Lang Resources & Evaluation (2017) 51:191–220
DOI 10.1007/s10579-016-9351-x

ORIGINAL PAPER

# CNLs for the semantic web: a state of the art

**Hazem Safwat**[1] · **Brian Davis**[1]

Published online: 1 April 2016


**Abstract** One of the core challenges for building the semantic web is the creation of ontologies, a process known as ontology authoring. Controlled natural languages (CNLs) propose different frameworks for interfacing and creating ontologies in semantic web systems using restricted natural language. However, in order to engage non-expert users with no background in knowledge engineering, these language interfacing must be reliable, easy to understand and accepted by users. This paper includes the state-of-the-art for CNLs in terms of ontology authoring and the semantic web. In addition, it includes a detailed analysis of user evaluations with respect to each CNL and offers analytic conclusions with respect to the field.

**Keywords** Controlled natural languages · Ontology authoring · Semantic web, state of the art

## 1 Introduction

The semantic web endeavours to extend the current web, by enriching information with well defined meaning, which is machine processable (Berners-Lee et al. 2001). This process is heavily dependent on the existence of ontologies, which describe the domain of interest. Formal data representation can be a significant deterrent for non-expert users or small organisations seeking to create ontologies and subsequently

This Journal article is an extended version of the paper in Safwat and Davis (2014).

✉ Hazem Safwat
hazem.abdelaal@insight-centre.org

Brian Davis
brian.davis@insight-centre.org

[1] Insight Centre for Data Analytics, National University of Ireland, Galway, Ireland

benefit from adopting semantic technologies. This challenges researchers to develop user-friendly means for ontology authoring. Controlled natural languages (CNLs) for knowledge creation and management offer an attractive alternative for non-expert users wishing to develop small to medium sized ontologies. Controlled natural languages are defined as "subsets of natural language whose grammars and dictionaries have been restricted in order to reduce or eliminate both ambiguity and complexity" (Schwitter and Tilbrook 2004). The goal of this article is to describe the state-of-the-art of CNLs solely within the semantic web context. For a broader review of the CNLs literature in general, we refer the reader to Kuhn (2014). In the remainder of this paper, Sect. 2, presents a background about CNLs in the semantic web domain, Sect. 3, provides a deep and fine grained analysis of CNLs for the semantic web within the context of ontology authoring listed in chronological order of publication. In Sect. 4, tools that use CNLs to perform ontology engineering and querying tasks in the semantic web domain are presented. Section 5, provides a detailed comparison and analysis of user evaluations for the CNLs. Section 6, shows the evaluation comparison of the tools, and finally Sect. 7 offers analytic conclusions and observations with respect to current and future trends with respect to CNLs.

## 2 Historical background of CNLs for knowledge representation

The original concept of CNL arose during the 1930s, when a number of influential linguists and scholars devoted considerable effort to establishing a minimal variety of English. The purpose was to make English accessible and usable by as many individuals as possible worldwide (Schwitter 2007). CNLs have found particular favor in large multinational corporations such as IBM, Rank, Xerox and Boeing usually within the context of user documentation production and machine translation (Schwitter 2007; Adriaens and Schreors 1992; O'Brien 2003). Traditionally, controlled natural languages (CNLs) are split into two major categories: (1) CNLs that improve human readability, mainly for non-native speakers, and (2) those that constrain the text for computational treatment. In this survey, we are mainly interested in the second category, where CNLs are used to develop user friendly means for ontology authoring, so that end-users can represent formal data within the context of the semantic web, without the need of formal training. Since, the knowledge representation language of the semantic web is the Web Ontology Language (OWL) which is based on fragments of the first order logic (Smith et al. 2004). Users with no formal background will find it difficult to interact with the ontology engineering process without a special training. This formal barrier may restrict the adoption if not ultimately result in rejection of the semantic technologies (Smart 2008). As humans are already familiar with natural languages, it will be required to develop natural language interfaces to facilitate the interaction of the end users with the semantic web resources while minimizing the need for training. However, it will be very difficult to interpret all the natural language expressions due to the ambiguity of the natural language, so using a controlled or restricted variants of natural language rather than the full or unrestricted version is a possible solution. The CNL community proposed this kind of interfaces to solve this issue. For instance, the

mathematical symbols used in logic to represent ontologies is very difficult to be understood by non-logicians, so another alternative syntax called Manchester OWL syntax (MOS) (Matthew et al. 2006) was proposed to replace the formal logic symbols with keywords such as some, only and not. The syntax is currently being used in tools such as *Protégé-OWL* (Holger et al. 2004). With respect to knowledge representation via CNLs, early work included Processable English (PENG). The PENG statements can be translated into Discourse Representations Structure (DRS), and also into first-order predicate logic (Schwitter 2002). Furthermore, the work is influential in that it was one of the first CNLs, designed with an incremental parsing approach. It has an editor with look-ahead features in order to guide the user. In regard to CNLs for the semantic web, early efforts involved extending PENG in Schwitter and Tilbrook (2004), whereby the authors present and discuss PENG-D, a variation of PENG, which targets the CNL to a knowledge representation language [via first order logic (FOL)] such as RDFS or OWL.

## 3 Semantic web related CNLs

*PENG* (Schwitter 2002) is a CNL designed to allow writing unambiguous and precise specifications using a restricted grammar and a domain specific lexicon, which consists of predefined function words, illegal words, and user-defined content words. The content words can be added or modified using the integrated lexical editor. PENG text is easy to be translated into first order logic (FOL) using discourse representation structures (DRS). Also, it uses a sophisticated look-ahead editor to facilitate writing by non linguists without the need to know the grammar rules of the CNL explicitly. This can be achieved by showing what kind of syntactic structures can be used after each word entered by the user. The restricted grammar defines the structure of joining PENG sentences into complex sentences using coordinators and subordinators. The structure is restricted using determiner, pronominal modifier, nominal head, post nominal modifier, negation, verbal head, compliment, adjunct, phrase-level coordination, phrasal level subordination, sentence level coordination, sentence level subordination and constructors. Furthermore, for making it easy to read by non-specialists, PENG avoids ambiguity by applying a set of interpretation principles.

*ClearTalk (CT)* (Skuce 2003) is a knowledge formulation language for the semantic web that offers a flexible degree of formality with an adequate expressiveness (Kuhn 2014). Documents could be automatically translated from CT to knowledge representation structure. It can be used by an English speaker who does not understand formalisms with almost no training. The grammar includes 100 rules discussed in Skuce (2003) that need to be be learnt before start writing. CT has syntactic restrictions where for instance, basic sentences have the general form of subject, predicate, complement and modifier phrases. The authors implemented a search engine using dtSearch[1] as a basic engine to find things a normal search engine cannot find, for instance, show all occurrences of a certain term and window size that are (a) ordered in a useful way, (b) summarized by showing levels only, and (c) seeking certain words, while integrating

[1] http://www.dtsearch.com/.

WordNet (the engine can require a second word to be present). CT knowledge base consists of three modules as follows; linearly organized documents; the lexicon containing all terms and their lexical information; the topic pages. All are indexed by the search engine that allows to search facts.

*PENG-D* (Schwitter and Tilbrook 2004) is an extension of PENG with all its components plus the support for ontology construction as a start for language layering purposes. Since the meta-modelling architecture of the semantic web is not standardized, it was difficult to layer ontology and rule languages on top of RDFS as some elements in the model will have dual roles in the RDF specification. On the other hand, OWL was not the ideal solution, because of the semantic layering incompatibility of the standardized Description Logic (DL) that is based on first order model theory, with the semantics of RDFS based on non-standard model theory. The interoperability between OWL Lite and Horn Logic is used to create a paradigm called Description Logic Programs (DLP) (Grosof et al. 2003). PENG-D has expressivity and formal properties equivalent to DLP, that provides a pathway to layer more expressive constructions on top of it. The authors claim that PENG-D was easy to write with the help of the look-ahead text editor, easier to read than RDF based notations, and easier to be translated into corresponding machine processable format.

*Sydney OWL syntax (SOS)* (Cregan et al. 2007) came after Manchester OWL syntax (Matthew et al. 2006) and PENG to overcome their limitations. While Manchester OWL syntax have been well received by non-logicians and is the syntax for Protégé-OWL, it was limited by less focus expressions for property and individual. Also, PENG grammar did not support bidirectionality from PENG sentences to FOL and vice versa. The scope of SOS is to be compatible with OWL 1.1 for expressing ontologies, and to form translations, and to cover anything that can be expressed using OWL 1.1. Furthermore, to provide a two translation between SOS and OWL 1.1 syntax without any information loss. However, the bidirectionality involves using a context sensitive grammar and generating the output during the parsing process. The authors considered assisting the users writing ontologies using SOS by adding an interactive functionality of the lookahead editor. The general issue for designing a CNL is that the authors of the language have to decide how the language will support naturalness, complex constructs, language support for user defined terms and definitions. The authors of SOS choose to be more closer to OWL 1.1 syntax while trying to observe a trade-off to make the expressions as natural as possible. However, this results in SOS having more statements.

