A Semantic Wiki System Based on F-Logic

Markus Schatten, Mirko Čubrilo, Jurica Ševa
Faculty of Organization and Informatics
University of Zagreb
Pavlinska 2, 42000 Varaždin, Croatia
{markus.schatten, mirko.cubrilo, jurica.seva}@foi.hr

Abstract. *In this paper we present the* ┬AOPIS *semantic wiki system based on F-logic that we implemented. The system uses the* FLORA-*2 reasoning engine and supports the dynamic creation of queries in restricted* FLORA-*2 syntax. We show how we were able to apply a simple tagging system in order to support the creation of an object oriented ontology. In the end we give a comparison between other semantic wiki systems and the developed system as well as give guidelines for future research.*
Keywords. semantic wiki, F-logic, taopis, flora-2

1 Introduction

The semantic web is an extension of the current (traditional) World Wide Web based primarily on HTML[1] that shall allow one to find, share, and combine information more easily. It relies on machine-readable information and metadata that allows computer programs or agents to reason about distributed knowledge.

Wiki systems on the other hand are systems that allow virtually any user to create, modify and store content. Every visitor of a wiki service on the Web can change articles and information which he encounters, add new articles and/or information as well as argue about the existing ones. An additional mechanism that is built into such systems is the possibility to interconnect terms used in articles. In other words, every term that is mentioned in one article of the system can be connected with other articles which elaborate it further. This mechanism allows easier search and explanation of the unknown terms to its users [1].

[1]HyperText Markup Language

Semantic wiki systems are an extenension to wiki systems that use concepts borowed from the semantic web. In this way semantics or meaning is added to knowledge created on the system through additional metainformation which eases search, integration and reasoning [6]. Prominent semantic wiki systems include SweetWiki [8], IkeWiki [3] and Semantic Media Wiki [7] to name a few.

F-logic[2] is a full-fledged logic that has model-theoretic semantics and a sound and complete proof theory. In this sense, F-logic stands in the same relationship to the object-oriented paradigm as classical predicate calculus stands to relational programming [2]. FLORA-*2* is a rule-based object-oriented knowledge base system designed for a variety of automated tasks on the semantic web, ranging from meta-data management to information integration to intelligent agents. The FLORA-*2* system integrates F-logic, HiLog, and Transaction Logic into a coherent knowledge representation and inference language which results in a flexible and natural framework that combines rule-based and object-oriented paradigms [10].

In this paper we introduce a semantic wiki system based on F-logic and particularly implemented using the FLORA-*2* reasoning engine. For the sake of demonstration we restricted the syntax of F-logic to allow basic querying and simple ontology development. Exports to FLORA-*2* and OWL[3] [9] ontologies were also implemented to support possible development of intelligent agents. The system is a subsystem of ┬AOPIS [11] that aims on becoming an imlementation of an autopoeitic information system [4] [1].

[2]Frame Logic
[3]Web Ontology Language

2 Introducing Object-Oriented Concepts to Semantic Wikis

Most semantic wiki systems allow users to add semantics to content published on the system through different metainformation. We decided to use a simple tagging system to allow users to approach content in an object-oriented manner. In order to consider a domain of interest in an object-oriented fashion one needs to be able to model specific concepts like objects, classes (types, concepts), relations, attributes, methods, states etc. Thus we provide the following conceptualization of semantic wiki systems.

Let the whole content stored on the semantic wiki system be a domain of interest D. Objects inside this domain are specific wiki pages having their classes, relations, attributes, methods etc. Any wiki page on creation is a generic object that users can specialize in order to reflect the domain of interest. Thus the domain is an extendable set of objects as shown in equation 1.

$$D = \{o_1, o_2, ..., o_n\} \tag{1}$$

To allow concretization of generic objects we introduce attribute-value tags that reflect specific characteristics of objects inside a domain. Any object can be thought of as a relation that consists of a finite number of attribute-value tuples, as shown in equation 2.

$$o_i = \{(a_1, v_1), (a_2, v_2), ..., (a_m, v_m)\} \tag{2}$$

We also introduce object's relations to be defined as labeled outgoing links on any wiki page whether to other wiki pages or to pages outside the semantic wiki system. These relations are reflected as additional attribute-value pairs whereby the label represents the attribute and the value the object (page or URL)[4] the relation links to.

