

# A Controlled Natural Language Interface for Semantic Media Wiki

Jie Bao<sup>1</sup>, Paul R. Smart<sup>2</sup>, Dave Braines<sup>3</sup>, Gareth Jones<sup>3</sup>, Nigel R. Shadbolt<sup>2</sup>

<sup>1</sup> Department of Computer Science,  
Rensselaer Polytechnic Institute,  
Troy, NY 12180, USA.  
baojie, @cs.rpi.edu

<sup>2</sup> School of Electronics and Computer Science,  
University of Southampton,  
Southampton, SO17 1BJ, United Kingdom.  
ps02v@ecs.soton.ac.uk, nrs@ecs.soton.ac.uk

<sup>3</sup> Emerging Technology Services,  
IBM United Kingdom Ltd, Hursley Park,  
Winchester, Hampshire, SO21 2JN, United Kingdom.  
dave\_braines@uk.ibm.com, garethj@uk.ibm.com

**Abstract.** Despite of their potential value as collaborative knowledge editing systems, semantic wikis present a number of usability challenges. In particular, there are several mismatches between the simple user interaction mechanisms of wikis (which are the key to the success of wikis) and the need for users to create, edit and understand structured knowledge content (e.g., in the form of RDF or OWL ontologies). In this paper, we present a Controlled Natural Language (CNL) approach to support collaborative ontology development on Semantic MediaWiki (SMW). In order to support the expressivity required for OWL ontology development we were obliged to extend the representational substructure of the SMW system with an OWL meta model using a template-based mechanism. To improve usability, we provided a guided input interface based on semantic forms and output several CNL verbalisers including Rabbit English, Rabbit Chinese (Yayan) and Ace English. As such, this work may provide a potentially effective mechanism for encouraging the large-scale collaborative creation of semantically-enriched online content.

## 1 Introduction

Semantic wikis extend the idea of collaborative content editing (made popular by systems such as Wikipedia) to the realm of semantically-enriched representations and formal knowledge models. While a conventional wiki includes structured text and untyped hyperlinks, a semantic wiki is based on the representation of metadata elements. Semantic MediaWiki (SMW) [10] is probably the most popular and mature semantic wiki. It relies on the same wiki engine as Wikipedia and uses constructs from the Resource Description Framework (RDF) and Web Ontology Language (OWL) for semantic annotation.

Despite of their potential value as collaborative knowledge editing systems, semantic wikis encounter a number of usability challenges. In particular, how can we enable users to create and edit structured knowledge content (in the form of RDF models and OWL ontologies) without renegeing on the kind of simple user interaction mechanisms that makes conventional wiki systems, such as Wikipedia, so popular? One answer to this question is to capitalize on the availability of Controlled Natural Languages (CNLs) that provide some support for ontology model development. CNLs such as Rabbit [5], Sydney OWL Syntax (SOS) [2] and Attempto Controlled English (Ace) [7], all support the creation of semantically-enriched knowledge models, while preserving the production and comprehension benefits of natural languages.

The advantages of each CNL, relative to other CNLs, are still a topic for debate and empirical research [14]. Each CNL may ultimately prove differentially attractive to different user communities, so by accommodating multiple CNLs within the semantic wiki system, we hope to provide a collaborative knowledge editing environment that is agnostic with regard to the kind of CNL interface used by end-users to interact with the system.

As such, CNL interfaces for semantic wiki systems may provide a potent mechanism for encouraging the large-scale participation of user communities in the creation of semantically-enriched online content. In this paper, we present our initial efforts to develop a CNL interface system for collaborative knowledge editing on semantic wikis using several CNLs for OWL including Rabbit English, Rabbit Chinese and Ace. In order to support the expressivity required for OWL ontology development we were obliged to extend the representational substructure of the SMW system. As our ultimate aim is to provide a system that is capable of supporting multiple extant CNLs, we have developed the system architecture and technology components with a view to accommodating multiple CNLs using a template-based meta model called SMW-mOWL. Ultimately, we hope to support end-users in defining their own CNL formalisms for structured knowledge entry in a collaborative knowledge editing environment.

By bringing CNL and semantic wikis together, this work provides some essential initial steps towards a powerful tool ideally suited for collaborative authoring of structured knowledge for end users and domain experts who have only limited knowledge engineering background, which are not supported effectively by software applications today. Our contributions are highlighted by:

- An extensible platform for CNL interface for semantic wikis that is capable of accommodating multiple, multilingual CNLs (section 3).
- An OWL meta model for knowledge representation on semantic wiki (section 4).
- A prototype implementation for form-based ontology authoring and controlled natural language generation in Rabbit English, Rabbit Chinese and Ace syntaxes (section 5).

