Published on Multilingual Online Translation (http://www.molto-project.eu)

D11.2 Multilingual semantic wiki

Contract No.:	FP7-ICT-247914
Project full title:	MOLTO — Multilingual Online Translation
Deliverable:	D11.2 Multilingual semantic wiki
Security (distribution level):	Public
Contractual date of delivery:	M23
Actual date of delivery:	2013-01-21
Type:	Prototype
Status version:	v1.0
Author(s):	Norbert E. Fuchs, Kaarel Kaljurand, Tobias Kuhn
Task responsible:	UZH
Other contributors:	

Abstract

This report describes our work to extend the existing semantic wiki engine AceWiki — which is based on the controlled natural language (CNL) ACE — with multilinguality features. In our approach, the underlying multilingual CNL grammar is implemented in Grammatical Framework (GF). The grammar facilitates precise automatic translation between different natural languages defined by the grammar, making the wiki content multilingual. The underlying grammar itself is integrated into the wiki and can be collaboratively edited. We discuss the current implementation of the system and its use cases.

Contents

1 Introduction

A wiki is a user-friendly collaborative environment for building knowledge bases in natural language. The most well-known example is Wikipedia, an encyclopedia that is being built by ∼100,000 people in hundreds of different languages. Numerous wikis have also been built for smaller domains. Semantic wikis [BSVW12] combine the main properties of wikis (ease of use, read-write, collaboration, linking) with knowledge engineering technology (structured content, knowledge models in the form of ontologies, automatic reasoning). Semantic wiki editors simultaneously work with the natural language content and its underlying formal semantics representation. The resulting wiki offers more powerful content management functions, e.g. dynamically created pages based on semantic queries and detection of semantic errors in the content, but has to somehow meet the challenge of keeping the user interface as simple as expected from wikis. The existing semantic wiki engines (e.g. Semantic Mediawiki[1], Freebase[2]) support the inclusion of semantics in the form of RDF-like subject-predicate-object triples, e.g. typed wikilinks (predicates) between two articles (the subject and the object).

Our approach to semantic wikis is based on controlled natural language (CNL). A CNL is a restricted version of a natural language. CNLs like Attempto Controlled English (ACE) [FKK08] have a precisely defined syntax, a formal (executable) semantics, and come with end-user documentation describing the syntax, semantics and usage patterns. CNLs and their editing tools support the creation of texts that are natural yet semantically precise, and can thus function well in human-machine communication. CNL-based wikis — such as AceWiki [Kuh08], on which our approach is based — can offer greater semantic expressivity compared to traditional semantic wikis (e.g. OWL instead of RDF), but their user interface does not become much more complex because it remains based on natural language and is therefore inherently easier to work with.

In this report we describe our work to extend the existing monolingual AceWiki with multilinguality features. In this approach, the underlying controlled natural language grammar is implemented in Grammatical Framework (GF) [Ran11]. The grammar facilitates a precise automatic translation between the concrete languages defined by the grammar, providing a multilingual interface to the wiki content. As an additional feature, the underlying multilingual grammar is integrated into the wiki itself and can be collaboratively edited. In our main use case, users speaking different languages collaboratively work on the same formal knowledge base, each in a CNL based on his/her own language. New wiki content is immediately available in all languages, and therefore all other users can immediately see it (in their language) and can edit or contribute on top of it. Our work demonstrates the combination of the existing technologies of ACE and GF, and is implemented by extending the existing ACE-based semantic wiki engine AceWiki with support to multilinguality and collaborative GF grammar editing. The main goal of this work is to explore natural language grammar based semantic wikis in the multilingual setting.

This report is structured as follows: in section 2 we review related work; in section 3 we introduce the core features of the existing tools and technologies employed in the rest of the paper (namely ACE, GF and AceWiki); in section 4 we discuss the main features of our wiki system; in section 5 we discuss a multilingual ACE-based wiki; in section 6 we list other potential uses cases; in section 7 we discuss the current implementation of the wiki engine; in section 8 we summarize our main results and outline future work.

2 Related work

The main CNL-based wiki that we are aware of is AceWiki, which is also the basis of our work and will be discussed below. In terms of multilinguality, our wiki system has some similarities with the OWL ontology editor described in [BSS+09] which allows users to view the ontology in three CNLs, two based on English and one on Chinese. [GRS12] describes the MoKi semantic wiki engine which offers a "lightly-structured access mode" for its structured content (OWL). In this mode the content is displayed as an uneditable ACE text, editing is supported for the simpler *isA* and *partOf* statements using templates that combine CNL

[1] http://semantic-mediawiki.org/
[2] http://www.freebase.com/

with HTML-forms, or using a native OWL syntax. As the main difference compared to these systems, our system uses the CNLs as the only user interface for both editing and viewing.

