Controlled Natural Languages and the Semantic Web

Paul Smart

School of Electronics and
Computer Science
University of Southampton
Southampton
SO17 1BJ
United Kingdom

27th July 2008

Report Documentation Page

Report Title	Controlled Natural Languages and the Semantic Web		
Project Title:	International Technology Alliance		
Number of Pages:	78	**Version:**	1.0
Date of Issue:	27/07/2008	**Due Date:**	N/A
Performance Indicator:	EZ~01~01~29	**Number of References:**	87
Reference Number:	ITA/P12/SemWebCNL		
Report Availability:	DISTRIBUTION UNLIMITED		
Abstract Availability:	DISTRIBUTION UNLIMITED		
Authors:	Paul Smart		
Keywords:	ontologies, semantic web, owl, rdf, controlled natural languages, natural language interfaces, semantic querying, ontology verbalization, ontology editors, ontology engineering, knowledge engineering		
Primary Author Details: Dr Paul Smart, Senior Research Fellow, School of Electronics and Computer Science, University of Southampton, Southampton, UK, SO17 1BJ, tel: +44 (0)23 8059 4492, fax: +44 (0)23 8059 3313, email: ps02v@ecs.soton.ac.uk		**Client Details:**	

Abstract:

The Semantic Web grew out of research into formal logics, and many of the representational and technological commitments of the Semantic Web reflect this heritage. Notwithstanding the obvious advantages for machine processability and reasoning, the logical underpinnings of the Semantic Web present a number of usability challenges for human end-users, especially when it comes to the communication and exploitation of domain-relevant knowledge/information. In this report, we review one approach to the resolution of these usability challenges – an approach that is based on the use of natural language user interfaces. Natural language interfaces exploit the medium of natural language in order to support end-users with respect to a range of capabilities; for example, the authoring of knowledge content, the retrieval of information from semantic repositories, and the generation of natural language texts from formal ontologies. The current report reviews the state-of-the-art with respect to natural language interfaces in all these capability areas. It also attempts to explore issues associated with the use of controlled natural languages as the representational basis for knowledge content on the Semantic Web. The idea that controlled natural languages could serve as a replacement for conventional Semantic Web ontologies is explored in some detail, as is the notion that natural language interfaces could contribute to the usability of the Semantic Web without requiring the wholesale replacement of pre-existing approaches. The report concludes with a discussion about the potential contribution of natural language interfaces to the emergence of truly hybrid (human/machine) intelligent systems. This idea sees natural language interfaces to the Semantic Web as an essential component of a future extended cognitive system – one that co-opts elements of human cognition with the computational and representational resources of a globally-extensive and semantically-scaffolded information environment.

Acknowledgements

This research was sponsored by the US Army Research laboratory and the UK Ministry of Defence and was accomplished under Agreement Number W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, the U.S. Government, the UK Ministry of Defence, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

Abstract

The Semantic Web grew out of research into formal logics, and many of the representational and technological commitments of the Semantic Web reflect this heritage. Notwithstanding the obvious advantages for machine processability and reasoning, the logical underpinnings of the Semantic Web present a number of usability challenges for human end-users, especially when it comes to the communication and exploitation of domain-relevant knowledge/information. In this report, we review one approach to the resolution of these usability challenges – an approach that is based on the use of natural language user interfaces. Natural language interfaces exploit the medium of natural language in order to support end-users with respect to a range of capabilities; for example, the authoring of knowledge content, the retrieval of information from semantic repositories, and the generation of natural language texts from formal ontologies. The current report reviews the state-of-the-art with respect to natural language interfaces in all these capability areas. It also attempts to explore issues associated with the use of controlled natural languages as the representational basis for knowledge content on the Semantic Web. The idea that controlled natural languages could serve as a replacement for conventional Semantic Web ontologies is explored in some detail, as is the notion that natural language interfaces could contribute to the usability of the Semantic Web without requiring the wholesale replacement of pre-existing approaches. The report concludes with a discussion about the potential contribution of natural language interfaces to the emergence of truly hybrid (human/machine) intelligent systems. This idea sees natural language interfaces to the Semantic Web as an essential component of a future extended cognitive system – one that co-opts elements of human cognition with the computational and representational resources of a globally-extensive and semantically-scaffolded information environment.

Contents

Figures

Tables

1 Introduction

This report forms part of the research effort associated with the International Technology Alliance (ITA) – a joint US/UK research programme that seeks to undertake fundamental research in the network and information sciences (Preece & Sieck, 2007). One of the focus areas for our ITA research concerns the use of Semantic Web[1] technologies to support military planning and decision-making. In particular, we have been exploring the use of semantic technologies to support the exchange of task-relevant information between coalition force elements (Braines et al., 2008a; Braines et al., 2008b; Kalfoglou et al., 2008), and we have also been examining the use of domain ontologies to provide a representational framework for the communication of plan-relevant information (Mott & Hendler, 2007).

One problem with the use of semantic technologies in applied research areas is that the end-user community often lacks the level of expertise required for the optimal or maximal exploitation of proposed technological solutions. In the case of the ITA, the end-user community consists of military personnel (e.g. commanders, intelligence analysts, military planners, etc.) who are not necessarily familiar with technologies such as domain ontologies, semantic queries, automated reasoners, and so on. This makes the use of semantically-enabled technologies somewhat problematic: technologies that might otherwise make a real contribution to intelligence analysis and military decision-making risk being rejected because of (often genuine) concerns about the training overheads associated with their use.

One of the solutions to this usability problem, which we have been exploring as part of our work in the ITA, concerns the use of Controlled Natural Languages (CNLs). CNLs are a subset of natural languages that impose a number of restrictions with respect to both the generation and interpretation of natural language expressions (Cregan et al., 2007; Fuchs et al., 2006a; Fuchs et al., 2006b; Hart et al., 2007; Kittredge, 2003; Kuhn, 2006). Recently, they have been used as a means of enabling human end-users to represent and communicate domain knowledge within the context of the Semantic Web, without necessarily introducing the kind of training overhead that might be required for the acquisition of professional ontology editing skills. As such, CNLs may provide an effective interface language for the Semantic Web – one that obviates some of the difficulties associated with the human exploitation of Semantic Web resources, without necessarily undermining the potential for machine-based processing of Semantic Web content.

Given the potential importance of CNLs as an interface language for the Semantic Web, this report aims to review the current state-of-the-art with respect to CNLs. In particular, we seek to understand the potential of natural language interfaces to address usability concerns in situations where the target user community is not necessarily familiar with the technological and representational solutions being developed as part of the Semantic Web initiative. The structure of the report is as follows: Section 2 provides a general introduction to some of the usability problems associated with the Semantic Web, including an analysis of why such usability problems might arise in the first place; Section 3 reviews some of the more common CNLs being used to address usability concerns in the context of the Semantic Web; Section 4 describes a number of ontology authoring

[1] see http://www.w3.org/2001/sw/

tools that capitalize on the availability of CNLs to provide the user with a supportive natural language environment for the creation and manipulation of machine-readable knowledge structures; Section 5 reviews natural language query systems that are used to retrieve information from the Semantic Web using natural language expressions; Section 6 describes various approaches to the generation of natural language texts using a combination of Natural Language Generation (NLG) technologies and semantically-enriched representations of domain-relevant knowledge; finally, Section 7 examines a number of perspectives regarding the precise relationship between (controlled) natural language representations and the formalisms that currently constitute the representational bedrock of the Semantic Web.

2 Good at Frisbee, Bad at Logic

The technological infrastructure of the Semantic Web supports the representation and dissemination of knowledge content in a form that is highly accessible to machines and supportive of a variety of automated, knowledge-based services (Berners-Lee et al., 2001). Such services can contribute to improved decision-making, situation awareness and information superiority, but the process of developing (and interacting with) ontologies often remains a difficult undertaking, typically the province of experienced ontology engineers. This apparent complexity in interacting with and exploiting the Semantic Web contributes to a potential usability problem that threatens to undermine the general acceptability of the Semantic Web to a variety of end-user communities. This section explores the origins of the proposed usability problem in terms of the apparent difficulty humans may have with the cognitive processing of logical information.

The study of human cognition has tended to emphasize its serial and deliberative aspects – the aspects that are most strongly associated with tasks involving highly sequential, stepwise problem-solving (e.g. logic and planning). Yet it is clear that the range of tasks to which the human mind is perhaps most suited is much broader than that. In fact, the kinds of tasks in which we, and most of our bio-behavioural brethren, seem to excel are precisely those that feature the rapid and adaptive execution of actions in the physical environment. They include, among other things, the ability to coordinate motor responses in light of changing patterns of multimodal sensory input, the ability to recognize complex stimulus configurations from background noise, and the ability to project from past experiences to a variety of actual (and counterfactual) future situations. All of these capabilities seem to be grounded in our possession of a computational device (a nervous system) that is capable of rapid pattern-recognition and pattern-completion, and it is precisely this class of device that seems to undergird most (if not all) of biological cognition. It is a class of device that is clearly adept at some tasks, but perhaps not so good at the kind of tasks that we typically regard as the hallmarks of human cognition:

> *"Such devices are adept at linking patterns of current sensory input with associated information… [They] prove extremely good at tasks such as sensori-motor coordination, face recognition, voice recognition, etc. But they are not well suited to deductive logic, planning and the typical tasks of sequential reasoning. They are, roughly speaking, 'Good at Frisbee, Bad at Logic'…" (Clark, 2004, pg. 28)*

Clearly, we are, at times, capable of engaging (very successfully) in tasks involving sequential reasoning and logical problem-solving, but the idea that our cognitive profile might not be terribly well suited to these kind of problems is highly significant. It is significant because the Semantic Web grew out of research into formal logics and many of the representational and technological commitments of the Semantic Web bear witness to this heritage. The knowledge representation language of the Semantic Web, for example, i.e. the Ontology Web Language (OWL), is based on a family of logics (called Descriptions Logics) that are decidable fragments of first-order logic (Smith et al., 2004). Inasmuch as human beings struggle with the principles of formal logic and logical reasoning, the logical underpinnings of the Semantic Web may present problems for human end-users, problems that may very well limit the exploitation of semantic technologies and lead to the rejection of otherwise robust, performance-enhancing solutions.

What evidence is there that human beings are actually bad at logic? Much of the study of human deductive reasoning has made use of logical systems – especially the propositional calculus – to evaluate people's performance with respect to logical problem-solving. The propositional calculus involves a small number of logical operators (i.e. *not*, *and*, *or*, *if…then*, *if and only if…then*), but most studies of human deductive reasoning have concentrated on the use of the conditional (i.e. *if…then*) operator (see Evans et al., 1993, for a full account of propositional reasoning research). The conditional operator supports conditional inferences of the form 'If P then Q', for example:

```
    If it is lunch-time (P), then Paula will eat some food (Q).
```

The inference rules for premises involving the conditional reveal two kinds of valid inference that can be made with statements of this sort: modus ponens and modus tollens. Modus ponens (see Figure 2-1) allows us to conclude that Q, when we are given P in the context of the premise 'If P then Q'. So, for example, if we are given the conditional '*if it is lunch-time, then Paula will eat some food*' and we are then told that '*it is lunch-time*', then we can validly conclude that '*Paula will eat some food*'.

Valid: Modus Ponens	
Premises	
If it is lunch-time, then Paula will eat some food.	*If P then Q,*
It is lunch-time.	*P*
Conclusion	
Therefore, Paula will eat some food.	*Therefore, Q*

Figure 2-1: Summary of modus ponens

The modus ponens form is the more obvious of the two valid inferences that can be made from the conditional. The other form, modus tollens, is not so obvious. In this case the rule states that if we are told that Q is false then we can infer that P is also false (see Figure 2-2).

Valid: Modus Tollens	
Premises	
If it is lunch-time, then Paula will eat some food.	*If P then Q,*
Paula does not eat some food.	*not Q*
Conclusion	
Therefore, it is not lunch-time.	*Therefore, not P*

Figure 2-2: Summary of modus tollens

Two other inferences can be drawn from premises involving the conditional, and they are both invalid inferences. They are called the 'affirmation of the consequent' and the 'denial of the antecedent'. In the affirmation of the consequent, we are told that Q is true and infer that P must also be true (see Figure 2-3). Clearly, this conclusion is invalid: there could be any number of reasons, other than the fact that it is lunch-time, why Paula may choose to eat food (it could be breakfast time, for example).

Invalid: Affirmation of the Consequent	
Premises	
If it is lunch-time, then Paula will eat some food.	*If P then Q,*
Paula eats some food.	*Q*
Conclusion	
Therefore, it is lunch-time.	*Therefore, P*

Figure 2-3: Summary of affirmation of consequent

In the case of the other invalid form, the denial of the antecedent, we are told that P is false and (invalidly) conclude that Q is also false (see Figure 2-4). As with the affirmation of the consequent, Paula may elect to eat food at various points in the day, and not all of them will necessarily be limited to lunch-time.

Invalid: Denial of the Antecedent	
Premises	
If it is lunch-time, then Paula will eat some food.	*If P then Q,*
It is not lunch-time.	*not P*
Conclusion	
Therefore, Paula does not eat some food.	*Therefore, not Q*

Figure 2-4: Summary of denial of the antecedent

How do people perform when presented with tasks requiring conditional reasoning of the form just described? There is a vast literature here (see Evans et al., 1993), but a number of things seem relatively clear: people will, at times, fail to make valid inferences, and they will often consider invalid inferences to be perfectly acceptable. In one early study, for example, Marcus and Rips (1979) examined the pattern of inferences for each of the four forms of the conditional (see Figure 2-5). Typically, most people will make the (valid) modus ponens inference, but the modus tollens inference appears much harder (only 50% of subjects make this inference). On the other hand, many subjects find the two invalid inferences perfectly acceptable – 21% of subjects make the denial of the antecedent inference and 33% make the affirmation of the consequent inference (both of which are, of course, logically invalid).

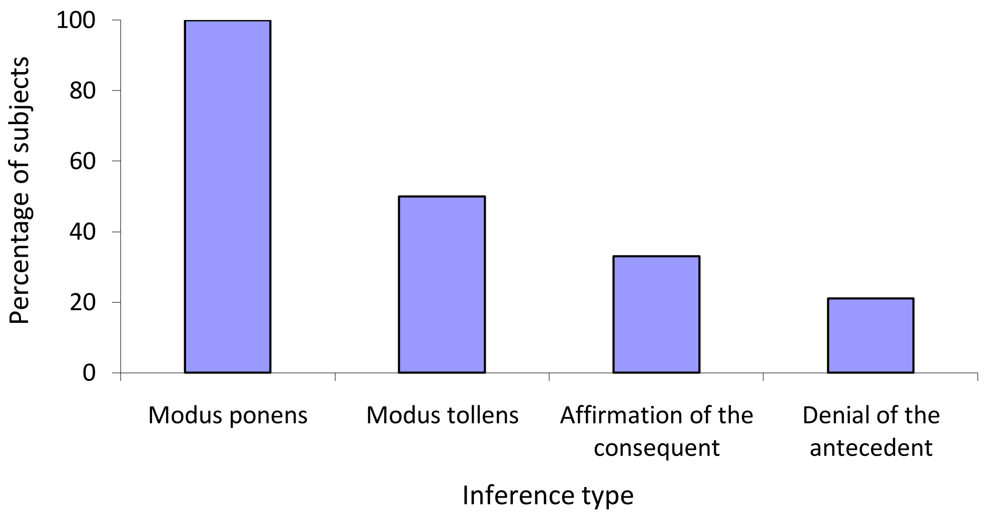


Figure 2-5: Percentage of subjects endorsing the various conditional inferences

A number of theories have been proposed to account for the performance of human subjects in conditional reasoning tasks. One theory argues that people are inherently rational, but that they make mistakes because they fail to fully understand the task, or because they misrepresent the task (Braine, 1978). Invalid inferences, on this account, occur because the interpretation of premises is made with respect to certain assumptions that feature as part of our everyday conversational discourse. Grice's (1975) cooperative principle, for example, maintains that a speaker will communicate what they think a listener needs to know in order to enable the listener to make sense of the speaker's utterances. The endorsement of this principle, by both the speaker and the listener, means that not all statements will be subject to an interpretation that supports logically valid conclusions outside a particular conversational context (e.g. in a contrived experimental setting). For example:

> *"If a speaker says 'It is raining, then Alicia will get wet' the hearer will assume, in the context of the conversation, that rain is the only likely event that will lead to Alicia getting wet. The hearer assumes that no other alternative Ps will play a role. So, during comprehension people make a reasonable assumption that modifies the premises. Having made this comprehension error, reasoning continues normally through the application of the various reasoning rules…" (Eysenck & Keane, 1995, pg. 414)*

Humans may therefore be capable of valid deductive reasoning (and the weight of empirical evidence does seem to suggest that humans are indeed rational), but a number of factors may conspire to undermine problem-solving success in certain situations. These include the kinds of assumptions people make when engaged in real-world (human-to-human) informational and social exchanges, and the level of experience they have with particular problem domains[2]. What is clear is

[2] One of the most researched effects on performance in the Wason selection task (a hypothetico-deductive reasoning task involving the use of conditionals) is the improved performance seen when subjects are given concrete content that draws on their practical experience with real-world problem domains (Eysenck & Keane, 1995). Abstract formulations of the Wason selection task (i.e. ones that do not make any contact with real-

that humans may not necessarily be well-equipped to deal with the idiosyncrasies of formal logic, particularly when it comes to the correct interpretation of logical statements and the execution of logically-valid inferences. Humans seem particularly vulnerable to the way in which logic problems are presented, and this no doubt exacerbates the more basic problem that people have with the mathematical symbology used to represent such problems.

Notwithstanding the research findings about people's ability (or inability) to cope with logical problems, it is clear that the logical grounding of the Semantic Web complicates the exploitation of Semantic Web resources. There can be little doubt that certain classes of users struggle with the technological and representational idiosyncrasies of the Semantic Web (see Section 3.1) and, on occasion, even experienced ontology engineers can experience difficulty with certain types of formalisms (Rector et al., 2004). What is required, it seems, is an ability to simplify the user's access to, and interaction with, Semantic Web resources. Ideally, we need to develop supportive interfaces that minimize training overheads and exploit a user's existing skills and competences with respect to the representation and communication of knowledge. It is precisely this kind of interface that has been proposed by members of the CNL research community, and it is towards this interface solution that we now turn.

world practical experiences) tend to lead to a pattern of errors that resembles those seen in conditional reasoning tasks.

3 Controlled Natural Languages

"In your heart you'd prefer to stick to Oldspeak, with all its vagueness and its useless shades of meaning. You don't grasp the beauty of the destruction of words."

George Orwell – 1984

3.1 Lost in Logic

As we saw in the previous section, the human mind may be (congenitally?) ill-equipped to deal with *certain types* of logic, and this may underpin (or at least compound) a basic problem in creating, interpreting, manipulating and otherwise interacting with the representational substructure of the Semantic Web. Irrespective of whether or not the logical underpinnings of the Semantic Web conflict with the cognitive profile of the human mind, it is clear (not least from our own pedagogical experiences within the University of Southampton and IBM UK) that competence with semantic technologies, particularly ontologies, can be difficult (and in some cases seemingly impossible) for certain classes of end-user to get to grips with (Kalyanpur et al., 2006; Rector et al., 2004).

What kinds of problems are typically experienced by users when working with ontologies? The most common problems identified by Rector et al (2004) include the following:

1. failure to make all information explicit — assuming that information implicit in names is "represented" and available to a semantic reasoner;
2. mistaken use of universal restrictions (i.e. ObjectAllValuesFrom), rather than existential restrictions (i.e. ObjectSomeValuesFrom), as the default form of local restriction;
3. open world reasoning ("the biggest single hurdle");
4. the effect of range and domain constraints as axioms ("the largest single source of errors after the open world reasoning problem");
5. the trivial satisfiability of universal restrictions — that "only" (i.e. ObjectAllValuesFrom) does not imply "some" (i.e. ObjectSomeValuesFrom), i.e. $\forall$ R C $\not\sqsubseteq$ $\exists$ R C;
6. the difference between defined and primitive classes and the mechanics of converting one to the other;
7. the difference between the linguistic and logical usage of "and" and "or";
8. confusion about the representation of "some not", i.e. $\exists$R($\neg$C) and "not some", i.e.$\neg$($\exists$RC);
9. expecting classes to be disjoint by default (this is essentially a sub-problem of problem 1); and
10. the difficulty of understanding subclass axioms used for implication.

A number of additional problems have been identified more recently by Kalyanpur (2006). They include:

- A $\doteq$ C was used, but A $\sqsubseteq$ C was meant;
- A $\sqsubseteq$ C;A $\sqsubseteq$ D was used, but A $\sqsubseteq$ C $\sqcup$ D was meant;
- domain(P,A); range(P,B) was used, but A $\sqsubseteq$ $\forall$ PB was meant;
- domain(P,A);domain(P,B) was used, but domain(P,A $\sqcup$ B) was meant.

One thing that clearly does not help users when it comes to the comprehension and production of logical statements is the use of conventional logical symbols, such as ∀, ∃ and ⊔ (I am sure some readers will already have encountered problems with the interpretation of the logical formulas expressed in the above lists!). The mathematical symbols used as part of formal logic can make ontological expressions difficult for non-logicians to read and interpret, and this has motivated a number of efforts to develop a more user-friendly format for the communication of ontological content (Horridge et al., 2006; Patel-Schneider et al., 2004). Manchester OWL Syntax (MOS) (Horridge et al., 2006), for example, was developed to provide an alternative syntax for ontologies in which the conventional symbology of formal logic was replaced by a number of intuitive keywords such as 'some', 'only' and 'not'. Recent user evaluation studies suggest that MOS is favourably received by non-logicians (Horridge et al., 2006), and it is currently being used in tools such as Protégé-OWL (Holger et al., 2004) to provide pseudo-natural language support for ontology developers. Clearly, formats such as MOS can help to attenuate some of the difficulties associated with the production and comprehension of semantically-potent statements, especially those using the notational apparatus of formal logic, but another means of bridging the apparent gap between the competences of human beings (at least those not fully conversant with formal logic) and the (re)presentational idiosyncrasies of Description Logic languages, e.g. OWL, is to use an interface language that draws on the somewhat obvious familiarity humans have with the medium of natural language. Let us turn therefore leave the realm of formal logic symbols (at least for a while) and look more closely at linguistically-oriented approaches to knowledge representation and communication.

3.2 Towards a More Natural Language

Natural language is, quite clearly, our preferred format for the external representation and communication of domain knowledge. It has a heritage that extends throughout the course of human evolution, and it is probably well-adapted to the cognitive profile of the human mind, serving as both a driver for cognitive change (i.e. driving the evolution of brains capable of coping with the demands of natural language), and perhaps itself evolving to meet the limitations and capabilities of the human cognitive system[3]. In light of these considerations, we would expect that natural language would be well-suited to its role as a representational and communicative vehicle for culturally-transmissible knowledge. Some commentators have even suggested that the functions of language, vis-à-vis human cognitive competence, go beyond its mere communicative role (Clark, 1997, 1998, 2006; Dennett, 1991). This supra-communicative view of language emphasizes the potential role that linguistic expressions may have in terms of augmenting, and indeed transforming, the otherwise limited capacities of the human mind:

> *"…language is in many ways the ultimate artefact. Not only does it confer on us added powers of communication; it also enables us to reshape a variety of*

[3] An interesting demonstration of this potential capability for linguistic evolution is provided by Hutchins and Hazelhurst (1991). They showed that symbolic artefacts (e.g. words) are capable of undergoing evolution so as to support the problem-solving capacities of successive generations of agents (connectionist networks in Hutchins and Hazelhurst's simulation). In a study where symbolic structures were selected based on their contribution to problem-solving success, Hutchins and Hazelhurst (1991) were able to show that later generations of problem-solving agents were able to learn environmental regularities that could not be learned by their predecessors. This was despite the fact that the problem-solving agents in question remained *unchanged* throughout the simulation (only the external symbolic structures used to *represent* domain knowledge were subject to evolutionary change).

difficult but important tasks into formats better suited to the basic computational capacities of the human brain. Just as scissors enable us to exploit our basic manipulative capacities to fulfil new ends, language enables us to exploit our basic cognitive capacities of pattern recognition and transformation in ways that reach out to new behavioral and intellectual horizons." (Clark, 1997, pg. 193)

Given the familiarity humans have with natural language, it makes sense to consider whether natural language interfaces could be developed to circumvent some of the apparent difficulties that humans have with the logic-infected representational formalisms of the Semantic Web. This is precisely the goal of a number of CNLs that have been applied to support user interaction with the Semantic Web (Cregan et al., 2007; Fuchs et al., 2006a; Funk et al., 2007a; Hart et al., 2007; Kuhn, 2006). CNLs are a subset of natural languages that impose a number of restrictions or constraints on both the generation and interpretation of natural language expressions. These constraints typically assume the form of rules that help to reduce both the ambiguity and complexity of the full natural language (Kittredge, 2003). In essence, CNLs attempt to provide a medium for knowledge representation that exploits the intrinsic familiarity of linguaform representations as the primary vehicle for human-human knowledge transfer. In conjunction with supportive user interfaces (e.g. Kaufmann et al., 2006), CNLs can potentially improve the usability of the Semantic Web, perhaps even enabling casual users (i.e. those who have limited or no expertise with semantic technologies) to create, edit and exploit Semantic Web content with a minimum amount of training. It is this improved usability potential that makes CNLs of such interest to a number of Semantic Web research programs (Smart et al., 2008a).

One (perhaps) obvious question at this point is why we would want to use controlled or restricted variants of natural language rather than the full or unrestricted version. The main reason is that natural language expressions have a number of sources of ambiguity that makes their precise interpretation difficult, if not impossible. One source of ambiguity derives from the use of anaphoric references. In natural language it is common to use words (such as pronouns) to refer back to some entity that was introduced earlier in the text. For example:

```
Margaret invited Susan for a visit, but she told her she had to go
to work.
```

In this sentence, the referents of the pronouns 'she' and 'her' are ambiguous. Do they refer to Margaret or Susan?

In some cases, background knowledge can be used to resolve anaphoric references. For example, our knowledge that the pronoun 'he' refers to something that is male and that 'Mark' is a male name, allows us to 'easily' identify who had to go to work in the following sentence:

```
Mark invited Susan for a visit, but he told her he had to go to
work.
```

Inferences of this kind are probably within the reach of extant NLP capabilities, but other types of anaphoric linkage require more sophisticated inferences. For example, in the sentence:

```
Kate lent Sally her car. She thought cycling was healthier.
```

we can infer that the ‘she’ probably refers to ‘Kate’ based on our commonsense knowledge of the world (Garnham, 1985).

