CLOnE: Controlled Language for Ontology Editing

Adam Funk[1], Valentin Tablan[1], Kalina Bontcheva[1], Hamish Cunningham[1], Brian Davis[2], and Siegfried Handschuh[2]

[1] University of Sheffield, UK
[2] Digital Enterprise Research Institute, Galway, Ireland

Abstract. This paper presents a controlled language for ontology editing and a software implementation, based partly on standard NLP tools, for processing that language and manipulating an ontology. The input sentences are analysed deterministically and compositionally with respect to a given ontology, which the software consults in order to interpret the input's semantics; this allows the user to learn fewer syntactic structures since some of them can be used to refer to either classes or instances, for example. A repeated-measures, task-based evaluation has been carried out in comparison with a well-known ontology editor; our software received favourable results for basic tasks. The paper also discusses work in progress and future plans for developing this language and tool.

1 Introduction

Creating formal data is a high initial barrier for small organisations and individuals wishing to create ontologies and thus benefit from semantic knowledge technologies. Part of the solution comes from ontology authoring tools such as Protégé[1], but they often require specialist skills in ontology engineering. Therefore defining a controlled language (CL) for formal data description will enable naive users to develop ontologies using a subset of natural language. Building on both controlled language and machine-translation (MT) work, processing input in a controlled language may achieve the high levels of accuracy necessary to make the ontology authoring process more widely feasible. The growing interest in Semantic Web applications and need to translate information into a machine-readable format create many uses for such applications.

The *Controlled Language for Ontology Editing* (CLOnE) allows users to design, create, and manage information spaces without knowledge of complicated standards (such as XML, RDF and OWL) or ontology engineering tools. It is implemented as a simplified natural language processor that allows the specification of logical data for semantic knowledge technology purposes in normal language, but with high accuracy and reliability. The components are based on

[1] `http://protege.stanford.edu`

K. Aberer et al. (Eds.): ISWC/ASWC 2007, LNCS 4825, pp. 142–155, 2007.

GATE's existing tools for IE (information extraction) and NLP (natural language processing). [1] CLOnE is designed either to accept input as valid (in which case accuracy is generally 100%) or to reject it and warn the user of his errors; because the parsing process is deterministic, the usual IE performance measures (precision and recall) are not relevant.

The prototype was developed and reported in [2] but not evaluated. The CLOnE system has since matured and the main contribution of this work is a *repeated-measures task-based* evaluation in comparison to a standard ontology editor (Protégé). CLOnE has been evaluated favourably by test users and is being further developed and applied in current and future projects. The remainder of this paper is organized as follows: Section 2 briefly discusses existing controlled languages and applications related to the Semantic Web field, Section 3 discusses our design and implementation and Section 4 presents our evaluation and discusses our quantitative findings. Finally, Section 5 and Section 6 offer conclusions and future work.

2 Related Work

"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." [3] The concept was originally proposed in the 1930s by linguists and scholars who sought to establish a "minimal" variety of English, in order to make it accessible and usable internationally for as many people as possible (especially non-native speakers). [3] CLs were later developed specifically for computational treatment. Early CLs include Caterpillar Fundamental English (CFE) [4] and have subsequently evolved into many variations and flavors such as Smart's Plain English Program (PEP) [5], White's International Language for Serving and Maintenance (ILSAM) [5] and Simplified English.[2] They have also found favor in large multi-national corporations such as IBM, Rank, Xerox and Boeing, usually within the context of user-documentation production, machine translation and machine-aided translation. [3,5]

Using CLs for ontology authoring and instance population has already evolved into an active research area. Previous work includes the translation of a CL, PENG-D, into first-order logic (FOL), in order to target the CL to a knowledge representation language such as RDFS or OWL, based on the proposal of FOL as the "semantic underpinning" of the semantic web. [6,7]

