Involving Domain Experts in Authoring OWL Ontologies*

Vania Dimitrova[1], Ronald Denaux[1], Glen Hart[2], Catherine Dolbear[2],
Ian Holt[2], and Anthony G. Cohn[1]

[1] School of Computing, University of Leeds, Woodhouse Lane, Leeds, LS2 9JT, UK
[2] Ordnance Survey Research, Romsey Rd, Southampton, SO16 4GU, UK
{vania,rdenaux,agc}@comp.leeds.ac.uk
{Glen.Hart,Catherine.Dolbear,Ian.Holt}@ordnancesurvey.co.uk

Abstract. The process of authoring ontologies requires the active involvement of domain experts who should lead the process, as well as providing the relevant conceptual knowledge. However, most domain experts lack knowledge modelling skills and find it hard to follow logical notations in OWL. This paper presents ROO, a tool that facilitates domain experts' definition of ontologies in OWL by allowing them to author the ontology in a controlled natural language called Rabbit. ROO *guides* users through the ontology construction process by following a methodology geared towards domain experts' involvement in ontology authoring, and exploiting intelligent user interfaces techniques. An evaluation study has been conducted comparing ROO against another popular ontology authoring tool. Participants were asked to create ontologies based on hydrology and environment modelling scenarios related to real tasks at the mapping agency of Great Britain. The study is discussed, focusing on the usability and usefulness of the tool, and the quality of the resultant ontologies.

Keywords: Ontology Authoring, Controlled Natural Language Interfaces, Evaluation of Ontology Building Tools, Geographical Ontologies.

1 Introduction

The need to construct ontologies – ranging from small domain ontologies to large ontologies linked to legacy datasets– hinders the ability and willingness of organisations to apply Semantic Web (SW) technologies to large-scale data integration and sharing initiatives [1,7,9]. This is due to the time and effort required to create ontologies [1,19]. Most ontology construction tools aggravate the situation because they are designed to be used by specialists with appropriate knowledge engineering and logic skills, but who may lack the necessary domain expertise to create the relevant ontologies. At present, it is knowledge engineers who usually drive the ontology authoring process, which creates an extra layer of bureaucracy in the development cycle [19].

* The work reported here is part of a research project, called Confluence, funded by the Ordnance Survey and conducted by an interdisciplinary team from the University of Leeds and Ordnance Survey. The main goal of the project is the development of the ontology construction tool ROO, presented in this paper.

A. Sheth et al. (Eds.): ISWC 2008, LNCS 5318, pp. 1 – 16, 2008.

Furthermore, this knowledge engineer led approach can hinder the ontology construction process because the domain expert and domain knowledge may become secondary to the process of efficient knowledge modelling. This is especially true where the domain expert has no understanding of the languages and tools used to construct the ontology. The development of approaches that facilitate the engagement of domain experts in the ontology construction process can lead to a step change in the deployment of the Semantic Web in the public and industrial sector.

Such an approach, drawn upon extensive experience in creating topographic ontologies at Ordnance Survey, the mapping agency of Great Britain, is described here. Ordnance Survey is developing a topographic domain ontology to empower the integration and reuse of their heterogeneous topographic data sets with third party data [9]. At the heart of Ordnance Survey's ontology development process is the *active involvement of domain experts* [20]. They construct *conceptual ontologies* that record domain knowledge in a human readable form with appropriate formality using a controlled language, Rabbit[1] [14], that is translated into OWL DL [8].

The paper presents ROO (Rabbit to OWL Ontology authoring), a user-friendly tool that guides the authoring of a conceptual ontology which is then converted to a logical ontology in OWL. The *distinctive characteristics* of our approach are: (a) catering for the needs of domain experts without knowledge engineering skills; (b) exploiting techniques from intelligent user interfaces to assist the ontology construction process by following an ontology authoring methodology (the current implementation follows the methodology used at Ordnance Survey for developing several large ontologies with the active involvement of domain experts [20]); (c) providing an intuitive interface to enter knowledge constructs in Rabbit. We describe an experimental study that examines the degree to which domain experts (i.e. not knowledge engineers) can build ontologies[2] following real scenarios based on work at Ordnance Survey.

An analysis of related work (§2) positions ROO in the relevant SW research. §3 presents the ROO tool and gives illustrative examples of user interaction taken from an experimental study reported in §4. §5 discusses the findings of the study, and outline implications for SW research.

2 Related Work

Recent developments of ontology authoring tools are increasingly recognising the need to cater for users without knowledge engineering skills. Controlled language (CL) interfaces have been provided for entering knowledge constructs in an intuitive way close to Natural Language (NL) interface (see [11,23] for recent reviews). ROO builds on the strengths and minimises the usability limitations of existing CL tools. Positive usability aspects have been followed in the design of ROO, such as: *look*

[1] Named after Rabbit in *Winnie the Pooh*, who is actually cleverer than Owl.

[2] Our expectation is not that domain experts will be able to completely author large complex ontologies without assistance (although this might be for small ontologies), but to establish that they can actively participate in the authoring process and construct significant portions of the ontology themselves. This means that domain experts can capture much of the ontology in a form that can be manipulated by knowledge engineers, who can in turn concentrate on the "hard modelling".

ahead to provide suggestions by guessing what constructs the users might enter [24]; showing the parsed structure to help the user recognise correct sentence patterns ([10,21,26]); providing a flexible way to parse English sentences using robust language technologies [8,11,24]; automatically translating to OWL ([17,4,11]); using templates to facilitate the knowledge entering process [22,24]; maintaining a text-based glossary describing parsed concepts and relationships [26]; and distributing the CL tool as a Protégé plug-in [10]. At the same time, we have tried to minimise the negative usability issues exhibited in existing CL tools, such as reliance on the user having knowledge engineering skills to perform ontology authoring (all existing tools suffer from this to an extent) and lack of immediate feedback and meaningful error messages [10,11,26].