Other forms of ambiguity are also prevalent. They include cases of semantic ambiguity:

```
John kissed his wife, and so did Sam. (Did Sam kiss John's wife or
his own?)
```

and lexical ambiguity:

```
I saw a bat. (Did I see an animal or an item of sports equipment?)
```

To address these issues, CNLs place restrictions on the generation and interpretation of natural language sentences in order to minimize or eliminate ambiguity:

> *“Controlled Natural Languages are subsets of natural language whose grammars and dictionaries have been restricted in order to reduce or eliminate both ambiguity and complexity.” (Schwitter, 2007)*

Such restrictions are clearly beneficial for the machine-based computational processing of linguistic expressions; they support machines with respect to the interpretation of sentential structures that would otherwise be difficult (or impossible) to interpret correctly.

In summary then, CNLs are restricted variants of natural language that establish a sensible contact with the linguistic competences and capabilities of natural (public) language users. They are restricted in the sense that they lack some of the flexibility and scope of natural languages, but such sacrifices seem trivial when compared to the very real advantages to be gained from the unambiguous interpretation of CNL statements, a feature that makes CNLs interpretable by machine processors and capable of serving as a natural language interface to the Semantic Web. Restricted variants of natural language have been around for some time. They were first introduced in the 1930s by linguists who sought to create a ‘minimal’ variety of English that would be accessible to non-native English speakers. They have also surfaced in a number of fictitious works, such as the NewSpeak in Orwell’s (1949) book, ‘Nineteen Eighty-Four’, and SpeedTalk in Heinlein’s (1949) novella, ‘Gulf’. In fact, during his lifetime Orwell both supported and ultimately rejected a form of constrained or constructed English. His concern was that language reflected more than just social conditions; it also contributed to the very creation of those conditions and could thus be used as an instrument of political power, influence and oppression – the removal of words representing the ideas of freedom, rebellion and independence in Orwell’s NewSpeak language is a direct reflection of the attempt to manipulate the minds of the masses by the totalitarian regime. Language is thus portrayed (by Orwell) as something that is directly related to the cognitive (and behavioural) potential of the individuals who wield it; it is a sentiment that is oddly reminiscent of more recent arguments about the transformative, cognitive-enhancing potential of natural language, i.e. that idea that language serves more than just a communicative function (see above).

3.3 Languages and Linguini

CNLs, like pasta, come in a rather bewildering variety of forms. To review every extant CNL, let alone every conceivable CNL, would take us far beyond of the scope of this report (see Pool, 2006, for a review of some 32 distinct CNLs). As such, the current section aims to review those languages that are most closely associated with the Semantic Web. In most cases, these languages have been used to support the authoring of Semantic Web ontologies by translating CNL expressions into OWL models (see Section 4). In some cases, the translation process is bidirectional; in which case the CNL can support round-trip authoring and ontology verbalization (see Section 6). Some CNLs were developed specifically for the purpose of creating OWL ontologies. These include ACE-OWL (Kaljurand & Fuchs, 2006a, 2006b), CLOnE (Funk et al., 2007b) and SOS (Cregan et al., 2007). All of these languages are described, at least in summary form, in the following sections.

3.3.1 Attempto Controlled English (ACE)

Attempto Controlled English (ACE) is a subset of English that is designed to support the unambiguous specification and communication of domain-relevant knowledge, including technical specifications. Whilst it is easily understood by speakers of the base language, it is designed to be unambiguously interpretable into Discourse Representation Structures (DRSs), which are syntactical variants of first-order logic. ACE thus has the same formal properties as first order logic and is consequently machine-processable. See the ACE project website[4] for more information about ACE.

Numerous studies have used ACE as an interface language for the Semantic Web (Fuchs et al., 2006b). Its applications include ontology authoring (Fuchs et al., 2006a; Kuhn, 2006), semantic querying (Bernstein et al., 2004), reasoning (Fuchs & Schwertel, 2003; Kuhn, 2007) and ontology verbalization (Kaljurand & Fuchs, 2007). Of all the CNLs currently available, ACE is probably the most extensively studied and used. Its popularity means that numerous tools have been developed to support the generation and computational processing of ACE texts. Notable among these is the Attempto Parsing Engine (APE), which consists of a definite clause grammar written in Prolog. The APE tool is available via a Web client interface[5], as well as an XML Web service[6], and it enables ACE texts to be converted into DRSs. In addition to APE, other tools include an ACE reasoner[7] (which allows users to reason about the content of ACE texts), an OWL verbalizer[8] (which converts an OWL ontology into an ACE text) and AceWiki (Kuhn, 2008) – a semantic wiki prototype that uses an ACE authoring tool to support collaborative ontology development via the medium of natural language.

ACE has recently been adopted as the controlled language for the EU FP6 Network of Excellence Reasoning on the Web with Rules and Semantics (REWERSE) initiative[9] (Fuchs et al., 2006a). This initiative focuses on the development of advanced reasoning capabilities for the Semantic Web; it specifically aims to "develop a coherent and complete, yet minimal, collection of inter-operable reasoning languages for advanced Web systems and applications" (see REWERSE homepage). REWERSE proposes a sublanguage of ACE, called ACE-OWL, which is specifically targeted at the ontology development process (Kaljurand & Fuchs, 2006a; Kuhn, 2006). The main advantage of ACE-OWL, as opposed to standard ACE, is that there is a bidirectional mapping between OWL (DL) and ACE-OWL (Kaljurand & Fuchs, 2006a). This allays concerns about the (excessive) expressivity of ACE relative to OWL (ACE is equivalent to first-order logic and therefore more expressive than OWL), and it opens up the possibility of round-trip ontology authoring in which a domain ontology could be

[4] http://attempto.ifi.uzh.ch/site/
[5] http://attempto.ifi.uzh.ch/ape/
[6] http://attempto.ifi.uzh.ch/site/docs/ape_webservice.html
[7] http://attempto.ifi.uzh.ch/site/main.php?text=tools/race_demo.html
[8] http://attempto.ifi.uzh.ch/site/docs/verbalizing_owl_in_controlled_english.html
[9] http://rewerse.net/

edited using other tools (e.g. Protégé). Some problems with ACE-OWL include the complexity associated with some forms of bidirectional mapping: since ACE-OWL aims to support bidirectional translation, it must be able to cope with some types of OWL formalism that do not translate well into linguistic constructions, at least ones that humans find easy to read. In addition, although ACE-OWL supports many OWL formalisms, it does not currently support enumerations (OWL's oneof) and has limited support for datatype properties (see Kaljurand, 2007, for more information).

3.3.2 Rabbit

Rabbit (Dolbear et al., 2007; Hart et al., 2007) is a CNL that was developed by the national mapping agency of Great Britain, the Ordnance Survey[10]. Like many of the CNLs described in this section, Rabbit can be converted to OWL in order to provide natural language support to ontology authors. Ontology development is not, however, the primary objective of Rabbit, and not all Rabbit expressions have their analogues in OWL. Rabbit is primarily a vehicle for capturing, representing and communicating knowledge in a form that is easy for domain experts (as well as other users) to use and understand. It is not, therefore, limited to the constructs that are provided by a particular ontology representation language (OWL in this case); rather, it aims to support the representation of all knowledge that is deemed important by a domain expert. In some cases, this means that the translation to OWL is only partially successful: some constructs, such as 'usually', are difficult, or even impossible, to represent in OWL. Not surprisingly, therefore, Rabbit is considered to be the "authoritative source of the ontology" (Hart et al., 2007); OWL is deemed to play an important role in exploitation, but it is Rabbit that serves as the primary representational mechanism for domain knowledge:

> *"Our intention is for Rabbit to exist in a complementary manner with OWL, with Rabbit being the primary means to author ontologies and OWL as the primary means to enable machine interpretation." (Hart et al., 2007)*

Rabbit is thus more than a "syntactic veneer" for OWL; it is a knowledge representation language in its own right (some of) which can be translated to OWL to ensure compatibility with ongoing Semantic Web efforts. With the addition of knowledge engineers and supportive editing tools (e.g. ROO – see Section 4.4), Rabbit is deemed to provide a strong foundation for knowledge acquisition and modelling that (crucially) "allows the domain expert to be in charge of the authoring process and other domain experts to understand the content" (Hart et al., 2007)[11].

As a knowledge representation language, Rabbit seems to serve a number of functions. Firstly, there is a need to represent knowledge in a way that a multitude of users (particularly domain experts) can understand. This may entail a number of syntactic simplifications as well as background assumptions regarding the semantics of particular expressions (e.g. all primitive classes sharing a superclass are assumed to be disjoint). Let us call this particular function the communicative function. Secondly, there is a need to capture domain knowledge that cannot always be expressed in

[10] http://www.ordnancesurvey.co.uk/

[11] It is important to point out that not all knowledge capture goals may be satisfied by allowing a domain expert to take charge of the authoring process. In some cases, domain experts do not have conscious introspective access to domain-relevant knowledge and this prohibits the direct expression of knowledge via verbal routes. In these cases, we often have to resort to alternative knowledge elicitation strategies (e.g. repertory grid techniques), strategies that are capable of exposing bodies of tacit or implicit knowledge (Shadbolt & Burton, 1990; Shadbolt et al., 1999).

OWL (recall that lossless conversions to OWL are not always possible). Assertions such as 'A River Channel usually contains Water' are deemed to reflect important facts about a domain, even though their semantics cannot be reliably represented in OWL. Let us, therefore, call this function the representational function. It is this function that undergirds the claim that Rabbit is the "authoritative source of the ontology". Together, the two functions (communicative and representational) make a compelling case for the use of Rabbit as a base language for knowledge representation, perhaps even motivating its use as a replacement language for OWL in the context of the Semantic Web[12]! I will have more to say about this issue in Section 7.

3.3.3 Controlled Language for Ontology Editing (CLOnE)

CLOnE (Funk et al., 2007b; Tablan et al., 2006) was developed at the University of Sheffield to provide a user friendly language for ontology authoring. Unlike Rabbit, it does not aim to provide an independent knowledge representation language; instead, it functions solely as a means of enabling users to create simple ontologies using natural language expressions. The key features of CLOnE are the following (Funk et al., 2007b):

1. CLOnE requires only one interpreter or runtime environment, the Java 1.5 JRE.
2. CLOnE is a sublanguage of English.
3. As far as possible, CLOnE is grammatically lax; in particular it does not matter whether the input is singular or plural (or even in grammatical agreement).
4. CLOnE can be compact; the user can create any number of classes or instances in one sentence.
5. CLOnE is easy to learn by following examples and a few guiding rules, without having to study formal expressions of syntax; nonetheless, the basic (declarative) implementation of CLOnE uses only 10 syntactic rules.
6. Any valid sentence of CLOnE can be unambiguously parsed.

A CLOnE text consists of CLOnE sentences, each of which consists of 'key-phrases' and 'chunks'. Key-phrases correspond to the fixed language expressions of CLOnE. They are stored in a gazetteer. Chunks are used to identify the elements of the ontology to be derived from a CLOnE text, i.e., instances, properties and values. All of the text that does not match a key-phrase is expected to contribute to a chunk.

The analysis of a CLOnE text consists of a pipeline of processing resources provided by the General Architecture for Text Engineering (GATE) framework[13] (see Figure 3-1). The first four components in this pipeline are fairly standard NLP tools which add linguistic annotations to the text. These are followed by a series of processing steps in which key-phrases and chunks are identified. Finally, the

[12] This is a claim that has been made by some proponents of CNLs in the ITA.

[13] GATE is a suite of components that supports text-based NLP and the development of language engineering applications. More information can be found on the GATE website (http://gate.ac.uk/). Good overviews of GATE can be found in Bontcheva et al (2004) and Cunningham et al (2002).

CLOnE Engine is a Java Annotation Patterns Engine (JAPE) transducer[14] that interprets the CLOnE sentence with respect to the background ontology. This is necessary in situations where a CLOnE sentence could match more than one syntactic rule. For example, consider the following syntactic rule (used to specify the relationships between instances or classes):

```
    CLASSES/INSTANCES are DESCRIPTION PREPOSITION CLASSES/INSTANCES.
```

This rule can clearly take either classes or instances as its arguments, and thus the sentence chunks (the capitalized components) need to be matched to the existing elements of the ontology. If the chunks are resolved to classes, then the CLOnE engine creates a new property with the relevant domain and range restrictions; if, however, the chunks are resolved to instances then a new property value is asserted between the existing instances.

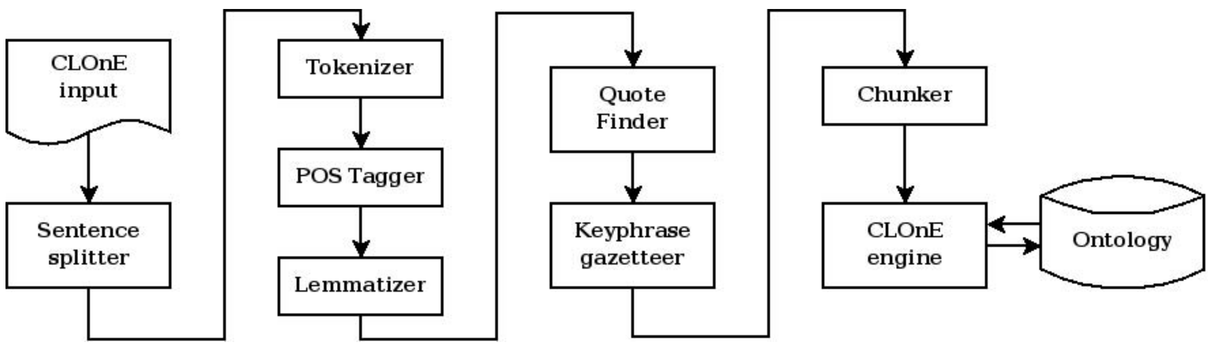


Figure 3-1: The CLOnE pipeline

Following the interpretation of a sentence, a single syntactic rule can be applied to the sentence to generate the equivalent OWL formalism. Ultimately, each valid CLOnE sentence is unambiguously mapped into OWL/RDF-S via using just 10 syntactic rules. This makes CLOnE one of the simplest CNLs currently available for ontology authoring. Such simplicity, however, comes at a price. The fact is that CLOnE can only be used to create a limited number of OWL formalisms, including a taxonomy of classes, instances of classes, properties and their associated values (Tablan et al., 2006). For any serious ontology engineer, this list of formalisms is worryingly small; it limits the kind of ontologies that can be developed using CLOnE to the simple and lightweight (ontologies that may not actually necessitate the use of a natural language interface in the first place!). Of course, simplicity is not always a vice when it comes to knowledge representation and the Semantic Web (Rousset, 2004), and Tablan et al (2006) argue that extending CLOnE to cover the full range of OWL formalisms would reduce the overall usability of the language:

> *"A CL [Controlled Language] that covers all the features of an ontology representation language such as OWL is possible but probably not desirable – as it would be difficult to learn and use. A simple controlled language, such as the one described here, requires essentially no training." (Tablan et al., 2006)*

Usability is, of course, a key evaluation metric for any CNL, but the general concern with minimally expressive languages, such as CLOnE, is that their usability becomes somewhat redundant in light of

[14] JAPE is essentially a language for creating regular expressions that are applied to linguistic annotations in a text corpus. A JAPE grammar consists of a set of phrases, each of which consists of a set of pattern/action rules (see the GATE User Guide for more information - http://www.gate.ac.uk/sale/tao/index.html).

the complexity of the ontological formalisms that they are capable of defining. Is it the case, for example, that a CNL interface is still warranted given the range of ontological formalisms that can be created with the CNL? Would it not be equally likely that a simple graphical interface could support users to the same extent without introducing the computational overhead associated with textual processing? This concern has been addressed, at least to some extent, in a recent user evaluation study involving CLOnE (Funk et al., 2007b). The study compared the use of CLOnE with the Protégé-OWL editor (Holger et al., 2004) using the System Usability Scale (SUS) (Brooke, 1996). Subjects were asked to complete a number of ontology editing tasks, such as creating classes, subclass hierarchies, instances of classes and relationships between instances of classes – essentially the tasks supported by CLOnE expressions. The results of the study suggested that CLOnE was rated more favourably than Protégé with respect to usability scores (unfortunately the results of statistical analyses to back-up this conclusion seem to be omitted), and it may therefore be the preferred tool when editing simple/lightweight ontologies. Obviously, this does not suggest the superiority of CLOnE relative to all graphical editing environments. Protégé was specifically developed to support the creation of complex ontologies and it is not necessarily designed to be used by the same group of users as targeted by the CLOnE CNL (casual or inexperienced users). Perhaps another Graphical User Interface (GUI) designed to support the same (limited) set of ontology editing tasks as CLOnE would fare somewhat better then a sophisticated ontology editor.

Another concern about the aforementioned user evaluation study relates to the characteristics of the subject pool. In this case, 60% of the subjects (9 out of 15) were selected from the GATE development team, and they were therefore intimately familiar with NLP technologies. Fortunately, the results do not suggest any bias between the GATE and non-GATE subjects (although, again the results of statistical analyses are lacking). Pre-test scores that aimed to evaluate the experience users had with Semantic Web technologies revealed that GATE users may have had more experience than their non-GATE counterparts (again no stats!); however, both groups seemed to have moderate levels of experience (scores ranging midway between no knowledge and perfect knowledge). It is not clear whether different results might be obtained with subject pools consisting of users with more or less experience of semantic technologies.

3.3.4 Sydney OWL Syntax (SOS)

SOS (Cregan et al., 2007) is a CNL for ontology development that is both complete (all of the OWL[15] axioms and assertions are supported) and consistent (all ontologies in SOS can be fully represented in OWL). SOS is a relatively new specification and there are thus few research papers at the present time demonstrating or evaluating its use in applied settings.

A number of features distinguish SOS from other CNLs described in this section. One feature is the aforementioned consistency of SOS with OWL (at least OWL 1.1). This distinguishes SOS from languages such as Rabbit and ACE that are more expressive than OWL. Another feature of SOS that distinguishes it from languages like Rabbit and CLOnE is the emphasis it places on variables in sentential structures. This can sometimes make SOS less readable than pure natural language, and it probably increases the cognitive burden on end users when it comes to processing SOS statements.

[15] SOS is specifically designed to support OWL 1.1 (http://www.w3.org/Submission/2006/SUBM-owl11-owl_specification-20061219/).

The following is an example of a sentence that uses variables to express an OWL inverse functional property (owl:InverseFunctional):

```
If X has Y as a son then Y is the son of only X.
```

Although the use of variables is probably best avoided, the creators of SOS claim that "some OWL statements could not be expressed clearly in CNLs without using variables" (Cregan et al., 2007). Such a claim is probably over-stated. It does seem possible to represent the full range of OWL formalisms without resorting to variables, and languages like Rabbit would seem to prove the point. The following, for example, expresses an inverse functional property in Rabbit

```
1 Postcode is assigned to an Address.
```
[16]

Other than its use of variables, there are a number of other features that potentially limit the usability of SOS. One minor point is that SOS is designed to support OWL 1.1 (Patel-Schneider & Horrocks, 2006), which is not yet a full World Wide Web Consortium (W3C) Recommendation. As such, the language may change over time as the OWL 1.1 specification evolves. Another problem concerns the limited use of NLP technologies or interpretation rules that might otherwise permit the use of anaphoric references. In SOS, every sentence is "translated as a unit, without reference to any other statement in the ontology, or any other background or linguistic knowledge" (Cregan et al., 2007). This makes the language somewhat stilted, although to be fair very few of the other CNLs support advanced linguistic processing or anaphor resolution[17]. SOS is also rigid with respect to the syntactic form adopted for specific expressions. For example, outside of SOS there are multiple ways of saying that a man must necessarily be a person, i.e. subClassOf(Man, Person). The following are just two examples:

```
If X is a Man then X is a Person.

Every Man is a Person.
```

SOS does not support this flexibility with respect to the expression of what are, in effect, semantically-equivalent statements – "there is only one Sydney OWL Syntax form for each OWL form" (Cregan et al., 2007).

3.4 Summary

This section has provided a brief (very brief!) overview of some of the more popular CNLs that have been used in the context of the Semantic Web. Other CNLs that have been omitted for reasons of brevity include Processable English (PENG) (Schwitter, 2005a, 2005b; Schwitter & Tilbrook, 2004, 2006), Lite Natural Language (Bernardi et al., 2007), Boeing's Computer-Processable Language (Clark et al., 2005) and Sowa's Common Logic Controlled English[18]. A comparative evaluation of all

[16] This example is taken from Dolbear et al (2007).
[17] Even if a CNL provides support for anaphoric resolution, it is unlikely that the use of anaphors could be preserved in the translation to OWL. This is because the resolution of anaphoric references depends on the serial organization of sentences within a text. Since ontologies do not make any commitment with respect to the serial order of axioms and assertions, it is unlikely that the original use of anaphors could be preserved in a CNL, especially in round-trip ontology authoring applications.
[18] See http://www.jfsowa.com/clce/specs.htm

available CNLs is beyond the scope of this report; nevertheless, it is possible to at least identify the kinds of evaluative criteria that might be explored in the context of future reviews. These include:

1. **Expressivity.** How expressive is the CNL? Is it at least as expressive as the family of logics used by the Semantic Web community? What types of semantic entailments can be supported by assertions made using the CNL?
2. **Usability.** How usable is the CNL? Have specific user evaluation studies been carried out? If so, are the studies empirically and analytically sound? Does the CNL require special training? Is the CNL easy to understand, not just easy to read?
3. **Support for Knowledge Types and Structures.** What types of knowledge can be captured by the CNL? Does it include support for the representation of instances (individuals in OWL), the representation of semantically heterogeneous relationships (properties, associations, etc.) and the representation of dynamic rule-based knowledge (as opposed to static knowledge structures that capture the relationships between various concepts)?
4. **Grammatical Flexibility.** Does the CNL impose specific grammatical constraints on the user? Just how constrained or controlled is the CNL? Does it permit a degree of flexibility when it comes to the expression of semantically-equivalent statements?
5. **Use of NLP Technologies**. What NLP technologies are exploited as part of the processing associated with the CNL? How mature, robust and reliable are these technologies? Are they freely available? Does the introduction of any NLP technologies impose particular constraints (or liberties) on the user when it comes to creating and editing a CNL text?
6. **Tool Support.** What tools and technologies are available to assist with the creation, manipulation and exploitation of CNL texts, constructs, models, etc.
7. **Semantic Web Compliance.** Is the CNL compliant with Semantic Web standards and technologies? Can it be used as an interface language for ontology construction? Does the language provide complete coverage of the range of formalisms used in ontology engineering? There are, in fact, two aspects to Semantic Web compliance. They are:
    a. **Consistency.** Are all models developed using the CNL capable of being fully represented in the ontology representation language?
    b. **Completeness.** Are all of the axioms and assertions of the ontology language supported?

A complete evaluation of CNLs with respect to these criteria would provide important insights into the statistical associations and dependencies between various features of the CNLs. For example, it would be interesting to know whether languages that score high on expressivity criteria score also low on usability criteria (in general it would be desirable to develop regression models that highlight the contribution of each variable to the usability criterion). Although we must wait for such studies to become available, there have been a number of attempts to compare CNLs. The most recent study, undertaken as part of the effort to work towards a common CNL syntax for OWL 1.1[19], is particularly relevant here since it targets 3 of the 4 languages (ACE, Rabbit and SOS) described in this section (Schwitter et al., 2008). In order to explore the similarities and differences between CNLs, Schwitter et al (2008) compared the rendering of OWL 1.1 axioms in ACE, Rabbit and SOS using a domain ontology for 'Buildings and Places', which was previously developed by domain experts at

[19] See http://code.google.com/p/owl1-1/wiki/OwlCnl for more information about the OWLCNL specification effort.

the Ordnance Survey[20]. As an example of the different renderings provided by the CNL syntaxes consider the rendering of the asymmetric object property 'is-larger-than' in each of the 3 CNLs:

```
ACE: If something X is larger than something Y then Y is not larger
than X.

RABBIT: The relationship "is larger than" is asymmetric.

SOS: If X is larger than Y then Y is not larger than X.
```

Two key differences should be immediately apparent from these renderings. Firstly, ACE and SOS rely on the use of variables to capture information, whilst Rabbit does not. Secondly, Rabbit speaks about the ontology and the nature of its properties; it uses the terminology used when talking about ontological constructs, e.g. asymmetric properties, and it is therefore somewhat more dependent on a knowledge of ontological constructs than is perhaps the case for the other two CNLs, i.e. SOS and ACE. There seems to be a genuine difference of opinion with respect to the importance of this difference between the CNLs. Proponents of Rabbit seem to assume that the use of ontological terms is necessary and that the user/reader needs to the educated in the meaning of such constructs, or at least aided in the ontology design process by an experienced knowledge engineer. Proponents of other approaches, such as ACE, seem to reject this assumption. They argue that the use of ontological terms undermines the usability of the language and that such terms can be avoided by the use of alternative linguistic expressions:

> *"We believe that ontological terms like 'property', 'range', or 'subclass' are unknown or unclear to most [users]…We show that it is possible to avoid them…For example, instead of saying something like 'man is a subclass of human' that uses the ontological term 'subclass', we can simply say 'every man is a human' which does not use any special terms." (Kuhn, 2008)*

Schwitter et al (2008) identify a number of other differences between ACE, Rabbit and SOS in the context of their comparative analysis. They include:

1. **Syntactic/Lexical Style:** ACE chooses to hyphenate noun phrases (e.g. 'river-stretch'), whereas Rabbit and SOS allow non-hyphenated forms (e.g. 'river stretch').
2. **Representation of mathematical constraints:** Rabbit assumes that the user will have some basic knowledge of the constructs and axioms used in ontology development, e.g. an understanding of what it means for something to be an asymmetric property; both ACE and SOS try to avoid making assumptions about the background knowledge of the CNL user.
3. **Assumed involvement of knowledge engineers in the ontology development process**: Rabbit explicitly endorses the cooperation between domain experts and knowledge engineers; ACE does not do this and, in fact, tries to eliminate knowledge engineers altogether. SOS is neutral with respect to this issue.

We have now completed our tour of extant CNLs. The next 3 sections focus on some of the capabilities of CNLs, as well as their full natural language counterparts, when it comes to specific Semantic Web activities, e.g. ontology editing, semantic querying and ontology verbalization.