A well known implementation of this approach (translating to FOL) is the use of the popular CL, *Attempto Controlled English*[3] (ACE) [8], as an ontology authoring language. It is a subset of standard English designed for knowledge representation and technical specifications, and constrained to be unambiguously machine-readable into discourse representation structures, a form of first-order logic. (It can also be translated into other formal languages.) ACE has

[2] `http://www.simplifiedenglish-aecma.org/Simplified_English.htm`
[3] `http://www.ifi.unizh.ch/attempto/`

been adopted as the controlled language for the EU FP6 Network of Excellence REWERSE[4] (Reasoning on the Web with Rules and Semantics). [9]. The Attempto Parsing Engine (APE) consists principally of a definite clause grammar, augmented with features and inheritance and written in Prolog. [10] This tool can be tested and demonstrated with a web browser through the *APE Webinterface*[5] and clients for the APE web service are also available.[6] REWERSE also proposes ACE OWL, a sublanguage of ACE, as a means of writing formal, simultaneously human- and machine-readable summaries of scientific papers. [11,12]

ACE itself however prohibits certain very natural constructions such as the use of *only* as an adverb. Since ACE OWL also aims to provide reversibility (translating OWL DL into ACE), OWL's *allValuesFrom* must be translated into a construction which can be rather difficult for humans to read. Futhermore, ACE OWL does not currently support enumerations (OWL's *oneOf*) and has limited support for datatype properties; it also imposes several further restrictions on ACE, such as the elimination of plural nouns. ACE itself stipulates a predefined lexicon but unknown words can be used if they are annotated with a POS (part of speech) tag; however this requires the user to be familiar with the lexicon. [13]

AquaLog[7] is an ontology-driven, portable question-answering (QA) system designed to provide a natural language query interface to semantic mark-up stored in knowledge base. Like CLOnE, it uses JAPE[8] grammars and other tools from the GATE framework [1]. AquaLog uses them to perform shallow parsing on the user's natural-language questions and translate them into a representation called Query-Triples (rather than discourse representation structures or first-order logic). [14] The Relation Similarity Service (RSS) module then maps the questions to ontology-compliant queries and structurally checks the generated query triples against the underlying ontology. If the RSS fails to discover matching relations or concepts within the KB, then it asks the user to disambiguate the relation or concept from a given set of candidates. This module also uses of string similarity metrics, lexical resources such as WordNet [15] and domain-dependent lexica in order to generate query-triples that comply with the underlying ontology. [16] However, existing linguistic rules pose difficulties with respect to complex queries (requiring extension of the NLP component) which the authors plan to remedy in future work.

Another CL implementation for knowledge engineering is the *Cypher*[9] software from Monrai Technologies which translates natural language input into RDF and SeRQL (Sesame RDF Query Language) according to grammars and

[4] `http://rewerse.net/`
[5] `http://www.ifi.unizh.ch/attempto/tools/`
[6] `http://www.ifi.unizh.ch/attempto/documentation/ape_webservice.html`
[7] `http://kmi.open.ac.uk/technologies/aqualog/`
[8] GATE provides the *JAPE* (Java Annotation Pattern Engine) language for matching regular expressions over annotations, adding additional annotations to matched spans, and manipulating the match patterns with Java code.
[9] `http://www.monrai.com/products/cypher/cypher_manual`

lexica defined by the user in XML. Because Cypher is recently developed and proprietary (but free of charge), no research papers exist to review.

Finally, *GINO* (Guided Input Natural Language Ontology Editor) provides a guided, controlled NLI (natural language interface) for domain-independent ontology editing for the Semantic Web. GINO incrementally parses the input not only to warn the user as soon as possible about errors but also to offer the user (through the GUI) suggested completions of words and sentences—similarly to the "code assist" feature of Eclipse[10] and other development environments. GINO translates the completed sentence into triples (for altering the ontology) or SPARQL[11] queries and passes them to the Jena Semantic Web framework. (The JENA Eyeball[12] model checker verifies the OWL for consistency.) Although the guided interface facilitates input, the sentences are quite verbose and do not allow for aggregation. [17]

3 Design and Implementation

Taking into consideration the strengths and weaknesses of other controlled language systems discussed above, we designed the CLOnE software and input language to offer the following advantages.

