Writing RSS Feeds in a Machine-Processable Controlled Natural Language

Rolf Schwitter and Marc Tilbrook
Centre for Language Technology
Macquarie University
Sydney, 2109 NSW, Australia

{schwitt|marct}@ics.mq.edu.au

Abstract

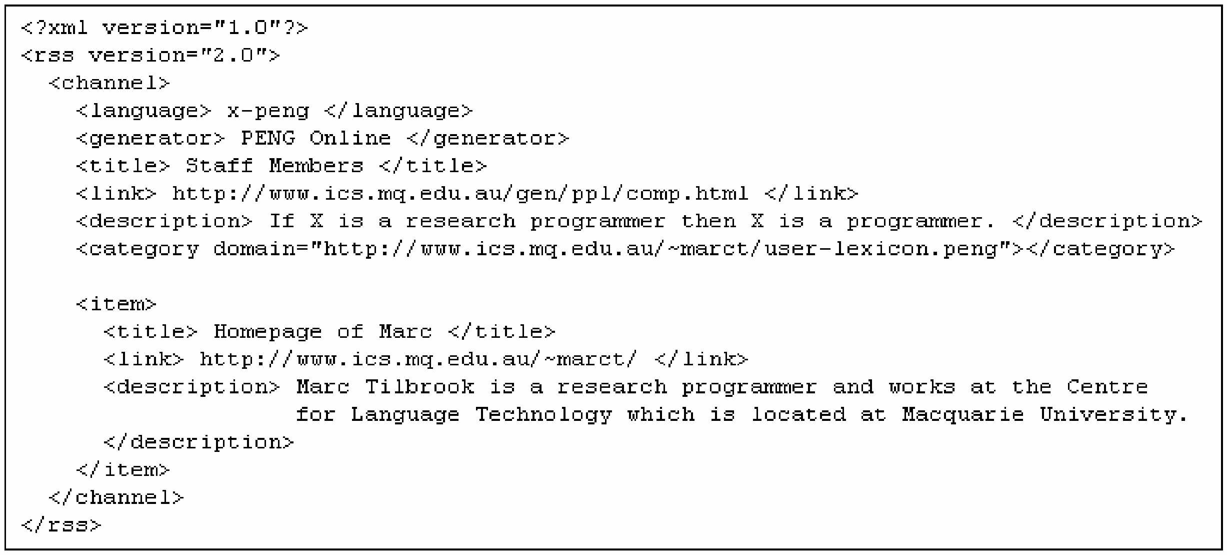
In this paper, we present PENG Online, a web-based authoring tool which allows authors to write RSS feeds in a machine-processable controlled natural language. The authoring tool is implemented as a Java applet and communicates in near real-time with a remote server which runs a controlled natural language processor and an inference engine. The authoring tool provides browser functionality for viewing web pages which can then be summarised and augmented in controlled natural language. The writing process is guided via predictive interface techniques which guarantee that the text conforms to the rules of the controlled natural language. The resulting descriptions in RSS format look informal at first glance and are easy to read and understand by humans, but they have the same formal properties as the underlying logic-based representation language. These properties build an ideal starting point for making inferences over the represented knowledge and for using the RSS feeds as a source for question answering.

1. Introduction

It has been convincingly argued that the current architecture for the Semantic Web, with its strong emphasis on RDF for syntactic and semantic compatibility, has severe problems when expressive Semantic Web (reasoning) languages are incorporated [Patel-Schneider, 2005]. An alternative approach is to use conventional first-order logic as the semantic underpinning for the Semantic Web. First-order logic is well understood and well established subsets of first-order logic offer tradeoffs with respect to expressive power, complexity and computability [Horrocks and Patel-Schneider, 2003]. For example, the direct mapping of description logic-based ontology languages and Horn rule languages into subsets of first-order logic provides immediate semantic interoperability and builds the prerequisite for efficient reasoning [Grosof et al., 2003]. Instead of relying on RDF, we suggest using a machine-oriented controlled natural language which is based on first-order logic as interface language to the Semantic Web. To promote our approach, we show how such a controlled natural language can directly be embedded into RSS and used to summarise and augment web pages with information which is easy to understand by humans and easy to process by machines.

RSS is a family of XML-based web feed formats designed for sharing and aggregating web content [RSS, 2002]. RSS feeds provide summaries of web content together with links to the full versions of the content. RSS feeds can be created automatically from existing websites with the help of specific markup tags or manually using feed creation software. Probably the most popular use of RSS is in RSS feed aggregators. RSS aggregators are programs which allow the user to subscribe to a number of RSS feeds, check periodically for new content, retrieve the content, and provide an aggregated view of the content. An RSS feed is an XML document consisting of an <rss> element with a single <channel> element, which contains information about the channel and its content. A channel may contain a number of <item> elements. In the most typical case, each item consists of a <title> element for the title of the summarised content, a <description> element for the summary of the web content written in natural language (possibly with entity-encoded HTML tags), and a <link> element which points to the full web content.