Although the goal of CL tools is to assist in entering knowledge constructs, the existing tools focus solely on the CL aspect - they *do not aim to provide assistance for the whole ontology construction process*. In this vein, the HALO project[3] makes an important contribution by offering holistic and intuitive support at all stages of ontology authoring [2]. This key design principle is also followed in ROO. HALO focuses on providing advanced functionality based on the state-of-the-art SW technologies, e.g. sophisticated NL parsing of source documents, graphical interface for entering ontology constructs and rule-based queries. In contrast, ROO offers simpler functionality and follows the Ordnance Survey's practice in ontology construction when taking design decisions. For example, we do not use information extraction techniques to pull out domain concepts from documents, as domain experts normally know what the key concepts are. Our experience shows that the major challenge is to perform abstraction and to a lesser degree reformulation (from NL to CL) and to formulate ontology constructs in a CL, which is the main focus in ROO. It provides *intelligent support* for ontology definition by offering *proactive guidance* based on monitoring domain experts' activities when performing ontology construction steps. Essentially, certain knowledge engineering expertise has been *embedded into* ROO to compensate for the lack of such skills in domain experts. This ensures rigour and effectiveness of the ontology development process, and can lead to better quality ontologies (ontology "quality" is described further in §4.3). Furthermore, ROO aims to improve users' understanding of the knowledge engineering process, and to gradually develop their ontology modelling skills. The study presented in this paper is an initial examination of some of these assumptions.

3 The ROO Tool

The design of ROO takes into account factors that may hinder the involvement of domain experts in the ontology authoring process. As identified through Ordnance Survey's experience in ontology construction, they are: the need to follow a systematic methodology for capturing the knowledge of domain experts; the difficulty in expressing knowledge constructs in a formal language; and the need to cater for the lack of knowledge engineering skills in domain experts.

ROO follows the main steps in Kanga, the Ordnance Survey's methodology for involving domain experts in the authoring of conceptual ontologies [20]. It includes

[3] www.projecthalo.com

the following steps: (a) identify the scope, purpose and other requirements of the ontology; (b) gather sources of knowledge (e.g. documents and external ontologies); (c) define lists of concepts, relationship and instances supplied with NL descriptions; (d) formalise *core* concepts and their relations in structured English sentences; (e) generate the OWL ontology. Once step (a) is complete, steps (b)-(d) are performed iteratively by domain experts, while step (e) is performed by ROO automatically. Note that the focus in Kanga is to capture domain experts' knowledge and encode it in OWL, so it can be further examined, validated and improved by knowledge engineers who can use ROO in combination with other ontology engineering tools, for example querying tools [3,18,28,31].

The formalisation, step (d), uses a controlled natural language, called Rabbit, developed in response to a need for domain experts to be able to understand and author ontologies [14]. Rabbit covers every construct in OWL 1.1 [14,8], allowing domain experts to express sufficient detail to describe the domain.

ROO[4] is an open source tool distributed as a Protégé 4 plugin [6]. ROO extends the Protégé 4 user interface by simplifying it as much as possible[5] - hiding advanced options from the user and using what we believe to be less-confusing terminology (e.g. instead of 'classes and properties', ROO shows 'concepts' and 'relations').

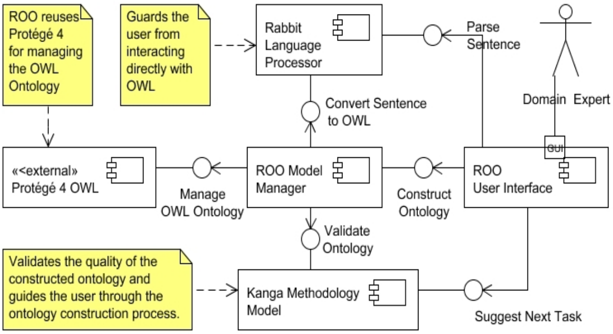

ROO reuses Protégé 4 for managing the OWL Ontology
Guards the user from interacting directly with OWL
Rabbit Language Processor
Parse Sentence
Domain Expert
Convert Sentence to OWL
«<external» Protégé 4 OWL
Manage OWL Ontology
ROO Model Manager
Construct Ontology
GUI
ROO User Interface
Validate Ontology
Validates the quality of the constructed ontology and guides the user through the ontology construction process.
Kanga Methodology Model
Suggest Next Task

Fig. 1. UML 2.0 component diagram shows the architectural elements, interfaces and inter-element connections in ROO

In order to explain the services provided by the Rabbit *Language Processor*, the Kanga *Methodology Model* and the ROO *Model Manager* (see Fig. 1), we show two typical user interactions with the system and explain how they are handled by ROO. The examples are taken from the experimental study described in §4.

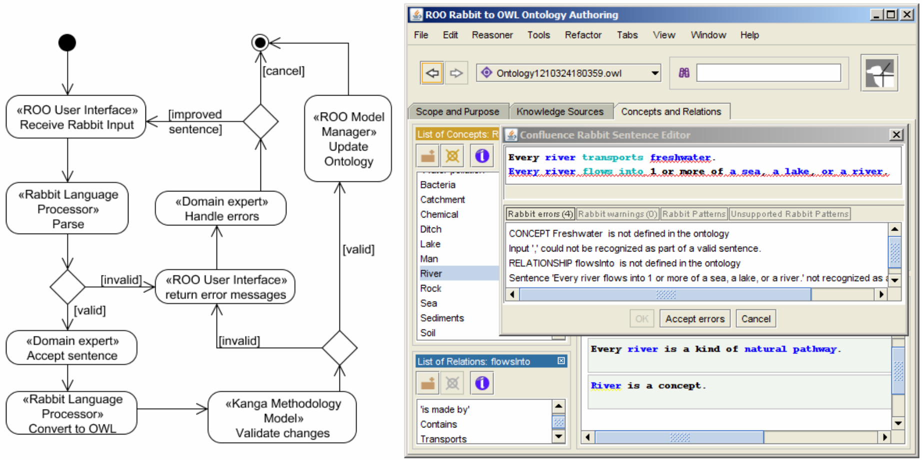
Domain experts edit the ontology using Rabbit sentences instead of directly editing OWL or the Manchester Syntax. Fig. 2 depicts how a domain expert enters sentences in ROO using the Rabbit *editor*. The user has entered two Rabbit sentences defining the concept `river`. The first one (`Every river transports freshwater`) is a valid Rabbit pattern but uses the concept `freshwater` which is not defined in the ontology. The Rabbit *Language Processor* recognises that `freshwater` is likely to be a domain concept and composes a corresponding error message. The user has typed the second sentence (`Every river flows into one or more of a sea, a lake, or a river`) while looking into the

[4] ROO is built as part of the Confluence project. http://sourceforge.net/projects/confluence

[5] The default Protégé 4 GUI components are still available for the more advance users, but are not used as a default in the ROO application.

existing Rabbit patterns (shown by clicking on the Rabbit *patterns* tab). However, the Rabbit pattern for *non-exclusive OR* is applied wrongly – instead of commas the user should have used `or`[6], and the sentence uses a relationship `flows into` which is not defined in the ontology. Corresponding error messages to help the user are generated, as shown in Fig. 2. The user then corrects the errors by adding the missing concept and relationship and correcting the Rabbit pattern. Every time the user makes changes, the input is re-parsed and, if necessary, error messages are generated accordingly. When the input does not contain errors, the user confirms the sentence. It is then translated into OWL by the Rabbit *Language Processor*, then validated by the Kanga *Methodology Manager*[7], and added to the ontology by the ROO *Model Manager*.