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.

CLOnE has been favourably evaluated by test users (see Section 4) as part of the SEKT project, and is being further developed for use in other projects (as indicated in Section 6).

Each valid sentence of the controlled language matches exactly one syntactic rule and consists of *keyphrases* (the fixed expressions in the language, which are indexed in a gazetteer, and punctuation marks) and *chunks* (which are similar to noun phrases and are used to name classes, instances, properties and values). A *quoted chunk*, a series of words enclosed in paired quotation marks (`"..."`), can contain reserved words that would otherwise be detected by the keyphrase

[10] `http://www.eclipse.org/`
[11] `http://www.w3.org/TR/rdf-sparql-query/`
[12] `http://jena.sourceforge.net/Eyeball/full.html`

gazetteer; tokens POS-tagged as determiners and prepositions are also used in some syntactic rules, but otherwise all text that does not match any keyphrases is expected to contribute to a chunk.

Procedurally, CLOnE's analysis consists of the GATE pipeline of processing resources (PRs) shown in Figure 1. This pipeline starts with a series of fairly standard GATE NLP tools which add linguistic annotations and annotation features to the document. These are followed by three PRs developed particularly for CLOnE: the gazetteer of keywords and phrases fixed in the controlled language and two JAPE transducers which identify quoted and unquoted chunks. (Names enclosed in pairs of single or double quotation marks can include reserved words, punctuation, prepositions and determiners, which are excluded from unquoted chunks in order to keep the syntax unambiguous.)

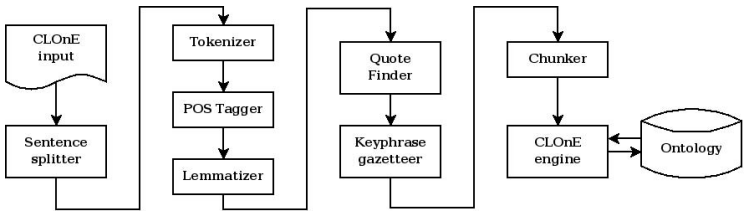

Fig. 1. The CLOnE pipeline

The last stage of analysis, the CLOnE JAPE transducer, refers to the existing ontology in several ways in order to interpret the input sentences. For example, one syntactic rule has the structure[13]

– (Forget that) CLASSES/INSTANCES are DESCRIPTION PREPOSITION CLASSES/INSTANCES.

which would match the following input sentences.

– `Persons are authors of documents.`
– `Carl Pollard and Ivan Sag are the authors of 'Head-Driven Phrase Structure Grammar'.`

This rule can take classes or instances as its arguments. The relevant Java code refers to the ontology and behaves differently according to each argument's status in the ontology (class, instance, or non-existent). For this rule,

– if the domain and range arguments both refer to classes, the code creates a new property between those classes (e.g. an *Author* property with the domain *Person* and range *Document*);

[13] In the rules, SMALL CAPITALS indicate chunks and chunklists (sequences of chunks separated by commas or the keyword `and`), except that PREPOSITION has the obvious meaning. Most of the rules have an optional *Forget that* prefix that corresponds to "undo" or "delete".

- if the domain and range arguments both refer to instances and a suitable property definition exists, the code defines a new property value between the instances (e.g. attributing to the instances *Carl_Pollard* and *Ivan_Sag* the property of authorship of the document *HSPG*); or
- in other cases (either argument does not exist, or one is a class and the other an instance), the code generates an error message.

Consequently, the same syntactic rule has different semantics according to the context. This means the user only has to learn one syntactic rule and can distinguish its two functions (semantic interpretations) intuitively, as in natural language but with high precision for the software.

The following list explains the other rules of the CLOnE language.