These RSS feeds are usually written in full natural language which is highly ambiguous and difficult to process by a machine. We suggest using a machine-oriented controlled natural language instead which looks seemingly informal on the surface but has formal properties which are equivalent to first-order logic. Figure 1 shows such an RSS feed where the <description> elements contain information written in controlled natural language.



```
<?xml version="1.0"?>
<rss version="2.0">
  <channel>
    <language> x-peng </language>
    <generator> PENG Online </generator>
    <title> Staff Members </title>
    <link> http://www.ics.mq.edu.au/gen/ppl/comp.html </link>
    <description> If X is a research programmer then X is a programmer. </description>
    <category domain="http://www.ics.mq.edu.au/~marct/user-lexicon.peng"></category>

    <item>
      <title> Homepage of Marc </title>
      <link> http://www.ics.mq.edu.au/~marct/ </link>
      <description> Marc Tilbrook is a research programmer and works at the Centre
                    for Language Technology which is located at Macquarie University.
      </description>
    </item>
  </channel>
</rss>
```

Figure 1: RSS Feed in Controlled Natural Language

2. Controlled Natural Languages

A controlled natural language is a subset of a full natural language with explicit restrictions on the grammar, lexicon, and style. These restrictions usually have the form of writing rules and help to reduce (or even exclude) ambiguity and to cut down the complexity of full natural language.

2.1 Human- versus machine-oriented Controlled Natural Languages

Traditionally, controlled natural languages fall into two categories: human-oriented and machine-oriented controlled natural languages. Human-oriented controlled natural languages (for example ASD Simplified Technical English [ASD, 2005]) aim at improving text comprehension for human readers while machine-oriented controlled natural languages (for example Common Logic Controlled English [Sowa, 2004]) focus on improving text processability for machines. An important difference between human-oriented and machine-oriented controlled natural languages is that the writing rules for machine-oriented controlled natural languages must be precise and computationally tractable [Huijsen, 1998]. However, as a rule of thumb, simplification works in both ways: human-oriented controlled natural languages are also easier to process by machines and machine-oriented controlled natural languages are also easier to understand by humans compared to full natural language.

2.2 PENG (Processable ENGlish)

PENG is a machine-oriented controlled natural language designed for writing unambiguous and precise specification texts for knowledge representation [Schwitter, 2002; Schwitter, 2004; Schwitter, 2005]. PENG covers a strict subset of standard English and is precisely defined by a controlled grammar and a controlled lexicon. Texts written in PENG are incrementally parsed using a unification-based grammar and then translated into first-order logic via discourse representations structures [Kamp and Reyle, 1993; Schwitter and Tilbrook, 2004]. The result is a logic theory which can be checked for consistency and used for question answering. In contrast to other machine-oriented controlled natural languages [Pullman, 1996; Fuchs et al., 1999; Holt et al., 1999; Sowa, 2004], the author of a PENG text does not need to know the grammatical restrictions of the language explicitly. The text editor of the PENG system dynamically enforces these restrictions while the text is written and displays the interpretation of a sentence in the form of a paraphrase in controlled natural language. The controlled lexicon of the PENG system consists of a base lexicon and a user lexicon. The base lexicon contains frequent content words (*proper nouns*, *common nouns*, *verbs*, *adjectives*, and *adverbs*) and pre-defined function words (*determiners*, *prepositions*, *coordinators*, *subordinators*, and *disambiguation markers*) which build the syntactic scaffolding of the controlled natural language. The user lexicon can be extended with domain-specific content words by the author while a text is written.

3. PENG Online

PENG Online is the web-based version of the PENG editor which can be used to create and update RSS feeds as well as other documents written in controlled natural language. This authoring tool offers built-in browser functionality for viewing web pages and provides an RSS mode for summarising the content of these web pages in controlled natural language.

3.1 Architecture

PENG Online is based on a client-server architecture which consists of – as Figure 2 below shows – three main components: an authoring tool (the PENG editor), a controlled natural language processor, and an inference engine. The web-based authoring tool is implemented as a Java applet which runs in a web browser and communicates with the Prolog server via a socket interface. The Prolog server implements the controlled natural language processor and the inference engine. Each time a Java client connects to the Prolog server, a child process is created to manage the session which allows for multiple user sessions.

After an RSS feed has been created with the help of the authoring tool, it can be stored on the local Web server together with the relevant part of the user lexicon. To link the RSS feed to the exported user lexicon, the <category> sub-element of the <channel> element is used. The <category> sub-element appears with a “domain” attribute and an URI as value which points to the user lexicon (see Figure 1 for an example).

In principle, any RSS aggregator can subscribe to such an RSS feed written in controlled natural language but the full benefit is only brought to bear via a PENG-compliant tool.