[20] http://www.ordnancesurvey.co.uk/ontology/v1/BuildingsAndPlaces.owl

4 Natural Language Ontology Editors

4.1 Overview

Natural languages promise to support human end-users with respect to the creation, editing, and exploitation of knowledge in a variety of application contexts. However, full (unconstrained) natural languages suffer from a number of problems, e.g. the vagaries of semantic and lexical ambiguity (see Section 3.2), and this necessitates the development of languages that impose specific constraints on the linguistic expressions that can be used to communicate domain-relevant knowledge. Such constraints, however, bring their own cost: they oblige users to learn the grammatical rules associated with the CNL. The main concern here is that the effort required to learn the CNL might exceed the effort required to learn other approaches to ontology development. It is, after all, not entirely clear based on the range of empirical studies currently available whether the time taken to learn a specific CNL, ACE say, is any more or less effortful than learning how to use a graphical knowledge editing environment such as Protégé.

One way in which users can be supported with respect to the full exploitation of CNLs is to design user interfaces that assist users with regard to the creation and editing of CNL expressions. A number of user interfaces are now available that attempt to guide the user through the process of entering CNL expressions as the basis for knowledge content creation. While these interfaces have not, in most cases, been subjected to rigorous user evaluation, it is probably fair to say that such development efforts are a step in the right direction when it comes to the provision of user-friendly ontology development tools.

The current section reviews the state-of-the-art with respect to natural language ontology authoring environments. The review is by no means extensive and many systems have been omitted for the sake of brevity. Nevertheless, the current section does provide a tantalizing glimpse of the current technologies currently available for ontology development using the medium of natural language.

4.2 GINO

GINO (Bernstein & Kaufmann, 2006) is an ontology editor that allows end-users to create RDF/OWL ontologies using the medium of natural language. GINO does not use any of the established CNLs, such as those discussed in Section 3.3; instead, it supports the entry of natural language expressions using a grammar that is, to a large extent determined by the elements of the loaded ontologies. GINO is an extension of GINSENG (see Section 5.7), which provides a natural language query interface to ontologies, and, as with GINSENG, GINO relies on a "simple static sentence structure grammar which is dynamically extended based on the structure and vocabulary of the loaded ontologies" (Bernstein & Kaufmann, 2006)[21]. This combined grammar (consisting of static and dynamically generated elements) is used to parse natural language sentences, as entered by the user, and it also serves to constrain and guide user input – informing the user of valid succession words/phrases in a sentence. The advantage of this guided input is threefold. Firstly it assists the user with respect to the entry of grammatically-valid natural language expressions; it therefore

[21] See Section 5.7 for more information about the dynamic, ontology-driven construction of CNL grammar rules.

obviates the need for the user to specifically learn the grammar associated with a CNL. Secondly, it prevents the user from entering statements that could not be interpreted or translated into an underpinning knowledge representation format, such as OWL; it therefore eliminates the problems associated with the potential mismatch between the representational capabilities of the natural language interface and the machine-readable ontology language. Thirdly, it prevents the user from entering information that might invalidate the logical consistency of the model. The statements entered by the end-user can be validated using a semantic reasoning agent, and the user can be alerted if any problems are encountered[22]. The main disadvantage of GINO's approach to the dynamic construction of a CNL grammar is that "GINO does not use any predefined lexicon beyond the vocabulary that is defined in the static sentence structure grammar and provided by the loaded ontologies" (Bernstein & Kaufmann, 2006). As such, the range of natural language expressions that the user can exploit at any particular time is limited by the ontologies that are loaded into the editing environment.

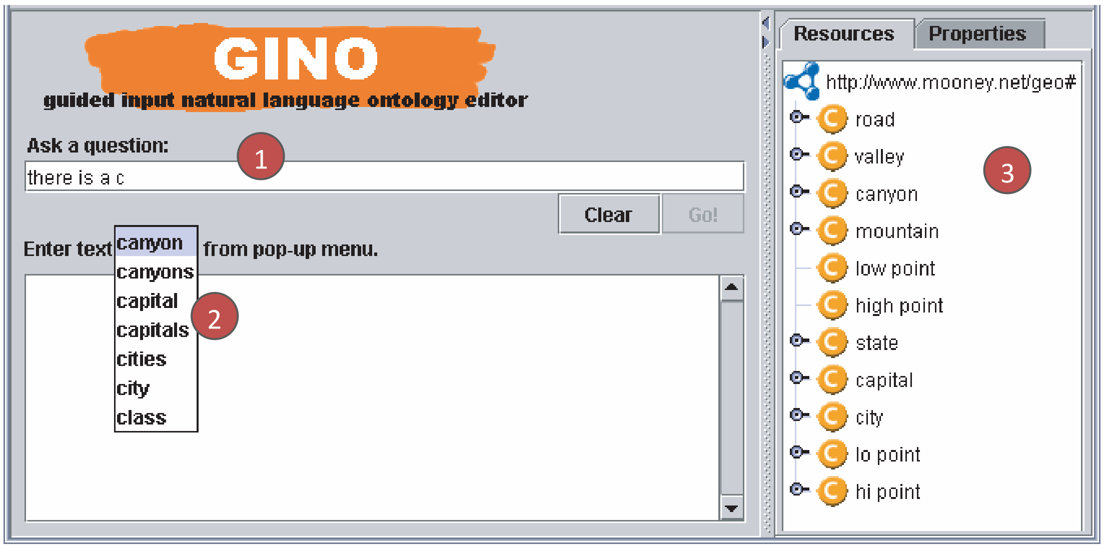


Figure 4-1: The GINO user interface

Figure 4-1 shows a screenshot of the GINO user interface. The user enters knowledge statements via the text field labelled (1) in Figure 4-1. Based on the underlying grammar, the system parses the user's entry and computes valid successor words/phrases. These successor elements are presented to the user in the form of a pop-up intellisense menu (2), from which the user can select an item using arrow keys, mouse actions or auto-completion capabilities. Obviously, as the user adds textual elements to the sentence, and thereby completes the knowledge statement, the range of choices for valid succession elements becomes progressively limited. In this way, GINO guides the user though the set of possible sentences collectively defining the totality of possible knowledge statements for a particular ontology. When the user completes a knowledge statement using the CNL syntax, the sentence is translated to RDF triples capturing the semantic content of the statement. Text entries that are invalid, according to the grammatical rules associated with the loaded ontologies, are not

[22] In fact, recent work on a symptom ontology (Baclawski et al., 2004) suggests that the user could be presented with more than just error information; they could be presented with explanatory information that pinpoints the precise nature of the problem and offers advice as to how to remedy the problem. In some cases, such feedback could be extended to include other types of information, e.g. communicating information about entailments, checking that textual entries correspond to the user's intended meaning, and so on.

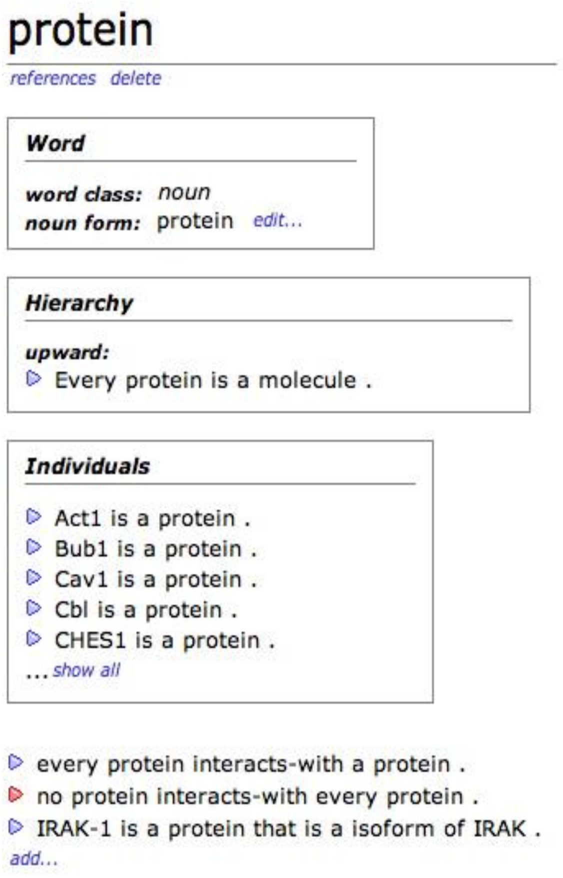
accepted by the system. Users who are familiar with other types of ontology editing environment, particularly graphical editing environments such as Protégé, can use the graph structure on the right hand side of the interface window (3) in order to edit or refine knowledge content using alternative, graphical, techniques.

4.3 AceWiki

AceWiki (Kuhn, 2008) is a prototype application that co-opts the principles of a semantic wiki[23] with the usability features of the ACE CNL. The main goal of AceWiki is to enable casual users, users not necessarily familiar with semantic technologies, to participate in the process of collaborative ontology development using a combination of a Web-based interface and natural language editing system.

Figure 4-2 depicts a screenshot of an AceWiki page describing the concept of a 'protein'. The first box shows the linguistic information associated with this entity. The other two boxes contain sentences that follow a certain pattern, i.e. hierarchy statements and individual assignments. Finally, the sentences at the bottom of the page assert additional information about the concept under discussion. All the text that is not in italics is ACE, and any sentences/statements that are preceded by a blue triangle also have a corresponding OWL representation. As such, we can see that most of the information on this wiki page, with the notable exception of the statement 'no protein interacts-with every protein', is represented in both ACE and OWL. AceWiki exploits the bidirectional mapping between ACE and OWL (Kaljurand & Fuchs, 2006a, 2006b) to accomplish this duality of representational formats.

[23] A wiki is a system that supports the online, collaborative entry of information using a Web-based interface. Wikis are used in a number of contexts, but perhaps the most well-known example of a wiki system is, of course, Wikipedia. The notion of a semantic wiki builds on the basic wiki idea and extends it 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 (Krötzsch et al., 2007) is probably the most popular and mature semantic wiki. It relies on the same wiki engine as Wikipedia and uses RDF/OWL to represent domain-relevant knowledge and information content.


protein

references delete

Word

word class: noun
noun form: protein edit...

Hierarchy

upward:
▷ Every protein is a molecule .

Individuals

▷ Act1 is a protein .
▷ Bub1 is a protein .
▷ Cav1 is a protein .
▷ Cbl is a protein .
▷ CHES1 is a protein .
...show all

▷ every protein interacts-with a protein .
▷ no protein interacts-with every protein .
▷ IRAK-1 is a protein that is a isoform of IRAK .
add...

Figure 4-2: AceWiki page describing the concept 'protein'

As with any wiki system, the contents of the AceWiki page shown in Figure 4-2 can be edited in a collaborative fashion by an end-user community. AceWiki provides a CNL-based editor, the AceWiki Editor (see Figure 4-3), to support users with respect to the creation of semantic contents using the ACE CNL. The editor interface is comprised of a number of UI components. The sentence field (1)[24] is a read-only text field that shows the beginning of an ACE sentence. The filter field (2) is a text field used for adding succession words to the ACE sentence presented in the sentence field (1). If the words entered by the user are accepted by the ACE language processor then they are moved to the sentence field (1). Clicking on the entries of the menu boxes (3) is an alternative way to construct a sentence. There is a menu box for each word class that is allowed at the current position. If a word is not yet known then it can be added on-the-fly by clicking on the respective menu entry (4).

[24] The numbers presented in brackets correspond to the number assigned to the user interface element in Figure 4-3.

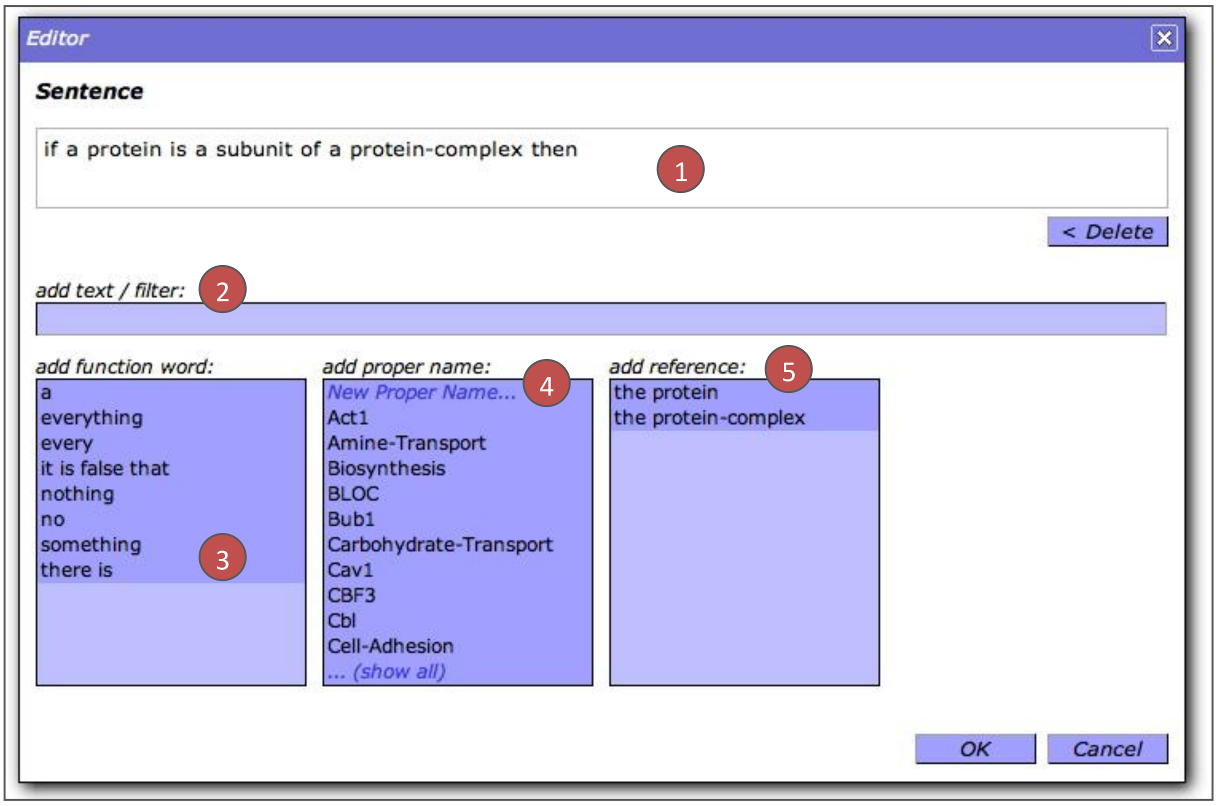


Figure 4-3: AceWiki Editor

Although AceWiki is still at an early stage of development, it shows great promise as a collaborative, user-friendly ontology editing system based on a CNL user interface. This set-up is potentially ideal for a number of situations where multiple agencies, devices and systems need to collaboratively and collectively contribute to the information space associated with coordinated, yet distributed, episodes of problem-solving activity. Systems like AceWiki provide a system that exploits a number of presentational formats, each of which is tailored to the perceptual and cognitive biases of a particular user group. In the case of the human end-user, the CNL interface supports the entry of knowledge and information content in a form that does not require any specialist knowledge engineering skills, and it does so in a way that does not renege on a commitment to support highly expressive, and epistemically-potent, representational formalisms. On the other hand, the link between CNL expressions and the formalisms of an ontology language (in this case the link between ACE and OWL) supports the involvement of non-human information acquisition systems and knowledge processors in the progressive enrichment and refinement of collective knowledge content[25].

[25] From the military perspective, it is important to remember that many military systems include a variety of intelligent agents that are capable of automatically processing and interpreting information. The provision of a machine-readable knowledge representation language, like OWL, supports the integration of these systems into a collective effort that subtends all elements of a coalition force. It is a collective effort that is driven by the need to enhance coalition situation awareness and enrich the information substrate upon which military decisions are ultimately founded.

4.4 Confluence Conceptual Ontology Builder (aka ROO)

ROO[26] (Rabbit to OWL Ontology) is an open-source Java-based plugin to Protégé that supports the user with respect to the creation and editing of ontologies using the Rabbit CNL (see Section 3.3.2). The tool is being developed by the Ordnance Survey and the University of Leeds, and its design is based on a systematic approach to ontology authoring, which has been developed based on the experiences of the Ordnance Survey with the creation of large-scale geo-spatial ontologies. Essentially, the Ordnance Survey has identified a number of factors that may negatively affect the ontology authoring process. They include:

1. the difficulty in expressing knowledge constructs in a formal language,
2. the lack of an appropriate methodology for capturing the knowledge of domain experts, and
3. the poor usability of existing ontology construction tools.

ROO has been designed to account for these factors. Firstly, the interface is based around the manipulation of Rabbit statements, not OWL constructs; the tool supports the translation of Rabbit models to OWL (at least OWL 1.1) via the GATE framework (Cunningham et al., 2002), but many of the technical details associated with OWL are effectively 'hidden' from the end user. Secondly, the design of the ROO interface is based on the main steps associated with the Ordnance Survey's proposed methodology for ontology development (Kovacs et al., 2006). This methodology, called Kanga, includes the following steps:

- identify the scope, purpose and other requirements of the ontology;
- gather sources of knowledge (e.g. documents and external ontologies);
- define lists of concepts, relationships and instances supplied with natural language descriptions;
- formalize core concepts and their relations in structured English sentences; and
- generate the OWL ontology.

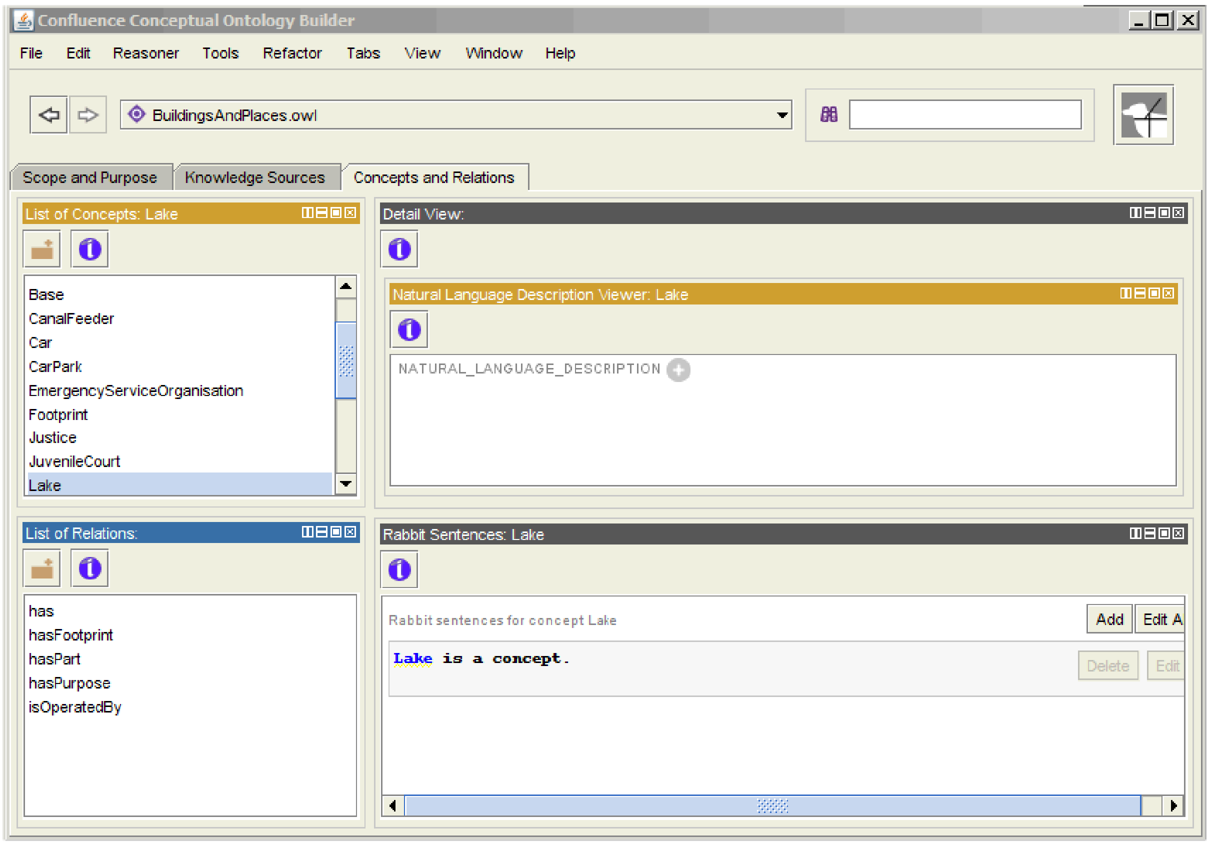
These steps are reflected in the kind of support and guidance provided by the ROO tool. ROO monitors the activities of the user (as well as the state of the underlying knowledge model) and provides contextually-appropriate suggestions as to what the user should aim to accomplish next:

> *"at the beginning of the ontology construction process, ROO will suggest that the scope and purpose of the ontology should be identified. Later, when a user has already defined several concepts and relationships, ROO will suggest to enter [sic] a natural language description for concepts missing such a description" (Denaux et al., 2008)*

Finally, ROO assists the user with respect to ontology construction by providing feedback in the form of syntax highlighting and error messages. Full scale usability studies have yet to be undertaken with ROO, yet it is clear that the combination of a supportive user interface and an intuitive linguaform knowledge representation language has great potential in terms of supporting end users with respect to the ontology authoring process.

[26] http://www.comp.leeds.ac.uk/confluence/index.html

The ROO interface consists of a number of tabs (e.g. 'Knowledge Sources') (see Figure 4-4) that broadly correspond to the elements of the Ordnance Survey's ontology development methodology. The 'Concepts and Relations' tab contains user interface components that support the user with respect to the creation and manipulation of OWL classes and properties. Figure 4-4 illustrates the 'Concepts and Relations' tab when the user has selected an item from the concepts list; while Figure 4-5 illustrates the 'Concepts and Relations' tab when the user has selected an item from the relations list. As can be seen from these figures, the ROO interface presents the user with a number of UI components. These support the user with respect to the entry of Rabbit CNL expressions; they also provide information about the state of the underlying ontology.



Base
CanalFeeder
Car
CarPark
EmergencyServiceOrganisation
Footprint
Justice
JuvenileCourt
Lake

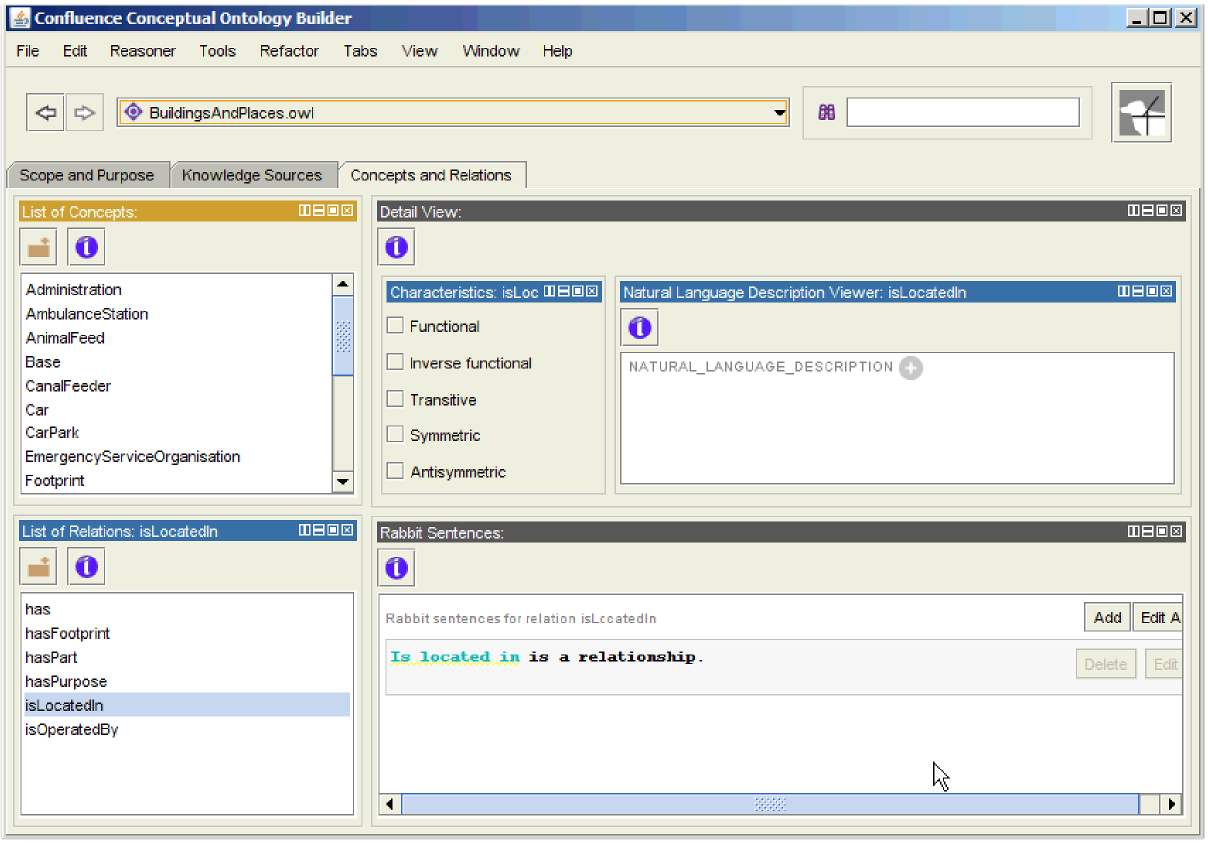
has
hasFootprint
hasPart
hasPurpose
isOperatedBy

Lake is a concept.

Figure 4-4: ROO Editor – concepts pane

As was mentioned above, ROO can export Rabbit models to OWL via GATE. This supports interoperation with other ontology editing tools, such as the Protégé-OWL editor (Holger et al., 2004). Denaux et al (2008) do not specifically state that ROO is capable of round-trip ontology authoring, i.e. that it can export ROO models to OWL which can then be edited in Protégé and subsequently re-imported in ROO; nevertheless, one assumes that this capability could be supported, especially since the alignment between Rabbit expressions (at least the OWL-compliant ones[27]) and OWL formalisms seems relatively tight.

[27] The ROO interface does not appear to support the expression of statements that would exceed the representational capacity of the OWL 1.1 specification; we therefore assume that the ROO editor only supports a subset of the formalisms capable of being expressed by Rabbit (the expressivity of Rabbit, recall, is greater than current versions of OWL - Hart et al., 2007).