*Attempto Controlled English*[2] (ACE) (Fuchs and Schwitter 1996b) is a well known CNL involving translation into first order logic (FOL). 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 (ACE can also be translated into other formal languages). ACE is a mature CNL and has been in development since 1995 for over 14 years (Kuhn 2010). It was first introduced by Fuchs and Schwitter (1996a). Over forty articles have been published by the Attempto group and over 500 articles contain the term "Attempto Controlled English" on Google Scholar, Kuhn (2010). ACE is a general purpose CNL and is not restricted to any specific

[2] http://www.ifi.unizh.ch/atempto/.

domain. The grammar of ACE is perhaps the most expressive in that it can parse a variety of syntactic phenomena in comparison to other CNLs. ACE caters for instance for relative clauses, coordinated noun phrases, coordinated adverbial and adjectival phrases, numerical and distributed quantifiers, negation, conditional sentences and some anaphoric pronouns.[3] ACE Web Ontology Language known as ACE OWL, a sublanguage of ACE, is a means of writing formal, simultaneously human-and-machine-readable summaries of scientific papers (Kaljurand and Fuchs 2006; Kuhn 2006). ACE has also served as the basis for other applications such as interface language for a first-order reasoner (Fuchs and Schwertel 2003), a query language for the semantic web (Bernstein et al. 2004), an application for the partial annotation of Webpages (Fuchs and Schwitter 2007) and the usage of ACE for producing summaries within the biomedical domain (Kuhn et al. 2006). A recent development is the translation of a complete collection of paediatric guideline recommendations into ACE (Shiffman et al. 2010).

*RABBIT Controlled English* (Hart et al. 2008) is a well known implementation. It is essentially an extension of controlled language for ontology editing (CLOnE) (Funk et al. 2007), but much more powerful with respect to grammar expressiveness and ontology authoring capabilities. Like CLOnE, Rabbit is implemented using the GATE framework (Cunningham 2002). Rabbit was developed by the national mapping agency in Great Britain—Ordnance Survey. It is primarily a vehicle for capturing, representing and communicating knowledge in a form that is easily understood by domain experts. There are three broad types of sentences in Rabbit; declarations; axioms; import statements. Interestingly, a given class or concept can refer to a specific ontology in Rabbit i.e. one can refer to the animal `Duck` within a specific ontology—`Waterfowl` as opposed to a default ontology. Therefore, more than one ontology can be referenced in the Rabbit language (Hart et al. 2008). Rabbit attempts to cater for property restrictions such as transitivity and symmetry, but as the authors themselves argue that such concepts are "not aligned to the way people think" and that there is no ideal solution to creating natural language equivalents to property restrictions. Arguably, these issues should be dealt with support from the ontology engineer and not the domain expert directly. The work of Dimitrova et al. (2008), gives a good overview of Rabbit's expressiveness with respect to Rabbit's syntax patterns and their corresponding ontology mappings such as existential quantifiers, union, disjointness and cardinality.

*OWL Simplified English* is a finite state language for ontology editing (Power 2012). The argument for the finite state approach is that the majority of the OWL expressions created by ontology developers were invariably right branching and hence could be recognised by a finite state grammar. Based on previous studies of ontology corpora, the authors show how the individuals, classes and properties tend to have distinct Part Of Speech (POS) tags. Individuals or instances tend to be either proper nouns, common nouns or numbers, while classes are composed mostly of common nouns, adjectives and proper nouns. Finally, properties tend to open with a verb or auxiliary verb in the present tense. In paper Power (2012), the authors describe a finite state network that is capable of interpreting the CNL sentences in

[3] http://attempto.ifi.uzh.ch/site/docs/ace/6.5/ace_constructionrules.html.

the grammar with minimal knowledge of content words. OWL Simplified English permits the acceptance of some technical phrases that violate normal English. The language can capture ontology operations such as simple negation, cardinality, object intersection but aims to reduce or eliminate structural ambiguity.

*Semantic Query and Update High-Level Language (SQUALL)* (Ferre 2014) is a CNL for semantic querying and update of RDF graphs on top of SPARQL 1.1. The authors claim that SQUALL is easier to learn, and to formulate complex queries and updates than other CNLs. This is because SQUALL combine expressiveness close to SPARQL 1.1, a natural syntax and semantics based on Montague grammars (Montague 1970) which is a context free generative grammar based on formal logic and calculus. The semantics of SQUALL are translated from this logical intermediate language into SPARQL. The lexical conventions of SQUALL at the lexical level does not differentiate between singular and plural, and between nouns and verbs. However, it does differentiate between them at the syntactic level as it uses URIs for non grammatical words. SQUALL has some ambiguity that the system can resolve using some rules related to priorities of, algebraic operators and smaller syntagms, over, sentence modifiers, and larger syntagms, respectively. Also, ambiguity of two constructs of the same syntagm is resolved by choosing the shorter construct.

*AIDA* (Kuhn et al. 2013) (Atomic, Independent, Declarative, Absolute) is a proposed approach that can be considered as a CNL for extending the nanopublications concept, to facilitate keeping track of latest research results in modern science using informal representations. AIDA means that natural sentences written in English has to follow a scheme where sentences have to be Atomic, Independent, Declarative and Absolute. This approach introduces a prototype of a nanopublication portal called nanobrowser, based on Apache wicket and the Virtuoso triple store. Nanobrowser looks like a semantic Wiki, where a particular scientific statement is presented with opinions from researchers, about related sentences with its meta-nanopublication, and this shows that AIDA links and relate nanopublications with each other. However, the problem of expressing a sentence in more than one way needs to be taken into account. To solve this, the authors proposed a mixture of clustering and crowdsourcing, so that nanopublications users can identify sentences that have similar meanings. In order to describe scientific results, AIDA assumes that sentences have their own independent existence. Therefore, each AIDA sentence get its own URI to make it first-class citizen in the RDF world. Furthermore, each AIDA sentence should be extractable from its URI without the need to consult any resources, and vice-versa (Table 1).

## 4 Tools based on semantic web related CNLS

This section will not discuss controlled languages, but will present tools like editors, wikis and frameworks that use CNLs to perform specific tasks. In Sect. 4.1, we will discuss the ontology engineering tools presented in the literature for the aim of ontology creation and authoring, Sect. 4.2, provides the ontology querying tools, and Sect. 4.3, shows the rest of the tools presented in the literature which use CNLs for different purposes. The end of the section includes a summary table (Table 2)-comparing all the tools.

**Table 1** Summary table comparing all the semantic web related CNLs listed in chronological order of publication

| CNL | Input | Output | Parsing | Aim | Advantage | Limitations |
|---|---|---|---|---|---|---|
| PENG | PENG texts | FOL | Incremental bottom-up chart parser discovers the word form that is currently under investigation and feeds this grammatical information to the look-ahead editor | Write precise and unambiguous specifications for knowledge representation | Predictive editing feature | Grammar is informed by FOL rather than DL considerations<br>The grammar is not bidirectional |
| CT | CT natural language | Formal logic notation and into other natural languages | Using syntactic and semantic restrictions | Knowledge representation and extraction | The author can choose to leave or remove ambiguity, depending on the need | Heavily restricted on both the syntactic and the semantic levels |
| PENG-D | PENG texts | RDF and OWL | Look-ahead text editor, a controlled language processor, an ontology component and an inference engine | Same as PENG plus the support for ontology construction | It has formal properties that are equivalent to DLP<br>Enable ontological definitions to be combined with rules<br>Predictive editing feature | The grammar is not bidirectional |

**Table 1** continued

| CNL | Input | Output | Parsing | Aim | Advantage | Limitations |
|---|---|---|---|---|---|---|
| SOS | Sydney OWL syntax | OWL 1.1 | Uses a context-sensitive grammar for bidirectionality which can store the required elements and employ an axiom schema instantiated during parsing then perform one-to-one mapping between controlled natural language and OWL syntax | Create and edit OWL ontologies<br>Provide English translations of OWL ontologies | Predictive editing feature<br>Bidirectional mapping to OWL 1.1<br>Coverage of the entire OWL language<br>Easy to implement in tools | Only one SOS form for each OWL form<br>Emphasis on variables makes it less readable than pure natural language |
| ACE | ACE texts | FOL | Attempto Parsing Engine (APE) consists of a definite clause grammar written in Prolog | Knowledge representation language to create and edit OWL ontologies | Syntactically and semantically expressive CNL | Does not focus on bidirectionality<br>Requires definition of a lexicon |
| Rabbit | Rabbit sentences | OWL | Rabbit parser implemented in java consists of a pipeline of linguistic processing resources | Create and edit OWL ontologies | It represents the whole ontology | Assumes that the user will have some basic knowledge of the constructs and axioms used in ontology development |