Domain experts can also ask a "guide dog" in ROO to *suggest tasks***,** which is a "wizard"-like feature which monitors the state of the ontology and the user's activities, and suggests the most appropriate actions. Fig. 3 shows how the system handles these requests. The user has already entered several concepts to the ontology. The user then asks for a next task. The Kanga *Methodology Model* then derives a list of possible tasks and sorts them according to the current ontology state and the user's recent activity. In Fig. 3, the user is prompted to enter Rabbit sentences for the concept `freshwater` which was created with the previous concept definition of `river` (see Figure 2) but did not yet give Rabbit definitions for it. Other task suggestions include reminding the user to enter missing natural language descriptions or pointing at other previously entered concepts which lack Rabbit definitions.


«ROO User Interface» Receive Rabbit Input
[improved sentence]
[cancel]
«ROO Model Manager» Update Ontology
«Rabbit Language Processor» Parse
«Domain expert» Handle errors
[valid]
[invalid]
«ROO User Interface» return error messages
[valid]
«Domain expert» Accept sentence
[invalid]
«Rabbit Language Processor» Convert to OWL
«Kanga Methodology Model» Validate changes
ROO Rabbit to OWL Ontology Authoring
File
Edit
Reasoner
Tools
Refactor
Tabs
View
Window
Help
Ontology1210324180359.owl
Scope and Purpose
Knowledge Sources
Concepts and Relations
List of Concepts: R
Bacteria
Catchment
Chemical
Ditch
Lake
Man
River
Rock
Sea
Sediments
Soil
List of Relations: flowsInto
'is made by'
Contains
Transports
Confluence Rabbit Sentence Editor
Rabbit errors (4)
Rabbit warnings (0)
Rabbit Patterns
Unsupported Rabbit Patterns
OK
Accept errors
Cancel

Every river transports freshwater.
Every river flows into 1 or more of a sea, a lake, or a river.

CONCEPT Freshwater is not defined in the ontology
Input ',' could not be recognized as part of a valid sentence.
RELATIONSHIP flowsInto is not defined in the ontology
Sentence 'Every river flows into 1 or more of a sea, a lake, or a river.' not recognized as a

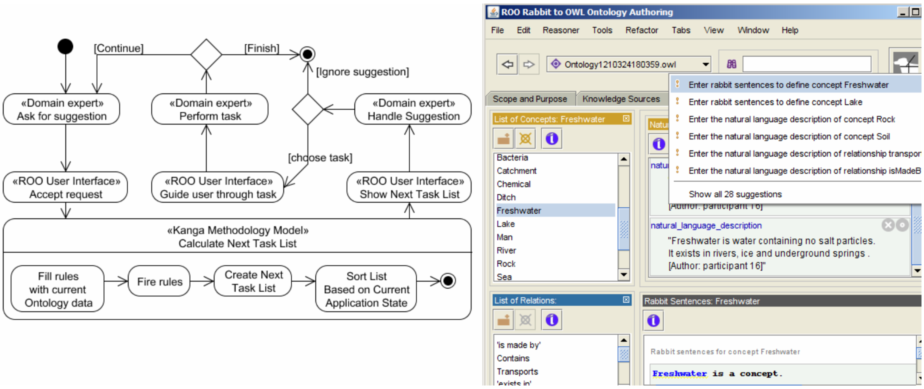
Every river is a kind of natural pathway.

River is a concept.

Fig. 2. State chart and screenshot showing how a Rabbit sentence is handled by ROO. The parsed syntax elements are highlighted, and possible errors/suggestions are reported to the user.

[6] The correct Rabbit pattern is: `Every river flows into 1 or more of a sea or a lake or a river.`

[7] This includes checks whether the input is appropriate to the current stage of the ontology construction; e.g. scope and purpose must be enterered before later stages can commence; Rabbit definitions require existence of NL descriptions.


[Continue]
[Finish]
[Ignore suggestion]
«Domain expert» Ask for suggestion
«Domain expert» Perform task
«Domain expert» Handle Suggestion
[choose task]
«ROO User Interface» Accept request
«ROO User Interface» Guide user through task
«ROO User Interface» Show Next Task List
«Kanga Methodology Model» Calculate Next Task List
Fill rules with current Ontology data
Fire rules
Create Next Task List
Sort List Based on Current Application State
ROO Rabbit to OWL Ontology Authoring
File
Edit
Reasoner
Tools
Refactor
Tabs
View
Window
Help
Ontology1210324180359.owl
Scope and Purpose
Knowledge Sources
List of Concepts: Freshwater
Bacteria
Catchment
Chemical
Ditch
Freshwater
Lake
Man
River
Rock
Sea
List of Relations:
'is made by'
Contains
Transports
'exists in'
Rabbit Sentences: Freshwater
natural_language_description

Enter rabbit sentences to define concept Freshwater
Enter rabbit sentences to define concept Lake
Enter the natural language description of concept Rock
Enter the natural language description of concept Soil
Enter the natural language description of relationship transpor
Enter the natural language description of relationship isMadeB
Show all 28 suggestions

"Freshwater is water containing no salt particles.
It exists in rivers, ice and underground springs .
[Author: participant 16]"

Rabbit sentences for concept Freshwater

Freshwater is a concept.

Fig. 3. State chart and screenshot depicting how ROO handles the suggestion of next tasks

The development of ROO has been guided by regular usability tests with potential users – domain experts in different domains. This has led to a fairly robust version that has been evaluated following real scenarios at Ordnance Survey. This evaluation is discussed in the rest of the paper.

4 Experimental Study

To assess the effectiveness of ROO, we conducted an experimental study following the criteria for evaluating ontology tools in [15]. The study addressed three groups of questions: (1) What is the interaction with the tool like? How usable is the tool? Can domain experts without knowledge engineering skills create OWL ontologies with ROO? (2) How well does ROO facilitate the ontology construction process? Do users develop ontology modelling skills as a result of the assistance the tool provides? (3) What is the quality of the resultant ontologies produced with ROO? Is the quality influenced by assistance provided by the tool?