## 2 Related Work

**Controlled natural language** Controlled natural languages (CNLs) have been shown effective ways to promote human comprehension of structured knowledge [5, 8]. In particular, in the Semantic Web context, several CNLs have been proposed to reduce the learning threshold of OWL, which has a description logic background that is often hard to capture for non-logician users. For example, a study on the CNL Rabbit [5] reveals that CNL can improve domain experts' understanding and ability to author ontologies in OWL. Kaufmann and Bernstein [8] reported a study on the usefulness of CNL interface for querying Semantic Web data. Some other examples of CNLs include CLOnE [4], Sydney OWL Syntax (SOS) [2] and Attempto Controlled English (Ace) [7]. Mellish and others [12] also reported natural language generation from OWL ontologies. For survey and comparison of CNLs, please see [15] and [14].

Several ontology authoring tools using CNLs have been developed. ROO [3] is an authoring plugin based on Rabbit for the Protege ontology editor. However currently ROO only supports desktop authoring, while the focus of our work is collaborative, wiki-based knowledge editing. The closest work to ours is the AceWiki [11], a wiki-based ontology authoring tool using Ace. Our wiki differs from AceWiki in several ways:

1) SMW, with an active user community and with many extensions available, provides an open infrastructure that allows our system to easily interact with other applications, e.g. query, visualization and ontology repository;

2) While Acewiki requires some familiarity to the Ace CNL, our system does not require user the deep knowledge of a particular CNL for our authoring model uses a form-based interface;

3) While Acewiki (and ROO too) only supports a single CNL, our system is designed to accommodate multiple CNL syntaxes including Ace.

**Collaborative Knowledge Modeling** Support for collaborative ontology development has received increasing attention, as witnessed by the systems such as CODE [6], OntoEdit <sup>1</sup>, WebODE [17] and COB-Editor [1]. In particular, a recent version of **Protege** offer a multi-user mode<sup>2</sup> (released in April 2006), which allows multiple users to edit the same ontology concurrently. The ontology in question resides on an ontology server, all the changes made by one user are seen immediately by other users. Protege 3.3 (released in June 2007) and later versions further support several collaboration features including change annotation, discussion thread, proposal, voting and chat [16]. **TopBraid Composer** (from version 1.2.0, August 2006) also supports a multi-user mode by allowing all users to work on a shared Sesame<sup>3</sup> repository.

Both Protege and TopBraid Composer provide advanced graphical user interfaces for ontology editing and integration with external tools, e.g., reasoners.

<sup>1</sup> <http://www.ontoknowledge.org/tools/ontoedit.shtml>

<sup>2</sup> <http://protege.cim3.net/cgi-bin/wiki.pl?MultiUserTutorial>

<sup>3</sup> <http://www.openrdf.org/>

They are suitable for ontology development for users with some knowledge engineering (KR) background and understanding of the semantics of OWL. On the other hand, the semantic wiki based approach we have adopted is intended to provide a light-weight solution for collective comprehension [9] and knowledge editing for users with, at best, limited, KR knowledge [13]. In addition, in our approach:

- The system inherits the inherent collaborative nature of wikis, and can co-opt with many existing wiki features for improving collaboration. For example, in our approach it is easy to track change history and provenance information of ontology editing, and revert an erroneous edit.
- The browser-based editing environment offers high portability and accessibility that desktop-based tools lack.
- While Protege and TopBraid Composer only allow *formal* knowledge modeling, the semantic wiki-based approach may support extra processes in the life cycle of knowledge engineering. For instance, a domain expert can start from *informal* or semi-structured descriptions on wiki, which will be further improved (potentially by other domain experts or knowledge engineers) into formal representation.
- As the key components of our system (CNL verbalisers, the OWL meta model, and form-based editor) are themselves stored as wiki pages, the system is highly transparent and easier to extend on the fly.