Now we are able to introduce special attributes labeled with common object-oriented programming constructs like `class`, `subClassOf`, `rule`, `inherits` etc. These attributes are used to provide additional semantics to the domain ontology.

[4]Unified Resource Locator

3 Introducing F-logic

The alphabet of an F-logic language $\mathcal{L}$ consists of the following [2]:

- a set of object constructors, $\mathcal{F}$;
- an infinite set of variables, $\mathcal{V}$;
- auxiliary symbols, such as, (,), [,], $\rightarrow$, $\twoheadrightarrow$, $\bullet\!\!\rightarrow$, $\bullet\!\!\twoheadrightarrow$, $\Rightarrow$, $\Rightarrow\!\!\!\!\!\!\Rightarrow$, etc.; and
- usual logical connectives and quantifiers, $\vee$, $\wedge$, $\neg$, $\longleftarrow$, $\forall$, $\exists$.

Object constructors (the elements of $\mathcal{F}$) play the role of function symbols in F-logic whereby each function symbol has an arity. The arity is a non-negative integer that represents the number of arguments the symbol can take. A constant is a symbol with arity 0, and symbols with arity ≥ 1 are used to construct larger terms out of simpler ones. An id-term is a usual first-order term composed of function symbols and variables, as in predicate calculus. The set of all variable free or ground id-terms is denoted by $U(\mathcal{F})$ and is commonly known as Herbrand Universe. Id-terms play the role of logical object identities in F-logic which is a logical abstraction of physical object identities.

A language in F-logic consists of a set of formulae constructed out of alphabet symbols. As in a lot of other logics, formulas are built out of simpler ones by using the usual logical connectives and quantifiers mentioned above. The most simple formulas in F-logic are called F-molecules.

A molecule in F-logic is one of the following statements:

- An is-a assertion of the form $C :: D$ (C is a nonstrict subclass of D) or of the form $O : C$ (O is a member of class C), where C, D and O are id-terms;
- An object molecule of the form O [a ';' separated list of method expressions] where O is a id-term that denotes and object. A method expression can be either a non-inheritable data expression, an inheritable data expression, or a signature expression:
  - Non-inheritable data expressions can be in either of the following two forms:

  - A non-inheritable scalar expression $ScalMethod@Q_1, ..., Q_k \rightarrow T, (k \geqslant 0)$.
  - A non-inheritable set-valued expression $SetMethod@R_1, ..., R_l \twoheadrightarrow \{S_1, ..., S_m\}$ $(l, m \geqslant 0)$.
- Inheritable scalar and set-valued expression are equivalent to their non-inheritable counterparts except that $\rightarrow$ is replaced with $\bullet\!\!\rightarrow$, and $\twoheadrightarrow$ with $\bullet\!\!\twoheadrightarrow$.
- Signature expression can also take two different forms:
  - A scalar signature expression $ScalMethod@V_1, ..., V_n \Rightarrow (A_1, ..., A_r)$, $(n, r \geqslant 0)$.
  - A set valued signature expression $SetMethod@W_1, ..., W_s \Rightarrow\!\!\!\Rightarrow (B_1, ..., B_t)$ $(s, t \geqslant 0)$.

All methods' left hand sides (e. g. Q_i, R_i, V_i and W_i) denote arguments, whilst the right hand sides (e. g. T, S_i, A_i and B_i) denote method outputs. Single-headed arrows ($\rightarrow$, $\bullet\!\!\rightarrow$ and $\Rightarrow$) denote scalar methods and double-headed arrows ($\twoheadrightarrow$, $\bullet\!\!\twoheadrightarrow$ and $\Rightarrow\!\!\!\Rightarrow$) denote set-valued methods.