4.1 Experimental Design

The study followed a *task-based*, *between-subjects* experimental methodology to compare ROO with a baseline system.

Baseline System. The study compares ROO with a similar tool that allows the user to author in a CL. From the available CL tools for ontology authoring, ACEView for Protégé [16] was chosen since the user interaction with it is the closest to the user interaction with ROO: both tools extend Protégé as plug-ins, support text input in a CL compatible with OWL-DL, provide error messages for sentence composition, and produce an ontology in OWL[8]. The main difference between ROO and ACEView is that ROO offers assistance with the whole ontology authoring process (§3).

[8] The other available CL ontology authoring tools are CLONE [11] and PENG [16]. They were used during a pilot but discarded for the actual study. CLONE is more suitable for users with some knowledge engineering skills, while the users in our study did not have such skills. The interaction with PENG is pattern-based and is notably different from the ROO interface.

Participants. The study involved 16 volunteers from the departments of Geography (8 students) and Earth and Environment (8 students) at the University of Leeds. The participants were chosen to closely resemble domain experts who may perform ontology modelling tasks at Ordnance Survey (Hydrology) or the Environment Agency for England and Wales (Flooding and Water Pollution). The main requirement for attending the study was to have knowledge and experience (confirmed with the modules attended and practical work done) in Hydrology, for Geography students, and Flooding and Water Pollution, for Environmental Studies students. In each domain, 4 participants used ACEView and 4 used ROO; this was assigned on a random basis. None of the participants was familiar with ontologies or ontology construction tools. They had not heard of RDF or OWL. None had previous background in encoding knowledge and for most participants "structuring knowledge" meant writing reports/essays in a structured way.

Scenarios. The study involved two ontology authoring scenarios.

Scenario 1 [Geography participants]: This scenario resembles ontology modelling tasks performed by domain experts at Ordnance Survey to describe geographical features whose spatial representations are included in Ordnance Survey's OS MasterMap®[9]. The participants were asked to describe several hydrology concepts: `River`, `River Stretch`, `River Bank`, `Ditch`, `Catch Drain`, `Balancing Pond`, `Canal` and `Reservoir`. These concepts are included in a large Hydrology ontology[10] defined by Ordnance Survey. The Geography participants were familiar with OS MasterMap®, which is used at the School of Geography at Leeds University.

Scenario 2 [Environmental Studies participants]: This scenario resembles ontology modelling tasks performed by domain experts at one of Ordnance Survey's customers –the Environment Agency of England and Wales– who can use OS MasterMap® for flooding and water pollution analysis. The participants were asked to describe: `River`, `Catchment`, `Flood Plain`, `Ditch`, `Water Pollution`, `Sediments`, `Colloids`, `Land Use` and `Diffuse Pollution`. These concepts were selected from a list derived by an Ordnance Survey researcher interviewing an expert from the Environment Agency as part of a project to scope a semantic data integration scenario. Many of these concepts required references to hydrology features from OS MasterMap® but the participants were unaware of this. None of the Environment subjects had knowledge of OS MasterMap®. Ontologies for geography and environment were also produced by Ordnance Survey and were used as comparators with the ontologies produced by the participants.

Procedure and Materials[11]**.** Depending on their background, the participants were sent the corresponding list of concepts, and were asked to prepare brief textual descriptions for these concepts by using specialised dictionaries or other sources. Each session was conducted individually and lasted 2 hours. It included several steps.

Pre-study questionnaire [20 min] included a brief introduction to the study and several questions to test the participants' ontology modelling background.

[9] OS MasterMap® www.ordnancesurvey.co.uk/osmastermap/ is a nationally contiguous vector map containing more than 450 million individual features down to street, address and individual building level, spatial data to approximately 10cm accuracy.

[10] www.ordnancesurvey.co.uk/ontology

[11] All materials are available from www.comp.leeds.ac.uk/confluence/study.html

Introduction to the scenario and training with the ontology authoring tool [10 min] was given to each participant by an experimenter, describing the main parts of the interface and entering of several definitions from a Building and Places[12] ontology. The examples used for the ACEView and ROO sessions were similar (the differences came from the CL and the errors given by each tool). The training with ROO also required entering the ontology's scope and purpose and knowledge sources.

Interaction with the tool [60 min] The participants had to use the tool allocated to them to describe the concepts following the descriptions they had prepared. Each session was monitored by an experimenter who provided some general help when the participants got stuck with the language. Help materials with printed examples of the corresponding CL were provided. The interactions were logged and video recorded. The experimenters kept notes of the user interaction.

Post-study questionnaire [20 min] included checking the participants' ontology modelling background (repeating questions from the pre-study questionnaire); a usability questionnaire using a seven-point Likert scale; and open questions about benefits, drawbacks, and future improvement of the tool used.

General impression and clarification [10 min] included a brief interview with each participant about their general impression of the CL used, interaction with the tool, and any additional aspects the participants wished to mention.

Data Collected. The following data was collected during the study: (a) *Questionnaires* – used for examining the usability of each tool and examining possible changes in the participants' understanding of ontology modelling; (b) *Log data, video records of the sessions, and experimenter's notes* – used for clarifying aspects of the interaction with each tool; (c) *Resultant OWL ontologies* – the quality of these ontologies was analysed following the O2 framework [12]. The data was analysed quantitatively and qualitatively. The quantitative analysis used Mann-Whitney U test[13] for discrete measurements and t-test for interval data.

4.2 Comparing the Interaction with ROO and ACEView

Interaction Patterns. Both tools have fairly simple interfaces and were easy to use. The first quarter of the interaction was usually slower as the participants had to learn to formulate sentences in the corresponding CL. During this time, the definition of the first concept `river` (common for both scenarios) was completed. Both tools offer a tab to show the *CL errors*, this was used extensively. Initially, most users did not realise that the error messages refer to incorrect CL grammar that the computer could not parse or translate into a logical form, rather than incorrect domain facts. From the second quarter, the users established a routine to describe a concept, including:

1. *Check the NL description for the currently entered concept and identify a statement with knowledge to be encoded.* The ACEView users had a printout of the descriptions they had prepared, while the ROO users followed the NL descriptions the tool prompted them to enter.

[12] www.ordnancesurvey.co.uk/ontology.

[13] Mann-Whitney U test is a powerful nonparametric test used as an alternative to the parametric t-test to compare two independent samples [27]. It is often used when the measurement is weaker than interval scaling or the parametric assumptions are not met.