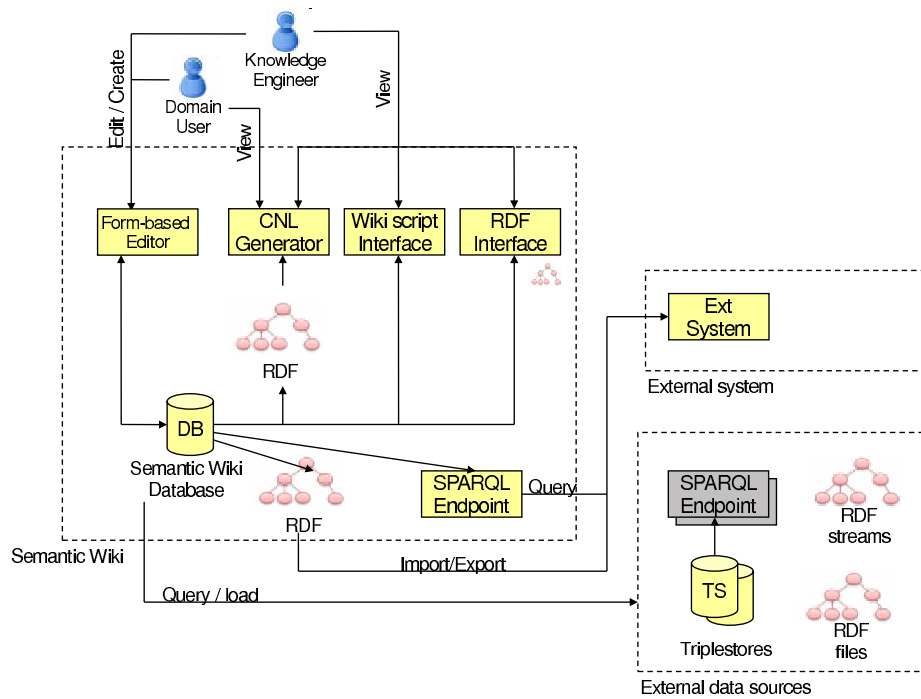
### 3 Architecture

Figure 1 illustrates the architecture of our system. The system consists of the following components:

- **Form-based Editor:** This is an editing interface that allows users to create and edit knowledge statements that will be translated into the target CNLs. Currently it contains a form-based OWL editor that allows both domain users and knowledge engineers to do guided entry of knowledge. To use such an interface, a user is not required to be familiar with either the ontology languages (e.g., RDF or OWL) or the CNL (e.g., Rabbit). The result from the CNL form interface will be saved in SMW-mOWL, a CNL-independent OWL meta model in SMW (Section 4).
- **Wiki Script Interface:** This interface supports the generation of knowledge description by either direct statements in the SMW script<sup>4</sup> or by providing knowledge statements in the SMW-mOWL meta model. It provides an alternative and more flexible approach for knowledge engineers, who have better

---

<sup>4</sup> A SMW script is a sequence of instructions that are interpreted by a wiki engine in order to accomplish a variety of wiki-based operations, e.g. creating semantic annotation, creating pages, determining content layout, and so on. For example, a “[[:Category:Person]]” script on a page “Jim” will generate a category membership annotation “Jim rdf:type Person” in the wiki database. For the complete syntax of SMW script, please refer to <http://semantic-mediawiki.org/>.



**Fig. 1.** Architecture of a generic CNL Interface For SMW

understanding of wiki script and OWL modeling, to input knowledge that maybe consumed by the CNLG module for CNL rendering.

- **CNL Generator:** This is a Controlled Natural Language Generation (CNLG) module that can generate CNL descriptions of the knowledge statements associated with a wiki page. It also allows users to provide usual informal contents on wiki pages.
- **RDF Interface:** This is an interface that supports an RDF-based view of components of an ontology (as wiki pages). It is intended for those users who are familiar with RDF models.
- **Database (DB):** This is the wiki database that stores both semantic (RDF triples) and non-semantic data (i.e., the free text contents of wiki pages).
- **Import/Export Modules:** A number of import/export modules handle the communication with external tools and knowledge technology components. For example, the knowledge base constructed can be accessed by external tools (e.g., a reasoner) via a SPARQL end point, or be dumped into an RDF file and further consumed by external data sources.

The system is designed to be highly extensible with additional CNL syntaxes in the future. The use of SMW-mOWL as the immediate format for knowledge representation and its correspondence to the OWL abstract syntax gives much flexibility in supporting multiple CNLs in the system. A new CNL can be added to the system as long as that language has a bi-directional mapping to/from OWL. In addition, it is straightforward to provide multi-linguistic CNLG interfaces as it only requires a new set of templates for each of the controlled natural language syntaxes being considered.

We have implemented a prototype semantic wiki system based on the system architecture presented in Figure 1 (see <http://tw.rpi.edu/proj/cnl/>).

## 4 OWL Meta-modeling on Semantic Wiki

This section explains SMW-mOWL, a meta modeling specification of OWL on SMW which we have developed as a part of this work.