Rule	Example	Action
(Forget that) There is/are CLASSES.	`There are agents and documents.`	Create new classes (or delete classes).
(Forget that) INSTANCES is a/are CLASS.	`"University of Sheffield" is a university. Alice Jones and Bob Smith are persons.`	Create (or delete) instances of the class.
(Forget that) CLASSES is/ are a type/types of CLASS.	`Universities and persons are types of agent. Dogs are a type of mammal. Forget that dogs are a type of cat.`	Make subclasses of the last class in the sentence (create new ones if needed). (CLOnE supports multiple inheritance.) The negative form unlinks the the subclass-superclass relationship (but does not delete the subclass).
(Forget that) CLASSES/ INSTANCES have CLASSES/ INSTANCES.	`Journals have articles. "Journal of Knowledge Management" has "Crossing the Chasm".`	For lists of classes, create properties of the form *Domain_has_Range*. For lists of instances, find a suitable property and instantiate it; if there is a class-instance mismatch or a suitable property does not exist, generate an error.
(Forget that) CLASSES have DATATYPE DESCRIPTION.	`Projects have string names. Deliverables and conferences have dates as deadlines.`	Create datatype properties of the form *Domain_has_Description*.

Rule	Example	Action
(Forget that) INSTANCE has DESCRIPTION with value VALUE.	`SEKT has name with value "Semantically-Enabled Knowledge Technology".`	Find a matching datatype property for the instance and instantiate it with the given range value.
(Forget that) CLASS/ INSTANCE is/are also called/known as SYNONYMS.	`Dogs are also called canines. Bob Smith is also called Bob.`	Create synonyms (RDF labels) for the class or instance, which can be used in later CLOnE statements.
Forget everything.	`Forget everything.`	Clear the whole ontology (and start over).
Forget CLASSES/ INSTANCES.	`Forget projects, journals and "Department of Computer Science".`	Delete all the classes and instances listed.

4 Evaluation

4.1 Methodology

We prepared the following documents for the users, which the deliverable [18] includes in full detail.

- The pre-test questionnaire asks each subject how much he thinks he already knows about ontologies, the Semantic Web, Protégé and controlled languages. We scored this questionnaire by assigning each answer a value from 0 to 2 and scaling the total to obtain a score of 0–100.
- The short manual introduces ontologies and provides "quick start" instructions for both pieces of software. Although much simpler, our manual was partly inspired by Protégé's *Ontology 101* documentation. [19]
- The post-test questionnaire for each tool is the *System Usability Scale* (SUS), which also produces a score of 0–100. [20]
- We devised a comparative questionnaire to measure each user's preference for one of the two tools. This form is scored similarly to SUS so that 0 would indicate a total preference for Protégé, 100 would indicate a total preference for CLOnE, and 50 would result from marking all the questions *neutral*. On the reverse side and in discussion with the facilitator, we offered each user the opportunity to make comments and suggestions.
- We prepared two similar lists of ontology-editing tasks, each consisting of the following tasks:
  - creating two subclasses of existing classes,
  - creating two instances of different classes, and
  - either (A) creating a property between two classes and defining a property between two instances, or (B) defining two properties between two pairs of instances.

For example, here is task list A:

- Create a subclass *Periodical* of *Document*.
- Create a subclass *Journal* of *Periodical*.
- Create an instance *Crossing the Chasm* of class *Article*.
- Create an instance *Journal of Knowledge Management* of class *Journal*.
- Create a property that agents are publishers of documents.
- Define a property that Hamish Cunningham, Kalina Bontcheva and Yaoyong Li are authors of *Crossing the Chasm*.

We recruited 15 volunteers with varying experience levels and asked each subject to complete the pre-test questionnaire, to read the manual, and to carry out each of the two task lists with one of the two tools. (Approximately half the users carried out task list A with CLOnE and then task list B with Protégé; the others carried out A with Protégé and then B with CLOnE.)