2. *Look for a CL pattern that matches the NL statement.* The ACEView users used only the printed list of CL examples provided, ROO users could, in addition, see the available patterns within the tool, and they gradually moved to using this;
3. *(Re)Formulate the NL statement in a CL pattern.* This usually involved simplifying the constructs or taking away unnecessary detail, e.g. simple patterns were easily created, more complex patterns were normally not written correctly in the first instance and required several iterations and checking the system feedback.
4. *Check for error messages* – if there are no error messages, continue with another NL statement (i.e. go to step 1). When there are error messages, the users would usually repeat steps 2-4. Some participants would be persist, reformulating the CL statement until there were no errors (and it was translated to OWL), while others would continue and leave the CL statement with errors (i.e. not encoded in OWL).

For both tools, the users were occupied mostly with steps 3 and 4 and would often refer to step 2 for a quick check. Two of the eight ACEView users entered sentences to describe all concepts from the given list (see scenarios), while none of the ROO users managed to complete the descriptions; in most cases the last two concepts were not defined. Table 1 summarises the main interaction problems.

Usability. Table 2 summarises the findings from the usability questionnaire. For both tools, the users were positive. ROO was found to be significantly less frustrating than ACEView, which may be due to the much more intuitive interface, much less confusing error messages, and the help offered from the "guide dog". The messages in ROO were more helpful, the tool was less complex than ACEView, and users would be more willing to use ROO again (note the very low significance).

Ontology Modelling Skills. The answers to six ontology modelling questions (covering the main steps and building blocks in conceptual models, definition of ontology, concepts, and relations) in the pre- and post-study questionnaires were compared by

Table 1. Summary of the main interaction problems identified in the study

Problem	Tool	Explanation
Error messages lack detail.	ACEView ROO	When the CL pattern entered was not recognised, the users would not always get informative error messages. In such cases, the users had to guess what may be misleading, e.g. ACEView: The sentence is not correct ACE syntax. ROO: Sentence is not recognised as correct Rabbit sentence.
Error messages confusing.	ACEView ROO	When the user entered sentences which could not be recognised, they sometimes received error messages that were misleading. ACEView messages included ??? to indicate unrecognised parts in the sentence or referred to grammatical constructs which some users found hard to follow. ROO gave at times misleading suggestions when the sentence was unrecognised.
Dealing with adjectives and compound noun phrases	ACEView ROO	Recognising a concept which includes a compound noun phrase (e.g. adjective-noun) can be a challenging problem. ACEView users often received the message "adjectives are not supported", in which case they had to use hyphenation (see above problem). ROO parses for compound noun phrases and in most cases could make helpful suggestions about what the concept might be, e.g. natural waterway, man-made feature. However, when the compound nouns were not recognised and this led to confusing error messages, e.g. natural body of water was not recognised as a possible concept.
Dealing with a specialised vocabulary	ACEView ROO	The parsers in both tools could not recognise some specialised vocabulary which did not allow entering certain concepts, such as: ACEView: sediment, irritation; ROO: watershed. ACEView deals with this by pre-entering classes. However, it would be hard to predict in advance what phrases a user may enter. A more flexible way would be to allow the user to enter a phrase which should be added to the vocabulary used by the NL parser.
Next task suggestion not always useful	ROO	On several occasions, users ignored the task suggestions and commented that not all of them were useful. E.g. ROO suggested that the participant enter definitions of secondary concepts, such as man or bacteria The Kanga methodology discerns between core concepts and secondary concepts. Only core concepts need to be formalised. However, the current ROO tool does not discriminate between core and secondary concepts yet.

Table 2. Summary of the comparison of the usability of both tools (post-study questionnaire)

Question (1-Strongly disagree; 4-Neutral; 7-Strongly agree)	ROO median	ACEView median	U (Mann-Whitney, 1-tail)	p	Significance
The error messages helped me write CL sentences	5	4.5	16.5	p≤0.1	LOW:
The error messages were confusing	2	4.5	11.5	p≤0.025	YES
The guide dog was helpful	5	—	—	—	—
The guide dog suggestions were not easy to understand	2	—	—	—	—
I did not follow the suggestions from guide dog	4	—	—	—	—
The interaction was demanding	3	4	39	p>0.1	NO
I had no idea what I was doing	2	1.5	16	p>0.1	NO
It took me too long to compose what I wanted	4	3	21	p>0.1	NO
The interaction was intuitive	5	3.5	11.5	p≤0.025	YES
The feedback was prompt and timely	5	4.5	24	p>0.1	NO
It was clear to me what to do in this tool	5	4.5	24	p>0.1	NO
The tool was frustrating	3	5	5.5	p≤0.01	YES (HIGH)
The tool was unnecessary complex	2.5	3.5	18	p≤0.1	LOW
I'd like to use the tool again	5	4	18.5	p≤0.1	LOW

to examine whether the users' ontology modelling skills had changed as a result of the interaction with the tool. Two evaluators with a sound ontology background worked independently and marked the users' answers. The following scheme was applied to each question: -1 (the understanding has worsened, e.g. because the user was confused); 0 (no change to the user's understanding on the questions), +1 (correct aspects are added but gaps exist), +2 (the understanding is improved, and now is correct and complete). The marker compared their results and the discrepancies were clarified in a discussion. The maximum score, if a user had not had any ontology modelling knowledge and has become an expert, would have been 12, while the worst score meaning a user was an expert and became totally confused would have been -6.

The ROO users scored significantly higher than the ACE users - ACEView score mean 0.38, STDEV 2.97; ROO score mean 5, STDEV 2.78; U (Mann-Whitney)=8.5, p≤0.01. This shows that the users' understanding in ontology modelling improves significantly more when using ROO than when using ACEView.

4.3 Quality of the Resultant Ontologies

The resultant ontologies were analysed following the ontology evaluation framework in [12] considering structural, functional, and usability ontology measures.

Ontology Structural Measures. Since the size of the ontologies is limited, we have used fairly simple structural metrics based on [29], calculated by Protégé 4[14].

There are no significant differences in the structural characteristics of the ontologies created, with exception to annotations per entity, as shown in Table 3.

Table 3. Summary of ontology structural measures

	Average Class Count	Average Object Property Count	Average Properties Relative to number of Classes	Average Annotations per Entity	Average Subclass Axiom per Class (Inheritance Richness)
ROO	21.875	8.250	0.367	2.625	0.634
ACE	28.125	11.875	0.420	0.582	0.877
p (t-test)			0.263	0.000	0.095
U (Mann-Whitney)	19.5	21.5			
p (Mann-Whitney)	0.104	0.147			

[14] We also attempted deeper graph-based structural metrics with the Protégé 3 plugin OntoCAT [5] but it could not properly analyse the produced ontologies due to version compatibility.

