-performance knowledge representation system whose roots lie in semantic networks and frame systems. It is capable of performing complex queries over very large knowledge bases in less than a second . For example, when used with a Unified Medical Language System (UMLS) knowledge base consisting of almost 2 million assertions, Parka was able to execute complex recognition queries in under 2 seconds. One of Parka's strong suits is that it can work on top of relational database systems, taking advantage of their transaction guarantees while still performing very fast queries. Parka represents the world with categories, instances, and n-ary predicates and can infer category membership and inheritance. It includes a built-in subcategory relation between categories (called *Isa*) and a categorization relation between an instance and a category (called *instance of).* It also includes a predicate for partial string matching, and a number of comparison predicates. Parka can support SHOE's most widely-used semantics directly. As with XSB, a renaming must be applied to guarantee the uniqueness of terms defined in different ontologies. The standard category axioms can be represented by Parka's *Isa* relation, the membership of an instance in a category can be represented by the *instance of* predicate, and all relation definitions can be accommodated by defining new predicates. Additionally, the Parka version of the SHOE KB API automatically computes and explicitly stores the inferred types for certain relations. A Parka knowledge base can be updated dynamically, which is advantageous for a system that must mirror the rapidly changing Web. In order to provide the best possible snapshot of the Web, the knowledge base must delete assertions that no longer appear in a resource. Thus a Parka knowledge base is best used to represent a single extended ontology perspective, preferably based on many ontologies.

IV.CONCLUSIONS:

A significant issue for theSemanticWeb is establishing the identity of individuals. The approach taken by SHOE is to choose a unique URL for each individual. In many cases, such as when the individual is a person or organization with a homepage, this is an acceptable solution. However, some people have multiple homepages (such as a personal page and professional page), and the URLs of a person can change over time (if the person changes jobs or internet service providers). SHOE and its big brother the HTML, have been presented as highly scalable OWLwhich provide substantial analytics capability. SHOE is an important entrant to the field of Semantic Web databases due to the richness of the interactive usage of Language compared to its competitors XML and the newly defined SGML . SHOE has been successfully fielded in commercial, government and academic settings. SHOE appears to be a reliable base upon which to build Semantic Web applications that require scaling and other.

REFERENCES:

[1] T. Berners-Lee and D. Connolly. *Hypertext Markup Language - 2.0*. IETF HTML Working Group, November 1995. At: http://www.ics.uci.edu/pub/ietf/html/rfc1866.txt.

[2] D. Freitag. Information extraction from HTML: Application of a general machine learning approach. In Proc. of Fifteenth American Association for Artificial Intelligence Conference (AAAI-98), pages 517–523, Menlo Park, CA, 1998. AAAI/MIT Press.

[3] Adam Farquhar, Richard Fikes, and James Rice. Tools for assembling modular ontologies in Ontolingua. In Proc. of Fourteenth American Association for Artificial Intelligence Conference (AAAI-97), pages 436–441, Menlo Park, CA, 1997. AAAI/MIT Press.

[4] T. Bray, J. Paoli, and C. Sperberg-McQueen. *Extensible Markup Language (XML)*. W3C (WorldWide Web Consortium), February 1988. At: http://www.w3.org/TR/1998/REC-xml-19980210.html.

[5] ISO (International Organization for Standardization). ISO 8879:1986(E). information processing – text and office systems – Standard Generalized Markup Language (SGML), 1986.

[6] J. Heflin and J. Hendler. A portrait of the Semantic Web in action. *IEEE Intelligent Systems*, 16(2):54–59, 2001.

[7] J. Heflin, J. Hendler, and S. Luke. Reading between the lines: Using SHOE to discover implicit knowledge from the Web. In *AI and Information Integration,Papers from the 1998 Workshop*, Menlo Park, CA, 1998. AAAI Press. Technical Report WS-98-14.

[8] V. Chaudhri, A. Farquhar, R. Fikes, P. Karp, and J. Rice. OKBC: A programmatic foundation for knowledge base interoperability. In Proc. of the Fifteenth National Conference on Artificial Intelligence (AAAI-1998), pages 600–607.

[9] K. Sagonas, T. Swift, and D. S. Warren. XSB as an efficient deductive database engine. In R. T. Snodgrass and M. Winslett, editors, *Proc. of the 1994 ACM SIGMOD Int. Conf. on Management of Data (SIGMOD'94)*, pages 442–453, 1994.

[10] K. Stoffel, M. Taylor, and J. Hendler. Efficient management of very large ontologies. In *Proc. of Fourteenth American Association for Artificial Intelligence Conference (AAAI-97)*, Menlo Park, CA, 1997. AAAI/MIT Press.

AUTHORS PROFILE

JAYARAJU CHIKKALA, Pursuing M.Tech(CSE) from Vignan's Lara Institute of Technology & Science, Vadlamudi, Guntur, A.P., India . Email: *glorious.ch@gmail.com*

DIVYA PEDASANAGATI , Pursuing M.Tech(CSE) from Vignan's Lara Institute of Technology & Science, Vadlamudi, Guntur, A.P., India. Email: *divya.lukky522@gmail.com*

MADHAVARAO EPURUI, Pursuing M.Tech(CSE) from Vignan's Lara Institute of Technology & Science, Vadlamudi, Guntur, A.P., India. Email*: madhavarao.epurru916@gmail.com*

RATNAM ASK, Assoc. Professor ,Head Department of Computer Science Engineering at Vignan's Lara Institute of Technology & Science, Vadlamudi, Guntur, A.P., India.
Email: *sriram.abburi@rediffmail.com*