### 4.1 Design Rationale

In order to accommodate multiple CNL syntaxes for OWL, such as Rabbit and Ace, within a semantic wiki system, we need to address a number of expressivity constraints associated with semantic wikis. SMW, for example, does not provide full support for OWL modeling formalisms, and this introduces a mismatch between the kind of knowledge statements that can be represented in that specific CNL and the knowledge statements that can be created in SMW.

In order to address this limitation, we developed a meta-model extension to SMW, called SMW-mOWL (where “m” stands for meta model). To reconcile the expressivity mismatch between SMW and OWL, SMW-mOWL represents an OWL ontology using a set of wiki pages, each of which encodes some ontology elements (i.e., classes, properties, individuals and axioms) as “semantic

template”<sup>5</sup> instances. The motivation behind this meta-modeling approach is based on a number of design considerations.

**OWL Abstract Syntax:**  
**Class(Rabbit partial intersectionOf (Animal  
restriction(eat someValuesFrom(FreshVegetable))))**

**Basic Information**

This page is a definition:  A definition gives both sufficient and necessary conditions

Label (English name of Rabbit):

Plural Form (plural form of the name of Rabbit):

In Ontology:

```

{{NamedClass
| is definition=No
| label=Rabbit
| plural=Rabbits
}}

```

**Relation to other classes**

Rabbit is  None  subClassOf  equivalentClass  complementOf  disjointWith  
the class

```

{{NamedClassRelation
| type=subClassOf
| class=Animal
}}

```

**The class must have some property values from**

Every Rabbit have some values of the property   
from the class

```

{{someValuesFrom
| on property=eat
| on class=FreshVegetable
}}

```

**Fig. 2.** SMW-mOWL Class Examples

**Relation to OWL Abstract Syntax (OWL-AS):** SMW-mOWL is intended to have a direct correspondence to OWL-AS. Entities (classes, properties and individuals) and axioms in SMW-mOWL are represented as corresponding wiki templates. This correspondence between OWL-AS and SMW-mOWL has a couple of advantages. Firstly, OWL-AS can be used as an intermediate syntax for knowledge exchange between SMW and other tools. Secondly, it provides an extensible framework for supporting multiple CNLs within the SMW environment. An example of such correspondences is shown in Figure 2, where one class expression in OWL-AS is mapped to three template instances.

<sup>5</sup> Semantic template extends the usual wiki template with the ability to turn wiki script into semantic annotations with a predefined skeleton. For example, if a template page “Template:T1” has content “[[value:{{para}}]]”, and it is used on a page “Ex1” with content “{{T1|para=v1}}”, then a triple “Ex value v1” will be added to the wiki database. A template can also be used to control look-and-feel of pages that use the template.

**Relation to Editing UI:** By utilizing what are called “semantic forms” within SMW, each template can be edited using a form-based interface. Thus, having the OWL meta-model immediately provides us with a light-weight OWL ontology editor within the SMW environment. Figure 2 illustrates an example of how several wiki templates (in this case represented by the ‘Basic Information’, ‘Relation to other classes’, and ‘The class must have some property values from’ sections), can be aggregated on a single wiki page in order to support ontology authoring capabilities. Each of the individual wiki templates exposes a form-based interface, comprising conventional form controls (textboxes, checkboxes, radio buttons, and so on) that support various editing operations. Auto-completion of semantic forms allows sentence editing using existing entities in the ontology.

**Query Convenience.** The use of a template-based mechanism for SMW-mOWL allows us to store the knowledge model in the SMW database and to use the query language for SMW (SMW-QL) to retrieve specific information from the model. The query libraries for SMW-mOWL are themselves implemented as a set of templates. For example, an instance of `Template:someValuesFrom` will be persisted as an instance<sup>6</sup> of the ternary property `owl:someValuesFrom` in the wiki of which the first element is the class where the template instance resides, the second element is the “on property” parameter, and the third element is the “on class” parameter. A query in the form “`{{ask: [[{{PAGENAME}}]] |?owl:someValuesFrom}}`” will fetch all such template instances related to a page “`{{PAGENAME}}`”. Further parsing and sentence generation work will be preformed by the specific CNLG templates.

## 4.2 SMW-mOWL Templates

In what follows, we present some design details for SMW-mOWL.