The results show that ontologies built with ROO have a *significantly better readability* than ontologies built with ACEView. Both systems store the entered sentences as annotations in the ontology. Since both Rabbit and ACE are quite readable for humans, these annotations can be used to understand the meaning of the OWL entities. The main reason why ROO ontologies are more readable is that ROO encourages users to provide additionally natural language descriptions for both concepts and relationships. When Rabbit sentences are translated and new classes and properties are added to the ontology, an appropriate `rdf:comment` is added containing the Rabbit sentence, with an `rdf:label` containing the Rabbit concept name. In contrast, ACEView does not add annotations when classes or properties are added.

We measured inheritance richness based on OntoQA[29]. ACEView ontologies had higher inheritance richness (Table 3), i.e. the classes built with ACEView had more connections to other classes. However, the functional measures (see Table 4 below) indicate that ACEView ontologies were more tangled than ROO ontologies. Domain experts seemed slightly more productive using ACEView than using ROO but the Mann-Whitney U-test does not provide conclusive significance.

Ontology Functional Measures. A domain expert who is also a knowledge engineer[15] at Ordnance Survey produced two benchmark ontologies to quantify the fitness-for-purpose of the participants' ontologies. A scoring system was devised:

+1 point for each axiom produced by the participant ontology that exactly matched[16] an axiom from the benchmark ontology;

+1 point for each additional valid axiom, i.e. axioms that were considered to be valid even though an equivalent did not exist in the benchmark;

-1 point deducted for each axiom in the benchmark but absent the user's ontology;

-1 point deducted for any axiom containing a modelling error.

The participants did not define axioms for all the concepts they were given. Where this was the case, we did not count any metrics for that concept for that participant. We only scored against axioms belonging to the concepts in the concept list given to the participants. The total score for each ontology was therefore the sum of the points added or deducted.

Table 4. Summary of the scores from the functional analysis of the resultant ontologies

Scenario	ROO (mean)	ACEView (mean)	U (p)
Geography	1.25	-3.5	3.5 ($p>0.1$)
Environment	3.75	-5	0 ($p\leq0.025$)
Combined	2.5	-4.25	9 ($p\leq0.1$)

Subjectively, the ACEView ontologies appeared to be more complete, whereas the ROO ontologies appeared to be better structured and with fewer modelling errors.

The data for each set of ontologies was analysed statistically using the Mann Whitney U test (Table 4). At a 95% confidence level this indicates that there is no significant difference between the sets of data collected for the geography ontologies but that ROO out-performs ACEView with respect to the environmental ontologies and overall (geography and environment combined). The weakest participant by far was a

[15] We were lucky that such an expert existed, making it possible to examine in depth the functional dimensions of the ontology.

[16] Some interpretation was required owing to variances in terminology.

ROO geographer who despite only recording axioms for three concepts achieved a negative overall score, but this alone would not have accounted for the overall differences even given the small sample sizes.

ACEView users tended to describe more concepts and add more axioms (Table 4). This applied to both the "in scope" concepts and also those out of scope. Some of the latter group were secondary concepts necessary to define the core concepts – for example `water body` used to super class `river` and `reservoir`. But others were irrelevant clutter, such as Scotland, and it was not clear why they were added.

ACEView users did better than ROO in getting exact axiom matches with the benchmark ontologies (with a mean that was 1.5 matches higher per person). They also had a higher mean for providing additional axioms, with an average of three more per person. However, ACEView users did very much worse when it came to the number of errors they made, that is the number of axioms that were deemed to be incorrect, averaging 8 errors per person more that ROO users. Even taking into account that ACEView users enter more axioms proportionately they enter 0.4 errors per axiom, compared to 0.13 errors for ROO users. Erroneous axioms were not included in the other axiom counts. If included, it would show that ACEView users are even more prolific – it seems to be a case of quantity over quality. Table 5 summarises the modelling problems that occurred.

Ontology Usability. None of the ontologies as produced would have been usable without modification. This is unsurprising given the fact that the users were essentially untrained in the language and knowledge modelling techniques. No user produced an ontology that provided a complete description of the concepts, but again this is unsurprising given the experience levels and time available. In simple terms the ROO ontologies were less complete, containing fewer concepts and fewer

Table 5. Types of modelling problems found with the functional analysis of ontologies

Problem	Tool	Explanation
Multiple tangled inheritance	ACEView ROO (much less frequently)	This was a very common error in ACE ontologies. In the worst case `Drainage` had five separate immediate simple super classes: `Artificial Object`, `Depression`, `Drainage`, `Long Trench` and `Narrow Trench`. An error was scored for each extra entanglement so in the case above a score of 4 would have been recorded. The axioms would have been included in the overall total of axioms. Although also occurring in ROO ontologies, the rate and degree of multiple inheritance was much lower.
Definition of an instance instead of a class	ACEView ROO	There were a number of occasions where a class was recorded as an instance. ACEView example: in one ontology `Flood-Plain` is declared to be an individual of class `sediment-deposition`. In examining the ACE log file the first mention of `flood-plain` is the sentence: `Flood-plain borders a river.` There is no use of `every` in the sentence so ACE assumes `Flood-Plain` is an individual, and so records the assertion `Flood-plain` is an individual of the anonymous class "`borders some River`". The next correct sentence: `Flood-plain is a sediment-deposition` adds `Flood-plain` as an individual of the class `sediment-deposition`. ROO example: user entered `Flood Plain is a Land Area` rather than `Flood Plain` *`is a kind of`* `Land Area`.
Generation of 'random' individuals	ACEView	ACEView also appears to generate "random" individuals. For example the sentence: `Scotland contains a farm and contains a forest and contains a reservoir.` Generates three individuals. It is probable that what the user meant was that `Scotland` (also an individual) contains *some* farms, forests and reservoirs. What is even less clear is why the user felt it necessary to add this out of scope information at all.
Repeated Knowledge	ACEView ROO (much less frequently)	In a number of cases ACEView users tended to enter axioms that were similar to axioms already entered. An example is: `Every flood-plain experiences flooding` and `Every flood-plain experiences periodic-flooding.` Such repetitiveness also occurred in the ROO ontologies, but much less frequently.

axioms. However, the greater number of modelling errors in the ACEView ontologies, combined with the amount of unnecessary clutter in terms of out-of-scope concepts and axioms would indicate that it would take longer to get them to a usable state. ROO ontologies were certainly better annotated and this helped significantly in terms of evaluating the usability of ontologies for a certain purpose.