Is located in is a relationship.

Figure 4-5: ROO Editor – relations pane

4.5 Discussion

The current section has reviewed a number of ontology authoring tools that use the medium of natural language to support end-users with respect to the creation of semantically-enriched knowledge content. Despite their differences, there are a number of features that almost all editors of this ilk have in common. Perhaps the most obvious commonality is the provision of syntax checking features, or features that, in some way or other, encourage (guide or constrain) the user into entering grammatically valid linguistic expressions. Such features are typically manifest in the form of error messages (e.g. ROO) or syntax highlighting features (e.g. GINO), although some editors also engage in what might be considered more intrusive forms of user guidance. Predictive interfaces, for example, attempt to limit user input to just those subsets of linguistic expressions that are deemed valid with respect to the underlying CNL grammar, and this usually means that users are presented with a choice of valid succession words during the course of sentence construction (e.g. GINO and AceWiki). Such strategies are particularly useful for ensuring that only grammatically-valid sentences can be constructed, and they clearly reduce the need for the user to learn the grammatical rules that are associated with the CNL in question:

> *"The writing process…is facilitated by predictive interface techniques: after the author enters a word form, the authoring tool displays look-ahead information indicating the available choices for the next word form, ensuring adherence to the lexicon and grammar. The author does not need to learn or remember the rules of the controlled natural language as these are taken care of by the authoring tool." (Cregan et al., 2007)*

Another issue for discussion in relation to CNL ontology editing tools is whether such tools should be limited to the creation of OWL ontologies, or whether they could be adapted to support the creation of other content, e.g. Semantic Web Rules Language (SWRL), SPARQL, etc. Kuhn (2008), for example, talks of a 'uniformity principle' in which the CNL serves as a common interface for the creation of multiple types of Semantic Web content, e.g. ontologies, queries, rules, etc. He suggests that the classical distinction between rule languages, query languages and knowledge representation languages, as epitomized by the various layers of the Semantic Web Layer Cake (see Figure 4-6), is motivated by the kind of distinctions that make sense to a knowledge engineer. These distinctions are not, however, the kind of distinctions that necessarily make sense to a human end-user, and this is because human users might (for example) conflate the representation of rule-like contingencies in a domain with the constraints and dependencies that characterize domain conceptualizations. Irrespective of whether it really makes sense to create languages based on differences in knowledge structure (rules, conceptual models) or epistemic function (rules, queries[28]), it is clear that we need not be committed to the use of different languages or presentational syntaxes at the level of the user interface. As Kuhn (2008) comments:

> *"For many users who are not familiar with formal conceptualizations, learning one formal language is already a hard task. We should not make this learning effort harder than necessary."*

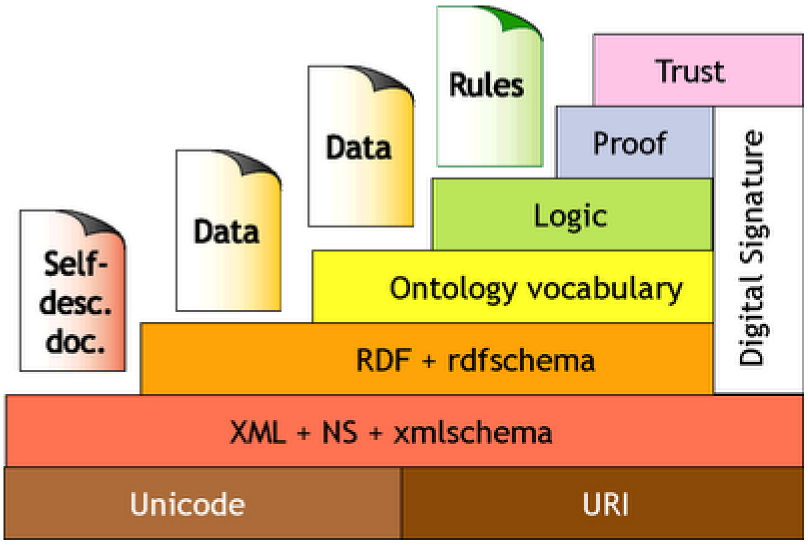


Figure 4-6: The Semantic Web layer cake

The basic claim, then, is that the different technological constituents of the Semantic Web could be unified at the user interface level by virtue of a common (re)presentational medium, i.e. the medium of natural language. This is the essence of Kuhn's (2008) uniformity principle. It is a suggestion that I find perfectly acceptable, although it leaves open the question as to how we might discriminate between linguistic expressions that need to be mapped into different Semantic Web languages. For example, should the sentence:

```
Every human is a mammal.
```

[28] In fact, when one considers the use of the SPARQL CONSTRUCT query form to make contingent assertions, the boundary between even semantic rule languages and semantic query languages begins to disappear.

be mapped into OWL:

```
ont:Mammal ⊑ ont:Human
```

SWRL:

```
ont:Human(human?)
=>
ont:Mammal(?human)
```

or even SPARQL:

```
CONSTRUCT
   {
         ont:Human rdfs:subClassOf ont:Mammal
   }
WHERE
   {
         ont:Human rdf:type owl:Class
   }
```

Clearly, we might rely on specific orthographic features here, e.g. if a sentence ends with a question mark then it is probably clear that we are dealing with a query and not an assertion. And in some cases it might be possible to adopt a strategy whereby a conversion into OWL has precedence over any other type of conversion, e.g. SWRL. If we revisit the statements at the bottom of the AceWiki page depicted in Figure 4-2, we can see that one statement is preceded by a red triangle. This statement is an ACE sentence that has no corresponding OWL representation; it cannot be converted to OWL. If we adopt an approach whereby OWL conversion failures are treated simply as the stimulus for other conversion efforts, then we can begin to see how some (perhaps all) of the expressivity of CNLs (which are, on occasion, more expressive than OWL) might be accommodated within a unified semantic representational system.

A key issue now comes into focus: inasmuch as the differential expressivity of CNLs and OWL (e.g. the mismatch between Rabbit and OWL, or ACE and OWL, etc.) is the basis for (often heated) exchanges about the relative superiority of one representational format over the other – especially when it comes to the representational bedrock of the Semantic Web, then perhaps the use of multiple languages (i.e. OWL, SWRL, SPARQL, etc.) in the way suggested by the uniformity principle can serve to soothe tempers on both sides of the frontline. For such approaches effectively eliminate the differences in expressivity along which empirical battle lines are drawn and theoretical fortifications constructed. Progress on the Semantic Web demands cooperation and consensus, not conflict and competition; there really is no room for Maginot lines in a science of the Semantic Web.

5 Natural Language Query Systems

5.1 Overview

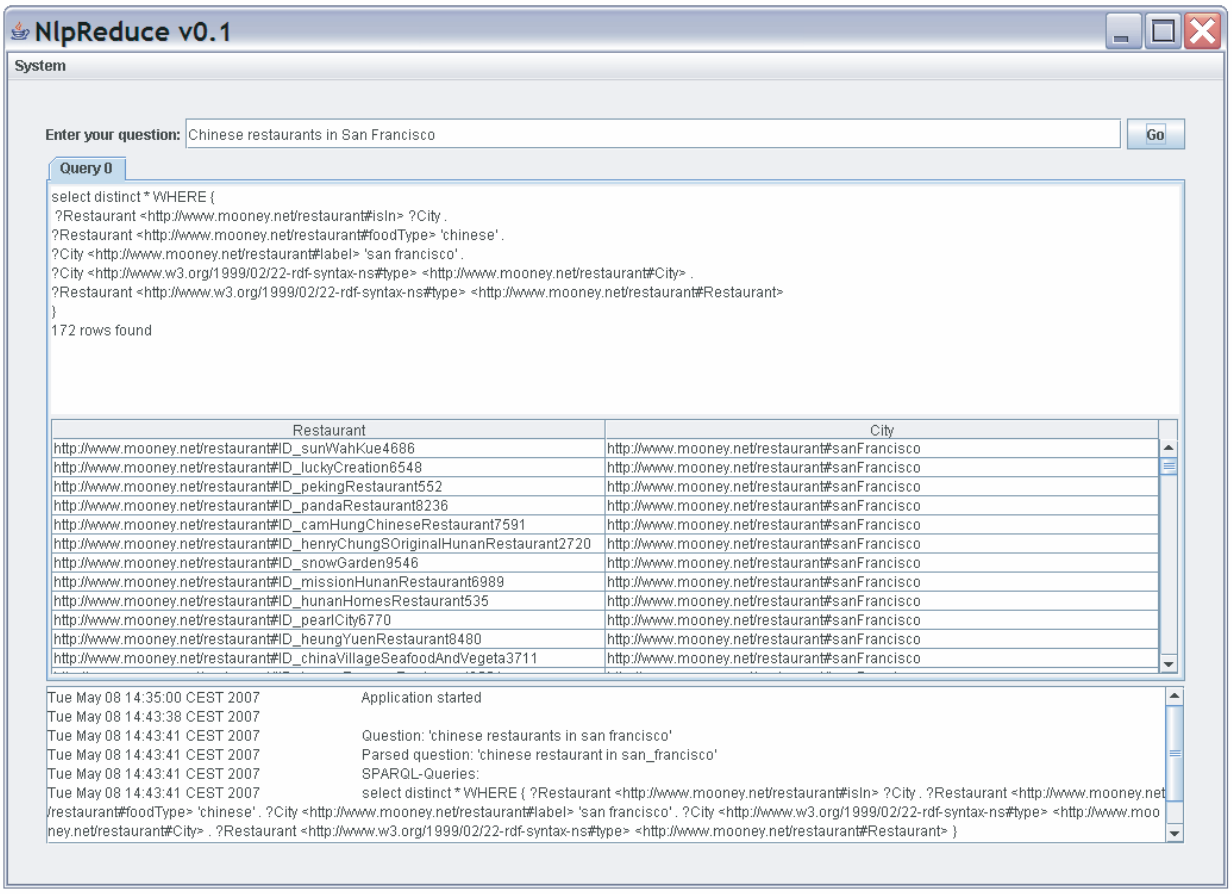
This section provides an overview of the current state-of-the-art with respect to natural language query systems, i.e. systems that support the retrieval of information content using natural language expressions. Natural language query systems for the Semantic Web tend to fall into three categories. They include:

1. **Pattern-Based Systems.** These systems attempt to detect certain patterns that typically recur in user queries. These patterns are used to interpret the nature of the information retrieval request implied by the query. Querix (see Section 5.6) is one example of this type of query system.
2. **Full Natural Language Query Systems.** These systems impose no grammatical constraints on the language that can be used to phrase an information retrieval request, e.g. a question. Instead, they use sophisticated NLP techniques to parse, interpret and transform user input into a Semantic Web-compliant query language. PANTO (see Section 5.5) is one example of this type of query system.
3. **Controlled Natural Language Query Systems.** These systems rely on a CNL, such as one of the CNLs described in Section 3 to structure, guide and constrain user input. The idea is that by restricting user input to a semantically unambiguous subset of possible query requests, a natural language query processor can interpret the user's query and convert it to a semantically-equivalent formal query. GINSENG (see Section 5.7) is an example of this type of system.

Subsequent sections provide an overview of some of the more common Semantic Web natural language query systems that are currently available.

5.2 NLP-Reduce

NLP-reduce is described as "a naive but domain-independent natural language interface for querying Semantic Web knowledge bases" (Kaufmann et al., 2007). It uses a reduced (hence the name) set of NLP techniques, including stemming and synonym expansion, but it refrains from any detailed processing of the query input. Essentially, the user's query is treated as a bag of words in which the words (and their associated synonyms) are mapped to entities in a target ontology. The NLP-Reduce user interface (see Figure 5-1) allows users to enter full natural language queries, sentence fragments, or just keywords. When the user clicks on the 'Go' button the application generates the SPARQL query and executes it against the underlying knowledge base. The results are presented to the user in the form of a datagrid display (see Figure 5-1).



Enter your question: Chinese restaurants in San Francisco

```
select distinct * WHERE {
 ?Restaurant <http://www.mooney.net/restaurant#isIn> ?City .
?Restaurant <http://www.mooney.net/restaurant#foodType> 'chinese' .
?City <http://www.mooney.net/restaurant#label> 'san francisco' .
?City <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.mooney.net/restaurant#City> .
?Restaurant <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.mooney.net/restaurant#Restaurant>
}
172 rows found
```

Restaurant	City
http://www.mooney.net/restaurant#ID_sunWahKue4686	http://www.mooney.net/restaurant#sanFrancisco
http://www.mooney.net/restaurant#ID_luckyCreation6548	http://www.mooney.net/restaurant#sanFrancisco
http://www.mooney.net/restaurant#ID_pekingRestaurant552	http://www.mooney.net/restaurant#sanFrancisco
http://www.mooney.net/restaurant#ID_pandaRestaurant8236	http://www.mooney.net/restaurant#sanFrancisco
http://www.mooney.net/restaurant#ID_camHungChineseRestaurant7591	http://www.mooney.net/restaurant#sanFrancisco
http://www.mooney.net/restaurant#ID_henryChungSOriginalHunanRestaurant2720	http://www.mooney.net/restaurant#sanFrancisco
http://www.mooney.net/restaurant#ID_snowGarden9546	http://www.mooney.net/restaurant#sanFrancisco
http://www.mooney.net/restaurant#ID_missionHunanRestaurant6989	http://www.mooney.net/restaurant#sanFrancisco
http://www.mooney.net/restaurant#ID_hunanHomesRestaurant535	http://www.mooney.net/restaurant#sanFrancisco
http://www.mooney.net/restaurant#ID_pearlCity6770	http://www.mooney.net/restaurant#sanFrancisco
http://www.mooney.net/restaurant#ID_heungYuenRestaurant8480	http://www.mooney.net/restaurant#sanFrancisco
http://www.mooney.net/restaurant#ID_chinaVillageSeafoodAndVegeta3711	http://www.mooney.net/restaurant#sanFrancisco

```
Tue May 08 14:35:00 CEST 2007          Application started
Tue May 08 14:43:38 CEST 2007
Tue May 08 14:43:41 CEST 2007          Question: 'chinese restaurants in san francisco'
Tue May 08 14:43:41 CEST 2007          Parsed question: 'chinese restaurant in san_francisco'
Tue May 08 14:43:41 CEST 2007          SPARQL-Queries:
Tue May 08 14:43:41 CEST 2007          select distinct * WHERE { ?Restaurant <http://www.mooney.net/restaurant#isIn> ?City . ?Restaurant <http://www.mooney.net
/restaurant#foodType> 'chinese' . ?City <http://www.mooney.net/restaurant#label> 'san francisco' . ?City <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.moo
ney.net/restaurant#City> . ?Restaurant <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://www.mooney.net/restaurant#Restaurant> }
```

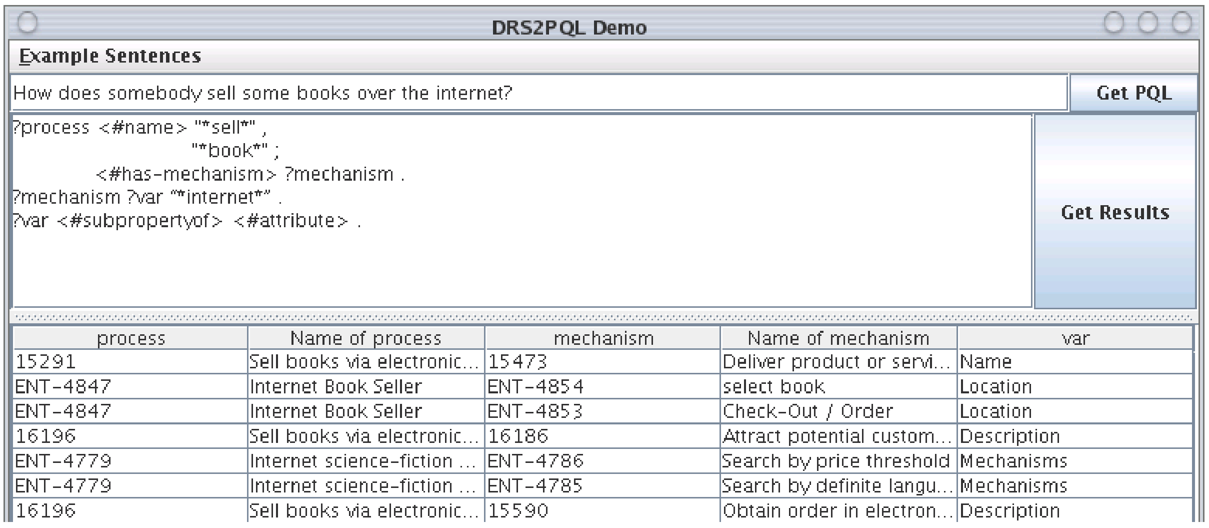
Figure 5-1: The NLP-Reduce user interface

Because NLP-Reduce eschews the use of sophisticated NLP techniques, its status as a natural language query interface for the Semantic Web is somewhat tenuous. Nevertheless, the tool is capable of generating SPARQL queries from unstructured textual input and preliminary observations suggest a surprising degree of success in retrieving query-relevant information[29]. Obviously, NLP-Reduce cannot answer queries which "require a dependency structure of the sentence elements" (Kaufmann et al., 2007) (e.g. "What is the largest state in the USA"); however, what it loses in interpretational power it gains in portability, as well as robustness to ungrammatical user input.

5.3 SWAT

SWAT (Bernstein et al., 2005a) is a tool that allows users to formulate queries in a specific CNL, namely ACE (see Section 3.3.1). It is unlike any of the natural language query interfaces reviewed in this section in the sense that it is the only one that uses a (predefined) CNL to specify a grammar for natural query generation and interpretation. The user interface for SWAT is illustrated in Figure 5-2.

[29] Kaufmann et al (2007) report a score of 76.4% for recall and 70.7% precision with respect to the Mooney Geoquery database.



How does somebody sell some books over the internet?

```
?process <#name> "*sell*" ,
                 "*book*" ;
        <#has-mechanism> ?mechanism .
?mechanism ?var "*internet*" .
?var <#subpropertyof> <#attribute> .
```

process	Name of process	mechanism	Name of mechanism	var
15291	Sell books via electronic...	15473	Deliver product or servi...	Name
ENT-4847	Internet Book Seller	ENT-4854	select book	Location
ENT-4847	Internet Book Seller	ENT-4853	Check-Out / Order	Location
16196	Sell books via electronic...	16186	Attract potential custom...	Description
ENT-4779	Internet science-fiction ...	ENT-4786	Search by price threshold	Mechanisms
ENT-4779	Internet science-fiction ...	ENT-4785	Search by definite langu...	Mechanisms
16196	Sell books via electronic...	15590	Obtain order in electron...	Description

Figure 5-2: The SWAT user interface

Behind the scenes, each ACE query entered by the user is translated into a DRS using the standard parsing engine for ACE, i.e. APE (see Section 3.3.1). Each DRS is subsequently converted to Process Query Language (PQL) using an ontology-based rewriting framework (see Figure 5-3), which consists of a set of "Ontology-Model Specific Keyword Rules" and "General Vocabulary Rules". Unfortunately, a full account of the rewriting framework would go beyond the scope of the current report (the interested reader is referred to Bernstein et al (2005a) for a detailed description); essentially the rules are used to map between elements of the DRS and the syntactic elements of a PQL query.

There are two main concerns with SWAT in its current form. The first is that the use of a CNL such as ACE assumes that the user must have expertise with the query language in order to be able to generate grammatically-valid queries. In response to this, Bernstein et al (2005a) point out that:

> *"learning a controlled language to phrase statements and queries is much easier than learning logic, and takes only a couple of days for the basics and two weeks for full proficiency, which is beyond what users need to write queries."*

In addition, user interfaces could be developed to support end-users with the entry of grammatically-valid ACE queries.

The second concern associated with SWAT relates to its support for PQL as opposed to a W3C recommended language, such as SPARQL. With respect to this issue, Bernstein et al (2005a) comment that PQL "can be easily mapped to query languages such as SquishQL"; however, it is not clear whether a similar mapping could be established for other types of semantic query language.

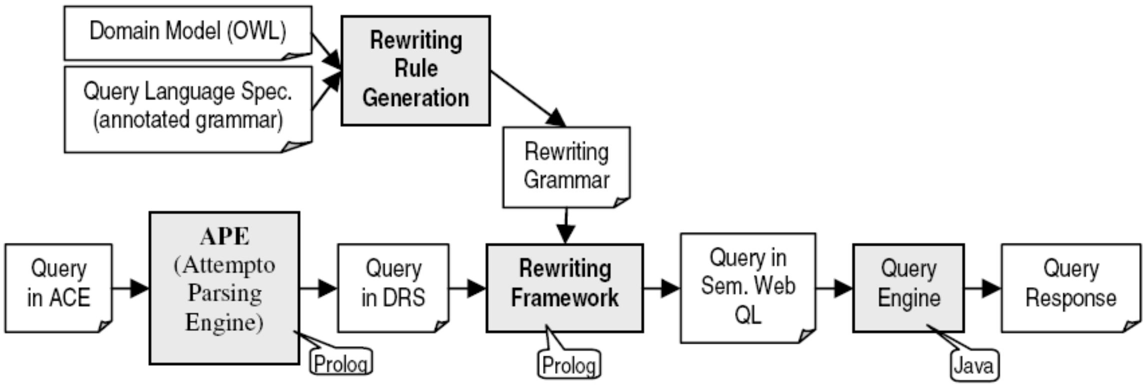


Figure 5-3: Query generation process for SWAT

5.4 AquaLog & PowerAqua

AquaLog[30] (Lopez & Motta, 2004; Lopez et al., 2005) is a portable question-answering system that was developed by the Knowledge Media Institute[31] as part of the Advanced Knowledge Technologies (AKT[32]) initiative. As a semantic question-answering system, AquaLog is able to provide answers to user-defined questions by drawing on the knowledge contained in an ontology. So, if a user was interested in finding out about humanitarian demining operations in Afghanistan, they could load an appropriate ontology, e.g. the SEMIOTIKS[33] Humanitarian Demining Ontology, and then query the ontology using unconstrained natural language expressions, e.g.:

```
What humanitarian demining organizations are operating in Helmand
province?
```

Clearly, the kind of question a user may ask in the context of AquaLog is constrained by the universe of discourse covered by the ontology, but AquaLog attempts to deal with vocabulary restrictions imposed by the selected ontology (any given ontology contains only a limited subset of the total terms used to refer to specific entities) by using resources such as WordNet (Fellbaum, 1998; Miller et al., 2004). This effectively expands the range of terms that AquaLog is able to deal with; it essentially enables AquaLog to deal with semantically-equivalent, but morphosyntactically distinct query expressions. For example, with respect to our previous example, a user may pose the following (equivalent) question:

What humanitarian demining **agencies** are **working** in Helmand province?

The task of translating a natural language query into an ontology-compliant query within AquaLog is a two-step process (see Figure 5-4). The first step involves the production of an intermediate, triple-based representation of the user's input query. These triples are subsequently processed by a module called the Relation Similarity Service (RSS), which attempts to make sense of the input query by looking at the structure of the ontology and using generic lexical resources such as WordNet. Essentially, the goal of the RSS is to establish a mapping between the elements of the triple-based

[30] http://kmi.open.ac.uk/technologies/aqualog/
[31] http://kmi.open.ac.uk/
[32] http://www.aktors.org/akt/
[33] See http://www.ecs.soton.ac.uk/research/projects/semiotiks for more information about the SEMIOTIKS project.

representation of a query and the elements of a domain ontology against which the query will be executed against:

> *"For instance, the query 'who is the secretary in KMi?' is parsed into <person/organization, secretary, kmi>, following purely linguistic criteria. Then, the first step for the RSS is to identify, in the target KB [Knowledge Base] that 'kmi' is actually a 'research-institute' called 'knowledge-media-institute'. Once a successful match is found, the problem becomes to find a relation which links the class research institute (or its superclass organization) to class person (or any of its subclasses, such as academic, student, etc...) or to class organization, by analyzing the taxonomy and relationships in the target KB. However, in this particular case there is a successful matching in the KB for secretary, even if secretary is not a relation but a subclass of person. The RSS reasons about the mismatch, re-classifies the intermediate query and generates the correct logical query, in compliance which the ontology, which is organized in terms of <secretary, works-for, kmi>." (Lopez et al., 2005)*

An interesting feature of the RSS is that it is interactive. In other words, when the RSS is not able to disambiguate between two possible relations that can be used to interpret the query, it prompts the user to help with the disambiguation. Once the RSS has resolved the elements of the triple-based representation with respect to the target domain ontology, the original natural language query can be converted into an ontology-compliant query. Unlike many of the natural language systems specifically developed for the Semantic Web, AquaLog does not convert natural language queries into a semantic query language such as RDQL or SPARQL: rather, it relies on the use of Onto-Triples (ontology-compliant triples) and a specialist query processor that interprets the Onto-Triples with respect to the target knowledge base.

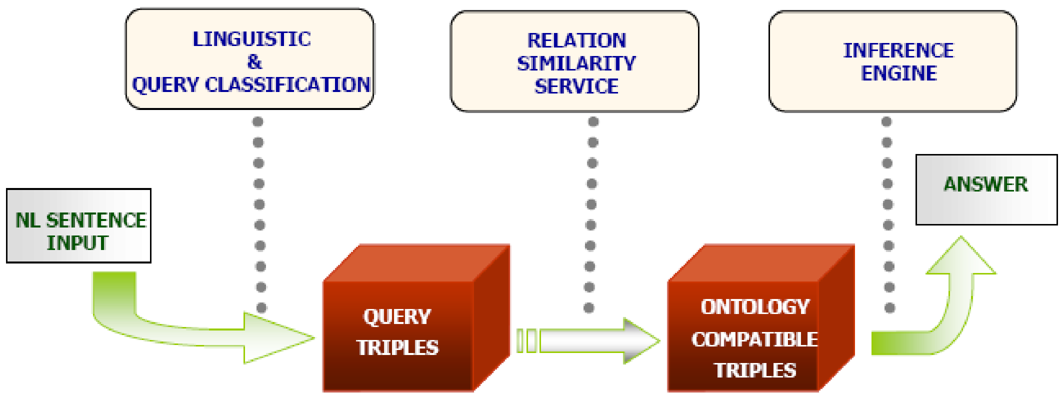

LINGUISTIC & QUERY CLASSIFICATION
RELATION SIMILARITY SERVICE
INFERENCE ENGINE
NL SENTENCE INPUT
QUERY TRIPLES
ONTOLOGY COMPATIBLE TRIPLES
ANSWER

Figure 5-4: AquaLog Data Model

One the limitations of AquaLog relates to its inability to query multiple ontologies at the same time. There are clear advantages in having a query system that is able to draw on the resources of multiple ontologies when providing answers to user's questions. For example, a query system might be able to exploit semantic mappings between ostensibly disparate ontologies in order to aggregate or fuse different subsets of information contained in multiple information repositories. In order to extend the capabilities of AquaLog with respect to multiple ontologies, Lopez et al (2006) developed

PowerAqua. PowerAqua is able to execute queries against multiple ontologies and is therefore well-placed to retrieve information from multiple ontologies using a common query interface.