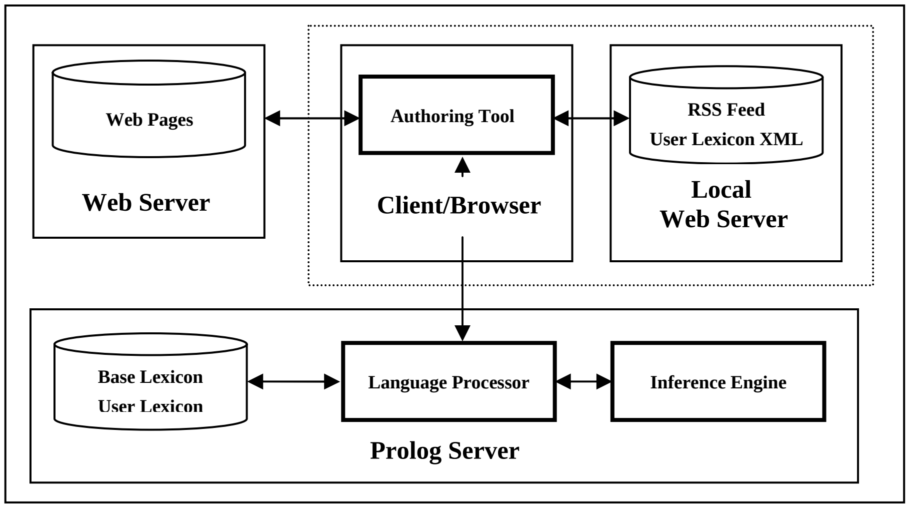
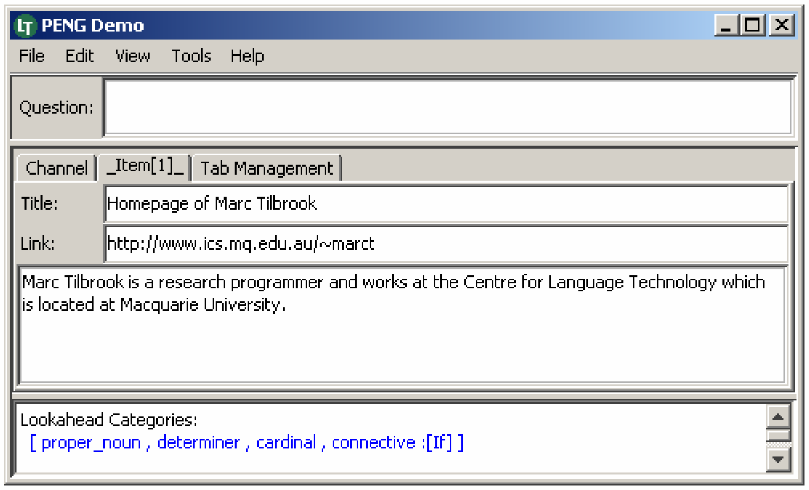


Figure 2: Architecture of PENG Online

3.2 The PENG Editor

The PENG editor features a standard mode and an RSS mode. The RSS mode mirrors the structure of a basic RSS feed with a channel element and any number of items. New items can easily be added and deleted via a management tab. As Figure 3 below illustrates, an item consists of a title field, a link field for the URL which points to the full web page, and a description field for the controlled natural language text. The textual information in the description fields of an RSS feed can be queried in controlled natural language via the question field.



Title: Homepage of Marc Tilbrook

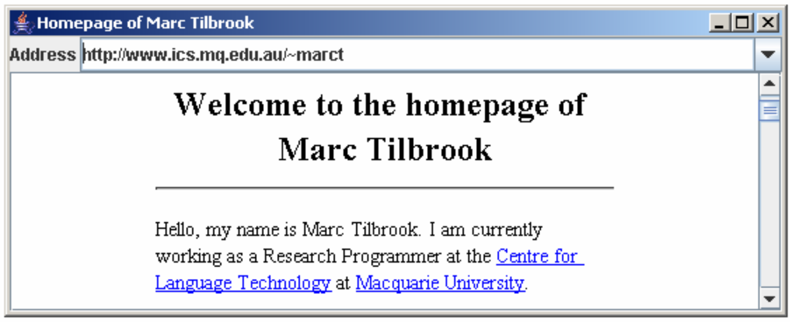
Link: http://www.ics.mq.edu.au/~marct

Marc Tilbrook is a research programmer and works at the Centre for Language Technology which is located at Macquarie University.

Lookahead Categories:
[proper_noun , determiner , cardinal , connective :[If]]

Figure 3: PENG Editor in RSS Mode

As already mentioned, the text editor provides browser functionality and allows the author to view a web page (see Figure 4) which can then be summarised in controlled natural language in the description field of the text editor (see Figure 3).