**Class Meta-Model.** Following the SMW convention, each OWL class is represented as a page within the namespace “Category”. In particular, there are two types of classes:

- *NamedClass*: such a class is assigned a name by end-users, e.g., the name “Category:Rabbit” represents the class “Rabbit” in the ontology. A named class may be associated with natural language labels which are used in serializing the class into a CNL text. The system supports multi-lingual labels.
- *AnonClass*: such a class is an anonymous class. It does not have a name and is typically an instance of the `owl:Restriction` class. Such a class is automatically assigned a unique number by the wiki system (e.g., Category:-1). The predominant role of an AnonClass is to represent categories of objects that are associated with various logical expressions, e.g., ‘the class of things

---

<sup>6</sup> SMW support n-ary property. The current implement also uses mashing to represent a n-ary property as a binary property.

that only eats instances of the VegetableFood class’. Natural language labels for an AnonClass are automatically generated by the wiki using a corresponding CNL grammar.

Basic OWL constructs are also represented as templates. For example, the Template:someValuesFrom represents the owl:someValuesFrom restriction. The form component in Figure 2 that is labeled ‘The class must have some property values from’ corresponds to the form-based interface to the Template:someValuesFrom template. It enables end-users to create classes (in this case AnonClasses) that are associated with the owl:someValuesFrom (or existential quantifier) restriction. The set of templates on a class page by default gives the necessary conditions of the class (“partial” relation in the OWL-AS); optionally, a class can be declared as a “definition”, and in this case templates on its page give both necessary and sufficient conditions of the class (“complete” relation in the OWL-AS). The set of class templates in SMW-mOWL is given in Figure 3.

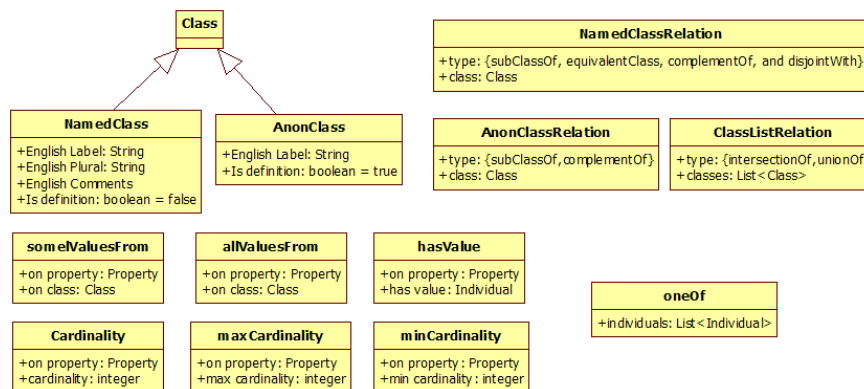


Fig. 3. SMW-mOWL Class Meta Model

**Property Meta-Model.** Following the SMW convention, each OWL property is represented as a page within the namespace “Property”. Basic information about a property is captured by the “Template:Property” template; relationships between properties are captured by the “Template:PropertyRelation” template (Figure 4).

Example: a property “eat”, which is a subproperty of “consume”, is represented as a wiki page “Property:Eat”

```

{{Property
|label           = eats
|pass label     = is eaten by

```

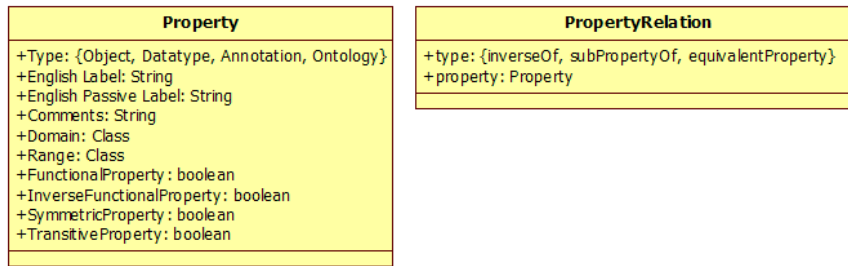


Fig. 4. SMW-mOWL Property Meta Model

```

|FunctionalProperty      = No
|InverseFunctionalProperty = No
|SymmetricProperty      = No
|TransitiveProperty     = No
}}
{{PropertyRelation
|type      = subPropertyOf
|property  = consume
}}

```

**Individual Meta-Model.** Instances of classes and properties (tuples) are represented in “Individual” pages, which are in the main namespace of the wiki, or in additional namespaces that the administrator of the wiki has specified. A property instance is always stored on its subject’s page; i.e. the page that represents an object appearing as the first element of the triple . Currently, we do not support the instantiation of an AnonClass. Individual templates are given in Figure 5.