**Table 1** continued

| CNL | Input | Output | Parsing | Aim | Advantage | Limitations |
|---|---|---|---|---|---|---|
| OWL Simplified English | Predefined function words and verbs listed by the user | OWL ontologies | Finite-state transducer | OWL ontology editing | Disallow structurally ambiguous sentences<br>Fast and reliable implementation of an editing tool using finite state grammar | Very restricted coverage of OWL axioms |
| SQUALL | SQUALL texts | SPARQL | Translating SQUALL text into an intermediate logical language in the form of Montague grammar and then to SPARQL | Query and update RDF graphs | Strong adequacy with RDF, and covers all constructs of SPARQL | SQUALL sentences may look unnatural because URIs are invariant with respect to number or person |
| AIDA | AIDA sentences | RDF | Each AIDA sentence get its own URI in the RDF world. URI encoding, uses the relations to link AIDA sentences and relate them to other entities | Allow informal representations of scientific assertions for semantic publishing of nanopublications | Sharing and interlinking of scientific findings, while being more flexible and applicable to a much wider range of scientific results | No concrete proposal yet to make the scientific results atomic<br>URIs encoding is against existing recommendations of keeping URLs opaque |

**Table 2** Summary table comparing all tools based on semantic web related CNLs listed in chronological order of publication

| Name | Input | Output | Parsing | Aim | Advantage | Limitations |
|---|---|---|---|---|---|---|
| *Ontology engineering tools* | | | | | | |
| WYSI-WYM | Natural language | OWL data | WYSIWYM engine in Prolog, and user interface in JAVA. Also, there is a NL generator to provide feedback for user interactions | Defining and editing knowledge bases | Allows direct control over semantic features through a NL feedback | The usability of the tool varies according to the structure of the ontology |
| Gino | NL expressions using a grammar derived by the loaded ontolgies | RDF triples or OWL axioms | Parse rules based on CNL grammar from 120 static rules that could be extended dynamically based on the ontology grammar. Then, the sentence is passed to the JENA engine for execution | Domain independent ontology editing interface in quasi NL<br>Extension of Ginseng to support procedural statements and introduce new entities | Guided input natural language editor<br>Easy adaptation to new ontologies using dynamic grammar generation | Exploiting NL expression is always limited by the vocabulary from the loaded ontologies |
| ROA | CLOne input | Modified or edited ontology | GATE NLP tools plus the gazetteer and 2 JAPE transducers | Introducing NLG into CLOnE to facilitate round-trip ontology authoring | The text generator reduces the learning curve for users | Works for less sophisticated knowledge engineering tasks |
| ROO | Rabbit sentences | OWL | Rabbit parser plus OWL conversion using OWL API | Embed into Protégé as a plug-in<br>Ontology authoring | Show feedback and provide a list of Rabbit templates to help the user | There are common error patterns when using a CNL to define ontological constructs |
| ACEView | ACE sentences | OWL/ SWRL | Plug-in for Protégé that relies on the OWL API | Translating to and from OWL/SWRL | Naming conventions and consistency<br>Structural complexity of class expressions<br>Verbalizing complex constructions into simpler syntax | No predictive editor is integrated<br>Weak semantic feedback to the user |

**Table 2** continued

| Name | Input | Output | Parsing | Aim | Advantage | Limitations |
|---|---|---|---|---|---|---|
| ACEWiki | ACE texts | OWL | Bidirectional mapping between ACE and OWL using APE parser | Ace Editor for wikis<br>Collaborative ontology development using NL | Easier to use and more expressive than other semantic wikis<br>Predictive editor | Still a prototype and not a real-world application<br>Supports a single grammar and allows users to control only the set of (monolingual) content words |
| ACEWiki-GF | ACE texts | OWL | Same as ACEWiki with replacing the ACE parser and English lexical editor, with a GF parser and source editor<br>It uses bidirectional mapping between ACE and OWL using APE parser | Ace Editor for wikis<br>Multilingual collaborative ontology development using NL | Same advantages as ACEWiki plus supporting multilingualism | Still a prototype and not real-world application |
| *Ontology querying tools* | | | | | | |
| PNL | PNL sentences | NL queries based on assertions | Builds upon RDF and first-order logic, and uses Prolog to calculate inferences | Querying and reasoning on the web | Unambiguous and has well-defined semantics<br>No need to learn a separate, query language as it is the same as its regular expression language | Unnatural capitalization mitigates the naturalness of the language<br>Complex rules have to be applied in order to resolve ambiguous syntax trees |

**Table 2** continued

| Name | Input | Output | Parsing | Aim | Advantage | Limitations |
|---|---|---|---|---|---|---|
| Ginseng | NL queries using a grammar derived by the loaded ontolgies | SPARQL | A grammar compiler that generates a set of dynamic rules referred by the loaded ontolgies, and a set of CNL static rules to provide the structure of a basic query | NL query interface to access OWL ontologies | Allows ontology annotation to create alternatives to the ontology identifiers<br>Works on domain independent ontologies<br>Guided input interface for queries | Writing queries is constrained by the vocabulary of the loaded ontologies |
| OWLPath | OWLPaths guided English queries | SPARQL | Parsing includes Ajax interface, suggester, grammar checker, SPARQL generator and ontology manager<br>The question ontology represents the grammar and the domain ontology represents the structure of concepts and relationships | Allows non-expert users to easily create SPARQL queries that could be issued over most existing ontology storage systems | Domain-independent system that can be applied to any ontology<br>It has a sophisticated grammar | The ontologies used for test purposes are relatively small and are processed in-memory<br>The question ontologies only include the most common constructors to allow formulating typical queries in the experimental scenarios |
| *Other tools* | | | | | | |
| GF | Grammar rules | CNL | Bidirectional mapping between the concrete language strings and their corresponding abstract trees | Framework for CNL editing and implementation<br>Implementing multilingual CNLs | Support multilingualism using grammar libraries<br>Compatible with many programming languages | The language coverage and the reasoning capabilities are limited to what the grammarian has encoded in the grammar |

**Table 2** continued

| Name | Input | Output | Parsing | Aim | Advantage | Limitations |
|---|---|---|---|---|---|---|
| PathONT | PathOnt CNL | RDF | Developed using the Java-2 platform<br>Based on the Galen top level ontology<br>Statements are mapped to RDF triples | Formalization of the gross description<br>Communication among clinicians and technicians involved in pathology examinations | Facilitates the interactions between end-users of medical information and between different applications<br>Multilingual as it supports both English and Korean languages | Covers only simple existential statements |
| Atomate | Atomate language | RDF | Written entirely in Java script and consists of world model, rule chainer, and triggered rules | Personal information assistance engine to define automatic tasks and reminders, while taking context and current activity into account | Integration of heterogeneous data from RSS feeds<br>Easy to extend its capabilities, add new data sources, and new types of information | Lacks a sophisticated DL inference engine<br>There is no capacities to learn, search, or act on the user's behalf beyond the rules set up by the user |

### 4.1 Ontology engineering tools

*What You See is What You Meant (WYSIWYM)* is a well-known implementation which employs the use of NLG to aid the knowledge creation process, with respect to ontology driven generation of CNLs or *conceptual authoring* (Power et al. 1998). It involves direct knowledge editing with natural language directed feedback. A domain expert can edit a knowledge based reliably by interacting with natural language menu choices and the subsequently generated natural language feedback which can then be extended or re-edited using the menu options.

Similar to WYSIWYM, *Guided Input Natural Language Ontology Editor (GINO)* provides a guided, controlled natural language interface (NLI) 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[4] with respect to morphological realisation and other development environments (Bernstein and Kaufmann 2006). GINO translates the completed sentence into triples (for altering the ontology) or SPARQL queries and passes them to the Jena semantic web framework. Although the guided interface facilitates input, the sentences are quite verbose and do not allow for aggregation. Static grammar rules exist for the controlled language but in addition, dynamic grammar rules are generated from the ontology itself.