5.5 PANTO

PANTO (Wang et al., 2007) provides a generic natural language query interface to Semantic Web ontologies. Like AquaLog/PowerAqua, PANTO is highly portable, meaning that it can be applied to any ontology or set of ontologies. When an ontology is selected as the underlying knowledge base, PANTO uses a Lexicon Builder component to automatically extract entities out of the ontology in order to build a Lexicon. It is this Lexicon that is used to make sense of the words that appear in a natural language query. In order to avoid the obvious restriction imposed by the ontology on the linguistic scope of the Lexicon (the fact that the name assigned to a class could have a multiplicity of morphosyntactic manifestations), PANTO uses WordNet (Fellbaum, 1998; Miller et al., 2004) to find synonyms of the names of ontology entities (concepts, properties, instances/individuals). PANTO can also make use of user-defined synonyms to expand the scope of the ontology's vocabulary.

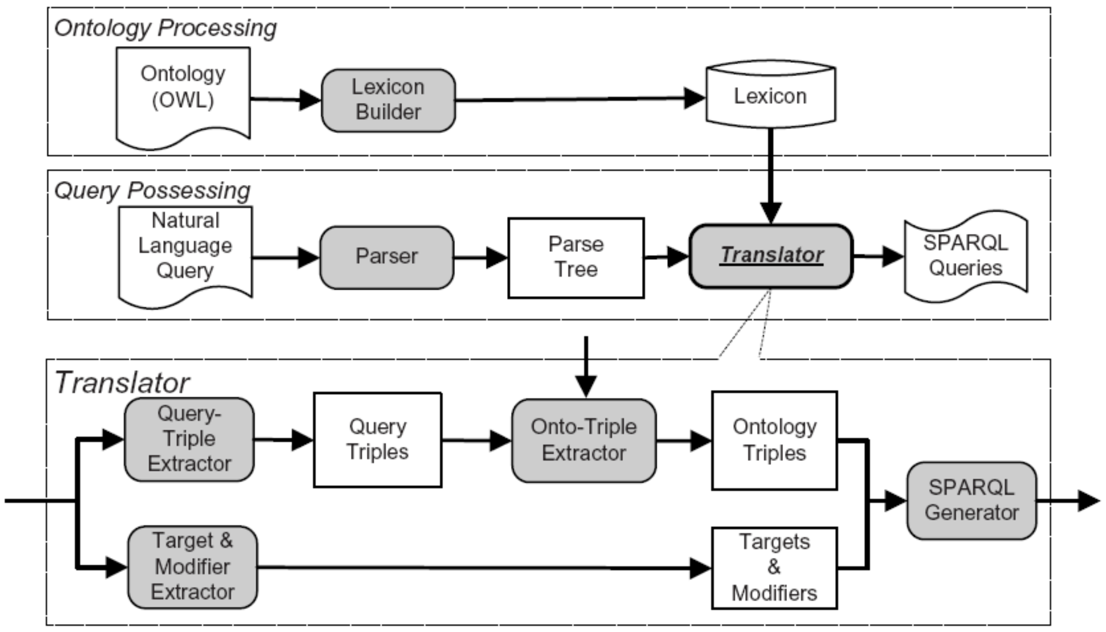


Figure 5-5: PANTO architecture

Once the user has entered a natural language query, PANTO invokes an off-the-shelf statistical parser, namely the StandfordParser (Klein & Manning, 2003), to produce a parse tree. This is subsequently transferred to the Translator, which converts the parse tree into SPARQL using two intermediate representations: QueryTriples and OntoTriples. QueryTriples are the first of two intermediate representations. They take the form of <subject-predicate-object> triples where the subject and object elements correspond to noun phrases (and their associated modifiers), and the predicate corresponds to the words used to join the subject and object together (typically it is a preposition or a verb phrase). So for example, in the query:

```
Which is the longest river that flows through Mississippi?
```

the PANTO Translator generates the following QueryTriple:

Subject	**Predicate**	**Object**
[longest river]	[flows through]	[Mississippi]

QueryTriples are generated without reference to the underlying ontology; that is, they rely solely on the linguistic analysis of the query sentence, specifically its decomposition into grammatical elements (noun phrases, verb phrases, prepositions, and the like). In order to convert the QueryTriple into an ontology-compliant query, PANTO clearly needs to map the various QueryTriple components to the semantic contents of a specific ontology (or set of ontologies). In order to do this, PANTO makes use of an OntoTriple Extractor component (see Figure 5-5), which converts the QueryTriples into ontology-compliant triples, called OntoTriples. OntoTriples are generated using the Lexicon and a number of string matching techniques, e.g. string distance matching metrics (morphological matching) and WordNet synsets (semantic matching). Once matches are found, the elements of the QueryTriple are replaced by their semantically-equivalent counterparts in the domain ontology. The result is a set of <subject-predicate-object> triples that conform to the triple-based structure of the target ontology. For example, the QueryTriple presented above yields the following ontology-compliant OntoTriples:

Subject	**Predicate**	**Object**
geo:River	geo:runThrough	geo:MississippiState
geo:River	geo:hasLength	xsd:long

The process is greatly simplified here, but essentially we end up with a triple-based structure that establishes a sensible linkage between the entities referred to in the user's original query and the semantic contents of the ontology. The final step is to convert the OntoTriple representation into a SPARQL query. This process is accomplished by the SPARQL Generator (see Figure 5-5). The SPARQL Generator takes the set of OntoTriples and converts them into SPARQL-compliant query elements. Clearly, the SPARQL Generator cannot use the OntoTriples alone to generate the query because such triples do not contain enough information about the kind of information a user actually seeks. For example, the natural language query presented above indicates that the user is interested in a particular river as the target of the query, namely the longest river in the state of Mississippi. In order to represent such information, PANTO utilizes a Target and Modifier Extractor component (see Figure 5-5). This component, as its name suggests, extracts a set of query targets, i.e. the words that correspond to variables in the SPARQL 'SELECT' statement, and modifiers that are used to filter the query targets in particular ways. In the case of our Mississippi river example, the target is clearly a river, and the modifier specifies the type of river we are interested in, namely the longest one. The resultant SPARQL query (as produced by the SPARQL Generator) is thus:

```
PREFIX geo: <http://apex.sjtu.edu.cn/nli/geo#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?river
WHERE
{
   ?river rdf:type geo:River .
   ?river geo:hasLength ?length .
   ?state rdf:type geo:State .
   ?river geo:runThrough ?state .
   ?state geo:border geo:MississippiState .
}
```

```
ORDER BY DESC(?length)
LIMIT 1
```

The details of the algorithms used to detect targets and modifiers and incorporate them into the final query need not concern us here (the interested reader is referred to Wang et al (2007) for more information); however, the essence of the query generation process should, by now, be clear. PANTO enables a user to write freeform natural language queries that, while directed towards a particular domain (as specified by the referenced ontology), are not restricted by the lexical scope or structural idiosyncrasies of the target ontology. PANTO thus appears as a powerful natural language query interface for the Semantic Web. It strikes a nice balance between portability and domain-specificity, without relying on the grammatical constraints of a CNL. Its use of NLP technologies and compliance with the W3C SPARQL specification are also highly laudable features.

In an evaluation of PANTO's performance, Wang et al (2007), derived measures of precision and recall with respect to a standard test database of natural language queries and domain ontologies. In particular, they used Mooney's Geoquery dataset[34], which has been used by a number of other studies to evaluate the performance of natural language query interfaces (Bernstein et al., 2005a; Bernstein et al., 2005b). The measure of precision in this context refers to the percentage of queries (relative to the number actually processed by PANTO) that PANTO was able to correctly[35] translate into SPARQL queries; recall, on the other hand, refers to the percentage of queries (relative to the original query dataset) that PANTO was able to produce output for. A summary of the performance results are presented in Figure 5-6 (adapted from Wang et al (2007)):

Original Mooney Queries	877
Precision	663 (88.05%)
Recall	753 (85.86%)
Error	90 (11.95%)

Figure 5-6: PANTO performance evaluation results

As we shall see (in Section 5.6), these results are roughly the same as another natural language query system that generates SPARQL-compliant queries from natural language input: Querix.

5.6 Querix

Querix (Kaufmann et al., 2006) is a domain-independent natural language query interface for the Semantic Web that bears a number of similarities to PANTO. For example, Querix allows a user to select a domain ontology against which to apply queries. Like PANTO, Querix uses WordNet (Miller et al., 2004) to obtain synonyms for the labels associated with entities in the ontology whenever the

[34] http://www.cs.utexas.edu/users/ml/nldata.html

[35] The notion of correctness here is somewhat subjective. It is based on a comparison of the PANTO-generated queries with manually-generated ones (this, of course, assumes that the manually-generated SPARQL queries were actually generated – 877 SPARQL queries seems like a lot of queries to create by hand!). It is not clear whether the authors based their comparison on the syntactic structure of the queries (were the queries syntactically identical?), or whether they used the results of query execution (did the queries return identical resultsets despite some, presumably minor, syntactic differences?).

user loads a new ontology. Querix also uses the StanfordParser (Klein & Manning, 2003) to generate parse trees for the natural language query entered by the user. Unlike PANTO, however, it adopts a different approach towards the processing of parse trees. Whereas PANTO "identifies BaseNPs [base noun phrases] and utilizes structure information inside and among the BaseNPs" (Wang et al., 2007), Querix extracts the sequence of main word categories (based on parser-generated Part-Of-Speech (POS) tags) in order to create a 'query skeleton'. A query skeleton is simply a sequence of word categories (such as nouns, verbs, prepositions, question words and conjunctions) that is based on order of words in the natural language query. So, for example, if the query is:

```
What are the population sizes of cities that are located in
California?
```

the query skeleton will be:

```
Q-V-N-P-N-Q-V-P-N
```

where

```
Q = question word

N = Noun

V = verb

P = preposition
```

Once a query skeleton has been generated from the parse tree, Querix attempts to match the elements of the query string with synonym-enhanced triples in the ontology. For each query, Querix relies on a set of heuristic patterns to identify <subject-predicate-object> triples in the domain ontology that correspond to sequences of elements in the query skeleton. So, for example, the N-P-N sequence in the aforementioned query skeleton (the sequence corresponding to "population sizes of cities"), might be matched to the following triple in the target ontology:

Subject	Predicate	Object
querix:City	querix:hasPopulationSize	xsd:integer

From here, Querix is able to generate SPARQL queries that can be executed against the target ontology. The details of this process, as one might imagine, are quite complex and largely beyond the scope this report; suffice to say, however, that Querix yields similar performance results[36] to PANTO when tested against the Mooney Geoquery dataset (see Figure 5-7).

[36] One caveat here relates to the number of queries used in the two studies. Kauffman et al (2006) opted to use a reduced set of 215 queries to represent the 879 queries associated with the Geoquery dataset.

Original Mooney Queries	201
Precision	151 (86.08%)
Recall	175 (87.11%)
Error	24 (13.71%)

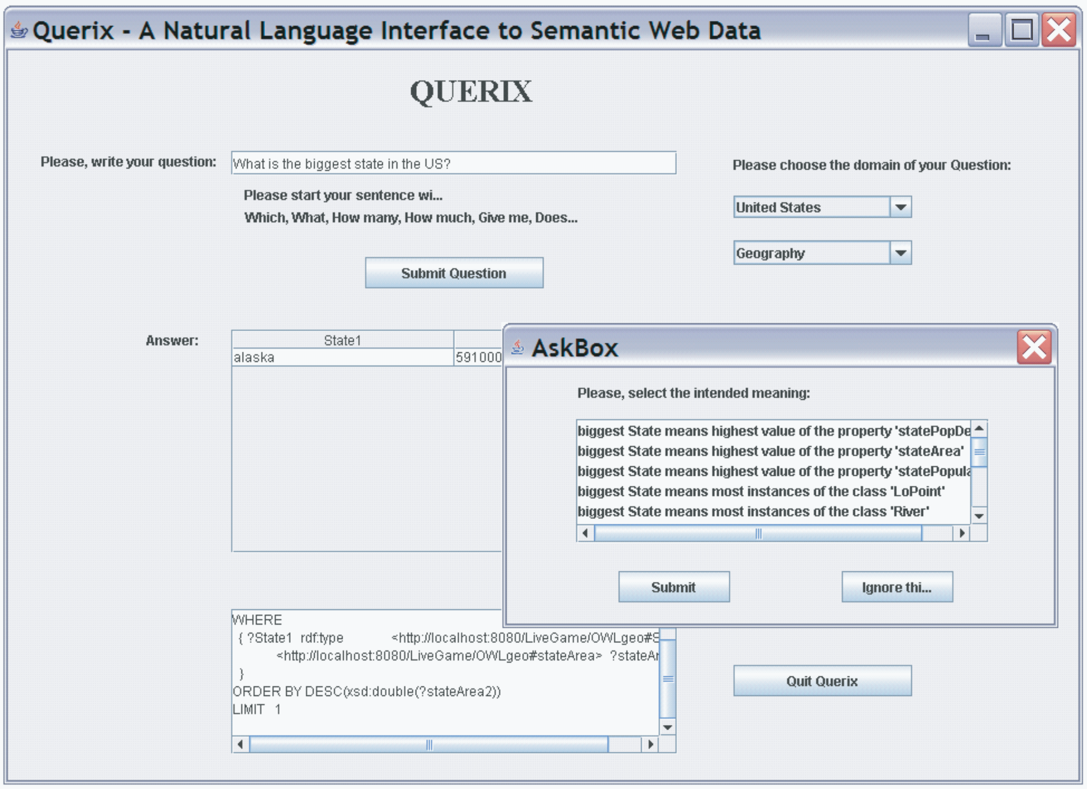
Figure 5-7: Querix performance evaluation results

It should be noted that despite the ostensible similarity of these results to those described with PANTO, there are some subtle differences in the experimental technique that undermine the validity of direct comparisons. Firstly, Kauffman et al (2006) used a reduced query sample in their evaluation of Querix (215). They also eliminated any queries that were not correctly analyzed by the StandfordParser (8 queries) or that were nonsense queries (e.g. “Which state lies in a city that…?”) (5 queries). This means that the total number of queries used in the Querix evaluation was 201 versus the 877 used in the PANTO evaluation study. Interestingly, the erroneous queries identified by Kauffman et al (2006) were not detected by Wang et al (2007). This is curious because Wang et al (2007) claim to have tested PANTO against a set of manually-generated SPARQL queries. They should therefore have identified at least the nonsense queries detected by Kauffman et al (2006). Remember as well that these nonsense queries were isolated from a subset of the total query sample. Assuming that the 215 queries selected by Kauffman et al (2006) is a representative sample, we can infer that there are approximately 20 nonsense queries in the sample of 877 queries selected by Wang et al (2007). In conjunction with the possibility of errors in the parsing process, this suggests that Wang et al’s (2007) results may have underestimated the true performance of PANTO with respect to the Mooney Geoquery dataset. It remains unclear why Wang et al (2007) failed to detect the invalid natural language queries in their test sample.

One final comment about Querix’s functionality: Querix clearly relies on NLP technologies to process the users query, but it does not attempt to fully resolve linguistic ambiguities using NLP techniques. Rather, Querix relies on a user interaction technique that co-opts the user in the disambiguation process. The technique relies on the use of clarification dialogs, one of which is illustrated in Figure 5-8. Basically, whenever Querix encounters an ambiguous phrase or word in the user’s input string, it presents the user with a dialog box asking the user to select from a list of intended meanings. For example, in the case of the input query:

```
What is the biggest state in the US?
```

The phrase ‘biggest state’ is ambiguous. It could mean the biggest state in terms of geographical area, or it could mean the most populous state. In order to resolve the ambiguity, Querix prompts the user to select the appropriate interpretation of the phrase and then proceeds to generate the appropriate SPARQL query.



Please, write your question: What is the biggest state in the US?

Please start your sentence wi...
Which, What, How many, How much, Give me, Does...

Please choose the domain of your Question: United States; Geography

Answer:

State1	
alaska	591000

AskBox

Please, select the intended meaning:

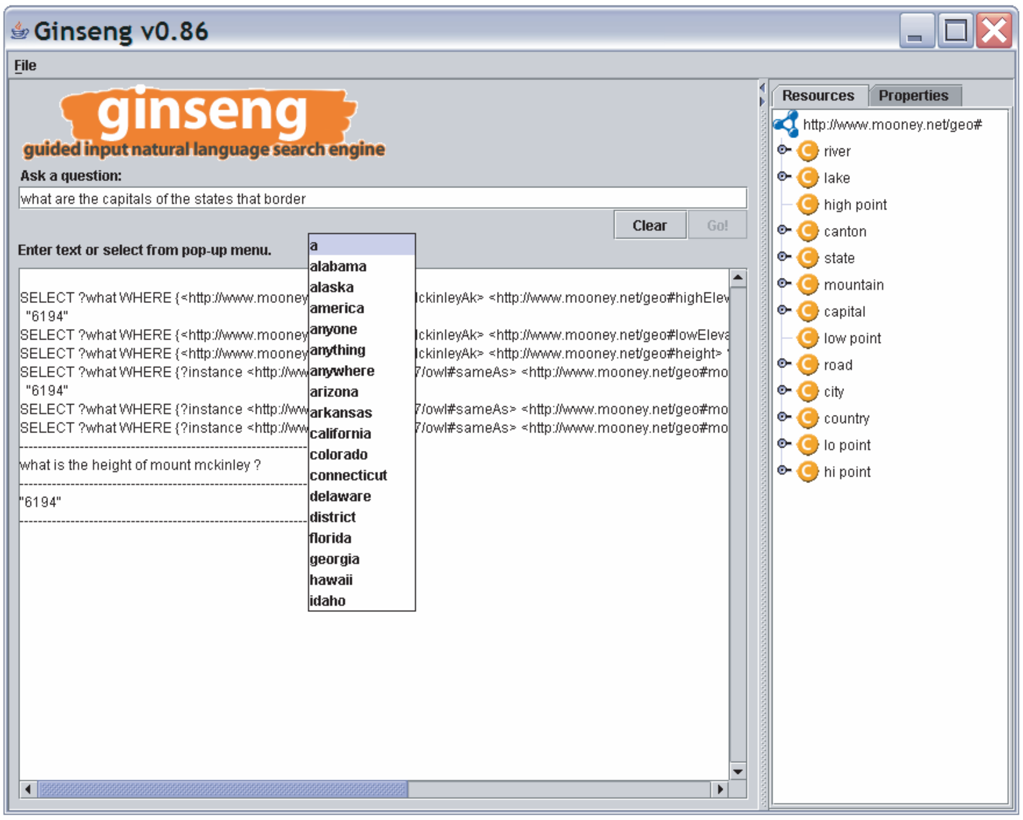
biggest State means highest value of the property 'statePopDe
biggest State means highest value of the property 'stateArea'
biggest State means highest value of the property 'statePopula
biggest State means most instances of the class 'LoPoint'
biggest State means most instances of the class 'River'

```
WHERE
 { ?State1  rdf:type          <http://localhost:8080/LiveGame/OWLgeo#S
        <http://localhost:8080/LiveGame/OWLgeo#stateArea>  ?stateAr
 }
ORDER BY DESC(xsd:double(?stateArea2))
LIMIT   1
```

Figure 5-8: The Querix user interface

5.7 GINSENG

GINSENG – a Guided Input Natural Language Search Engine – allows users to query ontologies using a controlled query language whose grammar is automatically generated from the target ontologies (Bernstein et al., 2005b; Bernstein et al., 2006). When a user starts GINSENG, all ontologies in a predefined search path are loaded into the execution environment, and a grammar compiler is used to generate a dynamic grammar rule for every class, property and individual contained in the ontology. The rules are subsequently used to control and guide user input as the user types their query into a text field in the user interface. Figure 5-9 shows a screenshot of the GINSENG application interface. The user types their query into the 'Ask a question' textbox, and, as they type, a pop-up listbox displays a list of valid succession words based on the aforementioned grammar rules. Only entries that are valid, according to the active grammar, can be entered into the system. As such, the user is prevented from entering any queries that cannot be interpreted in terms of the loaded ontologies. This is a potentially powerful approach because it limits the range of queries that can be entered by the user to just those for which a valid SPARQL query can be generated. Therefore, any question that is successfully entered by the user will be guaranteed to be converted to SPARQL (whether the query is correct or not is, of course, another matter).



```
what are the capitals of the states that border
```

```
SELECT ?what WHERE {<http://www.mooney                 ckinleyAk> <http://www.mooney.net/geo#highElev
 "6194"
SELECT ?what WHERE {<http://www.mooney                 ckinleyAk> <http://www.mooney.net/geo#lowEleva
SELECT ?what WHERE {<http://www.mooney                 ckinleyAk> <http://www.mooney.net/geo#height>
SELECT ?what WHERE {?instance <http://ww               7/owl#sameAs> <http://www.mooney.net/geo#mo
 "6194"
SELECT ?what WHERE {?instance <http://ww               7/owl#sameAs> <http://www.mooney.net/geo#mo
SELECT ?what WHERE {?instance <http://ww               7/owl#sameAs> <http://www.mooney.net/geo#mo
-------------------------------------------------------
what is the height of mount mckinley ?
-------------------------------------------------------
"6194"
-------------------------------------------------------
```

Figure 5-9: The GINSENG user interface

As mentioned above, GINSENG generates a set of dynamic grammar rules whenever a target ontology is loaded into the query environment. Dynamic grammar rules are, however, only one type of rule used by GINSENG. The other is (did you guess?) a static grammar rule. Static grammar rules provide the basic sentence structure for questions that can be entered into the system:

> *"The static grammar rules provide the basic sentence structures and phrases for English questions. It [sic] handles general question structures such as 'What are the capitals of the states that border Nevada?' as well as closed questions (e.g., 'Is there a city that is the highest point of a state?' typically resulting in an answer of 'yes' or 'no') or questions resulting in numbers (e.g., 'How many rivers run through Texas?'). Furthermore, it [sic] provides sentence construction rules for the conjunction or disjunction of two or more sentence parts." (Bernstein et al., 2006)*

Together, the dynamic and static grammar rules describe the parse rules for all the natural language queries that can be entered by the user (i.e. they specify the complete set of grammatically-valid sentences); they also, however, specify the query composition elements that are used in generating a semantic query (RDQL queries in this case) from the natural language query input. To make the approach a little more understandable, let us consider a concrete example involving a specific user-entered query:

```
    What state borders New York State?
```

Of course, we know that the user will have been supported in entering this question by the GINSENG user interface and (in the background) the set of dynamic and static rules defining the ontology-specific grammar. Let's review some sample rules that could have contributed to the generation of this sentence:

```
(1) <START> ::= <OQ> ?
                  | SELECT <<OQ>>
                  | WHERE (<<OQ>>)
(2) <OQ> ::= what <subject> <verb>
                  | <<subject>>
                  | <<subject:1>> <<subject>>) (<<subject:1>> <<verb>>
(3) <subject> ::= state
                  | ?state
                  | <rdf#type> <geo#state>
(4) <verb> ::= borders <object>
                  | -
                  | <geo#borders> <<object>>
(5) <object> ::= New York State
                  | -
                  | <geo#newYorkState>
```

For the moment ignore everything after the first line of each query rule, i.e. the statements after the '|' symbol (these elements of the rule are responsible for the generation of the RDQL statements comprising the appropriate query). Starting from rule (1), which as its name suggests is always the first rule to be applied, we can see that the right-hand-side of the rule returns a non-terminal symbol <OQ>. This symbol is then matched to other rules in the grammar and, as such, rule (2) is selected (this is the only rule in our simple grammar that has a <OQ> symbol on the left-hand-side of the rule). The right-hand-side of the rule, in this case, is 'what <subject> <verb>'. The <subject> and <verb> elements are again non-terminal symbols so they are matched to the left-hand-side of the other rules in the grammar, but the 'what' symbol is a terminal symbol, and it is presented to the user as a valid element of the natural language query. The general rule is that all non-terminal symbols are recursively matched to the left-hand-side of each rule in the grammar, while the non-terminal symbols are presented to the user to guide natural language query generation. As may already be clear, rules (1) and (2) are static grammar rules. The word 'what' is often used to begin a query sentence, and so it makes sense to include it as part of the ontology-independent rule set; the rest of the rules, however, are dynamic rules. These rules are generated by GINSENG whenever an ontology is loaded by the user.

Continuing with our example, once the user has selected 'which' from the popup listbox, the grammar matches the <subject> symbol to the left-hand-side of the rules in the grammar and finds a single match (rule 3 in this case). This rule returns a single non-terminal symbol, viz. 'state', so the user is again presented with a single option for the next word in the query sentence (bear in mind that this is very simple grammar!). With the selection of the 'state' symbol we have exhausted the number of rules that can fire as part of the <subject> symbol in rule (2), so we move onto the <verb> symbol of rule (3). This symbol matches rule (4), which gives us the word 'borders', and, finally, rule (5) gives us 'New York State'. It should be obvious that a greatly expanded set of grammatical rules would use a much richer set of options for natural language query generation, although the basic mechanism of rule matching, selection and execution would remain the same.

Now consider the process of semantic query generation. For each rule in the example grammar (see above), there are two lines each preceded by a '|' symbol. The first of these lines represents the "SELECT" part of the RDQL query; the second part represents the "WHERE" part. We know that all rules in the aforementioned grammar are selected by the sample sentence (because such rules were used to create the sentence in the first place), so let's look at the lines corresponding to the "SELECT" part of the query first. Rules (1) and (3) contribute to the "SELECT" part of the query to give us:

```
SELECT ?state
```

Now what about the "WHERE" part of the query? In this case all the rules are involved and following their sequential execution we get:

```
WHERE
   ( ?state <rdf#type> <geo#state> )
   ( ?state <geo#borders> <geo#newYorkState> )
```

The first thing to note here is that the <<subject:1>> symbol in rule (2) is used to back-reference the first occurrence of <<subject>> in the rule. This ultimately resolves into the symbol '?state' and it forms the first element of the two triple patterns in the query[37]. The second thing to note is that the "WHERE" component of rule (2) comprises two sets of non-terminal symbols separated by brackets. This is what gives us the two triple patterns associated with the "WHERE" part of the RDQL query.