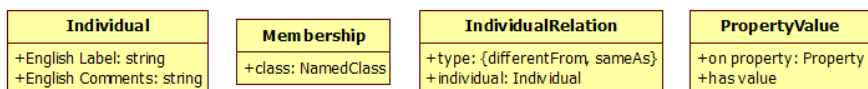


Fig. 5. SMW-mOWL Individual Meta-Model

In order to exemplify the template-based approach to OWL meta-modelling within the semantic environment, consider the following class expression, as represented in OWL-AS:

```
Class(Rabbit partial Animal
```

```
restriction(eat someValuesFrom(
    Vegetable restriction(hasStatus Fresh))))
```

This expression will be represented as the following set of wiki pages (names of the pages are in the bold font) within the CNL wiki system:

**Category:Rabbit**

```
{{NamedClass |label=Rabbit |plural=Rabbits }}
{{NamedClassRelation |type=subClassOf |class=Animal }}
{{someValuesFrom |on property=eat |on class=-1 }}
```

**Category:Animal**

```
{{NamedClass |label=Animal |plural=Animals}}
```

**Category:Vegetable**

```
{{NamedClass |label=Vegetable |plural=Vegetables}}
```

**Category:-1** (automatically numbered)

```
{{AnonClass}}
{{AnonClassRelation |type=subClassOf |class=Vegetables}}
{{hasValue |on property=hasStatus |has value=Fresh}}
```

**Property:eat**

```
{{Property |label=eat |type=Object}}
```

**Property:hasStatus**

```
{{Property |label=has status of |type=Object}}
```

**Fresh**

```
{{Individual}}
```

### 4.3 Limitations

Currently, the SMW-mOWL meta-model does not support several OWL features. These include:

- Built-in xsd datatypes, such as the datatypes for strings and numbers. Currently property values are by default treated as plain literals.
- Deprecation of vocabulary<sup>7</sup>.
- N-ary axioms: DisjointClass and EquivalentClass axioms between multiple classes; their binary forms are supported.
- Ontology management annotations, e.g., owl:imports.

<sup>7</sup> It's a rarely used feature in OWL

## Category:Rabbit

| Category:Rabbit [ Edit ] |                 |
|--------------------------|-----------------|
| <b>Label:</b>            | Rabbit          |
| <b>Plural:</b>           | Rabbits         |
| <b>In ontology:</b>      | Rabbit Ontology |

| "Category:Rabbit" in "Rabbit" controlled natural language  |
|--|
| <ul style="list-style-type: none"> <li>■ Rabbit is a Animal.</li> <li>■ No Rabbit is a NonRabbit.</li> <li>■ Rabbit and Hare are equivalent.</li> <li>■ Rabbit and Wolf are mutually exclusive.</li> <li>■ Every Rabbit is exactly one of Bugs Bunny OR Peter Rabbit.</li> <li>■ Every Rabbit is a White Rabbit or a Black Rabbit.</li> <li>■ Every Rabbit eats FreshVegetable,</li> <li>■ Every Rabbit has part Whisker.</li> <li>■ Every Rabbit has child(ren) only Rabbit or nothing.</li> <li>■ Every Rabbit has eye color of Red.</li> <li>■ Every Rabbit has leg(s) exactly 4.</li> <li>■ Every Rabbit has head at least 1.</li> <li>■ Every Rabbit has parent at most 2.</li> <li>■ Rabbit is a concept, plural Rabbits.</li> </ul> |

Fig. 6. A class represented in the Rabbit CNL

## 5 CNL Interface

One of the major design goals of the SMW-mOWL meta-model is to enable users to create, edit and view ontologies in different CNLs. This is enabled with the feasibility of bi-directional translations between our target CNLs and the subset of OWL that corresponds to the SMW-mOWL meta-model. The current system partially achieves this goal with supports CNL generation for two CNLs (Ace and Rabbit), and one of these (Rabbit) is available in a foreign language (viz. Chinese).

Please note that knowledge statements that will be consumed by the CNL interface can be edited either via the form-based editor (with some CNL hints), or by directly editing of either wiki script or the template-based script in SMW-mOWL. Please refer Fig. 2 for an example. Currently we do not support free-style entry of knowledge statements as CNL sentences.

### 5.1 Represent An Ontology in Rabbit (English and Chinese)

The OWL knowledge statements generated by users are stored as scripts conforming to SWM-mOWL on wiki pages. The meta model templates convert the meta model into a set of RDF triples that are persisted in the SMW triple store. A set of “Rabbit” templates will use semantic queries to fetch these triples for constructing Rabbit sentences. For example, the template `Template:CNL.Rabbit.getSomeRestrictionAssertion` will 1) get all restrictions related to the page in question by “someValuesFrom”; 2) parse the concept and property of the restriction; 3) query the natural language label of involved property and concepts; 4) construct a hyperlinked sentence following the Rabbit syntax.