Homepage of Marc Tilbrook
Address http://www.ics.mq.edu.au/~marct

Welcome to the homepage of Marc Tilbrook

Hello, my name is Marc Tilbrook. I am currently working as a Research Programmer at the Centre for Language Technology at Macquarie University.

Figure 4: Excerpt of Original Web Page

Figure 4 shows a simple excerpt of a web page written in full natural language. Not all of this information can and will be represented in the description field in controlled natural language. The idea is to produce a machine-processable summary of a web page that can easily be read by humans and efficiently be processed by a machine. This requires a careful tradeoff between expressiveness and processability of the controlled natural language.

3.2.1 The Description Field

The input to the description field is restricted by the language processor which generates look-ahead information for each word form that the author enters (see the message field of Figure 3 for an example). This look-ahead information consists of syntactic categories which predict what kind of input can follow the current word form. The look-ahead categories are implemented as hypertext links. By clicking on a look-ahead category the author is able to access help information. The author composes a sentence either by typing the word forms which fall under the look-ahead categories or by selecting word forms from a cascade of menus [Schwitter et al., 2003; Thompson et al., 2005]. Both input modes are driven by an incremental predictive chart parser.

Figure 3 shows the PENG editor in action. In this example, the author entered a first sentence into the description field to start summarising a web page, repeated here as (1):

(1) Marc Tilbrook is a research programmer and works at the Centre for Language Technology which is located at Macquarie University.

Please note that the look-ahead categories are generated on the fly and use linguistic information produced by the incremental chart parser of the controlled language processor. The processing of these look-ahead categories does not slow down the author while typing the text and happens in near real-time (ca. 140 milliseconds on average per word form).

The look-ahead categories in Figure 3 indicate that the author can continue the text, for example, using a proper noun as in (2), a determiner as in (3), a cardinal number as in (4), or a specific connective as in (5):

(2) *… which is located at Macquarie University.* ***Marc*** *…*

(3) *… which is located at Macquarie University.* ***The*** *…*

(4) *… which is located at Macquarie University.* ***Two*** *…*

(5) *… which is located at Macquarie University.* ***If*** *…*

Instead of typing an approved word form into the description field of the editor, the author can alternatively select a word form from the currently active look-ahead categories via the context menu:

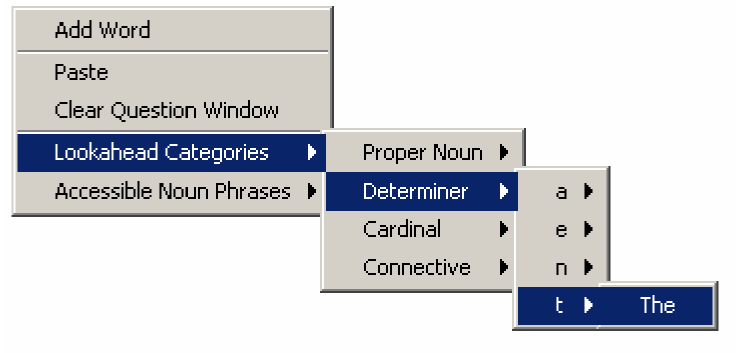


Figure 5: Active Look-ahead Categories

Once such a word form has been selected, it will be immediately inserted into the text at the current cursor position and the processing of the text is automatically resumed. Not only can approved word forms be inserted in this way, but also all noun phrases which are accessible in the text. Accessible noun phrases occur in the context menu and can be selected from there. Figure 6 shows that after processing of sentence (1) the following three noun phrases are available in the context menu:

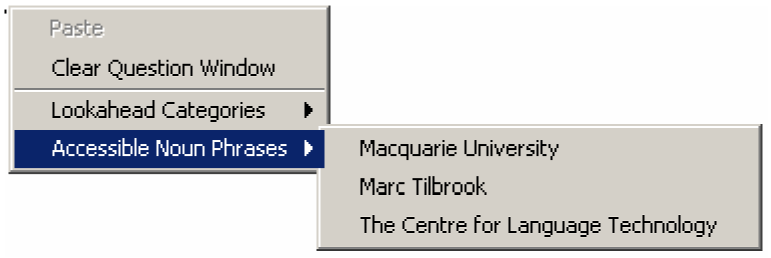


Figure 6: Accessible Noun Phrases

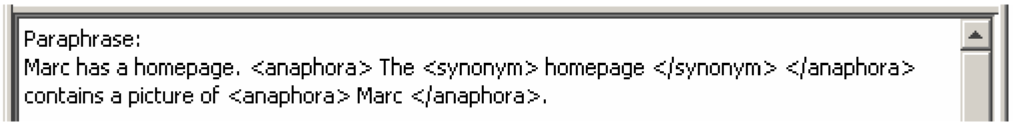
Please note that the noun phrase *a research programmer* is not accessible here, since it forms a property together with the copulative verb *be* and cannot be referred to by a definite noun phrase.