Finally, we can combine the "SELECT" and "WHERE" parts of our query to give us a syntactically-valid RDQL query that returns the desired information:

```
SELECT ?state
WHERE
   ( ?state <rdf#type> <geo#state> )
   ( ?state <geo#borders> <geo#newYorkState> )
```

As should be clear by now, the vocabulary used by GINSENG is limited by the ontology or ontologies from which the grammar rules are created. Clearly, if the system was restricted to using only the names used to identify ontology elements, e.g. class identifiers, then the range of words available for users to use in their queries would be highly constrained. Fortunately, however, GINSENG allows a user to annotate an ontology with synonym tags from the GINSENG namespace. These synonyms are processed by GINSENG and used to create alternatives to the identifiers used in the ontology[38].

In summary then, GINSENG uses a form of controlled language, but it is a controlled language where the set of permissible linguistic constructions is tied to the target ontology rather than to a particular set of predefined grammatical and linguistic conventions (as is the case with CNLs). Unlike query interfaces such as PANTO and Querix, GINSENG does not permit true natural language input; rather it uses the lexical and structural properties of the target ontology, in combination with synonym information, to specify the complete set of permissible natural language queries that can be executed against the loaded ontologies. Given the potential computational overhead associated with the use of full natural language processing, as well as the training overhead encountered with the use of CNLs, GINSENG provides a nice example of an alternative approach to developing natural

[37] Note the difference between <<subject>> and <<subject:1>> in this rule. The <<subject:1>> symbol back-references the first match to <<subject>>, which is '?state' in the context of rule (2); while <<subject>> leads to the selection of rule (3) and the incorporation of "<rdf#type> <geo#state>" into the triple pattern.
[38] This process could probably be enhanced by referencing external sources of synonymy information, e.g. WordNet synsets.

language query interfaces. Since the range of permissible queries is constrained by the grammar and user interface, we would expect recall (the percentage of queries successfully converted to the target query language) to be high. This is indeed the case with GINSENG, which has a 98.4% recall rate (Bernstein et al., 2005b). In addition, precision is also reported to be high in performance evaluation studies – 92.8% in Bernstein et al (2005b).

5.8 Summary

The current section has reviewed some of the more popular natural language query systems that are being used to retrieve information in the context of the Semantic Web. As yet, there are no large-scale comparative analyses that would assist with the evaluation of such systems in terms of their relative performance against (e.g.) precision, efficiency and usability criteria[39]. We can, however, begin to see that natural language query systems can be characterized in terms of a number of key features, such as their portability and support for unconstrained natural language input, and that such features might be used for comparative/evaluative purposes. The following represents a (partial) list of the features that might be used to compare query systems in future studies:

1. **Target query language:** What semantic query languages, e.g. SPARQL, are generated by the system?
2. **Portability:** How portable is the query system? Can it be easily applied to ontologies in different domains?
3. **Extent of NLP processing:** What kinds of NLP technologies are used as part of the query system?
4. **Use of restricted grammar or CNL:** Does the query system use a predefined CNL, e.g. ACE, or does it rely on a special purpose (perhaps dynamically-generated and ontology dependent) grammar?
5. **Support for multiple ontologies:** Can the system execute queries against multiple ontologies? Can it exploit ontology alignment information to provide a common query interface to multiple ontologies?
6. **Extent of user interaction:** How much user interaction is permitted by the tool? Does the tool involve the user in resolving semantically ambiguous user input?
7. **Recall:** What percentage of the natural language queries entered by a user can be translated (correctly or incorrectly) into a corresponding semantic query?
8. **Precision:** How accurate is the system in terms of correctly translating the user query into the target query language? Does the system always retrieve the right kind of information requested by the user[40]?

[39] Although see Kaufmann and Bernstein (2007) for the results of small-scale study exploring the usability of 3 of the natural language query interfaces described in this section (namely, NLP-Reduce, GINSENG and Querix).
[40] Note that the query system may accurately translate a user's query without actually retrieving the right resultset. The mismatch may arise if the grammar underlying natural language input does not permit a user to adequately communicate their information retrieval requirements.

9. **Usability:** How well does the tool perform in usability studies?

10. **Training requirements:** What is the training overhead associated with the tool? What kinds of users is the tool targeted towards, i.e. what is the target user community?

Table 5-1 provides a summary of the query systems presented in this section with respect to a subset of the aforementioned features.

Query System	Query Language	Portability	CNL	Precision
NLP-Reduce	SPARQL	High	No	Low
SWAT	PQL	Low	Yes (ACE)	Medium
AquaLog	Onto-Triples	High	No	Medium
PowerAqua	Onto-Triples	High	No	Medium
PANTO	SPARQL	High	No	High
Querix	SPARQL	High	No	Medium/High
Ginseng	SPARQL/RDQL[41]	High	Yes (grammar generated from ontology)	Very High

Table 5-1: Comparison of natural language query systems

[41] The query language targeted by GINSENG is RDQL according to Bernstein et al (2005b). However, Bernstein et al (2006) talk about the translation of input queries into SPARQL. This suggests that GINSENG has both an RDQL and SPARQL query generation capability.

6 Ontology Verbalization

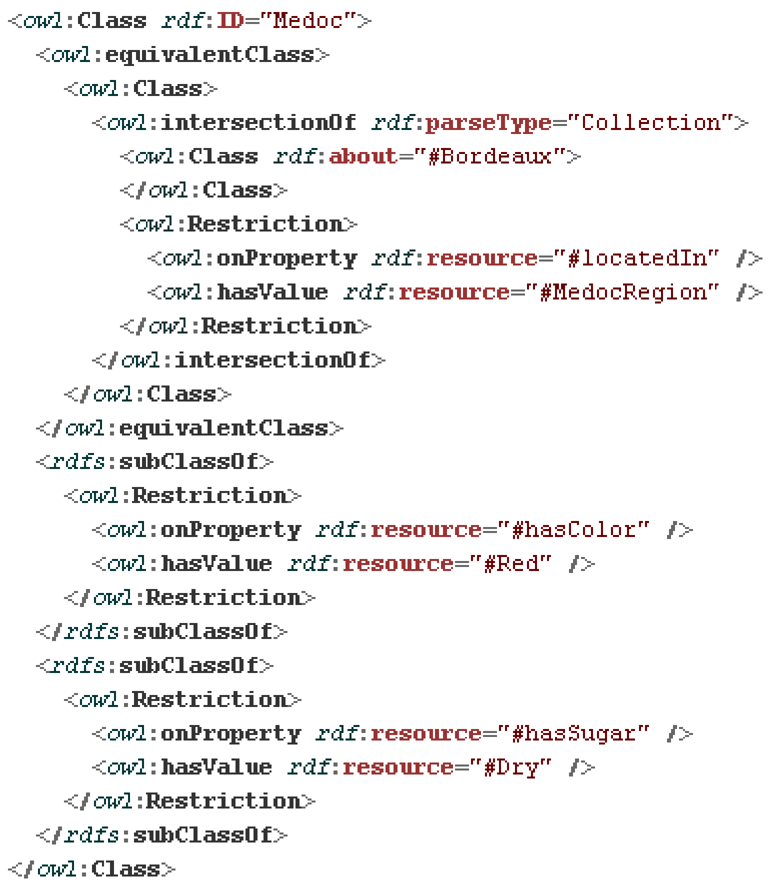
6.1 Talking My Language

The solutions scouted in the past two chapters should now convince us that there are rich opportunities for interfacing with the Semantic Web. Natural language ontology editors (see Section 4) support ontology authors with respect to the creation of Semantic Web content, and natural language query interfaces (see Section 5) would seem to enable even casual users to pose questions and retrieve information from Semantic Web repositories. But there is, I think, one additional barrier to surmount; one that perhaps more than anything else distances us from the Semantic Web. Thus far we have focused on techniques that enable human agents to communicate with the Semantic Web. The ontology editors in Section 4 allowed human end-users to express human knowledge in a form suitable for publication on the Semantic Web; the query designers in Section 5 enabled human end-users to express their epistemic needs in a form that a semantically-enabled system could attempt to service. These technologies enabled us to talk to the Semantic Web, but there is one thing they did not do: they did not allow the Semantic Web to talk to us.

In this section we complete our tour of natural language interfaces for the Semantic Web by visiting the realm of ontology verbalisers. An ontology verbaliser is essentially a system that transforms a pre-existing ontology into a series of natural language expressions. The advantages of such a transformation, at least in terms of human understanding, should be obvious. Consider, for example, the case of the 'Medoc' class in the Wine Ontology. Figure 6-1 illustrates the representation of this class using RDF/XML notations. Compare this representation with the semantically-equivalent natural language expression:

```
Medoc is a sweet, red color wine located in the Medoc region
```

It should be clear from this example that natural language expressions are (in general) far more suited for communicating knowledge to human beings than the logical formalisms and presentational formats typically countenanced by the Semantic Web community.



```
<owl:Class rdf:ID="Medoc">
  <owl:equivalentClass>
    <owl:Class>
      <owl:intersectionOf rdf:parseType="Collection">
        <owl:Class rdf:about="#Bordeaux">
        </owl:Class>
        <owl:Restriction>
          <owl:onProperty rdf:resource="#locatedIn" />
          <owl:hasValue rdf:resource="#MedocRegion" />
        </owl:Restriction>
      </owl:intersectionOf>
    </owl:Class>
  </owl:equivalentClass>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasColor" />
      <owl:hasValue rdf:resource="#Red" />
    </owl:Restriction>
  </rdfs:subClassOf>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="#hasSugar" />
      <owl:hasValue rdf:resource="#Dry" />
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>
```

Figure 6-1: Description of 'Medoc' class in the wine ontology

There are a number of other advantages associated with ontology verbalisation. They include the fact that natural language representations are easily amenable to existing search tools, such as Google. In addition a variety of NLP tools, such as spell checkers and speech synthesizers, can be immediately harnessed to cater for the orthographically challenged and visually impaired. In its most sophisticated forms, ontology verbalization can also provide multi-lingual capabilities that extend the reach of ontologies beyond those of the traditional English speaking world (e.g. see Aguado et al (1998)). Above all, ontology verbalization can be seen as an essential ingredient to the collaborative development and validation of ontologies. As Kalyanpur et al (2005) comment:

> *"…a need to provide an easily readable explanation of terms in the ontology arises from the fact that the intended purpose of ontologies is for information sharing, which could involve external parties that have little or no background knowledge of the ontology domain. In such cases, it becomes the responsibility of the domain experts creating the ontology to provide textual documentation for the terms within."*

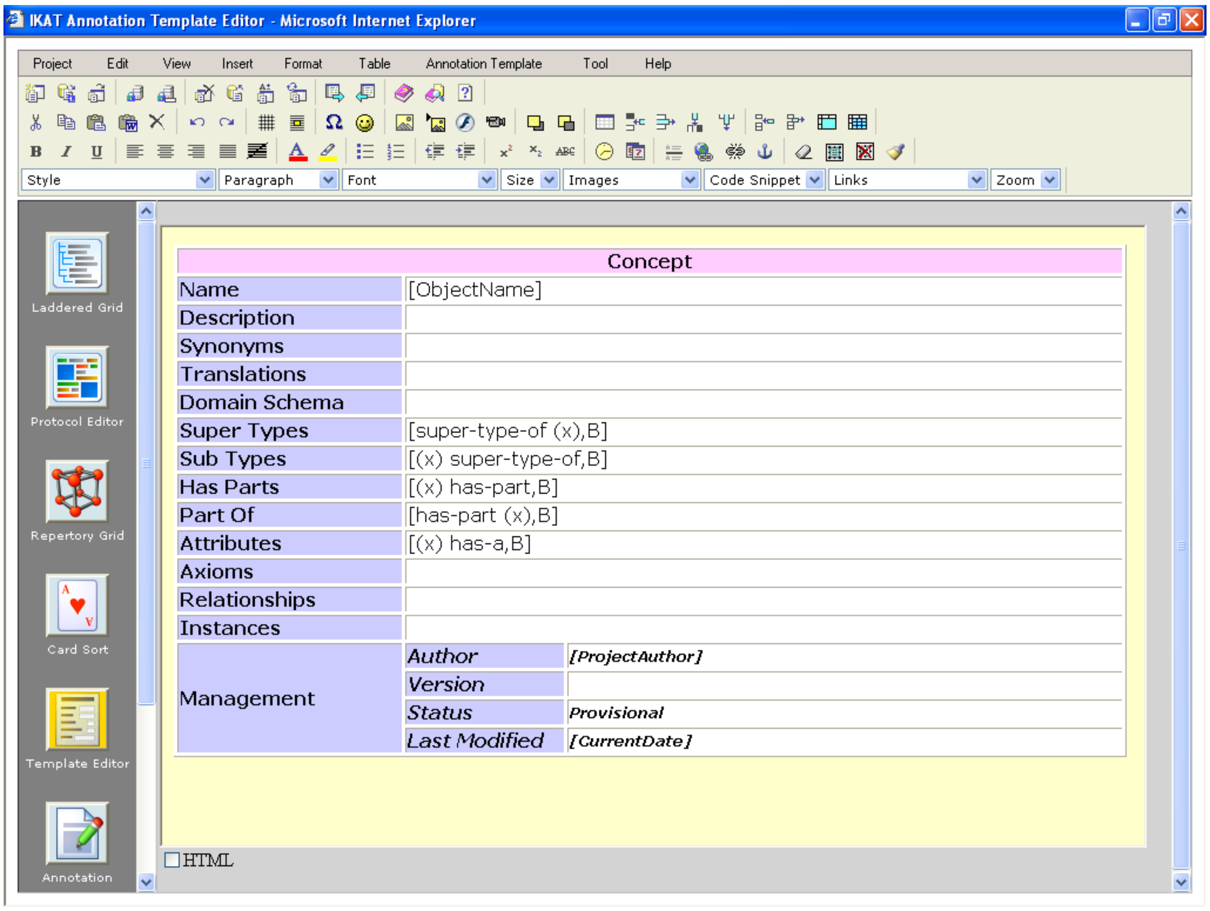
6.2 Tools and Techniques

In this section, we provide an overview of approaches to ontology verbalization. It should be noted that the techniques described herein represent the upper end of a continuum whose lower bounds are rooted in all manner of efforts to make ontological content more accessible to human end-users. First of all there is the OWL abstract syntax (Patel-Schneider et al., 2004), which aims to provide a more readable presentation format than the RDF/XML exchange format (see Figure 6-2). Then there is the Manchester OWL syntax (Horridge et al., 2006), which was "developed in response to a demand from a wide range of users, who do not have a Description Logic background". The need for richer and more readable representations of ontological content is a continuing theme throughout the Semantic Web research literature. In their review of problems that users encounter with OWL, for example, Rector et al (2004) express the need for a "pedantic but explicit" paraphrase language, one that assists end-users in understanding the *meaning* of semantic axioms.

```
ObjectProperty(eaten-by inverseOf(eats))
ObjectProperty(eats domain(animal))
ObjectProperty(is-part-of Transitive)

Class(animal partial
 annotation(rdfs:comment "Animals form a class."))

Class(branch partial
 annotation(rdfs:comment "Branches are parts of trees.")
 restriction(is-part-of allValuesFrom (tree)))

Class(carnivore complete
 annotation(rdfs:comment
           "Carnivores are exactly those animals that eat also animals.")
 intersectionOf(animal restriction(eats someValuesFrom (animal))))

Class(giraffe partial
 annotation(rdfs:comment "Giraffes are herbivores, and they eat only leaves.")
 herbivore
 restriction(eats allValuesFrom (leaf)))
```

Figure 6-2: OWL abstract syntax

6.2.1 Template-Based Techniques



Concept		
Name	[ObjectName]	
Description		
Synonyms		
Translations		
Domain Schema		
Super Types	[super-type-of (x),B]	
Sub Types	[(x) super-type-of,B]	
Has Parts	[(x) has-part,B]	
Part Of	[has-part (x),B]	
Attributes	[(x) has-a,B]	
Axioms		
Relationships		
Instances		
Management	*Author*	*[ProjectAuthor]*
	Version	
	Status	*Provisional*
	Last Modified	*[CurrentDate]*

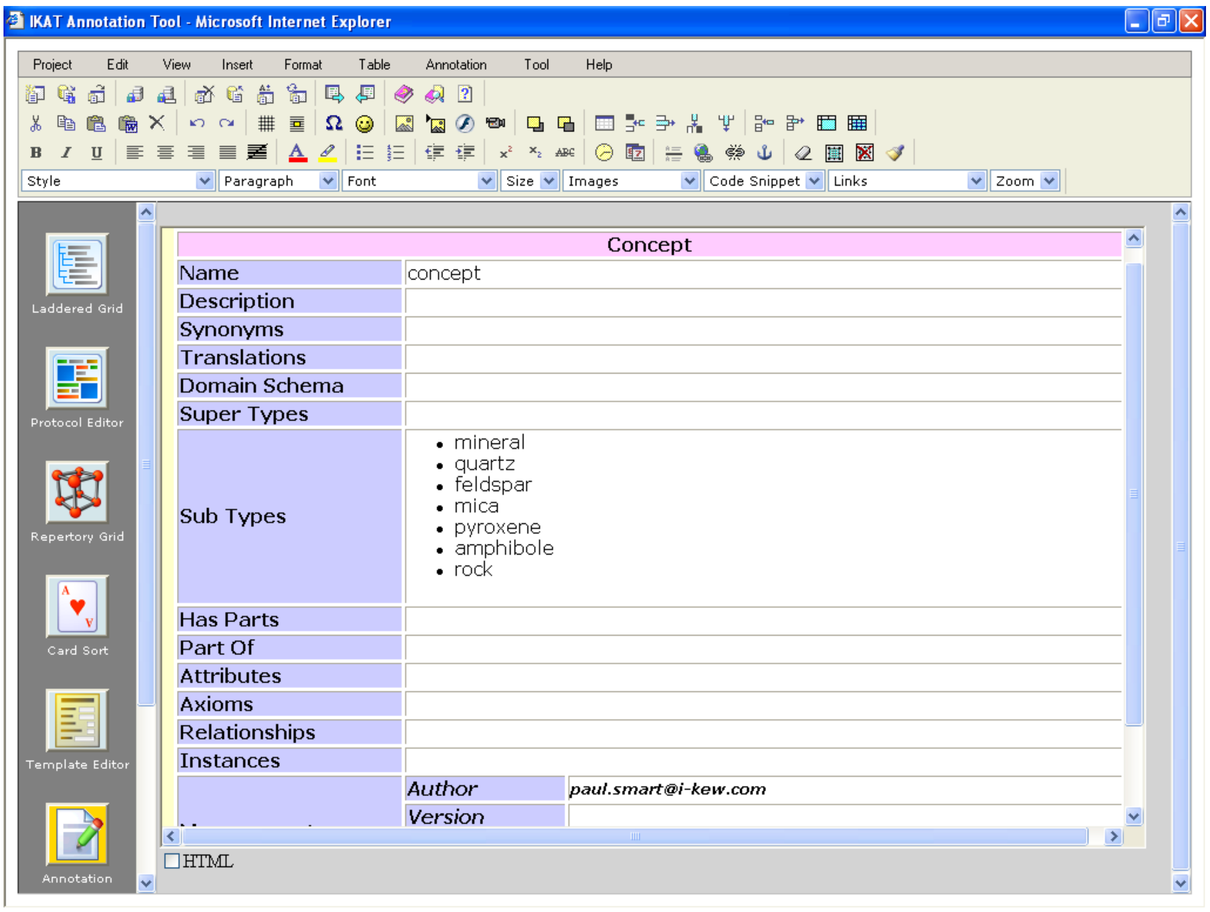
Figure 6-3: IKAT annotation template editor

Perhaps the simplest way to generate user-friendly output from ontologies is to use a template-based approach that allows users to embed knowledge content into some predefined publication format. Such is the approach adopted by the knowledge acquisition and modelling tool, PCPACK[42] (or its web-based counterpart, IKAT). PCPACK/IKAT includes a variety of knowledge editing and visualization tools all of which use a common knowledge repository. This means that any changes a user makes in one tool, e.g. the Laddered Grid Tool[43], are immediately reflected in all other tools, e.g. the Diagramming Tool[44]. To support the publication and dissemination of knowledge models, PCPACK/IKAT includes an Annotation Tool and an Annotation Template Tool. The Annotation Template Tool (see Figure 6-3), enables users to embed PCPACK/IKAT 'formulas' into an XHTML file; these are subsequently replaced by knowledge model content during the template instantiation process. The syntax used for the formulas directly references the structure of the knowledge model. So if a user wanted to create a bulleted list of the components of a particular object, they could do so using the following formula:

```
[hasPart-x, B]
```

Figure 6-4 shows the appearance of the web page once all the PCPACK/IKAT formulas have been replaced with knowledge model content.

[42] http://www.epistemics.co.uk/
[43] http://www.epistemics.co.uk/Notes/160-0-0.htm
[44] http://www.epistemics.co.uk/Notes/162-0-0.htm



Concept		
Name	concept	
Description		
Synonyms		
Translations		
Domain Schema		
Super Types		
Sub Types	• mineral • quartz • feldspar • mica • pyroxene • amphibole • rock	
Has Parts		
Part Of		
Attributes		
Axioms		
Relationships		
Instances		
	Author	***paul.smart@i-kew.com***
	Version	

Figure 6-4: IKAT annotation tool

Another approach to the generation of user-friendly output formats is to capitalize on the fact that OWL ontologies can be expressed as RDF/XML and to rely on pre-existing XML transformation solutions. Wilcock (2003), for example, describes a pipeline of XSLT transformations that can be used to generate natural language texts from RDF and DAML+OIL ontologies. The major limitation of this approach is, of course, the overhead incurred by the requisite adaptation of the transformation solution when dealing with new ontologies. In addition, because Wilcock's (2003) solution is grounded in the use of RDF and DAML+OIL ontologies, it is not clear whether the approach can extend to ontologies using more expressive formalisms, e.g. OWL.

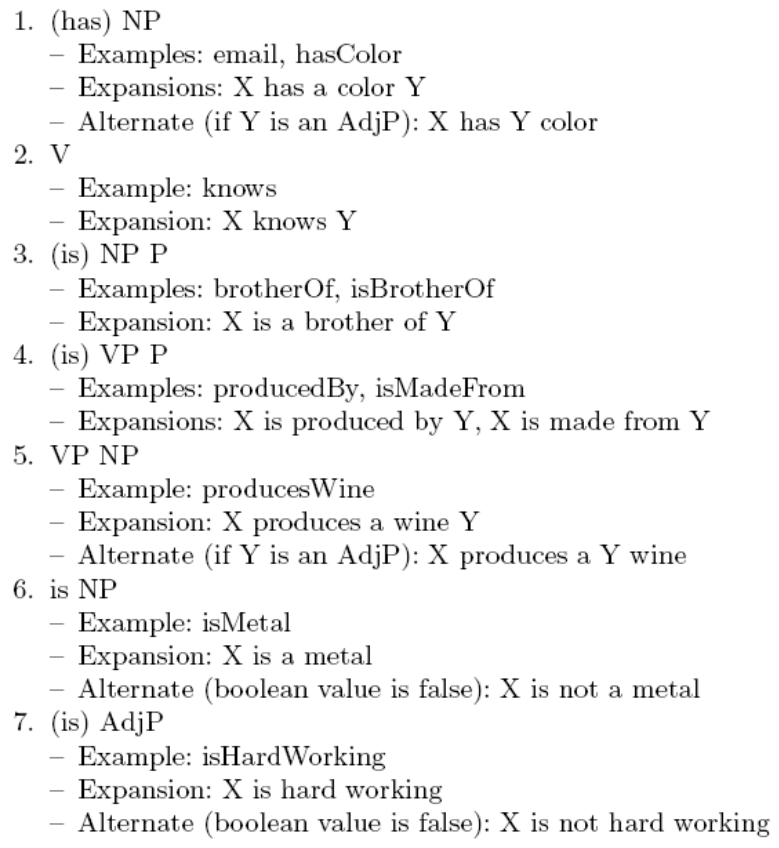
One final example of a template-based approach is provided by the ArtEquAKT report generator (Alani et al., 2003; Kim et al., 2002). In this case, a series of templates are used to guide the generation of natural language reports from a domain ontology.

6.2.2 SWOOP Natural Language Entity Renderer

A number of developers have attempted to integrate natural language capabilities into extant ontology editing environments. One example of this is the Class Description Display plugin[45] that provides pseudo-natural language descriptions of ontology elements in Protégé-OWL; another is the implementation of a Natural Language Entity Renderer (NLER) as part of the SWOOP Ontology

[45] http://www.coode. org/downloads/cdc/

Editor[46] (Hewlett et al., 2005; Kalyanpur et al., 2005). The NLER aims to generate natural language descriptions of OWL classes based on their semantic characterization in the domain ontology.



1. (has) NP
   - Examples: email, hasColor
   - Expansions: X has a color Y
   - Alternate (if Y is an AdjP): X has Y color
2. V
   - Example: knows
   - Expansion: X knows Y
3. (is) NP P
   - Examples: brotherOf, isBrotherOf
   - Expansion: X is a brother of Y
4. (is) VP P
   - Examples: producedBy, isMadeFrom
   - Expansions: X is produced by Y, X is made from Y
5. VP NP
   - Example: producesWine
   - Expansion: X produces a wine Y
   - Alternate (if Y is an AdjP): X produces a Y wine
6. is NP
   - Example: isMetal
   - Expansion: X is a metal
   - Alternate (boolean value is false): X is not a metal
7. (is) AdjP
   - Example: isHardWorking
   - Expansion: X is hard working
   - Alternate (boolean value is false): X is not hard working

Figure 6-5: Phrase structure categories for ontology properties[47]

Hewlett et al (2005) evaluated NLER in a small-scale user evaluation study involving 10 subjects, all of whom were unfamiliar with the Semantic Web. The results of their study suggest that users generally favoured NLER-generated class descriptions, but these claims are undermined by a number of serious methodological problems associated with the study:

1. The study used a sample of only 5 OWL classes, and it is doubtful whether such a limited sample could ever hope to be truly representative of the range of actual class definitions/descriptions available. No detail is provided about these sample classes, so it is impossible for the reader to actually comment on their actual representativeness. It also makes it difficult to know whether the sample was biased in favour of a particular NLG capability, such as the one used by the author's tool.