*Round trip ontology authoring (ROA)* builds on and extends the existing advantages of the CLOnE software to create and populate an ontology with the addition of a text generation component to form ROA environment. The aim of the text generator is to reproduce the CNL from an ontology, edit the text as required, then parse it back into the ontology until the user gets the desired results. Thus, NLG acts as a feedback to guide the user and reduces the need to learn the controlled language by following examples, style guides or CLOnE syntactic rules (Davis et al. 2008). The ROA pipeline consists of GATE NLP modules to annotate the input document, followed by Keyphrase gazetteer and to two JAPE transducers to identify quoted and unquoted chunks. Then, a controlled language for information extraction (CLIE) component is connected to the existing ontology to interpret the input sentences. Finally, the ontology is connected with the text generator component to act as an ontology verbaliser to present the ontology in textual form as an ambiguous subset of English (Davis et al. 2008).

*Rabbit to OWL ontology authoring (ROO)* (Dimitrova et al. 2008) is an editing tool that seeks to cater for the entire ontology engineering process. It was developed by the University of Leeds and is an open source Java based plug-in for Protégé. ROO supports the domain expert in creating and editing ontologies using Rabbit. The authors argue that CNL interfaces tend to ignore the ontology construction process. Domain experts are involved in the early stages of the ontology engineering process and engage in the conceptualisation of the ontology, while the ontology engineer is involved at the end stages and focus on the logical level of the ontology. A new intelligent model was integrated to ROO to understand the user actions and

[4] http://www.eclipse.org/.

give feedback accordingly. The model was introduced in Denaux et al. (2012) to resolve the modelling errors, by providing a framework for semantic feedback when adding a new fact to an existing ontology. The new framework extends the syntactic analysis performed by Rabbit through categorizing the new ontological facts into four categories concerning inconsistency and novelty of facts. This feedback approach was observed to be repetitive, confusing and sometimes redundant (Denaux et al. 2013). As a result, a new framework with dialogue interfaces was introduced in Denaux et al. (2013) as an extension to Rabbit. It provides more appropriate feedback according to different situations by keeping track of the ontology history. In addition, the inputs of the domain experts are analyzed and an intention is assigned to each input.

*ACEView* is an editor implemented as a plugin for the Protégé editor[5] mapping from ACE to OWL/SWRL and from OWL to ACE. It empowers Protégé with additional interfaces based on the ACE CNL in order to create, browse and edit an OWL ontology (Kaljurand 2008). The user can also query the ontology using ACE questions to access newly asserted facts from the knowledge base. ACE text is automatically parsed and converted into OWL/SWRL. ACEView comprises vocabulary and wordform view, asserted knowledge and entailed knowledge views. The most beneficial features are, ensuring that consistent naming conventions are used by placing restrictions on OWL entity names, restricting the complexity of the OWL class expressions, and verbalizing complex constructions into simpler syntax (Kaljurand 2008).

*AceWiki* (Kuhn 2008a) is a monolingual CNL based semantic wiki that takes advantage of ACE for its syntactically user friendly formal language, and of OWL frameworks for applying classification and querying. The AceWiki content is based on ACE predictive editor notation grammar called codeco (Kuhn 2012). The main benefit of codeco is that it can translate all AceWiki content to OWL.

*AceWiki-GF* (Kaljurand and Kuhn 2013) is a multilingual extension of the previously mentioned AceWiki, where users can get all the benefits of AceWiki plus multilinguality after using the Grammatical Framework (GF), discussed in the next section. The implementation was done by modifying the original AceWiki to include GF multilingual Ace grammar, GF parser, GF source editor, and GF abstract tree set. This study included an evaluation about the accuracy of translation in AceWiki-GF. The evaluation showed that the translation accuracy was acceptable, although some errors due to different reasons in terms of Resource Grammar Library (RGL), where incorrect use of RGL by mixing regular and irregular paradigms, using unnatural phrases to native speakers, and negative determiners.

### 4.2 Ontology querying tools

*Pseudo natural language (PNL)* (Marchiori 2004) is the first query logical system to provide natural, easy and friendly way for people to use the semantic web. The paper introduces the Metalog project,[6] to fill the axis of the people to the semantic

[5] http://Protege.stanford.edu/.

[6] http://www.w3.org/RDF/Metalog/. W3C, 1998–2004.

web. Metalog uses pseudo natural language (PNL) interface that is similar to natural language and extends RDF with the Metalog Model Level (MML). PNL is an unambiguous language with the principle one language, one query, designed to be easy to read for users and easy to write for developers, by sacrificing the total freedom of the natural language with the restrictions to the language. Metalog has a smart querying ability that accepts informal queries and normalize it to be mapped into an assertion.

*Guided Input Natural Language Search ENGine (GINSENG)* (Bernstein et al. 2006) provides a quasi-natural language guided query interface to the semantic web. Thus, it came out to reduce the gap between real world users and the logic-based semantic web. Ginseng allows users to query any semantic web knowledge base, using a guided input NL vocabulary loaded ontologies that grow with every additional added ontology, but without using any predefined vocabulary. Despite this can limit the users possibilities, it ensures that every query will have a correctly matched result. Ginseng guides the user with a set of possible queries while avoiding grammatical errors, by presenting to the user a choice pop up box that includes suggestions on how to correctly complete the current sentence, and hence the possible choices get reduced as the user continues typing. Ginseng translates queries into a RDF Data Query Language (RDQL) and displays the result. The architecture of ginseng consists of three modules as follows; multilevel grammar; an incremental parser; an ontology access layer through Jena. The multilevel grammar is a domain that contains about 120 independent rules, constructed manually, and divided into two types of rules (1) static grammar rules, to provide the basic structure of sentences and questions, (2) the dynamic grammar rules from the loaded OWL ontologies, created for each class, instance, object, and data-type properties. Furthermore, Ginseng provides ontology annotation option with Ginseng tags. The incremental parser uses the grammar to specify the complete set of parsed sentences without incorrect entries, and to generate the resulting query by creating a complete parse tree.

*OWLPath* (Valencia-Garca et al. 2011) is an ontology-guided input natural language query editor that combines the advantages of both the natural language interfaces NLIs and CNLs, to reduce the gap between users and the semantic web. It guides the user on how to complete a query using the question and the domain ontologies. The question ontology represents the grammar and the sentence structure, while the domain ontology represents the concepts and relationships in the domain. The main components of OWLPath system are as follows; Ajax interface that loads the domain related set of ontologies and let the users build the query; the suggester generates a list of terms shown in a pop-up list for the user to choose from; the grammar checker determine only the correct grammatical entries combinations; the SPARQL manager translates the query into SPARQL statements and parse it to the knowledge base through the ontology manager; and finally the results are shown to the user.

### 4.3 Other tools

*Grammatical Framework (GF)* is an implementation framework for multiple CNLs (Angelov and Ranta 2009; Ranta 2004). GF can cope with a variety of CNLs as well

as boost the development of new ones. In Angelov and Ranta (2009), the authors reverse engineer ACE for GF in order to demonstrate how portable CNLs are to the GF framework as well as how CNLs can be targeted to other natural languages. ACE is ported from English to five other natural languages. In short, the core advantage of GF is its multilingualism in that its primary task is domain specific knowledge based machine translation (MT) of controlled natural languages. It adds a syntax formalism to the logical framework which defines realisations of formal meanings as concrete linguistic expressions. The semantic model is called the *abstract syntax* while the syntactic realisation functionality is called *concrete syntax*. The authors state that GF is multilingual, in that one abstract syntax, acting as an interlingual, can be (given a concrete syntax for one or more source languages) re-targeted to several languages. The GF libraries now contain a collection of wide coverage grammars for over 15 natural languages. There is an increasing activity with respect to the GF development and a vibrant open source community, which continues to create language resources for GF. The success is also due to the European project, multilingual on-line translation (MOLTO).[7] This has boosted the uptake of GF and resulted in many comprehensive applications.

GF applications range from mathematical proofing, dialog systems, patent translation (España-Bonet et al. 2011), multilingual wikis and multilingual generation in the culture heritage domain (Angelov and Ranta 2009; Dannélls 2008; Dannélls et al. 2012). In addition, there have been recent efforts to cater for semantic web ontologies in GF. Although GF has no specific CNL, one could argue that its growing open source community may result in GF becoming the de-facto open source general framework for developing resources for engineering multilingual CNLs.

*PathOnt* (Kim et al. 2005) is a PATHological ONTology based application. It uses a controlled ontology from the terminology resources in GALEN (Rector et al. 1995) for the gross description medical ontology system. The need for this application is to solve the communication problem between pathologists and other professionals who misinterpret the meaning of the gross description. The system consists of three components; the PathOnt semantic to specify the required ontology for the gross description; PathOnt object for visualizing the macroscopic findings stored in the RDF file; PathOnt syntax that generates an XML form for the input update.