3.2.2 The Message Field

The message field displays a paraphrase for each sentence and clarifies the interpretation of the input. The paraphrase indicates, for example, if synonyms or anaphoric expressions have been used in the text. Let us assume that the author added the following two sentences to the description field:

(6) *Marc has a homepage. The home page contains a picture of Marc.*

And let us further assume that the noun *home page* has previously been defined as a synonym of its main form *homepage* in the user lexicon. After processing this information, the paraphrase in the message field will indicate – as Figure 7 illustrates – that the noun phrase *the home page* and the proper noun *Marc* are two anaphoric expressions which have been previously introduced in the text and that the synonym *home page* has been replaced, respectively normalized, by its main form *homepage*.



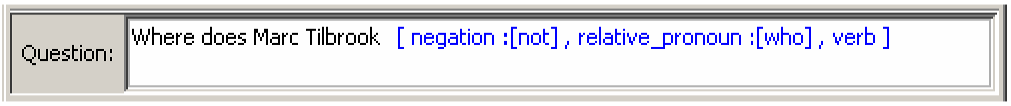
Paraphrase:
Marc has a homepage. <anaphora> The <synonym> homepage </synonym> </anaphora> contains a picture of <anaphora> Marc </anaphora>.

Figure 7: Paraphrase in Controlled Natural Language

Additionally, the message field displays the syntax tree for the last input sentence, the actual discourse representation structure for the entire text and its representation in first-order logic. Furthermore, the message field shows the model which was generated by the inference engine and indicates whether the current textual fragment is consistent or not (see Section 3.4 for details). Not all of this information is relevant for the author and parts of this information can therefore be selectively removed.

3.2.3 The Question Field

The purpose of the question field is to pose questions in controlled natural language about an RSS feed. The question field uses the same kind of look-ahead mechanism to guide the writing process as the description field. Figure 8 illustrates how the look-ahead information occurs in the question field while the author is typing a question.



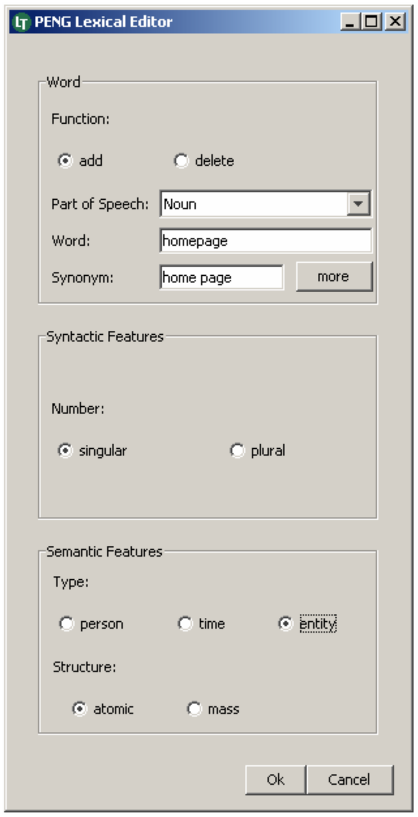
Question: Where does Marc Tilbrook [negation :[not] , relative_pronoun :[who] , verb]

Figure 8: Question Field with Look-ahead Categories

Once a question is completely formulated, it is translated into first-order logic via discourse representation structures and answered over the generated model using the inference engine of the PENG system.

3.2.4 The Lexical Editor

Part of the text editor is a lexical editor for adding user-specific content words. If the author enters a content word (i.e. *proper noun*, *common noun*, *verb*, *adjective* or *adverb*) into the text editor which is not yet available in the lexicon and is not in the list of illegal words, then this content word needs to be added to the user lexicon of the PENG system.


PENG Lexical Editor

Word

Function: [✓ add] delete

Part of Speech: Noun

Word: homepage

Synonym: home page

Syntactic Features

Number: [✓ singular] plural

Semantic Features

Type: person time [✓ entity]

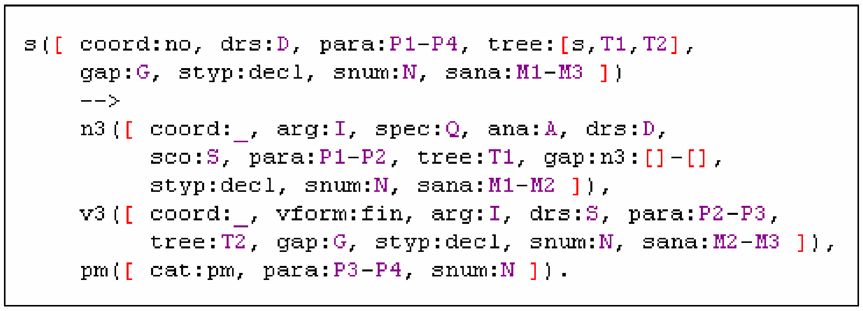
Structure: [✓ atomic] mass

Figure 9: Lexical Editor of PENG Online

As Figure 9 shows, the interface to the lexical editor has been designed in such a way that only minimal linguistic knowledge is required by the author to add a new content word to the lexicon. As soon as a new content word is available in the lexicon, the parsing process is resumed. User-defined content words can also be deleted from the user lexicon, but the author cannot delete words in the base lexicon of the PENG system which contains the most frequent 3000 words of English as well as all predefined function words.