2. The authors state that each subject was presented with three definitions of the same concept. One of these three definitions was generated by the technique described in the paper; however, the authors do not explicitly state how the two comparison natural language paraphrases were generated. The authors mention that the goal of the study was "to guage how subjects considered the output of our program relative to that of Protégé and OntoExpl"; one must therefore assume that these tools were used to generate the

[46] http://www.mindswap.org/2004/SWOOP/

[47] This list is reproduced verbatim from Hewlett et al (2005).

comparison sentences. Protégé, however, does not have a NLG plug-in unless the authors are referring to the Manchester OWL syntax or OWL abstract syntax – neither of which could really be considered suitable for users unfamiliar with the Semantic Web (i.e. the target user population cited by the study).

3. Related to the previous criticism is the fact that the authors only selected classes. They do not provide any indication as to how their technique copes with the description of properties or individuals; neither do they evaluate the performance of their technique (vis-à-vis properties/individuals) with respect to their sample of users.

4. The evaluation is based on a measure of subjective preference of one sentence over the others, but there are many measures of usability other than user preferences ratings (which could be based on rather arbitrary criteria). One would have thought that given the author's original motivation about usability (which is not the same as preference), it might be more appropriate to concentrate on the user's understanding of the concepts, rather than user's preference for one sentential format over other natural language expressions.

This list does not exhaust the criticisms that could be levied at the user study described by Hewlett et al (2005), but it is clear from the points raised here that more rigorous user evaluation studies need to be undertaken before any firm conclusions about the usability of NLER can be established.

Notwithstanding the problems with the user evaluation study, it is clear that NLER has a number of technological shortcomings. Firstly, the natural language generated by NLER can, on occasion, be quite complex, perhaps yielding natural language descriptions that are barely more understandable to end-users than the logical formalisms they are supposed to clarify. Another problem concerns the restriction of the technique to ontology classes: properties and individuals seem to be excluded[48].

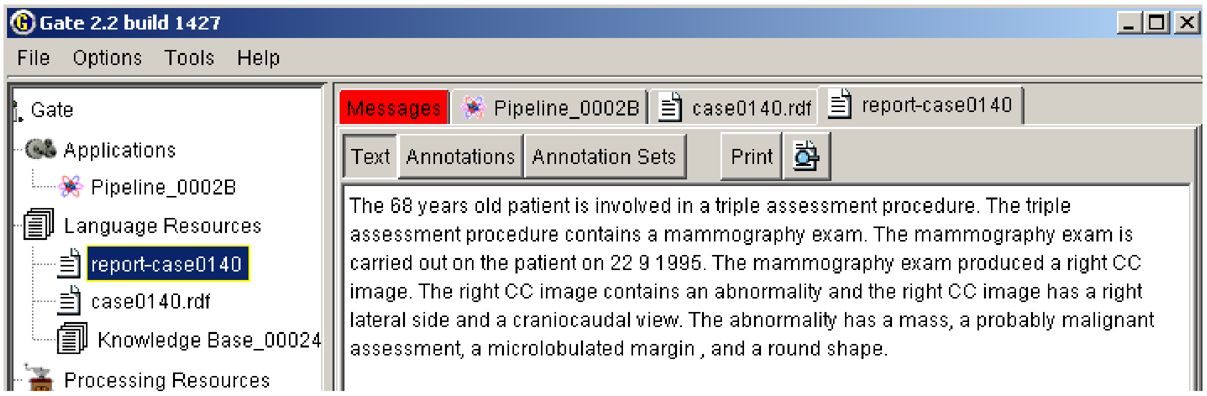
Perhaps the biggest limitation of the approach adopted by NLER relates to its dependency on a set of naming conventions to identify the property type associated with class descriptions. Recall that NLER attempts to categorize a property based on its lexical properties, e.g. whether it is a noun phrase optionally preceded by the word 'has' (see Figure 6-5). Although this approach could work under certain conditions, namely those in which ontology authors rigorously adhered to the required naming conventions, it is unlikely that such conditions could ever be established in the context of the Semantic Web. One possibility is that the naming conventions could be proposed as part of a suite of best-practice ontology authoring guidelines that facilitate the realization of a number of ontology processing goals.

6.2.3 MIAKT Report Generator

A number of ontology verbalization capabilities have been explored as part of the Advanced Knowledge Technologies[49] (AKT) initiative, a six-year research program that sought to investigate state-of-the-art solutions to a number of knowledge engineering challenges. One of these verbalization capabilities is the MIAKT Report Generator (Bontcheva & Wilks, 2004), which was developed in the context of the MIAKT[50] initiative (Shadbolt et al., 2004). The MIAKT Report

[48] Subsequent implementation efforts may have addressed this particular shortcoming (see Halaschek-Wiener et al., 2006).
[49] http://www.aktors.org/akt/
[50] http://www.aktors.org/miakt/

Generator aims to provide an automatic, ontology-driven report generation capability in the domain of biomedical informatics, specifically in the area of breast cancer screening and diagnosis. It takes, as input, a medical ontology and an RDF description of patient data, and uses a suite of GATE-based resources to create a textual report that summarizes the patient data in a natural language format (see Figure 6-6). More details about the technical approach can be found in Bontcheva and Wilks (2004).



The 68 years old patient is involved in a triple assessment procedure. The triple assessment procedure contains a mammography exam. The mammography exam is carried out on the patient on 22 9 1995. The mammography exam produced a right CC image. The right CC image contains an abnormality and the right CC image has a right lateral side and a craniocaudal view. The abnormality has a mass, a probably malignant assessment, a microlobulated margin , and a round shape.

Figure 6-6: MIAKT patient report

6.2.4 Verbalizing OWL in ACE

Given ACE's popularity as a CNL, it is perhaps not surprising that an ontology verbalization solution would be available to translate domain ontologies into ACE texts. Kaljurand and Fuchs (2007) describe the technological realization of this capability.

In describing their solution, Kaljurand and Fuchs (2007) cite a number of desirable features for an OWL-ACE verbalization capability:

1. Firstly, the verbalization should be reversible, i.e. it should be possible to serialize the ontology to ACE and then convert it back into an OWL ontology that is identical, or at least semantically-equivalent, to the original.
2. Secondly, the verbalization should be understandable English, i.e. it should be easy to read and it should preserve the semantics of the original ontology. This means that Kaljurand and Fuchs (2007) prefer certain linguistic constructions to others. For instance, "Every man is a human" is deemed to be more preferable than the semantically equivalent "If there is a man then he is a human".
3. Thirdly, the OWL-ACE mapping must be compatible with the semantics of ACE expressions. So, for example, the OWL subclass axiom subClassOf(Human, Mammal), must be mapped to a universally quantified sentence[51] (e.g. 'every human is a mammal') and not to a sentence like 'a human is a kind of mammal', which, in ACE at least, is interpreted as having only existential quantification.
4. Finally, Kaljurand and Fuchs (2007) attempt to leave the structure of the original ontology as intact as possible. The motivation behind this preservation of ontological structure is to

[51] Recall that in OWL 'subclassof' means necessary implication: if 'Human' is a subclass of 'Mammal' then all instances of Human are also instances of Mammal, without exception. If something is a human then this implies that it is also a mammal. It is for this reason that the mapping of subclass axioms into ACE must respect the sentential realization of universal quantification.

ensure that ontology engineers can see how OWL formalisms were mapped to ACE sentential structures.

The verbalization capability itself is based on a number of verbalization rules that map OWL class descriptions onto ACE noun phrases and OWL properties into active and passive verbs (see Table 6-1). Some OWL constructs cannot be directly verbalized in ACE because of the semantic ambiguities that would arise with a direct serialization into natural language. For example, the word 'range' has a number of meanings in natural language, none of which are necessarily consistent with its intended meaning in OWL, i.e. to specify the 'type' of the information object targeted by an OWL property. It would therefore be confusing, to say the least, to encounter the following sentence as part of everyday discourse:

```
'Is located in' has a range that is a spatial region.
```

OWL Properties and Classes	**Examples of Corresponding ACE Verbs and Noun Phrases**
Named property	*Transitive verb*, e.g. like
InverseObjectProperty(R)	*Passive verb*, e.g. is liked by
Namedclass	*Common noun*, e.g. cat
owl:Thing	something; thing
ObjectComplementOf(C)	something that is not a car; something that does not like a cat
ObjectIntersectionOf(C_1…C_n)	something that is not a cat and that owns a car and that…
ObjectUnionOf(C_1….C_n)	something that is a cat or that is a camel or that…
ObjectOneOf(a)	Proper name, e.g. John; something that is John
ObjectSomeValuesFrom(R C)	something that likes a cat
ObjectExistsSelf(R)	something that likes itself
ObjectMinCardinality(n R C)	something that owns at least 2 cars
ObjectMaxCardinality(n R C)	something that owns at most 2 cars
ObjectExactCardinality(n R C)	something that owns exactly 2 cars

Table 6-1: Verbalizing OWL property and class expressions as ACE verbs and noun phrases

Space limitations prohibit a detailed description of the ACE-OWL verbalization capability (more details can be found in Kaljurand and Fuchs (2007)); nevertheless, it appears that this capability can, at least on occasion, generate sentences that are considerably more readable (and probably more understandable) that their Description Logic counterparts. Take, for example, the following logical description of a complex class:

```
ObjectIntersectionOf(
cat
ObjectComplementOf(
   ObjectSomeValuesFrom(like
      ObjectIntersectionOf(
         dog
         ObjectUnionOf(
            ObjectSomeValuesFrom(attack mailman)
            ObjectOneOf(Fido))))))
```

This can be verbalized into ACE as:

```
something that is a cat and that does not like a dog that attacks a
mailman or that is Fido
```

Despite the apparent appeal of an ACE-OWL verbalization capability, Kaljurand and Fuchs (2007) identify a number of problems with their approach. Perhaps the most significant problem is, they suggest, the various naming conventions adopted for class and property names.

> *"the most visible deficiency of the described verbalization is caused by the naming conventions used in OWL ontologies".*

Most ontologies feature a range of orthographic and morphological styles when it comes to the naming of ontology elements. So, for example, we see such things as 'FifteenMinutes', 'NAMEDArtery' 'hasTopping', 'offeredIn', 'isPairedOrUnpaired', 'is_Located_In' and 'High_Or_Medium_Priority_Action'. Such variability is a problem for any ontology verbalization scheme (not just ACE-OWL) because it is not necessarily clear how we can abstract away from the orthographic idiosyncrasies of particular ontologies to yield a system that is generically capable of translating class, property and individual names into semantically coherent and linguistically-acceptable phrases. As was mentioned at the end of Section 6.2.2, one means of resolving this problem is to suggest a series of naming conventions which could be proposed as part of a suite of best-practice ontology authoring guidelines. These could be used by particular user communities, e.g. military coalitions, to ensure adequate ontology verbalization capabilities with respect to a community-specific domain of interest.

6.2.5 CLOnE Revisited

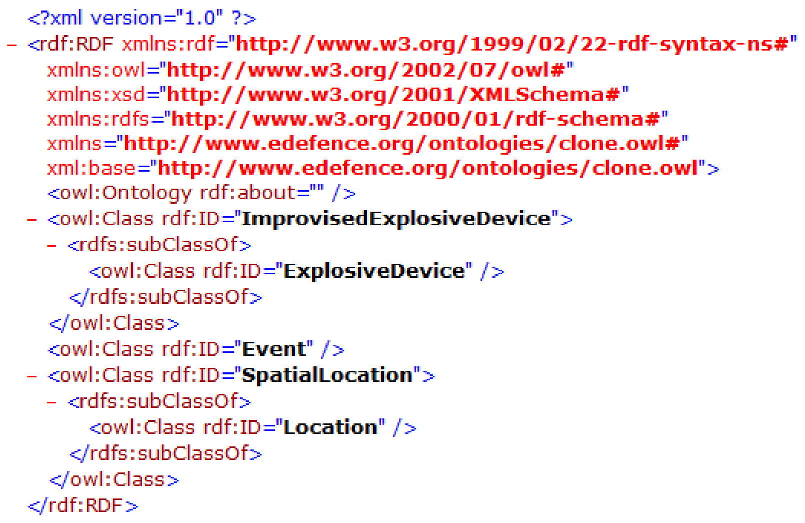
The CNL, CLOnE, was presented in Section 3.3.3 as a language supporting the creation of simple ontologies using natural language expressions. In addition to its use as an ontology authoring language, CLOnE can also be used for the purposes of ontology verbalization (Tablan et al., 2006). The CLOnE ontology verbalizer supports the generation of CLOnE sentences using a combination of XML configuration files and GATE processing resources. The XML configuration files contain templates that are matched to triples in the ontology using simple pattern-matching rules. Figure 6-7 shows a sample configuration template used to generate the text for top-level (i.e. root classes). The template consists of three components: an <in> element that is matched to the triple structure of the ontology (the triple specifications within the <in> element correspond to conditions determining whether or not the template will be used for text generation); an <out> element that specifies the text to be generated when the conditions of the template are satisfied (i.e. the triple specifications in the <in> element are successfully matched against the ontology); and an <ignoreIf> element that can be used to specify additional conditions for the application of the template (if the triple specifications within the <ignoreIf> element are matched to the ontology, then any matches in the <in> element are ignored).

```
<!-- Template for defining top classes -->
<template>
  <in>
    <triple id="t1">
      <property ns="rdf" name="type"/>
      <object ns="owl" name="Class"/>
    </triple>
  </in>
  <out>
    <singular>
      <phrase>There are <ref ref="t1.subject" number="plural"/>. </phrase>
    </singular>
    <plural>
      <phrase>There are <ref ref="t1.subject" number="plural"/>. </phrase>
    </plural>
  </out>
  <ignoreIf>
    <triple id="t2">
      <subject ref="t1.subject"/>
      <property ns="rdfs" name="subClassOf"/>
    </triple>
  </ignoreIf>
</template>
```

Figure 6-7: CLOnE language generation template

The way in which the templates are used to generate natural language sentences is relatively simple. Each <out> section of a template contains <phrase> elements that reference values to the triple(s) matched to triple specifications in the <in> section. Each <phrase> element also contains text elements that are copied to the output stream when the template is executed. The contents of the <reference> elements within the <phrase> element are replaced with values referenced in the <in> section. Using just the template presented in Figure 6-7, the ontology presented in Figure 6-8 will be serialized to the following CLOnE sentences:

```
There are Events. There are Locations. There are ExplosiveDevices.
```



```
  <?xml version="1.0" ?>
- <rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns="http://www.edefence.org/ontologies/clone.owl#"
    xml:base="http://www.edefence.org/ontologies/clone.owl">
    <owl:Ontology rdf:about="" />
  - <owl:Class rdf:ID="ImprovisedExplosiveDevice">
    - <rdfs:subClassOf>
        <owl:Class rdf:ID="ExplosiveDevice" />
      </rdfs:subClassOf>
    </owl:Class>
    <owl:Class rdf:ID="Event" />
  - <owl:Class rdf:ID="SpatialLocation">
    - <rdfs:subClassOf>
        <owl:Class rdf:ID="Location" />
      </rdfs:subClassOf>
    </owl:Class>
  </rdf:RDF>
```

Figure 6-8: CLOnE sample ontology

The use of natural language templates to verbalize OWL in CLOnE opens up the possibility of round-trip ontology authoring in which an existing ontology (perhaps developed in some other editing environment) can be serialized to CLOnE and then edited using natural language expressions. The problem with this proposed capability is that CLOnE only supports a limited subset of OWL formalisms (see Section 3.3.3). This means that any formalisms in the original ontology that are not supported by CLOnE will be lost during the course of the verbalization process. CLOnE verbalization could be used to support round-trip authoring with CLOnE-derived ontologies, but it is unlikely to be of much use in situations where ontologies are being edited in a distributed environment involving the use of multiple editing tools.

6.3 Summary

Ontology verbalization is a useful capability because it makes the knowledge content of the Semantic Web much more accessible to human end-users than would otherwise be the case. Whereas much attention in the Semantic Web and CNL community has been devoted to the development of natural language interfaces that support the creation of Semantic Web resources, it is important to remember that, in many cases, manual modes of semantic annotation and knowledge creation may not be as prevalent in the future as they are today. Instead, much of the Web and Semantic Web may be populated automatically[52], and our primary requirement will be to access Web[53] contents in a manner that makes sensible contact with our idiosyncratic cognitive and perceptual profile. Ontology verbalization may be seen as one capability that speaks to such a requirement.

[52] This is particularly pertinent when one considers the likely capabilities of future sensor and fusion systems to automatically interpret, process and disseminate semantically-enriched information.

[53] As an aside, it is worth noting that much of the current Web (the individual Web pages browsed by end users) is not manually created. Rather, it is generated from back-end data repositories whose content may or may not have been automatically generated.

7 Controlled Natural Languages and the Semantic Web

We have now witnessed CNLs being used in a variety of application contexts. For the most part, CNLs have emerged as a highly attractive addition to our technological toolkit for dealing with the Semantic Web. There is, however, a sense in which a more radical perspective has emerged. Rather than seeing CNLs as a mere interface language to the Semantic Web, as just one of many potential visualizations of Semantic Web content, there are some researchers in the CNL community who seem to suggest that CNLs should have a more central role in the representational infrastructure of the Semantic Web. The essence of the argument goes something like this:

> *CNLs are often more expressive than the knowledge representation languages (e.g. OWL) typically used on the Semantic Web. In many cases, CNLs are equivalent to first-order logic (e.g. ACE, Rabbit) and this means that they are more expressive than current versions of OWL, which correspond to decidable fragments of first-order logic. This greater expressivity, coupled with the fact that CNLs seem to be somewhat easier for human beings to use, seems to suggest that they are preferable to ontologies as the de facto knowledge representation language for the Semantic Web. If CNLs are at least representationally-equivalent to ontology languages, and they are also easier for humans to learn and to use, then why not dispense with ontologies altogether, or at least treat ontologies as secondary, derivative resources that are specifically generated to cater for machine-based processing.*

The rhetoric is somewhat more grandiose than what might be typically encountered in the research community, but the sentiment is real and not necessarily uncommon: one sees elements of this proposal being expressed in the ITA, and it is also apparent in the Ordnance Survey's use of Rabbit (instead of OWL) as the primary representational vehicle for geospatial knowledge. In the final section of this report I shall attempt to explore some of the issues associated with this rather radical thesis. I shall argue that although CNLs are generally useful, and that further research and technology development is certainly worthwhile, they should not be viewed as a replacement for the current representational bedrock of the Semantic Web.

7.1 Usability

The primary value of natural language interfaces to the Semantic Web, it would seem, lies in their ability to support human end-users with respect to the creation, manipulation and utilization of distributed knowledge resources. By establishing a rather direct contact with the linguistic capabilities of human agents, the hope is that we are able to circumvent at least some of the problems (e.g. the apparent difficulty humans have with formal logic) that may hinder (and indeed prohibit) the widespread adoption and uptake of semantic technologies by non-specialist user communities. The core aim of much research in the CNL community is therefore one of enhanced usability – of making Semantic Web technologies easier to use and more widely accessible to a multiplicity of end-user communities.

The usability function is, of course, the most obvious goal of CNL research, and we have now charted many ways in which natural language interfaces may contribute to this usability function. We have seen, for example, that CNLs can be used to create ontologies (see Section 4) and specify semantic queries (see Section 5) without requiring the user to learn the corresponding languages for knowledge representation and semantic query formulation (OWL and SPARQL, respectively). It seems that by tapping into the rich bodies of experience that human beings have with the medium of natural language, we are able to make semantic technologies easier to use and exploit.

And yet, for all that, I think there are a number of causes for concern. The truth is that relatively few usability studies have been undertaken to date with respect to natural language interfaces, and those that have been undertaken are (more often than not) full of methodological flaws and inadequacies. One problem seems to be the overriding tendency of workers in the CNL community to report the results of provisional usability studies in the context of some other more technically-oriented paper. This would be fine if the 'provisional' usability study was in fact the precursor to a more empirically-rigorous study; the results of such studies are, however, seldom reported in the literature (if they are even undertaken in the first place!). The main concern associated with the absence of good usability studies is that it is not at all clear whether CNLs actually do make the Semantic Web more usable, as opposed to simply more *accessible*. Obviously, the use of natural languages would be expected to make the Semantic Web more accessible to language-using agents, but this does not necessarily mean that the Semantic Web suddenly becomes a much more user-friendly environment in which to work. Firstly, it is not clear whether the time and effort required to learn a CNL is any lesser (or greater) than the time and effort required to become proficient with semantic technologies. Even in cases where CNL interfaces seem to support users with respect to capabilities such as ontology editing, it is not entirely clear whether such usability benefits stem from the intrinsic features of the CNL, or from the deficiencies of the tool with which it is compared. In this respect it should be noted that tools such as Protégé-OWL were not necessarily designed to support the same kind of users as CNL interfaces (i.e. casual users), and, as such, direct comparisons between these editing environments are largely invalid. Furthermore, most usability studies tend to record user preference for one tool over another. These preference ratings are not necessarily a reliable indicator of usability – just because users prefer a tool does not necessarily mean that it is more usable – and, in any case, such measurements often fail to consider the trade-offs that might be made with respect to tool functionality and usability features (a CNL tool might be more usable, but may not provide the same functionality as an alternative tool).

A second problem with the claim that CNLs enhance the usability of semantic technologies relates to the difference between the mere readability of CNL expressions as opposed to their semantic comprehensibility. Obviously a few CNL representations would seem to be much easier for (most) human beings to read than an equivalent (semantically-equivalent) collection of esoteric logical formulae. Who could possibly disagree with that? But readability is not the same as comprehensibility, and it is not always clear whether humans understand the logical significance of CNL representations in quite the way that is implied by their formal semantic representation. Even in cases where human agents know the grammatical rules underpinning the production and interpretation of sentences in the target CNL, do they really understand the 'meaning' associated with such sentences? It is not entirely clear to me that they do, and one of the reasons for doubt lies in the possibility that humans may be congenitally ill-equipped to deal with the vagaries of formal logic (see Section 2). Inasmuch as CNLs provide representational formalisms, albeit homely and familiar linguistic ones, that have their analogues in the paraphernalia of formal logic, then why assume that sentential structures expressing logical statements should be any more understandable to human beings than the formal expressions they are intended to replace? Ultimately, if the real root of the Semantic Web usability problem lies at a level below the level of symbols, words and mathematical expressions, if it is, in fact, grounded in the deficiencies of the human cognitive system, then no amount of linguistic massaging may help to solve the problem. A logical formalism by any other name might prove to be just as impenetrable to our logic-lubberly interpretive faculties.

7.2 Expressivity

Another argument that is sometimes presented in debates about the relative merits of CNLs as the representational bedrock of the Semantic Web, concerns their greater expressivity with respect to current versions of OWL. Obviously, there are cases where CNLs are (considerably) more expressive than OWL (ACE and Rabbit are, by now, two familiar examples). But claiming that this makes the CNL a viable substitute for OWL is a strong claim. The first thing to say is that OWL as an ontology representation language is constantly evolving, and the range of formalisms (and representational capabilities) being proposed as part of the OWL 1.1[54] specification exceeds those seen in OWL 1.0 by some considerable margin. This may, to some extent at least, close the gap between OWL and CNLs in terms of their relative expressivity.

Another point worth mentioning with respect to expressivity is that OWL is sometimes talked about as though it defines the limits of our representational vista for the Semantic Web. The representational inadequacies of OWL have been known for some time and this has resulted in the invention of additional ways of representing and communicating knowledge. Chief among these is the strategy of using rule languages, such as SWRL, to capture knowledge-rich contingencies that would be difficult, or impossible to express using OWL by itself. Clearly, the expressivity of OWL is limited, but why suppose that it, or indeed any knowledge representation language, need function in isolation. It seems entirely plausible to me that a whole range of representational formalisms might be required (not all of them currently part of OWL, or even formal logic) in order to capture the full extent of human knowledge on the Semantic Web. Once we accept this claim (i.e. that a multiplicity of representational devices might be required on the Semantic Web), then arguments about the greater expressivity of CNLs as the basis for wholesale revisions of our technological approach to the Semantic Web seem to be increasingly untenable[55].

In contrast to the cases where CNLs are more expressive than OWL, we should not forget the cases where CNLs are *less* expressive than OWL. We see examples of this 'inferior' expressivity in the case of CLOnE (see Section 3.3.3), which, as you may recall, supports users with respect to the creation of simple, lightweight ontologies. It is tempting in such cases, I suspect, to see the reduced expressivity as a negative feature, something that counts against the CNL. I am largely opposed to such a view. Inasmuch as CNLs aim to enhance the usability of the Semantic Web for casual users, I suspect they do not necessarily require highly expressive features – if such features were so important to end users perhaps they would be better off using an editing environment specifically designed for the creation of complex ontological expressions. In any case, we have seen that humans are not always

[54] http://www.w3.org/Submission/owl11-overview/

[55] This does not, of course, negate the value of using CNLs as a uniform interface for the creation of knowledge content using a variety of Semantic Web languages (see Section 4.5 for a more detailed discussion of this issue).

well-equipped to deal with the idiosyncrasies of formal logic systems, and it would seem natural to assume that one reason for introducing CNLs is to assist human users with respect to these shortcomings. If this is the case, then why suppose that increased expressivity is such a good thing? Perhaps such moves simply serve to complicate the use of the CNL and reduce its usability to casual or novice users. Given our apparent tendency to get 'lost in logic', efforts to increase the logical expressivity of knowledge representation languages does not always, to me at least, seem such a good idea.

7.3 Machine Minds

The Semantic Web was originally conceived as an extension to the conventional Web. The idea was that the Semantic Web would provide a semantically-enriched substrate on which a number of advanced capabilities could be founded. These capabilities capitalize on the fact that the Semantic Web was (perhaps primarily) a mechanism for enabling machine access to knowledge content; it made knowledge accessible to machines in a way that the messy, unstructured linguistic substratum of the conventional Web never could (and perhaps never will). Amidst the seemingly endless tirade of complaints about the awkwardness, complexity and downright intractability of the Semantic Web, especially from those ill-versed in the black arts of formal logic, it is easy to forget that that Semantic Web was not necessarily designed to appeal to human sensibilities. Humans already have their Semantic Web – it is the conventional WWW, the web of natural language, of images and video, the homely media with which we are all accustomed to using as part of our everyday communication, entertainment and worldly edification.