*Atomate It* (Van Kleek et al. 2010) is a web based reactive personal information assistance engine, that allows end-users to use data feeds to drive reactive automation. For instance, Atomate can integrate the information out of the RSS/Atom feeds from social networks into RDF model to derive useful behaviors, and thus important reminders can be created, taking into account the rules specified by the user. The CNL interface design is based on GINO and Ginseng interfaces (Van Kleek et al. 2010). The rules in Atomate consists of antecedent to represent the execution conditions, and consequent to specify the actions to be taken. Atomates data flows from the lost of the data sources provided by the user in the RDF model to the Atomates feeder. The feeder creates a new entity for each new data source or

[7] http://www.molto-project.eu/.

updates the existing ones. Then, based on these updates the rule chainer retrieves all the rule entities from the world model, and fire all rules whose triggered antecedents depend on the changed entities. Atomate is implemented to be an add-on in Firefox browser, so that sites can access data through javascript libraries and APIs and add new data sources without hard integration work.

## 5 Evaluation of CNLs

According to Kuhn (2013), one of the main methods used to evaluate CNLs is *paraphrase-based*, where a group of users provide their feedback and observations about the usability of a CNL. The feedback can be provided via a questionnaire or conversations. With respect to related work, in the next sections we will review existing CNL research, but in the context of user evaluation. The end of the section includes a summary table (Table 3) comparing all the evaluations for these CNLs. Some CNLs do not not have any evaluations mentioned in the literature, and hence they are not listed below.

The authors of *ClearTalk* conducted an evaluation to check if CT helps students to learn (Skuce 2003). The experiment was informal, where 80 students were divided into two groups to answer questions about applets within 45 min, with both groups have text books, and one group only has access to the KB. The results show that the group with access to the KB got 50 % higher mark, which proves the main function of the KB that makes students find facts faster. However, some of the difference might be due to better understanding of the subjects.

Recently, an evaluation for *ACE* was presented in Kuhn (2013), where it describes an evaluation framework for CNLs based on ontographs. Ontographs are a graphical notation to enable tool independent and reliable evaluation of the human understanding of a given knowledge representation language. They serve as a common basis for testing and comparing the understandability of two different formal languages and facilitate the design of tool-independent and reliable experiments. An experiment was performed by 64 participants to compare the syntax of ACE versus a simplified version of *Manchester OWL syntax*, to test which syntax is better in terms of, understandability, learning time, and users acceptance. The results showed that users were able to do better classification using ACE with approximately 5 % more accuracy than Manchester OWL, and 4.7 min less time for learning. Also, in terms of understandability ACE got 0.67 higher score than Manchester OWL (Kuhn 2013).

In Engelbrecht et al. (2009), the authors undertake an evaluation to assess whether domain experts without any ontology authoring development can author and understand declaration and axiom sentences in *Rabbit*. The experiment included 21 participants from the ordnance survey domain and a Rabbit language expert. The participants were given a text that describes a fictional world and were asked to make knowledge statements, then they are analysed for correctness by independent experts and compared to equivalent statements created by the Rabbit expert. Interestingly, on average 51 % of the generated Rabbit sentences contained at least

**Table 3** Summary table comparing all the evaluations for semantic web related CNLs in chronological order of publication

| Name | Compared | Evaluation goal | Experiment | Participants | Results |
|---|---|---|---|---|---|
| CT | No | If CT helps students to learn | Informal test let students answer questions related to IT field | 80 students | The group with access to the KB find facts faster and got 50 % higher mark. Other than that no other statistical results |
| ACE | Simplified version of Manchester OWL syntax (MLL) | Test which is better in terms of, understandability and learning time using ontographs | Users are given instructions about using ontographs then asked to answer a questionnaire | 64 students with no background in Logic or CS | Learning time: ACE got 4.7 min less<br>Understandability: ACE got 0.67 higher score |
| Rabbit | No | Test the understandability of users with no background of ontology authoring | Convert a text about fictional world topic into rabbit sentences, then the ouput is compared with the statements created by the Rabiit expert | 21 users from Ordnance survey domain and a Rabbit language expert | On average 51 % of the generated rabbit sentences contains at least 1 error |
| AIDA | No | Exp1: difficulty of creating nanopublications using AIDA | Online questionnaire that explains how to rewrite sentences using AIDA and asks for the difficulty | 16 participants with no knowledge about AIDA | On average a sentence is created in 90 s<br>Understandability: easy but not very easy<br>Rewriting: medium difficulty |

one error. Furthermore, the most common error was the omission of the quantifier at the beginning of every sentence.

For the evaluation of *AIDA*, the authors did two evaluations related to the initial stage of the approach (Kuhn et al. 2013). The first evaluation was done to measure the difficulty of creating nanopublications for scientific results. The experiment involved 16 participants with background in biology and medicine who never knew about AIDA. A random sample was taken from Pubmed abstracts (Kuhn et al. 2013) that have a conclusion section. The evaluation was through an online questionnaire consisting of three parts; the first part explains AIDA concept; the second part showed five short texts to be written in one to three AIDA sentences each; and the last part asked about the difficulty to understand AIDA concept and to do rewriting tasks. The results showed that an average sentence required 90 s to be created including the time to learn about AIDA concept. Out of 163 sentences created by the user, 70 % were perfectly complied with the AIDA restrictions. All participants mentioned that understanding AIDA concept was easy but not very easy, and the rewriting task was of medium difficulty. The second test was to evaluate the quality of automatically extracting AIDA nanopublications from text resources and relate them to each other. The authors used GeneRif[8] dataset, which contains sentences about gene and protein functions. Results showed that 71 % of the resulting AIDA sentences were fully complied with AIDA restrictions.

## 6 Evaluation of the tools based on semantic web related CNLs

According to Kuhn (2013), *task-based experiments* are used to conduct evaluations of tools that use CNLs to perform different tasks. Users are provided with instructions sheet to read and understand, then they are asked to perform some tests using the tool. The statistical data from the tests are observed and recorded to check the tool effectiveness. With respect to related work, in the next sections we will review user evaluation of tools. The end of the section includes a summary table (Table 4) comparing all the evaluations for these tools. Some tools do not not have any evaluations mentioned in the literature, and hence they are not listed below.

### 6.1 Evaluation of ontology engineering tools

An evaluation of *WYSIWYM* was carried out two times. The first evaluation was presented in the CLEF[9] project developed for the medical domain (Hallett et al. 2007). The experiment was conducted by 15 participants mainly medics and bio-informaticians to test usability, understandability and the difficulty of using the tool. Participants were given a short demonstration on how to construct a simple query using the interface, and then asked to create a set of four SQL queries for a database in the medical domain. The sets were given to each participant in a different order to

[8] http://www.ncbi.nlm.nih.gov/gene/about-generif.

[9] http://www.clinical-escience.org/, Retrieved 2008-05-22.

**Table 4** Summary table comparing all the evaluations for the tools based on semantic web related CNLs in chronological order of publication

| Name | Compared | Evaluation goal | Experiment | Participants | Results |
|---|---|---|---|---|---|
| *Ontology engineering tools* | | | | | |
| WYSIWYM | No | Usefulness and difficulty of using the tool | Reproduce description of English paragraphs in WYSIWYM | 16 researchers and Ph.D. students from social science domain | On 1–5 scale (5 is very useful and very difficult)<br>Usefulness = 3.94<br>Difficulty = 2.69 |
| ROA | Protégé | Usability for ontology editing tasks | Ask users to work on task lists using Protégé and ROA | 20 participants from research and industry with no background about Protégé | Mean SUS score of usability:<br>ROA = 74 %<br>Protégé = 41 % |
| ROO | ACEView | Usability and usefulness of the tool | Create ontologies based on hydrology and environmental models using ROO and ACEView | 16 participants from geography and environment studies | Usability: users are more willing to use ROO than ACEView<br>Usefulness on 1–7 scale (7 is strongly agree)<br>ROO = 5<br>ACEView = 0.38 |
| ACEWiki | No | Difficulty of using the tool for-non technical users | Add knowledge to ACEWiki given some instructions | 20 participants mostly 7 students and graduates with no background about logic and semantic web | Medium difficulty = 75 % of the users.<br>Difficult = 25 % of the users |
| ACEwiki-GF | No | Difficulty and translation accuracy of ACEWiki-GF | Each user evaluate the output of the tool in his native language through an online questionnaire | 30 participants each one is fluent in one of the languages used in the evaluation | Translation error rate was less than 5 %<br>Difficulty on a 0–4 scale (4 is very easy) = 2.93 |

**Table 4** continued

| Name | Compared | Evaluation goal | Experiment | Participants | Results |
|---|---|---|---|---|---|
| *Ontology querying tools* | | | | | |
| Ginseng | SQL | Usability, speed and precision/recall of the paresd queries | Write geographical queries using Ginseng and SQL interface | 20 students from CS department with knowledge about SQL queries | Speed: on average Ginseng was 1 min faster<br>Usability: Ginseng is easier to learn<br>Precision = 92.8 %<br>Recall = 98.4 % |
| OWLPath | No | Time to generate a query | Create queries related to tourism based ontology | Four Ph.D. students with background in ontologies and SPARQL | It takes less time to generate a query using OWLPath than to do it manually |
| *Other tools* | | | | | |
| Atomate | No | Understandability and difficulty usefulness | Feedback about the design and creating rules using the tool | 15 UI researchers for design review feedback and 33 participants for the rule creation process | Easy to use: 65 % of the users<br>Difficult: 35 % of the users<br>Usefulness on 1–7 scale (7 is very useful) = 5.5 |

ensure that tasks complexity does not affect the process. The results showed that from the second task onwards all participants achieved 100 % success in composing all the queries in a mean completion time 3–9 min per query, and became faster with each task, especially after the first task. For testing the understandability, the participants were given a paper-based questionnaire of complex queries and asked to select the correct meaning for each query from a list of three options. The results showed that on average the participants choose the correct interpretation 84 % of the time.