3.3 The Controlled Natural Language Processor

When the author types a word form into the text editor, this word form is immediately sent to the incremental chart parser of the controlled language processor. The chart parser uses a unification-based phrase structure grammar as syntactic scaffolding [Schwitter, 2003; Schwitter and Tilbrook, 2004]. As Figure 10 shows, the phrase structure rules of the grammar are highly parameterised. The beauty of this approach is that it allows us to deal with syntactic, semantic and pragmatic information concurrently and in the same logic-based framework. The grammar currently consists of about 150 such phrase structure rules. During parsing the incremental chart parser generates a chart which can be used to harvest the look-ahead information for the text editor. The other important information in the chart is the discourse representation structure which represents the meaning of the text and which can be translated in linear time into a set of first-order logic formulas. These first-order logic formulas can then be further processed by the inference engine of the PENG system as we will briefly describe in the next section.

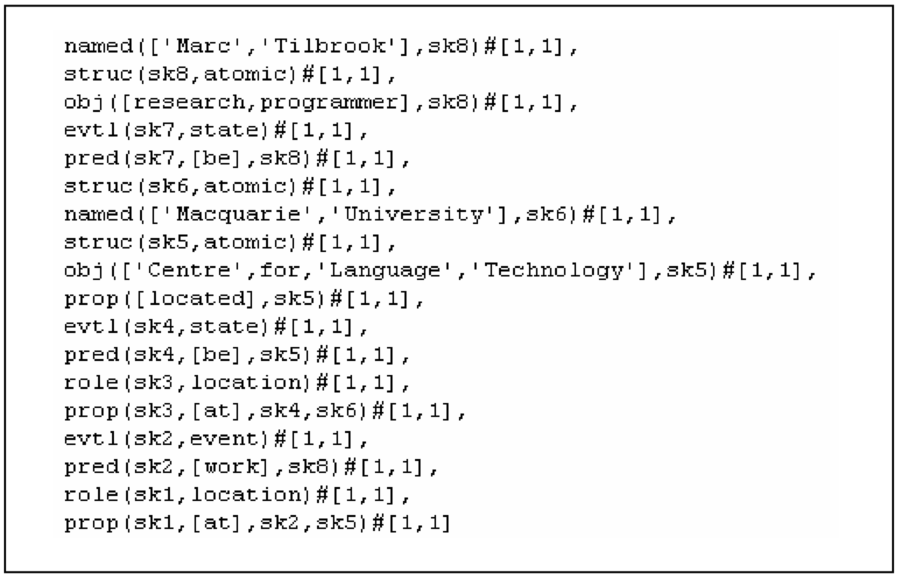


```
s([ coord:no, drs:D, para:P1-P4, tree:[s,T1,T2],
    gap:G, styp:decl, snum:N, sana:M1-M3 ])
    -->
    n3([ coord:_, arg:I, spec:Q, ana:A, drs:D,
         sco:S, para:P1-P2, tree:T1, gap:n3:[]-[],
         styp:decl, snum:N, sana:M1-M2 ]),
    v3([ coord:_, vform:fin, arg:I, drs:S, para:P2-P3,
         tree:T2, gap:G, styp:decl, snum:N, sana:M2-M3 ]),
    pm([ cat:pm, para:P3-P4, snum:N ]).
```

Figure 10: Example of a Phrase Structure Rule

3.4 The Inference Engine

The current version of the PENG system uses a modified version of a SATCHMO-style model generator as inference engine. SATCHMO can handle full first-order logic and uses well-established logic programming techniques and powerful extensions such as integrity constraints and disjunction [Manthey and Bry, 1988; Bry and Yahya, 2000; Loveland and Yahya, 2003]. The SATCHMO-style inference engine fits well into the same logic programming paradigm as the controlled language processor and can be fine-tuned for our specific purposes. The first-order formulas generated by the controlled language processor are automatically translated into a set of clauses in SATCHMO rule format. A SATCHMO rule has the form *antecedent* ---> *consequent*. The antecedent of such a rule is either true, a single atomic formula or a conjunction of atomic formulas. The consequent is either false, a single atomic formula or a disjunction of atomic formulas. SATCHMO takes a theory in this rule format and checks its satisfiability by generating a model (for details see [Abdennadher et al., 1995]). For example, for the complex sentence (1) SATCHMO generates the following consistent model:



```
named(['Marc','Tilbrook'],sk8)#[1,1],
struc(sk8,atomic)#[1,1],
obj([research,programmer],sk8)#[1,1],
evtl(sk7,state)#[1,1],
pred(sk7,[be],sk8)#[1,1],
struc(sk6,atomic)#[1,1],
named(['Macquarie','University'],sk6)#[1,1],
struc(sk5,atomic)#[1,1],
obj(['Centre',for,'Language','Technology'],sk5)#[1,1],
prop([located],sk5)#[1,1],
evtl(sk4,state)#[1,1],
pred(sk4,[be],sk5)#[1,1],
role(sk3,location)#[1,1],
prop(sk3,[at],sk4,sk6)#[1,1],
evtl(sk2,event)#[1,1],
pred(sk2,[work],sk8)#[1,1],
role(sk1,location)#[1,1],
prop(sk1,[at],sk2,sk5)#[1,1]
```