We measured each user's time for each task list and in most cases (12 of 15) for each sublist. After each task list we asked the user to complete the SUS questionnaire for the specific tool used, and finally we asked him to complete the comparative questionnaire.

4.2 Background

Our methodology constitutes a *repeated-measures, task-based* evaluation: each subject carries out a similar list of tasks on both tools being compared.

We chose the SUS questionnaire as our principal measure of software usability because it is a *de facto* standard in this field. Although superficially it seems subjective and its creator called it "quick and dirty", it was developed properly as a Likert scale. [20] Furthermore, researchers at Fidelity Investments carried out a comparative study of SUS and three other published usability questionnaires as well as an internal questionnaire used at that firm, over a population of 123 subjects, to determine the sample sizes required to obtain consistent, accurate results. They found that SUS produced the most reliable results across all sample sizes; they noted a jump in accuracy to 75% at a sample size of 8, but recommended a sample of at least 12–14 subjects. [21]

4.3 Quantitative Findings

Table 2 summarizes the main measures obtained from our evaluation.[14] In particular, CLOnE's usability scores are generally above the baseline of 65–70% [22] and are generally distributed higher than Protégé's, and scores on the comparative questionnaire are generally favourable to CLOnE. Table 3 shows confidence intervals for the SUS scores by tool and task list (A, B or combined); these indicate that for each task list and for the combined results, the larger population which our sample represents will also produce mean SUS scores for CLOnE that are both higher than those for Protégé and above the SUS baseline.[15]

[14] The deliverable [18] provides full details of the measurements discussed here.

[15] A data sample's *95% confidence interval* is a range 95% likely to contain the mean of the whole population that the sample represents. [23]

Table 2. Summary of the questionnaire scores

Measure	min	mean	median	max
Pre-test scores	25	55	58	83
CLIE SUS rating	65	78	78	93
Protégé SUS rating	20	47	48	78
CLIE/Protégé preference	43	72	70	93

Table 3. Confidence intervals (95%) for the SUS scores

Tool	Confidence intervals		
	Task list A	Task list B	Combined
Protégé	33–65	30–58	37–56
CLIE	75–93	67–79	73–84

We also generated Pearson's and Spearman's correlations coefficients[16] for a wide range of data from the experiments; Table 4 shows the highlights of these calculations. In particular, we note the following points.

- The pre-test score has no correlation with task times or SUS results.
- The task times for both tools are moderately correlated with each other but there are no significant correlations between task times and SUS scores, so both tools are technically suitable for carrying out both task lists.
- As expected, the C/P preference score correlates moderately strongly with the CLOnE SUS score and moderately negatively with the Protégé SUS score. (The SUS scores for the two tools also show a weak negative correlation with each other.) These figures confirm the coherence of the questionnaires as a whole.

4.4 Sample Quality

Although our sample size ($n = 15$) satisifies the requirements for reliable SUS evaluations (as discussed in Section 4.2), it is also worthwhile to establish the consistency of the two partitions of our sample, as enumerated in Table 5:

[16] A *correlation coefficient* over a set of pairs of numbers is analogous to a scatter plot: +1 signifies a perfect correlation and corresponds to a graph in which all points lie on a straight line with a positive slope; −1 signifies a perfect inverse correlation (the points lie on a straight line with a negative slope); 0 indicates a complete lack of correlation (random distribution of the points). Values above +0.7 and below −0.7 are generally considered to indicate strong correlations.