The second evaluation was conducted in Hielkema et al. (2008) by 16 researchers and Ph.D. students from the social sciences domain. Users were shown a 6 min background video for the main functionalities of the WYSIWYM interface. Descriptions of four resources as paragraphs of English were provided to the users. The goal was to reproduce the descriptions using the WYSIWYM tool. Each subject also received the descriptions in varied order. The descriptions were further divided into eight to ten sub tasks. The successful completion of certain sub-tasks was dependent on the preceding sub-task. Task completion times, number of operations, as well as errors including "avoidable" errors (which imply the result of an error introduced from a previous sub-task), were measured. The results were encouraging, where users mean completion times decreased significantly. Hence, users gained speed over time. The results of the subjective feedback on the tool indicated that the tool perceived positively, where the mean scores on a 1–5 (very useful and very difficult) scale was 3.94 for usefulness, and 2.69 for the difficulty level. However, the results in Hallett et al. (2007) was more positive than Hielkema et al. (2008), since the participants of the CLEF project were mainly medics who understand their domain very well. On the other hand, the social science domain tends to be more varied with many different theories and approaches. Consequently, the underlying domain ontology can have a large significant impact on usability. More importantly, users from the social sciences field reported that they were overwhelmed by the large number of options available i.e. thirty properties per one object (Hielkema et al. 2008).

*ROA* evaluation is conducted against Protégé (Davis et al. 2008), where 20 users were recruited from both research and industrial background, but with no background in either GATE and Protégé tools. The participants were provided by Protégé manual, text generator examples, and two task lists. They were divided into two groups, each group was asked to work on each task list, using either ROA or Protégé, opposite to the other group. Finally, they were asked to complete both a SUS (Brooke 1996) questionnaire and a comparative questionnaire for each tool. The results showed that the mean SUS score for ROA is 74 %, and 41 % for Protégé.

An evaluation study of *ROO* was conducted against *ACEView* in Dimitrova et al. (2008) to compare both tools in terms of usability, usefulness and the quality of resultant ontologies. The study involved 16 students from the domains of geography and environmental studies. Student were asked to create ontologies based on hydrology and environmental models, respectively. Both ontology creation tasks were designed to resemble real tasks performed by domain experts at the Ordnance Survey. Ontologies for both domains were produced by the Ordnance Surveys OS

MasterMap10.[10] The usability results showed that messages in ROO were more helpful, the tool was less complex than ACEView, and users would be more willing to use ROO again. In terms of usefulness, the mean score for ROO users was 5 which is higher than ACEView users who scored 0.38, as the understanding of ontology modelling improves significantly more when using ROO than when using ACEView. The quality of the resultant ontologies, showed that ontologies built with ROO have better readability than those built with ACEView, as ROO encourages users to add annotations for concepts and relationships. Another study presented as an extension of ROO in Denaux et al. (2012) showed that 91 % of the feedback messages were helpful to the users, and 78 % were informative. However, feedback caused confusion and overwhelming for 10 % of the cases.

*ACEWiki* was evaluated to test whether people with no background about ontologies and logic will be able to learn and deal with ACEWiki, without the help of an expert and without spending long time (Kuhn 2008b). The experiment was conducted online, where 20 participants mostly students and graduates with no background about semantic web or logic, were provided by instructions sheet and then asked to add whatever knowledge they like to ACEWiki, following certain restrictions. The participants created 186 sentences, 148 of them were correct, and the other 38 were not. The participants spent on average 11 min for creating the first correct sentence, and 8.2 min overall for each correct sentence. The feedback for the difficulty level of using ACEWiki was mostly of medium difficulty, and 25 % of the users found it difficult.

*ACEWiki-GF* evaluation (Canedo et al. 2013) to determine, how much using ACEWiki-GF is effective and efficient to help two users of different languages understand each other. The experiment is to let each user write an article in his native language, and in the post editing stage users read the automatically translated articles written by other users and evaluate, whether the sentences are true or false in their language. The evaluation took place online through the ACEWiki-GF online tool, where 30 participants were asked to create a new wiki page and write true and false statements. Then after finishing, the users were asked to fill a questionnaire about their feedback about the system. The 30 participants created in total 316 sentences on average of 37 min. One hundred and seventy one of the sentences were measured as true, and 145 as false. The results show that the translation error rate for ACEWIki-GF is less than 5 %. The feedback from the users for the difficulty level of using ACEWiki-GF in general was 2.93 on a 0 (very difficult) to 4 (very easy) scale. The result was 2.77 for the difficulty level of the sentence editor.

### 6.2 Evaluation of ontology querying tools

*Ginseng* was evaluated against SQL using SUS evaluation in terms of usability, speed, precision/recall and the ability to parse a large number of real world queries (Bernstein et al. 2006). The evaluation was held by 20 students from the CS

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

department with knowledge of SQL queries. In the experiment, half of the users have to query into Ginseng, and the other half were provided with SQL interface. The results showed that, in terms of speed, Ginseng was faster than SQL with average difference of 1 min. Also, Ginseng was rated to be better integrated and easier to learn. In terms of parsing power, one knowledge base from geographical Mooney (Tang and Mooney 2001) was used with 880 queries. Ginseng could execute 40 % of the queries out of the box. In addition, the queries that could be parsed resulted in precision of 92.8 % and a recall of 98.4 %. However, the usability evaluation was limited by a specific subject (CS students), and it was not performed across huge datasets. The authors intend to improve these limitations by extending Ginsengs property tags generation which can be automated with WordNet and machine learning techniques. In contrast to PENG, where knowledge has to be entered into the system using a complete NL processing engine, Ginseng query existing semantically annotated content using a simple querying grammar, where it can be dynamically extended by any OWL ontology structure.

*OWLPath* evaluation was conducted to test the performance analysis and the user experience (Valencia-Garca et al. 2011). The performance analysis was in terms of time elapsed between selecting the next entry in the query, and showing the next choice in the pop-up list, taking into account the time for SPARQL statement generation. All the tests were performed on the local machine to avoid internet latency, taking the average time over ten runs, the results showed that the elapsed time did not change for larger number of words as the number of relations decrease as well. In addition, the time for generating the SPARQL statements is short, since the RDF triples of the words are generated when the user enters each word. On the other hand, regarding the user experience evaluation, precision and recall were not relevant for evaluation, since the OWLPath system is very accurate, as the resulting queries forced by the pop-up list through the ontologies were always valid. However, the authors designed an experiment to test the advantages of building queries using OWLPath. The experiment involved four Ph.D. students with strong background in ontologies and SPARQL to create ten queries related to a tourism-based ontology. The results showed that, it takes less time to generate a query using OWLPath interface than to do it manually.

### 6.3 Evaluation of the other tools

The authors of *Atomate* conducted two evaluations to test whether the users will be able to understand and create rules, and to check whether the users will be interested to use the system in the present or the future (Van Kleek et al. 2010). The first study was a design review with 15 UI researchers to get early feedback before the rule creation process. The second study, involved three colleagues from their lab who were asked to create nine rules ranged from simple to complex, after watching an explanation video. For the design review study, the authors got further feedback for making the rule creation process more clear. For the rule creation study, 33 participants did the study. Twenty six of them completed all the rules and the survey. Fourteen of the participants had programming experience. The first six rules were correct over 75 % of the time, while the rest of the rules were more problematic. For the complexity of creating the rules, 65 % of the users found it

easy, while the rest 35 % found it difficult. Regarding the usefulness of the system, on a scale of 1–7 (7 is very useful), the mean response of the participants was 5.5.