Figure 11: Generated Model

This model is straightforward to build for sentence (1) and does not use any additional ontological knowledge. But we can easily add terminological information to an RSS feed. The best place to do this is the <description> element of the main <channel> element (see Figure 1 for a simple example). In the text editor, we can specify a class hierarchy in controlled natural language using the description field of the channel tab, for example:

(7) If X is a research programmer then X is a researcher.
(8) If X is a researcher then X is a scientist.
(9) If X is a research programmer then X is not a staff member.

This information is available for all subsequent items in the RSS feed. The generated model can also be used as a starting point for question answering. In this scenario, the author poses a question about the RSS feed in controlled natural language. A question *Q* is first translated into first-order logic via discourse representation structures, similar to a declarative sentence. The question is then interpreted as a logical consequence of a set of clauses *S* if and only if $S \cup \{Q \rightarrow \bot\}$ has no model. In the case of *wh*-questions variable bindings during model generation result in answers to questions. That means the PENG system answers questions on the basis of the information provided by the generated model. Given the model in Figure 11, the following questions can be answered:

(10) Where does Marc Tilbrook work?
(11) Is Marc Tilbrook a research programmer?
(12) Who works at the Centre for Language Technology?
(13) Where is the Centre for Language Technology located?

If the terminological information expressed in sentence (7)-(9) is additionally available, then we can also answer questions such as:

(14) Is Marc Tilbrook a researcher?
(15) Is Marc Tilbrook a staff member?
(16) Who is a scientist and works at the Centre for Language Technology?

Note that the truth of simple sentences as well as each constituent of a sentence can be investigated in this way. The answers to these questions are then displayed in controlled natural language in the message field of the text editor.

4. Conclusions

In this paper, we presented PENG Online, an authoring tool which allows authors to summarise and augment web pages with machine-processable information. The authoring tool provides browsing functionality to view selected web pages and offers predictive writing support to help with summarising these web pages in controlled natural language. The authoring tool communicates with a remote server, that implements a controlled language processor, which translates a text incrementally into discourse representation structures and subsequently into first-order logic. This representation can be checked automatically for consistency with the help of a model generator and the generated model can be used to answer questions posed in controlled natural language. The resulting text is stored as an RSS feed together with a link to the user lexicon which is exported as an XML document. Any RSS aggregator can subscribe to such an RSS feed, but the full processing power is only available via PENG Online or another PENG compliant tool.

Acknowledgments

The research reported here is supported by the Australian Research Council, Discovery Project No. DP0449928, and by the Australian Defense Science and Technology Organisation (DSTO). We would also like to thank to three anonymous reviewers for their valuable feedback.

References

[Abdennadher et al., 1995] S. Abdennadher, F. Bry, N. Eisinger, T. Geisler, The Theorem Prover Satchmo: Strategies, Heuristics, and Applications (System Description). *Research Report PMS-FB-1995-3,* May, Institute for Informatics, Ludwig Maximilians University, Munich, Germany, 1995.

[ASD, 2005] ASD Simplified Technical English, Specification ASD-STE100, *A Guide for the Preparation of Aircraft Maintenance Documentation in the International Aerospace Maintenance Language*, Issue 3, January 2005.

[Bry and Yahya, 2000] F. Bry, A. Yahya, Minimal Model Generation with Positive Unit Hyper-Resolution Tableaux, in: *Journal of Automated Reasoning*, Vol. 25, Issue 1, July pp. 35-82, 2000.

[Fuchs and Schwitter, 1996] N. E. Fuchs, R. Schwitter, Attempto Controlled English (ACE), in: *Proceedings of CLAW 96*, First International Workshop on Controlled Language Applications, University of Leuven, Belgium, March 1996, pp. 124-136, 1996.

[Fuchs et al., 1999] N. E. Fuchs, U. Schwertel, R. Schwitter, Attempto Controlled English – Not Just Another Logic Specification Language, *Lecture Notes in Computer Science* 1559, Springer, 1999.