5 Discussion and Conclusions

To the best of our knowledge, the study presented here is the first attempt to evaluate how domain experts without knowledge engineering skills can use CL-based tools to complete ontology modelling tasks close to real scenarios (existing studies have either used people with knowledge engineering skills and simple tasks [11] or looked into recognising CL constructs [14]). The results enable us to address key questions concerning the authoring of ontologies where a domain expert takes a central role: Can we use CL to involve domain experts in ontology construction? To what degree can a tool support help the authoring process and substitute for a knowledge engineer? What further support is needed?

Involvement of Domain Experts. Accepting that the users who participated in our study had minimal training in the languages and the tools, it is fair to conclude from the resultant ontologies that domain experts alone, even with tool assistance, would be unable to author anything more than simple ontologies without some formal training. Nevertheless, almost a quarter of the participants entered axioms that matched roughly 50% of the axioms in the benchmark ontologies. This would indicate that with even a minimal amount of training these domain experts could become quite competent as authors. It is always likely that for complex ontologies knowledge engineering skills will be required. However, if the domain expert is able to author most of the ontology, they will be more easily able to engage with the knowledge engineer who can then express the more difficult aspects. Furthermore, the study indicated that if methodical, intelligent support for ontology authoring is *embedded* in the authoring tool, domain experts can gain an understanding of the ontology modelling process, that can gradually lead to the development of knowledge engineering skills.

The study confirmed that domain experts are able to start authoring relatively quickly and without the need to learn obscure terminology and esoteric languages such as OWL. In fact, it is unlikely that the study would have been possible if OWL had been used rather than Rabbit (or ACE) given the need to provide training in OWL. That no real training was provided to participants is, at the very least, indicative of the benefits to domain experts in using intuitive CL interfaces. We are confident that a central involvement by domain experts in the authoring process is not possible if the only way of expressing the ontology is in a logic-based language expressed using esoteric terms and symbols, without a lengthy process of turning the domain expert into a fully fledged knowledge engineer, something that few domain experts have the time or inclination to do.

Existing Tool Support. The various processes involved in authoring an ontology include: (a) identification of concepts and relationships (classes and properties); (b) development of an overall structure for the ontology; (c) capturing of axioms for each concept; (d) development of patterns to express certain model constructs; (e) optimisation and rationalisation; (f) testing and validation; (g) documentation. This list is not exhaustive, nor does it attempt to imply a priority of one process over another. ROO and ACEView currently provide degrees of support for (a), (c), (d) and (g). The study gives strong evidence that offering intuitive error messages, making users aware of the knowledge constructs they are creating, and offering methodical guidance can have a positive effect on the usability and efficacy of ontology construction tools. It also indicates that this additional functionality tailored to domain experts (as in ROO) can have impact on the quality of the resultant ontologies - domain experts make fewer errors, detect unwanted concepts and relationships, avoid repetition, and document the ontology more consistently and in more detail.

Required Tool Support. The interaction with both tools suggests that additional support should be provided. This may have implications for ontology authoring in general, including the newly emerging collaborative ontology editing environments [21] where support is even more critical. Patterns of modelling errors can be recognised and pointed out with the error message or the task guidance (the guide dog in ROO). For instance, definition of an instance instead of a class can be detected based on the CL pattern (e.g. `is a` vs `is a kind of` in Rabbit), as the error can turn the OWL ontology from OWL-DL to OWL Full; likely repetition or redundancy can be recognised by using synonyms (e.g. `is part of`, `consists of`, `contains`, `comprises`) and indicated in a 'warning' message; both multiple tangled inheritance and isolated classes can be detected with structural analysis and warnings generated or advice given. The study also indicated that flexible CL parsing should be provided, such as recognising similarity between NL and CL sentences (e.g. no need to ask the user to specify a determiner, as in ACEView, as this is not normally needed in a correct NL sentence; missing '`Every`' can be spotted easily and pointed out in a meaningful error message); recognising compound noun phrases and the underlying structure (e.g. the parsers can recognise that `natural body of water` may require two concepts linked with subsumption, so the user may be asked whether `natural body of water` is a kind of `body of water`); or enabling the users to add missing specialised terminology (e.g. `sediment`) that can then be considered by the parser in future sentences.

Although there is evidence that the guidance offered in ROO is beneficial, it has to be improved further. For instance, the suggestions should take into account the current task better to avoid distracting and confusing the user (e.g. a task context could be retained in ROO and only activities/concepts relevant to that context would be suggested). The ontology status should be better monitored more closely and potential limitations pointed out (e.g. some of the structural metrics can indicate unpopulated parts of the ontology). Lastly, more proactive help should be offered (instead of waiting for the user to click on the guide dog, certain suggestions could be brought to the user's attention automatically). The study confirmed that systematic support based on an ontology methodology is beneficial. The current implementation of ROO can be considered as a proof of concept that a methodology can be embedded in the planning process. An interesting research question would be to define ontology construction methodologies explicitly, e.g. by using an ontology and rules. For instance, ROO could be easily adapted to work with methodologies which Kanga is similar to, e.g. Uschold and King's method [30] or METHONTOLOGY [13]. It would then be possible to choose the most appropriate methodology for the current ontology authoring task, or to compare the effect of different methodologies.

At the time of writing ROO implements only the core Rabbit constructs. We intend to complete all Rabbit constructs and implement some of the additional support outlined above. This will give us a much more robust and usable tool, that can then be the basis for a larger user study in real settings, facilitating further examination of the extent to which domain experts can be involved in ontology authoring.

Acknowledgements. The authors would like to thank Kaarel Kaljurand, for kindly providing us with the ACEView tool. Thanks go to Paula Engelbrecht for sharing her material on the Environment Agency and Ilaria Corda for helping with the initial design of ROO. Special thanks go to the participants in the experimental study.