## 7 Conclusion

With respect to CNLs for ontology authoring we make the following analytic conclusions:

- The evaluations conducted for the CNLs still need more work and further analysis. There is a need for the CNL community to agree on a concrete methodology for the evaluation of CNLs. It was clear that the authors of each CNL or each CNL tool developed a different (some times even ad-hoc) methodology for their respective evaluation according to the available resources. This makes difficult to compare two or more CNL with each other at a later stage. Furthermore, researchers should make efforts to ensure the optimum number or at least the minimum number of subjects/users is met for both the task-based and paraphrase based evaluations. The optimum number of users for both evaluations is an open question for research. However, we refer to the research output from Nielsen (2006), where it was found that five users is a sufficient number to find most of usability problems, while for quantitative (aiming at statistics) evaluation, 20 users is a reasonable confidence interval. Many evaluations do not include any statistical evidences, that make it difficult to compare with another CNL of interest. With respect to the paraphrased based approaches ontographs presented in Kuhn (2013), may have potential towards a clear process for comparing CNLs. Furthermore, the literature did not show much evaluations for different languages other than English.
- Grammatical Framework, (GF) appears to be gaining momentum in the CNL research community. It is possible that GF, may take on the role of a general architecture for developing controlled languages. Furthermore, research within the CNL community is turning its attention towards multilingual controlled languages, with recent efforts to generate ACE, using GF, for several European languages.
- There has been an increasing tendency towards conducting proper user evaluation for CNLs. While some CNL researchers have conducted task based evaluations, there have been less comparative evaluations across tools. In general, the CNL community should invest more in conducting strong user evaluations and not to lose track of the end goal—the creation of more user friendly ontology editing interfaces.
- A major question is whether a CNL is appropriate for the task? Although, in the context of ontology authoring, CNLs like CLOnE and ACE offer an attractive alternative to ontology editors, we argue that a CNL is not a panacea for formal knowledge engineering. We argue that for these scenarios, there should be a pre-existing use case for a *human orientated* CNL, in other words a restricted vocabulary or syntax for a technical domain either legal, clinical or aeronautics such as ASD Simplified Technical English.[11] Without such a use case (despite it

[11] http://www.asd-ste100.org/.

being possible to adapt a human-orientated CNL to a machine processable CNL), there would be little incentive for users to interact with it. Factors to be taken into account when designing CNLs include, the knowledge creation task complexity, target user (specialist or non expert), the domain (open or specific), available corpora, sample texts, pre-existing language resources or vocabularies, ontologies, multilingualism, requirements for language generation capabilities, and finally, availability of an NLP engineer or computational linguist for development of general purpose CNLs.

– Other issues include whether to adopt a shallow or deeper NLP approach? CLOnE and RABBIT (Hart et al. 2008) are based on a suite of shallow linguistic analysis tools while Grammatical Framework (GF) and Attempto Controlled English (ACE) are more lexicalised. Furthermore, they are both more powerful with respect to knowledge modelling. Both GF and ACE are bidirectional, which is extremely useful for surface realisation. In addition, GF, which is based on the functional language paradigm, has an exhaustive bank of application grammars for multiple languages. ACE on the other hand is "logic-based" and has built-in discourse representation structures which are "unification-based". However, both RABBIT and CLOnE, respectively, as GATE applications, have a number of semantic web and linked data processing resources available as GATE resources (Cunningham et al. 2002). In summary, deciding on what CNL or tools to use depends very much on the complexity of both the knowledge creation task and the language modelling task of the CNL as well as the target knowledge representation language and whether there is a need to reuse existing ontologies or vocabularies.
– As research into CNLs has been invigorated to a certain degree by the semantic web initiative, semantic web researchers with an interest in CNLs, should observe lessons learned by previous work in designing CNLs. Corpus analysis and empirical approaches should be a necessary step when designing a CNL (Grover et al. 2000).

**Acknowledgments** This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under Grant Number SFI/12/RC/2289.

## References

Adriaens, G., & Schreors, D. (1992). From cogram to alcogram: Toward a controlled English grammar checker. In *Proceedings of the 14th Conference on Computational Linguistics* (pp. 595–601). Morristown, NJ, USA: Association for Computational Linguistics.

Angelov, K., & Ranta, A. (2009). Implementing controlled languages in GF. In *CNL* (pp. 82–101).

Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web. *Scientific American*, *284*(5), 34–43.

Bernstein, A., & Kaufmann, E. (2006). GINO—A guided input natural language ontology editor. In *5th International Semantic Web Conference (ISWC2006)*.

Bernstein, A., Kaufmann, E., Fuchs, N., & Bonin, J. (2004). Talking to the semantic web: A controlled English query interface for ontologies. In *14th Workshop on Information Technology and Systems* (pp. 212–217).

Bernstein, A., Kaufmann, E., Kaiser, C., & Kiefer, C. (2006). Ginseng: A guided input natural language search engine for querying ontologies. In *2006 Jena User Conference*.

Brooke, J. (1996). SUS: A "quick and dirty" usability scale. In P. Jordan, B. Thomas, B. Weerdmeester, & A. McClelland (Eds.), *Usability evaluation in industry*. London: Taylor and Francis.

Canedo, L., Fuchs, N. E., Kaljurand, K., Koponen, M., Kuhn, T., Rautio, J., et al. (2013) Deliverable D11.3. Evaluations of ACE-in-GF and of AceWiki-GF. Technical report, MOLTO project, May 2013. http://www.molto-project.eu/biblio/deliverable/evaluations-ace-gf-and-acewiki-gf.

Cregan, A., Schwitter, R., & Meyer, T. (2007). Sydney OWL syntax towards a controlled natural language syntax for OWL 1.1. In *Proceedings OWLED 2007* (p. 10). Innsbruck.

Cunningham, H. (2002). GATE: A general architecture for text engineering. *Computers and the Humanities*, *36*, 223–254.

Cunningham, H., Maynard, D., Bontcheva, K., & Tablan, V. (2002). GATE: A framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*.

Dannélls, D. (2008). Generating tailored texts for museum exhibits. In *Proceedings of the 6th edition of LREC 2008, Workshop on Language Technology for Cultural Heritage Data (LaTeCH), Marrakech, Morocco* (pp. 17–20).

Dannélls, D., Damova, M., Enache, R., & Chechev, M. (2012). Multilingual online generation from semantic web ontologies. In *Proceedings of the 21st International Conference Companion on World Wide Web* (pp. 239–242). ACM.

Davis, B., Iqbal, A., Funk, A., Tablan, V., Bontcheva, K., Cunningham, H., et al. (2008). Roundtrip ontology authoring. In A. P. Sheth, S. Staab, M. Dean, M. Paolucci, D. Maynard, T. W. Finin, et al. (Eds.), *International Semantic Web Conference, volume 5318 of Lecture Notes in Computer Science* (pp. 50–65). Springer.

Denaux, R., Dimitrova, V., Cohn, A. G. (2013). Interacting with ontologies and linked data through controlled natural languages and dialogues. In *Do-Form: Enabling Domain Experts to Use Formalised Reasoning @ AISB, Exeter*.

Denaux, R., Thakker, D., Dimitrova, V., & Cohn, A. G. (2012). *Interactive semantic feedback for intuitive ontology authoring*. In *7th International Conference on Formal Ontology in Information Systems, Graz*.

Dimitrova, V., Denaux, R., Hart, G., Dolbear, C., Holt, I., & Cohn, A. (2008). Involving domain experts in authoring OWL ontologies. In *Proceedings of the 7th International Semantic Web Conference (ISWC 2008), Karlsruhe, Germany*. Springer.

Engelbrecht, P., Hart, G., & Dolbear, C. (2009). Talking rabbit: A user evaluation of sentence production. In N. Fuchs (Ed.), *Controlled Natural Language volume 5972 of Lecture Notes in Computer Science*(pp. 56–64) Berlin, Heidelberg: Springer.

España-Bonet, C., Enach, R., Slaski, A., Ranta, A., Marquez, L., & Gonzalez, M. (2011). Patent translation within the molto project. In *Workshop on Patent Translation, MT Summit XIII* (pp. 70–78).

Ferre, S. (2014). SQUALL: The expressiveness of SPARQL 1.1 made available as a controlled natural language. *Data and Knowledge Engineering*, *94*, 163–188.

Fuchs, N. E., & Schwertel, U. (2003). Reasoning in attempto controlled english. In F. Bry, N. Henze & J. Małuszyński (Eds.), *Principles and practice of semantic web reasoning, Lecture Notes in Computer Science* (Vol 2901, pp. 174–188). Springer.