[Grosof et al., 2003] B. Grosof, I. Horrocks, R. Volz, S. Decker, Description Logic Programs: Combining Logic Programs with Description Logic, in: *Proceedings of WWW 2003*, Hungary, ACM, pp. 48-57, 2003.

[Huijsen, 1998] W. O. Huijsen, Controlled Language – An Introduction, in: *Proceedings of CLAW 1998*, Pittsburgh, pp. 1-15, 1998.

[Holt et al., 1999] A. Holt, E. Klein, C. Grover, Natural language for hardware verification, in: *Proceedings of ICoS-1 workshop: Inference in Computational Semantics*, Institute for Logic, Language and Computation (ILLC), Amsterdam, August 1999, pp. 133-137, 1999.

[Horrocks and Patel-Schneider, 2003] I. Horrocks, P. F. Patel-Schneider, Three theses of representation in the semantic web, in: *Proceedings of WWW 2003*, Hungary, ACM, pp. 39-47, 2003.

[Kamp and Ryle, 1993] H. Kamp, U. Reyle, *From Discourse to Logic*, Kluwer Academic Publisher, 1993.

[Loveland and Yahya, 2003] D. Loveland, A. H. Yahya, SATCHMOREBID: SATCHMO(RE) with BIDirectional relevancy, in: *New Generation Computing*, Vol. 21, Issue 3, pp. 177-207, 2003.

[Manthey and Bry, 1998] R. Manthey, F. Bry, Satchmo: A theorem prover implemented in Prolog, in: E. Lusk and R. Overbeek (eds.), *Proceedings CADE-88*, Vol. 310 of LNCS, pp. 415-343, 1988.

[Patel-Schneider, 2005] P. F. Patel-Schneider, A Revised Architecture for the Semantic Web Reasoning, in: *Proceedings of PPSWR'05*, Dagstuhl, Germany, LNCS 3703, pp. 32-36, 2005.

[Pulman, 1996] S. G. Pulman, Controlled Language for Knowledge Representation, in: *Proceedings of CLAW 96*, First International Workshop on Controlled Language Applications, University of Leuven, Belgium, March 1996, pp. 233-242, 1996.

[RSS, 2002] *RSS 2.0 Specification*, Technology at Harvard Law, Internet technology hosted by Berkman Center, available at: http://blogs.law.harvard.edu/tech/rss, 2002.

[Schwitter, 2002] R. Schwitter, English as a Formal Specification Language, in: *Proceedings of the Thirteenth International Workshop on Database and Expert Systems Applications (DEXA 2002)*, W04: Third International Workshop on Natural Language and Information Systems - NLIS, 2-6 September 2002, Aix-en-Provence, France, pp. 228-232, 2002.

[Schwitter, 2003] R. Schwitter, Incremental Chart Parsing with Predictive Hints, in: *Proceedings of the Australasian Language Technology Workshop 2003*, December 10, University of Melbourne, Australia, pp. 1-8, 2003.

[Schwitter et al., 2003] R. Schwitter, A. Ljungberg, D. Hood, ECOLE - A Look-ahead Editor for a Controlled Language, in: *Proceedings of EAMT-CLAW03*, Controlled Translation, Joint Conference combining the 8th International Workshop of the European Association for Machine Translation and the 4th Controlled Language Application Workshop, May 15-17, Dublin City University, Ireland, pp. 141-150, 2003.

[Schwitter, 2004] R. Schwitter, Dynamic Semantics for a Controlled Natural Language, in: *Proceedings of the Fifteenth International Workshop on Database and Expert Systems Applications (DEXA 2004)*, NLIS'04: 4th International Workshop on Natural Language and Information Systems, 30 August - 3 September 2004, Zaragoza, Spain, pp. 43-47, 2004.

[Schwitter and Tilbrook, 2004] R. Schwitter, M. Tilbrook, Dynamic Semantics at Work, in: *Proceedings of LENLS2004* (in conjunction with the 18th Annual Conference of the Japanese Society for Artificial Intelligence, 2004), in Kanazawa (Japan), May 31, pp. 49-60, 2004.

[Schwitter, 2005] R. Schwitter, A Layered Controlled Natural Language for Knowledge Representation, in: S. Cardey, P. Greenfield and S. Vienney (eds.), *Machine Translation, Controlled Languages and Specialised Languages*: Special Issue of Linguisticae Investigationes, Vol. 28, No. 1, pp. 85-106, 2005.

[Sowa, 2004] J. F. Sowa, Common Logic Controlled English, *Draft*, 24 February 2004, available at: http://www.jfsowa.com/clce/specs.htm, 2004.

[Thompson et al., 2005] C. W. Thompson, P. Pazandak, H. R. Tennant, Talk to Your Semantic Web, in: *IEEE Internet Computing*, Vol. 9, No. 6, pp. 75-79, 2005.