4 F-logic and Semantic Wikis

For a proof of concept application we restricted the syntax of F-logic to support only non-inheritable scalar expressions as well as is-a assertions but this restriction should be easily extended to support the whole syntax of F-logic. We consider any attribute-value pair from equation 2 to denote a non-inheritable scalar method where the attribute (a_i) is the methods name that has no arguments $(k = 0)$ and values (v_i) to be outputs.

Special attribute labels like `class` and `subClassOf` are used to create is-a assertions. For example a wiki page tagged with the attribute `class` and value `student` is considered to be an object that is a member of class `student`. If the same object is additionally taged with the attribute `subClassOf` and value `person` than the class `student` is considered to be a nonstrict subclass of class `person`. All other tags provided on a wiki page are the special attributes of this object, thus the object mentioned previously if additionally tagged with tags like `name:Foo`, `surname:Bar`, `address:Linus Lane 27` would yield the following sentence in a F-logic knowledge base:

$$\begin{aligned} student &:: person \wedge \\ o_x &: student\ [\\ name &\rightarrow 'Foo'; \\ surname &\rightarrow 'Bar'; \\ address &\rightarrow 'Linus\ Lane\ 27'\ ] \end{aligned}$$

Where o_x denotes the logical object-id of the wiki page under consideration. Thus, in this proof of concept application classes and class hierarchy are created dynamically by tagging specific objects. Future versions shall include inheritable scalar expressions that would allow users to specify tags with attribute `class` and value `class` thus defining classes as instances of class class (allowing meta-modeling in this way). Set valued expressions can be also easily implemented by allowing a user to specify the same atrribute but different values. In the end signature expressions can be implemented using web services and script extensions that act as methods of a specific object.

5 Implementation and Syntax

We implemented a proof of concept object-oriented semantic wiki into the `ɹAOPīs` system. The implementation was made using $\mathcal{F}$LORA-*2* , Python, PostgreSQL and PHP. $\mathcal{F}$LORA-*2* was used as the reasoning engine that was interfaced with PostgreSQL using the Python programming language [5] in order to support dynamic creation of queries. Python was also used to generate $\mathcal{F}$LORA-*2* and OWL ontologies due to it's strong support for string processing. PHP acts mainly as a presentation layer.

The following syntax for dynamic creation of queries was introduced:

```
[query flora2_query.  ]
{[header] header_formatting [/header]}
answer_formatting
[/query]
```

whereby `flora2_query` is a normal (restricted) FLORA-2 query with defined return variables, `header_formatting` is the optional header (possibly formatted using `ᴛAOPĪs` code), and `answer_formatting` is a `ᴛAOPĪs` formatting that can contain variables used in the FLORA-2 query. The `answer_formatting` is repeated for any answer returned by the FLORA-2 reasoning engine by using the generated FLORA-2 ontology of the semantic wiki as a knowledge base.

For example a query like:

```
[query ?_:member[ name->?f_name ].  ]
Member's first name:  [b]?f_name.[/b]
[/query]
```

Would yield a list similar to:

Member's first name: **Markus**
Member's first name: **Mirko**
Member's first name: **Jurica**

Presuming that the members of the semantic wiki would have the corresponding first names.

6 Discussion

Whileas the idea of using semantic web concepts in wiki systems isn't new, the idea of using an object-oriented approach seems to be firstly described in this paper. Most other approaches used description logics and particularly OWL as their background logic [3], [7], [8] as well as annotations and tags on particular content. The problem with such approaches lies mostly in ignoring the primary users of wiki as well as tagging systems.

Wiki systems are used by ordinary people who most certainly have no good understanding of semantic technologies. They use wikis to quickly create content in a collaborative environment. Tagging systems are also used by ordinary people to organize content they encounter for their selves in order to easier retrieve content. The semantics that emerge by combining tags of different users are more of a side effect than the main purpose. If users are free to tag any content in the way they want more metainformation and semantics will emerge as when tagging is restricted to special types of tags that can be used. This metainformation will self-organize due to the autopoiesis of the social system surrounding a dynamic web application like a tagging or a wiki system [1].