Please note that natural language label of a class, property or individual may be different from its wiki page name; this allows multi-lingual support as well as complex expressions that have no explicitly given name.

One example of the automatically generated Rabbit sentences of a class is shown in Figure 6.

**Internationalization Support.** To demonstrate the multi-linguistic capability of the system, a Yayan (a.k.a. Rabbit Chinese) CNL generator (CNLG) has been implemented. Yayan is based on Rabbit English, although it includes adaptations for accommodating several special characteristics of the Chinese language. These adaptations include:

- Removal of capitalization - since written Chinese is not an alphabet-based language, there is no need for capitalization.
- Removal of plural forms of class labels - nouns in Chinese usually do not distinguish between singular and plural forms.
- The introduction of unit words - Chinese requires a special unit word for each noun when describing its quantity. For example, “Rabbit” (tùzi in Chinese<sup>8</sup> has the unit word “zhī”, thus “one rabbit” will be represented as “yī zhī tùzi” (where “yī” means one). Similarly, for “one fish” (yú, with unit word “tiáo”), the Chinese expression is “yī tiáo yú”. Thus, for each class (corresponding to a noun) in Yayan, a unit word label is required.
- CNL generation for cardinality restrictions - to generate sentences in Chinese that are natural to average users, some cardinality restrictions require special CNL language patterns. For example, to express restriction (hasLeg cardinality(4)) (has four legs), one needs to split the verb part (has, yǒu) and the noun part (leg, tuǐ) of the property and insert the number (four, sì) and the unit word (“tiáo” for “tuǐ”) in between, thus rendering “yǒu sì tiáo tuǐ”. Such language patterning information is embedded in the label of the property “hasLeg”, which will be used by the Yayan CNLG.
- Paraphrase from Rabbit English - Some language patterns in Rabbit cannot be literally translated into natural Chinese. These include: disjointWith, all-ValuesFrom, (min, max) Cardinality, and FunctionalProperty. Paraphrasings of all these are implemented.

<sup>8</sup> For ease of reading, we use pinyin to present Chinese here. The implemented system supports Chinese characters in Unicode.

A draft of Yayan syntax in BNF, derived from the Rabbit English BNF syntax, is available at <http://tw.rpi.edu/proj/cnl/Yayan>. A Yayan CNLG has also been developed using a template-based approach that can generate Yayan sentences from the knowledge statements in SMW-mOWL.

## 5.2 Represent Ontology in Ace (English)

Ace is a CNL whose expressivity matches that of first order logic. Its expressivity is therefore greater than that of OWL, or other CNLs that directly support OWL, e.g. SOS [2], and this has resulted in the specification of a restricted variant of Ace (Ace-OWL) that has been used as the basis for expressing OWL ontologies in natural language. Ace-OWL differs from Rabbit in a number of ways. For example, in Rabbit and Ace-OWL an "inverseOf" relation between the properties "eats" and "is eaten by" is represented in the following way:

Rabbit: The relationship "eats" is the complement of "is eaten by"  
 Ace: If something X is-eaten-by something Y then Y eats X

We have implemented an Ace-OWL CNLG in the wiki system, using a set of Ace templates that translates SMW-mOWL expressions into Ace-OWL statements, designed in the similar fashion as that of Rabbit. Some constraints of the Ace-OWL CNL grammar are respected:

- Blanks in name labels are replaced by hyphens .
- Since Ace does not support class equivalence relations, an equivalence statement has to be represented by two "subclassOf" statements, e.g., "Every Child is a Kid" and "Every Kid is a Child". These statements provide a shorthand way to assert equivalence between "Child" and "Kid" classes.

## 6 Conclusions

This paper summarizes our initial efforts to develop a generic CNL interface for semantic wiki systems. Thus far we have developed an architecture to support the collaborative editing of community knowledge using semantic wikis. We have also developed a meta modeling extension to SMW in order to accommodate the expressivity features of many of CNLs. Finally, we have implemented a prototype of our system (see <http://tw.rpi.edu/proj/cnl/>) that can support multiple and multilingual CNLs (Ace and Rabbit in English and Yayan in Chinese).