The Semantic Web is clearly different from this conventional Web (otherwise why bother with the new phraseology?). But it is different not in terms of its technological infrastructure – the Semantic Web for the most part uses the same set of communication protocols and network infrastructure components as the conventional Web. Rather, the difference lies in the intended purpose of the Semantic Web, i.e. the need to support automated information processing and artificial intelligence capabilities. It is here, I think, that the most compelling objection to the radical claim advanced above can be found. CNLs are designed for human use and consumption, and they are not necessarily well-adapted to the world of automated information transformation, reasoning and distributed intelligence. One problem is that in order to be useful to machines, the CNL (the communicative vehicle conveying domain-relevant knowledge) has to be translated into some other format that is more supportive of machine processing. Another problem is that CNLs may not be ideally suited to cases where we need to support machine-machine communication and the dissemination, processing and exploitation of information sources from advanced sensor systems and intelligent devices. These are the kinds of systems that are likely to become increasingly prevalent in the networks of the future, and it is such systems that the technological infrastructure of the Semantic Web is (primarily) designed to support.

7.4 Pragmatics

Besides the scientific arguments against a greater role for CNLs in the representation of knowledge on the Semantic Web, there are a host of pragmatic and political issues at stake. Firstly, out of all the CNLs that are currently available (see Section 3), which one should we in fact use? Inasmuch as these languages boast different features and capabilities (see Schwitter et al., 2008), how should we align or integrate knowledge contained in each of the various languages? Do we not confront here a situation akin to the days before the advent of OWL as a 'standard' for knowledge representation on the World Wide Web?

Another practical issue concerns the types of query mechanism we should use when querying CNL models. How should we query CNL models in a way that makes sensible contact with existing techniques and technologies, e.g. SPARQL? What would the CNL equivalent to SPARQL actually look like and how easy would it be for machines to process the syntactic idiosyncrasies of this new query language? Would we need to disband a variety of relatively mature semantic technology components whose development has been driven by the technological evolution of the Semantic Web?

Finally, how do we get *machines* to automatically generate natural language representations of domain relevant knowledge, some of which may have been acquired without *any* form of human intervention (e.g. intelligent sensor systems)? Can machines ever communicate automatically acquired (or inferred) knowledge to humans in a form that is consistent with our human expectations regarding narrative and rhetorical structure?

7.5 Conclusion

The radical thesis outlined in the introduction to this section advocated the use of CNLs as the primary means for representing knowledge on the Semantic Web. The thesis is grounded on a number of claims about the relative advantages of CNLs as a knowledge representation language for the Semantic Web. One of the core claims is that CNLs contribute to the usability of the Semantic Web and that this is a key element in the increased uptake and acceptability of the technology to human end users. My response to this usability claim is that we should not be too hasty in concluding that CNLs (or indeed natural languages!) are as user friendly (from a knowledge representation perspective) as they might at first appear. There have been very few user evaluation studies that have been undertaken to confirm that CNLs, with all their apparent advantages in terms of readability, are also understandable to humans, at least in a way that supports logically-sound reasoning. All too often it is easy to confuse the apparent ease with which humans interact with linguaform resources with the notion that they must, as a result, find linguistic serializations of logically-specified knowledge easier to understand. We should be wary of such a conclusion, not least because some of the studies in cognitive psychology seem to demonstrate failures in logical reasoning with what are clearly sentential structures communicating basic facts about a domain of discourse (see Section 2)[56]. If humans prove so deficient at understanding the logical implications of linguistically-specified knowledge in these rather constrained contexts, why assume that CNL representations of domain-relevant knowledge, in the context of the Semantic Web, will fare any better?

Perhaps the greatest reason for doubting claims about the supposed superiority of CNLs relative to ontology representational languages (i.e. OWL) relates to the fact that the Semantic Web is not necessarily intended to support human knowledge processing. What is a good representational substrate for epistemic processing in the case of human agents is not necessarily a good basis for

[56] In particular, linguaform representations might encourage humans to make background assumptions and inferences based on their experiences with the cooperative dynamics of conversational discourse. If so, then such representations may lead to inconsistencies in semantic interpretation, especially where we encounter the interaction of human and machine agents as part of some collaborative problem-solving activity.

machine intelligence. The point here is that just because language is supportive of human cognition, it does not follow that linguaform representations can support advanced intelligence capabilities in a network environment, especially one populated by a variety of heterogeneous machine entities (e.g. sensors, agents, services, intelligent devices, etc.). Why assume, in fact, that the external manifestation of our inner imaginings, the serialization of our inner cognitive worlds into the sentential apparatus of natural language, could ever be anything other than a rather shallow reflection of some deeper representational and computational reality that undergirds much of our epistemic and cognitive potential. Natural language is such a salient part of our cognitive profile that it is difficult at times to avoid the conclusion that natural language must occupy a central part of our cognitive prowess. But why assume that the structures of natural language are really the most appropriate medium for the instantiation of intelligence in a distributed network environment, especially one that subtends a panoply of disparate epistemic resources and services?

I suspect this failure to appreciate the true nature, and indeed potential, of the Semantic Web underpins many of the more radical claims about the use of CNLs as a primary knowledge representation language for the Semantic Web. Once one understands the issues at stake for the instantiation of more intelligent and automatic services in the networks of the future, one begins to appreciate the importance of emphasizing representational structures and devices that selectively favour machine cognition, perhaps at the expense of a little human usability. Nevertheless, it would be a mistake to downplay or undermine the genuine contribution that CNLs and their associated interfaces can bring to the Semantic Web. Ultimately, I suggest, that such interfaces may provide the basis of a human-machine interface that allows for the emergence of a new type of hybrid cognitive system, one that is not bounded by the traditional boundaries of skin or skull, but rather one that co-opts a variety of internally- and externally-located cognitive resources into a new form of network-enabled capability – a future hybrid intelligence reflecting the true potential of the network-enabled mind (Smart et al., 2008b).

8 References

Aguado, G., Banon, A., Bateman, J., Bernardos, S., Fernandez, M., Gomez-Perez, A., Nieto, E., Olalla, A., Plaza, R., & Sanchez, A. (1998) *ONTOGENERATION: Reusing domain and linguistic ontologies for Spanish text generation*. Workshop on Applications of Ontologies and Problem Solving Methods at ECAI1998, Brighton, UK.

Alani, H., Kim, S., Millard, D. E., Weal, M. J., Hall, W., Lewis, P., & Shadbolt, N. R. (2003) Automatic Ontology-Based Knowledge Extraction from Web Documents. *IEEE Intelligent Systems, 18*(1), 14-21.

Baclawski, K., Matheus, C., Kokar, M., Letkowski, J., & Kogut, P. (2004) *Towards a symptom ontology for semantic web applications*. 3rd International Semantic Web Conference, Hiroshima, Japan.

Bernardi, R., Calvanese, D., & Thorne, C. (2007) *Lite Natural Language*. 7th International Workshop on Computational Semantics (IWCS-7), Tilburg, Netherlands.

Berners-Lee, T., Hendler, J., & Lassila, O. (2001) The Semantic Web. *Scientific American, 284*(4), 34-43.

Bernstein, A., & Kaufmann, E. (2006) *GINO-A Guided Input Natural Language Ontology Editor*. 5th International Semantic Web Conference (ISWC2006), Athens, Georgia, USA.

Bernstein, A., Kaufmann, E., Fuchs, N. E., & von Bonin, J. (2004) *Talking to the Semantic Web: A Controlled English Query Interface for Ontologies*. 14th Annual Workshop on Information Technologies and Systems (WITS 2004). , Washington DC, USA.

Bernstein, A., Kaufmann, E., Gohring, A., & Kiefer, C. (2005a) Querying ontologies: A controlled english interface for end-users. *Proc. 4th International Semantic Web Conference (ISWC05)*, 112–126.

Bernstein, A., Kaufmann, E., & Kaiser, C. (2005b) *Querying the Semantic Web with Ginseng: A Guided Input Natural Language Search Engine*. 15th Workshop on Information Technology and Systems (WITS 2005), Las Vegas, Nevada, USA.

Bernstein, A., Kaufmann, E., Kaiser, C., & Kiefer, C. (2006) *Ginseng: A Guided Input Natural Language Search Engine for Querying Ontologies*. Jena User Conference Bristol, UK.

Bontcheva, K., Tablan, V., Maynard, D., & Cunningham, H. (2004) Evolving GATE to meet new challenges in language engineering. *Natural Language Engineering, 10*, 349-373.

Bontcheva, K., & Wilks, Y. (2004) *Automatic report generation from ontologies: the MIAKT approach*. 9th International Conference on Applications of Natural Language to Information Systems, Manchester, UK.

Braine, M. D. S. (1978) On the relationship between the natural logic of reasoning and standard logic. *Psychological Review, 85*(1-21).

Braines, D., Kalfoglou, Y., Smart, P. R., Bao, J., Shadbolt, N. R., & Hendler, J. (2008a) *Semantic Web Techniques to Support Interoperability in Distributed Networked Environments*. Annual Conference of the International Technology Alliance (ACITA'08), London, UK.

Braines, D., Kalfoglou, Y., Smart, P. R., Shadbolt, N. R., & Bao, J. (2008b) *A Data-Intensive Lightweight Semantic Wrapper Approach to Aid Information Integration*. 4th International Workshop on Contexts and Ontologies (C&O) hosted by the 18th European Conference on Artificial Intelligence (ECAI'08), Patraz, Greece.

Brooke, J. (1996) SUS: A 'quick and dirty' usability scale. In P. Jordan, B. Thomas, B. Weedmeester & A. McClelland (Eds.), *Usability Evaluation in Industry*. Taylor and Francis, London, UK.

Clark, A. (1997) *Being There: Putting Brain, Body and World Together Again*. MIT Press, Cambridge, Massachusetts.

Clark, A. (1998) Magic words: how language augments human computation. In P. Carruthers & J. Boucher (Eds.), *Language and Thought: Interdisciplinary Themes*. Cambridge University Press, Cambridge, UK.

Clark, A. (2004) Towards a science of the bio-technological mind. In B. Gorayska & J. Mey (Eds.), *Cognition and Technology: Co-existence, Convergence and Co-evolution*. John Benjamins Publishing Company, Amsterdam, Netherlands.

Clark, A. (2006) Language, embodiment, and the cognitive niche. *Trends In Cognitive Sciences, 10*(8), 370-374.

Clark, P., Harrison, P., Jenkins, T., Thompson, J., & Wojcik, R. (2005) *Acquiring and Using World Knowledge Using a Restricted Subset of English*. 18th International FLAIRS Conference (FLAIRS 2005), Florida , USA.

Cregan, A., Schwitter, R., & Meyer, T. (2007) *Sydney OWL Syntax—towards a Controlled Natural Language Syntax for OWL 1.1*. OWL Experiences and Directions Workshop, Innsbruck, Austria.

Cunningham, D. H., Maynard, D. D., Bontcheva, D. K., & Tablan, M. V. (2002) *GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications*. 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02), Philadelphia, USA.

Denaux, R., Holt, I., Dimitrova, V., Dolbear, C., & Cohn, A. G. (2008) *Supporting the Construction of Conceptual Ontologies with the ROO Tool*. OWL: Experiences and Directions (OWLED'08), Washington D.C., USA.

Dennett, D. (1991) *Conciousness Explained*. Little, Brown & Company, Boston, Massachusetts, USA.

Dolbear, C., Hart, G., Kovacs, K., Goodwin, J., & Zhou, S. (2007) *The Rabbit Language: Description, Syntax and Conversion to OWL.* Ordnance Survey, Southampton, UK.

Evans, J. S. B. T., Newstead, S. E., & Byrne, R. M. J. (1993) *Human Reasoning: The Psychology of Deduction*. Lawrence Erlbaum Associates, Hove, UK.

Eysenck, M. W., & Keane, M. T. (1995) *Cognitive Psychology: A Student's Handbook* (3rd ed.). Lawrence Erlbaum Associates, Hove, UK.

Fellbaum, C. (Ed.). (1998) *Wordnet: An Electronic Lexical Database*. MIT Press, Cambridge, Massachusetts, USA.

Fuchs, N. E., Kaljurand, K., Kuhn, T., Schneider, G., Royer, L., & Schroder, M. (2006a) *Attempto Controlled English and the Semantic Web.* University of Zurich, Zurich, Switzerland.

Fuchs, N. E., Kaljurand, K., & Schneider, G. (2006b) *Attempto Controlled English Meets the Challenges of Knowledge Representation, Reasoning, Interoperability and User Interfaces*. 19th International FLAIRS Conference (FLAIRS'2006), Florida, USA.

Fuchs, N. E., & Schwertel, U. (2003) *Reasoning in Attempto Controlled English*. International Workshop on Principles and Practice of Semantic Web Reasoning, Mumbai, India.

Funk, A., Davis, B., Tablan, V., Bontcheva, K., & Cunningham, D. H. (2007a) *Controlled Language IE Components.* University of Sheffield, Sheffield, UK.

Funk, A., Tablan, V., Bontcheva, K., Cunningham, H., Davis, B., & Handschuh, S. (2007b) *CLOnE: Controlled Language for Ontology Editing*. 6th International Semantic Web Conference (ISWC2007), Busan, Korea.

Garnham, A. (1985) *Psycholinguistics*. Routledge, London, UK.

Grice, H. P. (1975) Logic and conversation. In P. Cole & J. L. Morgan (Eds.), *Syntax and Semantics: Vol III. Speech Acts*. Seminar Press, New York, USA.

Halaschek-Wiener, C., Golbeck, J., Parsia, B., Kolovski, V., & Hendler, J. (2006) *Image browsing and natural language paraphrases of semantic web annotations*. 1st International Workshop on Semantic Web Annotations for Multimedia (SWAMM), Edinburgh, Scotland.

Hart, G., Dolbear, C., & Goodwin, J. (2007) *Lege Feliciter: Using Structured English to represent a Topographic Hydrology Ontology*. OWL Experiences and Directions Workshop, Innsbruck, Austria.

Hewlett, D., Kalyanpur, A., Kovlovski, V., & Halaschek-Wiener, C. (2005) *Effective Natural Language Paraphrasing of Ontologies on the Semantic Web*. End User Semantic Web Interaction Workshop, International Semantic Web Conference (ISWC 2005), Galway, Ireland.

Holger, K., Ferguson, R. W., Noy, N. F., & Musen, M. A. (2004) *The Protege OWL Plugin: An Open Development Environment for Semantic Web Applications*. 3rd International Semantic Web Conference - ISWC 2004, Hiroshima, Japan.

Horridge, M., Drummond, N., Goodwin, J., Rector, A., Stevens, R., & Wang, H. H. (2006) *The Manchester OWL Syntax*. OWL Experiences and Directions Workshop (OWLED'06) at ISWC2006, Athens, Goergia, USA.

Hutchins, E., & Hazelhurst, B. (1991) Learning in the cultural process. In C. G. Langdon, C. Taylor, J. D. Farmer & S. Rasmussen (Eds.), *Artificial Life II*. Addison-Wesley Publishing Company, Redwood City, CA, USA.

Kalfoglou, Y., Smart, P. R., Braines, D., & Shadbolt, N. R. (2008) *POAF: Portable Ontology Aligned Fragments*. International Workshop on Ontologies: Reasoning and Modularity (WORM 2008) hosted by the 5th European Semantic Web Conference (ESWC'08), Tenerife, Spain.

Kaljurand, K. (2007) *Attempto Controlled English as a Semantic Web Language.* University of Tartu, Tartu, Estonia.

Kaljurand, K., & Fuchs, N. E. (2006a) *Bidirectional mapping between OWL DL and Attempto Controlled English*. 4th Workshop on Principles and Practice of Semantic Web Reasoning, Budva, Montenegro.

Kaljurand, K., & Fuchs, N. E. (2006b) *Mapping Attempto Controlled English to OWL DL*. 3rd European Semantic Web Conference (EWSC), Budva, Montenegro.

Kaljurand, K., & Fuchs, N. E. (2007) *Verbalizing OWL in Attempto Controlled English*. OWL Experiences and Directions Workshop, Innsbruck, Austria.

Kalyanpur, A., Halaschek-Wiener, C., Kolovski, V., & Hendler, J. (2005) *Effective NL paraphrasing of ontologies on the semantic web.* Department of Computer Science, University of Maryland, Maryland, USA.

Kalyanpur, A., Parsia, B., Sirin, E., & Cuenca-Grau, B. (2006) *Repairing unsatisfiable concepts in owl ontologies*. 3rd European Semantic Web Conference (ESWC'06), Budva, Montenegro.

Kaufmann, E., & Bernstein, A. (2007) *How Useful are Natural Language Interfaces to the Semantic Web for Casual End-Users?* 6th International Symantic Web Conference (ISWC 2007), Busan, Korea.

Kaufmann, E., Bernstein, A., & Fischer, L. (2007) *NLP-Reduce: A" naive" but domain-independent natural language interface for querying ontologies*. 4th European Semantic Web Conference (ESWC2007), Innsbruck, Austria.

Kaufmann, E., Bernstein, A., & Zumstein, R. (2006) *Querix: A Natural Language Interface to Query Ontologies Based on Clarification Dialogs*. International Semantic Web Conference (ISWC2006), Athens, Geogia, USA

Kim, S., Alani, H., Hall, W., Lewis, P. H., Millard, D. E., Shadbolt, N., & Weal, M. (2002) *Artequakt: Generating Tailored Biographies with Automatically Annotated Fragments from the Web*. Semantic Authoring, Annotation and Knowledge Markup (SAAKM) 2002 Workshop at the 15th European Conference on Artificial Intelligence (ECAI 2002), Lyon, France.

Kittredge, R. I. (2003) *Sublanguages and controlled languages*. Oxford University Press, Oxford, UK.

Klein, D., & Manning, C. D. (2003) *Accurate unlexicalized parsing*. 41st Annual Meeting of the Association for Computational Linguistics, Sapporo, Japan.

Kovacs, K., Dolbear, C., Hart, G., & Mizen, A. (2006) *A Methodology for Building Conceptual Domain Ontologies.* Ordnance Survey Research Labs, Southampton, UK.

Krötzsch, M., Vrandecic, D., Völkel, M., Haller, H., & Studer, R. (2007) Semantic Wikipedia. *Journal of Web Semantics, 5*, 251-261.

Kuhn, T. (2006) *Attempto Controlled English as an Ontology Language*. Reasoning on the Web with Rules and Semantics, Munich, Germany.

Kuhn, T. (2007) *AceRules: Executing Rules in Controlled Natural Language*. Web Reasoning and Rule Systems (RR2007), Innsbruck, Austria.

Kuhn, T. (2008) *AceWiki: A Natural and Expressive Semantic Wiki*. Semantic Web User Interaction Workshop at CHI 2008, Florence, Italy.

Lopez, V., & Motta, E. (2004) *Ontology-driven question answering in AquaLog*. 9th International Conference on Applications of Natural Language to Information Systems, Salford, UK.

Lopez, V., Motta, E., & Uren, V. (2006) *PowerAqua: Fishing the Semantic Web*. 3rd European Semantic Web Conference (ESWC2006), Budva, Montenegro.

Lopez, V., Pasin, M., & Motta, E. (2005) *AquaLog: An Ontology-portable Question Answering System for the Semantic Web*. 2nd European Semantic Web Conference (ESWC 2005), Heraklion, Greece.

Marcus, S. L., & Rips, L. J. (1979) Conditional reasoning. *Journal of Verbal Learning and Verbal Behaviour, 18*, 199-233.

Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., & Miller, K. J. (2004) Introduction to WordNet: An On-line Lexical Database. *International Journal of Lexicography, 3*(4), 235-244.

Mott, D., & Hendler, J. (2007) *Progress on the Collaborative Planning Model*. 1st Annual Conference of the International Technology Alliance (ACITA), Maryland, USA.

Patel-Schneider, P. F., Hayes, P., & Horrocks, I. (2004) *OWL Web Ontology Language Semantics and Abstract Syntax.* W3C.

Patel-Schneider, P. F., & Horrocks, I. (2006). OWL 1.1 Web Ontology Language Overview. from http://www.w3.org/Submission/owl11-overview/

Pool, J. (2006) *Can controlled languages scale to the Web?* 5th International Workshop on Controlled Language Applications (CLAW'06), Cambridge, Massachusetts, USA.

Preece, A., & Sieck, W. R. (2007) The International Technology Alliance in Network and Information Sciences. *IEEE Intelligent Systems, 22*(5), 18-19.

Rector, A., Drummond, N., Horridge, M., Rogers, J., Knublauch, H., Stevens, R., Wang, H., & Wroe, C. (2004) *OWL Pizzas: Practical Experience of Teaching OWL-DL: Common Errors & Common Patterns*. European Conference on Knowledge Acquisition (EKAW-2004), Whittlebury, UK.

Rousset, M. C. (2004) *Small can be beautiful in the semantic web*. 3rd International Semantic Web Conference (ISWC), Hiroshima, Japan.

Schwitter, R. (2005a) *Controlled Natural Language as Interface Language to the Semantic Web*. 2nd Indian International Conference on Artificial Intelligence (IICAI-05), Pune, India.

Schwitter, R. (2005b) *A Controlled Natural Language Layer for the Semantic Web*. 18th Australian Joint Conference on Artificial Intelligence, Sydney, Australia.

Schwitter, R. (2007) *Controlled Natural Languages.* Centre for Language Technology, Macquarie University, New South Wales, Australia.

Schwitter, R., Kaljurand, K., Cregan, A., Dolbear, C., & Hart, G. (2008) *A Comparison of three Controlled Natural Languages for OWL 1.1*. OWL: Experiences and Directions (OWLED'08), Washington D.C., USA.

Schwitter, R., & Tilbrook, M. (2004) *Controlled Natural Language meets the Semantic Web*. Australasian Language Technology Workshop, Sydney, Australia.

Schwitter, R., & Tilbrook, M. (2006) *Let's Talk in Description Logic via Controlled Natural Language*. Logic and Engineering of Natural Language Semantics (LENLS2006), Tokyo, Japan.

Shadbolt, N. R., & Burton, M. (1990) Knowledge elicitation: a systematic approach. In J. R. Wilson & E. N. Corlett (Eds.), *Evaluation of Human Work: A Practical Ergonomics Methodology* (2nd ed., pp. 406-440). Taylor and Francis, London, England.

Shadbolt, N. R., Lewis, P., Dasmahaptra, S., Dupplaw, D., Hu, B., & Lewis, H. (2004) *MIAKT: combining grid and web services for collaborative medical decision making*. 2nd e-Science All Hands Meeting, Nottingham, England, UK.

Shadbolt, N. R., O'Hara, K., & Crow, L. (1999) The experimental evaluation of knowledge acquisition techniques and methods: history, problems and new directions. *International Journal of Human-Computer Studies, 51*(4), 729-755.

Smart, P. R., Russell, A., Mott, D., Braines, D., Bao, J., Giammanco, C., Houghton, P., Shadbolt, N. R., & Hendler, J. (2008a) *Interfacing with the Semantic Web: Graphical and Natural Language Approaches to Knowledge Editing, Ontology Alignment and Information Retrieval.* School of Electronics and Computer Science, University of Southampton, Southampton, UK. (Ref: ITA/P12/SWI)

Smart, P. R., Shadbolt, N. R., Hendler, J., Engelbrecht, P., Giammanco, C., Nixon, M., & Harnad, S. (2008b) *Knowledge In Action: Knowledge Representation and the Future of Web-Enabled Intelligence.* School of Electronics and Computer Science, University of Southampton, Southampton, England. (Ref: ITA/P12/KnowAction)

Smith, M. K., Welty, C., & McGuiness, D. L. (2004). OWL Web Ontology Language Guide. World Wide Web Consortium: http://www.w3.org/TR/owl-guide/

Tablan, V., Polajnar, T., Cunningham, H., & Bontcheva, K. (2006) *User-friendly ontology authoring using a controlled language*. 5th International Conference on Language Resources and Evaluation (LREC), Genoa, Italy.

Wang, C., Xiong, M., Zhou, Q., & Yu, Y. (2007) *PANTO: A Portable Natural Language Interface to Ontologies*. European Semantic Web Conference (ESWC2007), Innsbruck, Austria.

Wilcock, G. (2003) *Talking OWLs: Towards an Ontology Verbalizer*. Workshop on Human Language Technology for the Semantic Web and Web Services at ISWC2003, Sanibel Island, Florida, USA.

Appendix A Acronyms & Abbreviations

ACE	Attempto Controlled English
AKT	Advanced Knowledge Technologies
APE	Attempto Parsing Engine
CLOnE	Controlled Language for Ontology Editing
CNL	Controlled Natural Language
DAML	DARPA Agent Mark-up Language
DIFDTC	Data and Information Fusion Defence Technology Centre
DL	Description Logic
DRS	Discourse Representation Structure
GATE	General Architecture for Text Engineering
GINO	Guided Input Natural Language Ontology Editor
GINSENG	Guided Input Natural Language Search Engine
GUI	Graphical User Interface
IKAT	Internet-enabled Knowledge Acquisition Toolkit
ITA	International Technology Alliance
JAPE	Java Annotation Patterns Engine
JRE	Java Runtime Environment
KB	Knowledge Base
KRL	Knowledge Representation Language
MIAKT	Medical Imaging with Advanced Knowledge Technologies
MoD	Ministry of Defence
MOS	Manchester OWL Syntax
NLER	Natural Language Entity Renderer
NLG	Natural Language Generation
NLP	Natural Language Processing
OIL	Ontology Inference Layer
OWL	Web Ontology Language
PANTO	Portable Natural Language Interface to Ontologies

PENG	Processable English
POS	Part Of Speech
PQL	Process Query Language
RDF	Resource Description Framework
RDF-S	RDF Vocabulary Description Language
RDQL	RDF Data Query Language
REWERSE	Reasoning on the Web with Rules and Semantics
ROO	Rabbit to OWL Ontology
RSS	Relation Similarity Service
SEMIOTIKS	Semantically-Enhanced Information Extraction for Improved Knowledge Superiority
SOS	Sydney OWL Syntax
SPARQL	Simple Protocol and RDF Query Language
SUS	System Usability Scale
SWAT	Semantic Web ACE Transformer
SWRL	Semantic Web Rules Language
W3C	World Wide Web Consortium
WWW	World Wide Web
XHTML	eXtensible Hypertext Mark-up Language
XML	eXtensible Mark-up Language
XSLT	eXtensible Stylesheet Language Transformation