In such a highly distributed and complex environment it is highly likely that certain inconsistencies will emerge. Tools for managing such inconsistencies are subject to our future research.

7 Conclusion and Future Research

By using a simple tagging system and a F-logic reasoning engine we were able to implement a proof of concept collaborative tool for object-oriented ontology development. This implementation trows new light on the implementation of semantic wiki systems since most such systems use description logics as their background logic. We argued how it is possibe to apply object oriented programming concepts to semantic wiki systems by defining a wiki page to be a genetir object that is to be shaped through user-defined tags. By adding additional semantics to attribute-value tags we were able to implement non-inheritable scalar expressions as well as is-a assertions. We also showed how to implement other expressions from F-logic.

By using the FLORA-2 reasoning engine as well as the power of the Python programming language we were able to implement a semantic wiki system based on F-logic. The proof of concept application supports export to FLORA-2 and OWL ontologies to facilitate posible agent technologies, attribute-value tagging as well as dynamic queries. Future development shall include full F-logic syntax (without restrictions but FLORA-2 modules), an improoved querying mechanism as well as tools for ontology amalgamation.

In the end we argued that current semantic wiki system often ignore the power of simplicity of normal (non-semantic) wiki systems. Semantic wiki systems should allow their users to do anything they like, semantics will emerge due to the autopoiesis of the social system surrounding it. If more metainformation is available and users are free to use any tag they like there is a certain probability that inconsistencies will emerge. Tools to manage such inconsistencies have to be developed and this is the main aim of our future research.

References

[1] Bača M, Schatten M, Deranja D: Autopoietic Information Systems in Modern Organizations. Organizacija, Revija za management, informatiko in kadre (Journal of Management, Informatics and Human Resources). Kranj, Slovenia, 2007, 40, 3, pp. 157-165.

[2] Kifer M, Lausen G, Wu J: Logical Foundations of Object-Oriented and Frame-Based Languages, Journal of the Association for Computing Machinery, New York, NY, USA, 1995, 42, 4, pp. 741-843.

[3] SalzburgResearch: IkeWiki, available at `http://ikewiki.salzburgresearch.at/`, Accessed: 14^{th} May 2008.

[4] Schatten M, Brumec J, Višić M: Strategic Planning of an Autopoietic Information System, 18^{th} International Conference on Information and Intelligent Systems (IIS2007) Proceedings, Fakultet organizacije i informatike (Faculty of Organization and Informatics), Varaždin, Croatia, 2007, pp. 435-440.

[5] Schatten, Markus. Reasonable Python or How to Integrate F-logic into an Object-Oriented Scripting Language, Proceedings of the 11^{th} International Conference on Intelligent Engineering Systems, IEEE, Budapest, Hungary, 2007, pp. 297-300.

[6] Schatten M, Žugaj M: Organizing a Fishnet Structure, Proceedings of the ITI 2007 29^{th} International Conference on Information Technology Interfaces, SRCE University Computing Centre, Cavtat, Croatia, 2007, pp. 81-86.

[7] Semantic-mediawiki.org: Semantic MediaWiki, available at `http://semantic-mediawiki.org/`, Accessed: 14^{th} May 2008.

[8] SweetWiki: Senamtic WEb Enabled Technology Wiki, available at `http://argentera.inria.fr/wiki/data/Main/MainHome.jsp`, Accessed: 14^{th} May 2008.

[9] W3C: OWL Web Ontology Language Semantics and Abstract Syntax, W3C Recommendation 10^{th} February 2004, Patel-Schneider P. F, Hayes P., Horrocks I (eds.), available at `http://www.w3.org/TR/owl-semantics/`, Accessed: 14^{th} May 2008.

[10] Yang G, Kifer M, Zhao C: FLORA-2 - A Rule-Based Knowledge Representation and Inference Infrastructure for the Semantic Web, 2^{nd} International Conference on Ontologies, Databases and Applications of Semantics (ODBASE), Catania, Italy, 2003, pp. 671-688.

[11] TaOPis: The Autopoietic Information System, available at `http://autopoiesis.foi.hr`, Accessed: 14^{th} May 2008.