The current prototype implementation requires several further usability improvements. Several potential approaches have been identified and will be explored in the future:

- The development of improved parsing and editing capabilities, i.e., a better user interface to support the direct entry of CNL sentences.
- The use of "semantic forms" for end-user entry and editing of knowledge content will be extended to hide the handling of anonymous classes.

- A better ontology repository module to support multiple ontologies within a single wiki environment.
- The integration of a graphical interface to support users with various kinds of editing operations.
- The OWL meta model and CNLGs will be extended to support OWL 2, an extension to OWL.
- In addition, we aim to extend the meta-model system, briefly described above, in order to accommodate other CNLs (e.g. SOS, CLCE, etc.) within a single semantic wiki system.

## Acknowledgement

This work is partially supported by Research was sponsored by the U.S. Army Research Laboratory and the U.K. Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. We thank Zhenning Shangguan for part of the implementation work of the CNL wiki, David Mott and James Hendler for useful discussions, and Li Ding for reviewing the draft of the paper.

## References

1. Jie Bao, Zhiliang Hu, Doina Caragea, James Reecy, and Vasant Honavar. Developing frameworks and tools for collaborative building of large biological ontologies. In *the 4th International Workshop on Biological Data Management (BIDM), DEXA Workshops*, pages 191–195. 2006.
2. Anne Cregan, Rolf Schwitter, and Thomas Meyer. Sydney owl syntax - towards a controlled natural language syntax for owl 1.1. In *OWLED*, 2007.
3. Vania Dimitrova, Ronald Denaux, Glen Hart, Catherine Dolbear, Ian Holt, and Anthony G. Cohn. Involving domain experts in authoring owl ontologies. In *International Semantic Web Conference*, pages 1–16, 2008.
4. Adam Funk, Valentin Tablan, Kalina Bontcheva, Hamish Cunningham, Brian Davis, and Siegfried Handschuh. Clone: Controlled language for ontology editing. In *ISWC/ASWC*, pages 142–155, 2007.
5. Glen Hart, Martina Johnson, and Catherine Dolbear. Rabbit: Developing a control natural language for authoring ontologies. In *ESWC*, pages 348–360, 2008.
6. P. Hayes, R. Saavedra, and T. Reichherzer. A collaboration development environment for ontologies. In *Proceedings of the Semantic Integration Workshop, Sanibel Island, Florida*,. 2003.
7. Kaarel Kaljurand and Norbert E. Fuchs. Bidirectional mapping between owl dl and attempto controlled english. In *PPSWR*, pages 179–189, 2006.
8. Esther Kaufmann and Abraham Bernstein. How useful are natural language interfaces to the semantic web for casual end-users? In *ISWC/ASWC*, pages 281–294, 2007.
9. Chrysovalanto Kousetti, David Millard, and Yvonne Howard. A study of ontology convergence in a semantic wiki. In *WikiSym 2008*, September 2008.
10. Markus Krötzsch, Denny Vrandečić, Max Völkel, Heiko Haller, and Rudi Studer. Semantic wikipedia. *J. Web Sem.*, 5(4):251–261, 2007.

11. Tobias Kuhn. AceWiki: Collaborative Ontology Management in Controlled Natural Language. In *Proceedings of the 3rd Semantic Wiki Workshop*. CEUR Workshop Proceedings, 2008.
12. Chris Mellish and Jeff Z. Pan. Natural language directed inference from ontologies. *Artif. Intell.*, 172(10):1285–1315, 2008.
13. Frederik Pfisterer, Markus Nitsche, Anthony Jameson, and Catalin Barbu. User-centered design and evaluation of interface enhancements to the semantic mediawiki bibtex. In *Semantic Web User Interaction Workshop (SWUI 2008)*, April 2008.
14. Rolf Schwitter, Kaarel Kaljurand, Anne Cregan, Catherine Dolbear, and Glen Hart. A comparison of three controlled natural languages for owl 1.1. In *OWL: Experiences and Directions (OWLED)*, page online.
15. Paul R Smart. Controlled natural languages and the semantic web. Technical Report ITA/P12/SemWebCNL, School of Electronics and Computer Science, University of Southampton, <http://eprints.ecs.soton.ac.uk/15735/>, July 2008.
16. Tania Tudorache and Natasha Noy. Collaborative protege. In *Workshop on Social and Collaborative Construction of Structured Knowledge (CKC) at 16th International World Wide Web Conference (WWW2007)*, 2007.
17. Julio César Arpírez Vega, Óscar Corcho, Mariano Fernández-López, and Asunción Gómez-Pérez. WebODE: a scalable workbench for ontological engineering. In *K-CAP*, pages 6–13. 2001.