Fuchs, N., & Schwitter, R. (1996a). Attempto controlled English (ACE). In *CLAW96: Proceedings of the First International Workshop on Controlled Language Applications, Leuven, Belgium*.

Fuchs, N., & Schwitter, R. (1996b). Attempto controlled English (ACE). See citeseer.ist.psu.edu/article/fuchs96attempto.html.

Fuchs, N. E., & Schwitter, R. (2007). Web-annotations for humans and machines. In *Proceedings of the 4th European Semantic Web Conference, Lecture Notes in Computer Science* (pp. 458–472). Berlin: Springer.

Funk, A., Tablan, V., Bontcheva, K., Cunningham, H., Davis, B., & Handschuh, S. (2007). Clone: Controlled language for ontology editing. In *ISWC/ASWC* (pp. 142–155).

Grosof, N., Horrocks, I., Volz, R., & Decker, S. (2003). Description logic programs: Combining logic programs with description logic. In *Proceedings of the Twelfth International World Wide Web Conference (WWW 2003)* (pp. 48–57).

Grover, C., Holt, A., Klein, E., & Moens, M. (2000). Designing a controlled language for interactive model checking. In *Proceedings of CLAW 2000* (pp. 29–30). Seattle, WA.

Hallett, C., Scott, D., & Power, R. (2007). Composing questions through conceptual authoring. *Computational Linguistics*, *33*(1), 105–133.

Hart, G., Johnson, M., & Dolbear, C. (2008). Rabbit: Developing a control natural language for authoring ontologies. In *5th European Semantic Web Conference (ESWC2008)* (pp. 348–360).

Hielkema, F., Mellish, C., & Edwards, P. (2008). Evaluating an ontology-driven wysiwym interface. In M. White, C. Nakatsu, & D. McDonald (Eds.), *INLG*. The Association for Computer Linguistics.

Holger, K., Ferguson, R. W., Noy, N. F., & Musen, M. A. (2004). The protege OWL plugin: An open development environment for semantic web applications. In *3rd International Semantic Web Conference—ISWC 2004, Hiroshima, Japan*.

Kaljurand, K. (2008). ACE view—An ontology and rule editor based on attempto controlled English. In *5th OWL Experiences and Directions Workshop (OWLED 2008), Karlsruhe, Germany*.

Kaljurand, K., & Fuchs, N. (2006). Bidirectional mapping between OWL DL and attempto controlled English. In *Fourth Workshop on Principles and Practice of Semantic Web Reasoning, Budva, Montenegro*.

Kaljurand, K., & Kuhn, T. (2013). A multilingual semantic wiki based on attempto controlled English and grammatical framework. In P. Cimiano, O. Corcho, V. Presutti, L. Hollink & S.Rudolph (Eds.), *The Semantic Web: Semantics and Big Data* (pp. 427–441). Berlin: Springer.

Kim, H.-G., Ha, B.-H., Lee, J.-I., & Kim, M.-K. (2005). A multi-layered application for the gross description using semantic web technology. *International Journal of Medical Informatics*, *74*(5), 399–407.

Kuhn, T. (2006). Attempto controlled English as ontology language. In F. Bry & U. Schwertel (Eds.), *REWERSE Annual Meeting*.

Kuhn, T. (2008). AceWiki: A natural and expressive semantic wiki. In *Semantic Web User Interaction at CHI 2008: Exploring HCI Challenges*.

Kuhn, T. (2008). AceWiki: A natural and expressive semantic Wiki. In *Proceedings of Semantic Web User Interaction at CHI 2008: Exploring HCI Challenges, CEUR Workshop Proceedings*.

Kuhn, T. (2010). *Controlled English for Knowledge Representation (to Appear)*. Ph.D. thesis, University of Zurich.

Kuhn, T. (2012). A principled approach to grammars for controlled natural languages and predictive editors. *Journal of Logic, Language and Information, 22*(1), 33–70.

Kuhn, T. (2013). The understandability of OWL statements in controlled English. *Semantic Web*, *4*(1), 101–115.

Kuhn, T. (2014). A survey and classification of controlled natural languages. *Computational Linguistics*, *40*(01), 121–170.

Kuhn, T., Barbano, P. E., Nagy, M. L., & Krauthammer, M. (2013). Broadening the scope of nanopublications. In *Proceedings of ESWC 2013* (pp. 487–501). Montpellier.

Kuhn, T., Royer, L., Fuchs, N., & Schroeder, M. (2006). Improving text mining with controlled natural language: A case study for protein interactions. In *Data Integration in the Life Sciences* (pp. 66–81). Springer.

Marchiori, M. (2004). Towards a people's web: Metalog. In *Proceedings of WI 2004* (pp. 320–326). Beijing.

Matthew, H., Drummond, N., Goodwin, J., Rector, A. L., Stevens, R., & Wang, H. (2006). The Manchester OWL syntax. In *Proceedings of OWLED 06* (pp. 10). Athens, GA.

Montague, R. (1970). Universal grammar. *Theoria*, *36*, 373–398.

Nielsen, J. (2006). Quantitative Studies: How Many Users to Test? Jakob Nielsen's, Alertbox. http://www.useit.com/alertbox/quantitative_testing.html. 26 June 2006.

O'Brien, S. (2003). Controlling controlled english an analysis of several controlled language rule sets. *Proceedings of EAMT-CLAW*, *3*, 105–114.

Power, R. (2012). Owl simplified English: A finite-state language for ontology editing. In *CNL* (pp. 44–60).

Power, R., Scott, D., & Evans, R. (1998). What you see is what you meant: Direct knowledge editings with natural language feedback. In H. Prade (Ed.), *13th European Conference on Artificial Intelligence (ECAI'98)* (pp. 677–681). Chichester, England: John Wiley and Sons.

Ranta, A. (2004). Grammatical framework: A type-theoretical grammar formalism. *Journal of Functional Programming*, *14*(02), 145–189.

Rector, A., Zanstra, P., & Solomon, W. (1995). The GALEN consortium, GALEN: Terminology services for clinical information systems. In M. Laires, M. Ladeira & J. Christensen (Eds.), *Health in the New Communication Age: Health care telematics for the 21st Century* (pp. 90–100). IOS Press, Amsterdam.

Safwat, H., & Brian, D. (2014). A brief state of the art of CNLs for ontology authoring. Fourth Workshop on Controlled Natural Language (CNL14). Springer LNAI (vol. 8625, pp. 190-200). 20–22 Aug, Galway, Ireland.

Schwitter, R. (2002). English as a formal specification language. In *Proceedings of the 13th International Workshop on Database and Expert Systems Applications, DEXA '02* (pp. 228–232). Washington, DC, USA: IEEE Computer Society.

Schwitter, R. (2007). Controlled natural languages. Technical report, Centre for Language Technology, Macquarie University.

Schwitter, R., & Tilbrook, M. (2004). Controlled natural language meets the semantic web. In *Proceedings of the Australasian Language Technology Workshop 2004* (pp. 55–62). Sydney, Australia.

Shiffman, R. N., Michel, G., Krauthammer, M., Fuchs, N., Kaljurand, K., & Kuhn, T. (2010). Writing clinical practice guidelines in controlled natural language. In N. E. Fuchs (Ed.), *Proceedings of the Workshop on Controlled Natural Language (CNL 2009) volume 5972 of Lecture Notes in Computer Science* (pp. 265–280). Berlin/Heidelberg, Germany: Springer.

Skuce, D. (2003). A Controlled Language for Knowledge Formulation on the Semantic Web. http://www.site.uottawa.ca:4321/factguru2.pdf.

Smart, P. (2008). Controlled natural languages and the semantic web. Technical report, School of Electronics and Computer Science, University of Southampton, Southampton, England.

Smith, M. K., Welty, C., & Mc Guinness, D. L. (2004). OWL Web Ontology Language Guide. W3C Recommendation. http://www.w3.org/TR/owl-guide/.

Tang, L. R., Mooney, R. J. (2001). Using multiple clause constructors in inductive logic programming for semantic parsing. In *ECML-2001* (pp. 466–477). Freiburg, Germany.

Valencia-Garca, R., Garca-Sanchez, F., Castellanos-Nieves, D., & Fernandez-Breis, J. T. (2011). OWLPath: An OWL ontology-guided query editor. *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, *41*(1), 121–136.

Van Kleek, M., Moore, B., Karger, D., Andre, P., & Schraefel, M. C. (2010). Atomate it! End-user context-sensitive automation using heterogeneous information sources on the Web. In *Proceedings of WWW 2010* (pp. 951–960). Raleigh, NC.