References

1. Alani, H., Dupplaw, D., Sheridan, J., O'Hara, K., Darlington, J., Shadbolt, N., Tullo, C.: Unlocking the Potential of Public Sector Information with Semantic Web Technology. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825. Springer, Heidelberg (2007)
2. Angele, J., Moench, E., Oppermann, H., Staab, S., Wenke, D.: Ontology-Based Query and Answering in Chemistry: OntoNova @ Project Halo. In: Fensel, D., Sycara, K.P., Mylopoulos, J. (eds.) ISWC 2003. LNCS, vol. 2870. Springer, Heidelberg (2003)
3. Cimiano, P.: ORAKEL: A natural language interface to an F-logic knowledge base. In: Proceedings of Natural Language Processing and Information Systems, pp. 401–406 (2004)
4. Cregan, A., Meyer, T.: Sydney OWL Syntax - towards a Controlled Natural Language Syntax for OWL 1.1. In: Proceedings of OWLED 2007 (2007)
5. Cross, V., Pal, V.: An ontology analysis tool. International Journal of General Systems 37(1), 17–44 (2008)
6. Denaux, R., Holt, I., Dimitrova, V., Dolbear, C., Cohn, A.G.: Supporting the construction of conceptual ontologies with the ROO tool. In: OWLED 2008 (2008)
7. Dimitrov, D.A., Heflin, J., Qasem, A., Wang, N.: Information Integration Via an End-to-End Distributed Semantic Web System. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273, pp. 764–777. Springer, Heidelberg (2006)
8. Dolbear, C., Hart, G., Goodwin, J., Zhou, S., Kovacs, K.: The Rabbit language: description, syntax and conversion to OWL. Ordnance Survey Research Labs Techn. Rep. IRI-0004 (2007)
9. Dolbear, C., Hart, G.: Combining spatial and semantic queries into spatial databases. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L.M. (eds.) ISWC 2006. LNCS, vol. 4273. Springer, Heidelberg (2006)
10. Fuchs, N.E., Kaljurand, K., Schneider, G.: Attempto Controlled English meets the challenges of knowledge representation, reasoning, interoperability and user interfaces. In: Proceedings of FLAIRS 2006 (2006)
11. Funk, A., Tablan, V., Bontcheva, K., Cunningham, H., Davis, B., Handschuh, S.: CLOnE: Controlled Language for Ontology Editing. In: Aberer, K., Choi, K.-S., Noy, N., Allemang, D., Lee, K.-I., Nixon, L., Golbeck, J., Mika, P., Maynard, D., Mizoguchi, R., Schreiber, G., Cudré-Mauroux, P. (eds.) ASWC 2007 and ISWC 2007. LNCS, vol. 4825. Springer, Heidelberg (2007)

12. Gangemi, Catenacci, C., Ciaramita, M., Lehmann, J.: Modelling ontology evaluation and validation. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011. Springer, Heidelberg (2006)
13. Gomez-Perez, A., Fernandez-Lopez, M., Juristo, N.: Methontology: from ontological art toward ontological engineering. In: Proceedings of AAAI 1997 Spring Symposium Series on Ontological Engineering, pp. 33–40 (1997)
14. Hart, G., Johnson, M., Dolbear, C.: Rabbit: Developing a Control Natural Language for Authoring Ontologies. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) ESWC 2008. LNCS, vol. 5021, Springer, Heidelberg (2008)
15. Hartman, J., Spyns, P., Giboin, A., Maynard, D., Cuel, R., Suárez-Figueroa, M., Sure, Y.: Methods for ontology evaluation, Knowledge Web Deliverable, D1.2.3 (2004)
16. Kaljuran, K.: Attempto Controlled English as a Semantic Web Language. PhD thesis, Faculty of Mathematics and Computer Science, University of Tartu (2007)
17. Kaljurand, K., Fuchs, N.E.: Bidirectional mapping between OWL DL and Attempto Controlled English. In: Workshop on Principles and Practice of Semantic Web Reasoning (2006)
18. Kaufmann, E., Bernstein, A., Fischer, L.: NLP-Reduce: A "naive" but domain-independent natural language interface for querying ontologies. In: Franconi, E., Kifer, M., May, W. (eds.) ESWC 2007. LNCS, vol. 4519. Springer, Heidelberg (2007)
19. Klischewski, R.: Ontologies for e-document management in public administration. Business Process Management Journal 12, 34–47 (2006)
20. Kovacs, K., Dolbear, C., Hart, G., Goodwin, J., Mizen, H.: A Methodology for Building Conceptual Domain Ontologies. Ordnance Survey Research Labs Techn. Report IRI-0002 (2006)
21. Kuhn, T.: AceWiki: A Natural and Expressive Semantic Wiki. In: Proceedings of Workshop on Semantic Web User Interaction, held at CHI 2008 (2008)
22. Lopez, V., Sabou, M., Motta, E.: PowerMap: Mapping the Semantic Web on the Fly. In: Sure, Y., Domingue, J. (eds.) ESWC 2006. LNCS, vol. 4011. Springer, Heidelberg (2006)
23. Schwitter, R., Kaljurand, K., Cregan, A., Dolbear, C., Hart, G.: A comparison of three controlled natural languages for OWL 1.1. In: Proc. of OWLED 2008 workshop (2008)
24. Schwitter, R.: A. Ljungberg and D Hood, ECOLE - A Look-ahead Editor for a Controlled Language. In: Proc. of EAMT-CLAW 2003, pp. 141–150 (2003)
25. Schwitter, R.: English as a formal specification language. In: Hameurlain, A., Cicchetti, R., Traunmüller, R. (eds.) DEXA 2002. LNCS, vol. 2453. Springer, Heidelberg (2002)
26. Schwitter, R.: Representing Knowledge in Controlled Natural Language: A Case Study. In: Negoita, M.G.R., Howlett, J., Jain, L.C. (eds.) KES 2004. LNCS (LNAI), vol. 3213, pp. 711–717. Springer, Heidelberg (2004)
27. Siegel, S., Castellan, J.N.: Nonparametric Statistics for the Behavioral Scences, 2nd edn. McGraw-Hill, New York (1988)
28. Tablan, V., Damljanovic, D., Bontcheva, K.: A natural language query interface to structured information. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) ESWC 2008. LNCS, vol. 5021. Springer, Heidelberg (2008)
29. Tartir, S., Arpinar, I.B., Moore, M., Sheth, A.P., Aleman-Meza, B.: Ontoqa:Metric-based ontology quality analysis. In: Proc. of W. on Knowledge Acquisition from Distributed, Autonomous, Semantically Heterogeneous Data and Knowledge Sources (2006)
30. Uschold, U., King, M.: Towards and methodology for building ontologies. In: Workshop on Basic Ontological Issues in Knowledge Sharing, held at IJCAI 1995 (1995)
31. Wang, C., Xiong, M., Zhou, Q., Yu, Y.: PANTO: A Portable Natural Language Interface to Ontologies. In: Franconi, E., Kifer, M., May, W. (eds.) ESWC 2007. LNCS, vol. 4519. Springer, Heidelberg (2007)