Pearson's formula assumes that the two variables are linearly meaningful (e.g. physical measurements such as temperature), whereas Spearman's formula stipulates only ordinal significance (ranking) and is often considered more suitable for subjective measurements (such as many in the social sciences). [23,24]

Table 4. Correlation coefficients

Measure	Measure	Pearson's	Spearman's	Correlation
Pre-test	CLIE time	-0.06	-0.15	none
Pre-test	Protégé time	-0.13	-0.27	none
Pre-test	CLIE SUS	-0.17	-0.17	none
Pre-test	Protégé SUS	-0.16	-0.15	none
CLIE time	Protégé time	0.78	0.51	+
CLIE time	CLIE SUS	0.26	0.15	none
Protégé time	Protégé SUS	-0.17	-0.24	none
CLIE time	Protégé SUS	0.19	-0.01	none
Protégé time	CLIE SUS	0.42	0.44	weak +
CLIE SUS	Protégé SUS	-0.31	-0.20	none
CLIE SUS	C/P Preference	0.68	0.63	+
Protégé SUS	C/P Preference	-0.62	-0.63	−

by tool order and task-tool assignment: subjects who carried out task list A on CLOnE and then B on Protégé, in comparison with those who carried out A on Protégé then B on CLOnE; and

by sample source: subjects drawn from the GATE development team, in comparison with others.

Tool order was divided almost evenly among our sample. Although the SUS scores differed slightly according to tool order (as indicated in Table 3), the similar task times suggest that task lists A and B required similar effort. We note that the SUS scores for each tool tend to be slightly lower for task list B than for A, and we suspect this may have resulted from the subjects' waning interest as the evaluation progressed. (To eliminate the possibility of this effect completely with the same reliability, we would need twice as many subjects, each carrying out one task list with one tool—a *between-subjects* experiment, in contrast to our *within-subject* experiment.) But because Table 3 in particular shows consistent results between CLOnE and Protégé for each task list, we conclude that our study was fair.

We must also consider the possibility of biased subjects drawn from colleagues of the developers and facilitator. As Table 5 shows, colleagues in the GATE team constituted 60% of the user sample. The measures summarized in Table 6, however, show in particular that although members of group G generally rated their own expertise higher (in the pre-test questionnaire) than those in group NG, both groups produced very similar ranges of SUS scores for each tool and of C/P preferences scores. These measures allay concerns about biased subjects: we conclude that groups G and NG were equally reliable so the sample as a whole is satisfactory for the SUS evaluation (according to the research [21] explained in Section 4.2).

4.5 Subjects' Suggestions and Comments

The test users made several suggestions about the controlled language and the user interface.

Table 5. Groups of subjects by source and tool order

Source		Tool order		Total
		PC	CP	
G	GATE team	4	5	9
NG	others	4	2	6
Total		8	7	15

Table 6. Comparison of the two sources of subjects

Measure	Group	min	mean	median	max
Pre-test	G	25	62	67	83
	NG	33	44	42	58
CLIE SUS	G	65	78	78	90
	NG	65	78	78	93
Protégé SUS	G	25	46	48	70
	NG	20	47	46	78
C/P Preference	G	50	71	68	90
	NG	43	74	79	93

- Several subjects complained that they needed to spell and type correctly the exact names of the classes and instances, such as "Journal of Knowledge Management" and that CLOnE is intolerant of typos and spelling mistakes. They suggested spell-checking and hinting (as provided in the Eclipse UI) to alleviate this cognitive load.
- A related suggestion is to highlight the input text with different colours for classes, instances and properties, perhaps after the user clicks a *Check* button, which would be provided in addition to the *Run* button. The *Check* button could also suggest corrections and give error messages without affecting the ontology.
- Users complained that it was difficult to tell why some input sentences failed to have any effect, because CLOnE does not explicitly indicate unparsable sentences. (This has already been corrected in subsequent work.)
- Some suggested that CLOnE should automatically clear the input text box after each run, but they also realized this would make it more difficult to correct errors, because it is currently easy to prefix the keyword `forget` to incorrect sentences from the previous input without having to retype them.
- Some suggested making the *Run* button easier to find and putting the ontology viewer and the input box on the screen at the same time instead of in alternative panes.
- A few users said it would be useful to have an *Undo* button that would simply reverse the last input text, instead of having to `forget` all the sentences individually.

5 Conclusion

Our user evaluation consistently indicated that our subjects found CLOnE significantly more usable and preferable than Protégé for the straightforward tasks that we assigned. Of course we make no claims about the more complicated knowledge engineering work for which Protégé but not CLOnE is designed and intended. The users considered CLOnE picky about spelling and typing, but they found the language basically easy to learn and use.

Our subjects made several interesting and useful suggestions for improvements to CLOnE, many of which we already envisage developing in future work on this software beyond SEKT, as Section 6 will indicate. In particular, we will embed CLOnE in wikis and perhaps other user interfaces which will eliminate some of the constraints imposed by running it in the GATE GUI, which is really intended for developing language engineering applications (such as CLOnE itself) rather than for interactively editing language resources (such as documents and ontologies). The use of CLOnE in NEPOMUK will especially address these issues.

6 Continuing and Future Work

We have discussed using CLOnE to generate ontologies from input text. The reverse of the process involves the generation of CLOnE from an existing ontology by shallow NLG (natural language generation). A prototype component for generating CLOnE has already been implemented as a GATE resource. The textual output of the generator is configured using a an XML file, which contains text templates that are instantiated and filled in with ontological values. The NL generator and the authoring process both combine to form a Round-Trip Ontology Authoring environment: one can start with an existing or empty ontology, produce CLOnE using the NL generator, modify or edit the text as requirement and subsequently parse the text back into the ontology using the CLOnE environment. The process can be repeated as necessary until the required result is obtained. Current developments in relation to the NL generator involve extending modifying the XML templates with respect to the generator's linguistic output so that it complies with the grammar and subsequent extensions such as alternatives for expressing the same message. This is essential in order to ensure that the generation component does not interfere with ontological data created or modified by CLOnE. This work is currently being carried out in collaboration with DERI Galway[17] on the integration of CLOnE as a web service for Round Trip Ontology Authoring as a part the Nepomuk[18] Solution (The Social Semantic Desktop). We are also investigating the provision of CLOnE as a local service within one of the promised semantic wikis in addition to the possible use of CLOnE for Semi-Automatic Semantic Annotation. Shallow, template-based

[17] `http://www.deri.ie/`
[18] `http://nepomuk.semanticdesktop.org/xwiki/`

NLG of CLOnE will also be exploited in KnowledgeWeb[19] for the verbalization suggestions for semi-automatic ontology integration.

In the TAO project, CLOnE is currently being extended and embedded in applications to allow controlled-language querying of knowledge bases. This will allow non-expert users (who are not familiar with SeRQL and other sophisticated semantic search tools) to ask questions in an extended version of CLOnE; they are translated by the application into SeRQL or SPARQL and evaluated by OWLIM.

We are also assisting the EPSRC-funded Easy project to use CLOnE to provide a controlled natural language interface for editing IT authorization policies (access to network resources such as directories and printers) stored as ontologies. This project will probably involve specialized extensions to the controlled language as well as a *virtuous circle* or round-trip information flow, to be completed by generation of CLOnE, similar to the work in NEPOMUK. [25]

Acknowledgements

This research has been partially supported by the following grants: KnowledgeWeb (EU Network of Excellence IST-2004-507482), TAO (EU FP6 project IST-2004-026460), SEKT (EU FP6 project IST-IP-2003-506826, Líon (Science Foundation Ireland project SFI/02/CE1/1131) and NEPOMUK (EU project FP6-027705).

References

1. Cunningham, H., Maynard, D., Bontcheva, K., Tablan, V.: GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. In: ACL 2002. Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (2002)
2. Tablan, V., Polajnar, T., Cunningham, H., Bontcheva, K.: User-friendly ontology authoring using a controlled language. In: 5th Language Resources and Evaluation Conference (2006)
3. Schwitter, R.: Controlled natural languages. Technical report, Centre for Language Technology, Macquarie University (2007)
4. Caterpillar Corporation: Dictionary for Caterpillar Fundamental English. Caterpillar Corporation (1974)
5. Adriaens, G., Schreurs, D.: From COGRAM to ALCOGRAM: Toward a controlled English grammar checker. In: COLING 1992. Conference on Computational Linguistics, Nantes, France, pp. 595–601 (1992)
6. Schwitter, R., Tilbrook, M.: Controlled natural language meets the semantic web (2004)
7. Horrocks, I., Patel-Schneider, P.: Three theses of representation in the semantic web (2003)
8. Fuchs, N., Schwitter, R.: Attempto Controlled English (ACE). In: CLAW 1996. Proceedings of the First International Workshop on Controlled Language Applications, Leuven, Belgium (1996)

[19] `http://knowledgeweb.semanticweb.org/`

9. Fuchs, N.E., Kaljurand, K., Kuhn, T., Schneider, G., Royer, L., Schröder, M.: Attempto Controlled English and the semantic web. Deliverable I2D7, REWERSE Project (2006)
10. Hoefler, S.: The syntax of Attempto Controlled English: An abstract grammar for ACE 4.0. Technical Report ifi-2004.03, Department of Informatics, University of Zurich (2004)
11. Kaljurand, K., Fuchs, N.E.: Bidirectional mapping between OWL DL and Attempto Controlled English. In: Fourth Workshop on Principles and Practice of Semantic Web Reasoning, Budva, Montenegro (2006)
12. Kuhn, T.: Attempto Controlled English as ontology language. In: Bry, F., Schwertel, U. (eds.) REWERSE Annual Meeting 2006 (2006)
13. Kaljurand, K.: Writing owl ontologies in ace. Technical report, University of Zurich (2006)
14. Bernstein, A., Kaufmann, E., Göring, A., Kiefer, C.: Querying Ontologies: A Controlled English Interface for End-users. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 112–126. Springer, Heidelberg (2005)
15. Fellbaum, C. (ed.): WordNet - An Electronic Lexical Database. MIT Press, Cambridge (1998)
16. Lopez, V., Motta, E.: Ontology driven question answering in AquaLog. In: Meziane, F., Métais, E. (eds.) NLDB 2004. LNCS, vol. 3136, Springer, Heidelberg (2004)
17. Bernstein, A., Kaufmann, E.: GINO—a guided input natural language ontology editor. In: Cruz, I., Decker, S., Allemang, D., Preist, C., Schwabe, D., Mika, P., Uschold, M., Aroyo, L. (eds.) ISWC 2006. LNCS, vol. 4273, Springer, Heidelberg (2006)
18. Funk, A., Davis, B., Tablan, V., Bontcheva, K., Cunningham, H.: Controlled language IE components version 2. Deliverable D2.2.2, SEKT (2006)
19. Noy, N.F., McGuinness, D.L.: Ontology development 101: A guide to creating your first ontology. Technical Report KSL-01-05, Stanford Knowledge Systems Laboratory (2001)
20. Brooke, J.: SUS: a "quick and dirty" usability scale. In: Jordan, P., Thomas, B., Weerdmeester, B., McClelland, A. (eds.) Usability Evaluation in Industry, Taylor and Francis, London (1996)
21. Tullis, T.S., Stetson, J.N.: A comparison of questionnaires for assessing website usability. In: Usability Professionals' Association Conference, Minneapolis, Minnesota (2004)
22. Bailey, B.: Getting the complete picture with usability testing. Usability updates newsletter, U.S. Department of Health and Human Services (2006)
23. John, L., Phillips, J.: How to Think about Statistics. W. H. Freeman and Company, New York (1996)
24. Connolly, T.G., Sluckin, W.: An Introduction to Statistics for the Social Sciences, 3rd edn. Macmillan, NYC (1971)
25. Chadwick, D., Sasse, A.: The virtuous circle of expressing authorization policies. In: Semantic Web Policy Workshop, Athens, Georgia (2006)