The research on GF has not yet focused on a wiki-like tool built on top of a GF-implemented grammar or application. Tool support exists mostly for users constructing single sentences (not texts) and working alone (not in collaboration). A notable exception is [MMB08], which investigates using GF in a multilingual wiki context, to write restaurant reviews on the abstract language-independent level by constructing GF abstract trees.

Even though the mainstream wiki engines generally allow for the wiki articles to be written in multiple languages, these different language versions exist independently of each other and only article-level granularity is offered by the system for interlinking the multilingual content. Some recent work targets that problem though, e.g. the EU project CoSyne[3] develops a technology for the multilingual content synchronization in wikis by using machine translation.

Some collaborative environments host editable "source code" which automatically generates some part of the content. Although not CNL-based these systems share with us the idea of an underlying editable grammar. Adessowiki [LMKR09] is a collaborative environment that carries simultaneously documentation, programming code and the code execution results (figures, numerical tables). Many of the envisioned use cases are similar to ours.

3 Background technologies

3.1 Attempto Controlled English

Attempto Controlled English (ACE) [FKK08] is a general purpose language based on first-order logic with English syntax, i.e. ACE can be viewed as both a natural language understandable to every English speaker, as well as a formal language with a precisely defined syntax and semantics understandable to automatic theorem proving software. ACE offers many language constructs, the most important of which are countable and mass nouns (e.g. 'man', 'water'); proper names ('John'); generalized quantifiers ('at least 2'); indefinite pronouns ('somebody'); intransitive, transitive and ditransitive verbs ('sleep', 'like', 'give'); negation, conjunction and disjunction of noun phrases, verb phrases, relative clauses and sentences; and anaphoric references to noun phrases through definite noun phrases, pronouns, and variables. Texts built from these units are deterministically interpreted via Discourse Representation Structures (DRS) [KR93] which can be further mapped to formats supported by existing automatic reasoners (e.g. OWL, SWRL, FOL, TPTP). The ACE sentence structures and their unambiguous interpretation is explained in the end-user documentation of *construction* and *interpretation* rules.

The grammar of ACE and its mapping to DRS is fixed and cannot be changed, but users can customize ACE in their applications by specifying a content word lexicon of nouns, verbs, adjectives, adverbs and prepositions and their mapping to logical atoms.

While originally designed for software specifications, in the recent years ACE has been developed with the languages and applications of the Semantic Web in mind. [Kal07] describes ACE fragments suitable for mapping to and from languages like OWL, SWRL, DL-Query. ACE View [Kal08] and AceWiki are ACE-based tools for building OWL ontologies.

3.2 Grammatical Framework

Grammatical Framework (GF) [Ran11] is a functional programming language for building multilingual grammar applications. Every GF program consists of an *abstract syntax* (a set of functions and their categories) and a set of one or more *concrete syntaxes* which describe how the abstract functions and categories are linearized (turned into surface strings) in each respective concrete language. The resulting grammar describes a mapping between concrete language strings and their corresponding abstract trees (structures of function names). This mapping is bidirectional — strings can be *parsed* to trees, and trees *linearized* to strings. As an abstract syntax can have multiple corresponding concrete syntaxes, the respective languages

[3] `http://www.cosyne.eu/`

can be automatically *translated* from one to the other by first parsing a string into a tree and then linearizing the obtained tree into a new string.

While GF can be used to build parsers and generators for formal languages, it is optimized to handle natural language features like morphological variation, agreement, and long-distance dependencies. Additionally, the GF infrastructure provides a *resource grammar library* (RGL), a reusable grammar library of the main syntactic structures and morphological paradigms currently covering about 30 natural languages [Ran09]. As the library is accessible via a language-independent API, building multilingual applications remains simple even if the programmers lack detailed knowledge of the linguistic aspects of the involved languages. These features make GF a good framework for the implementation of CNLs, especially in the multilingual setting [RED12]. The development of GF has focused on parsing tools, grammar editors, and extending the grammar library to new languages.

3.3 AceWiki

AceWiki[4] [Kuh10] is a CNL-based semantic wiki engine. It uses ACE as the user interface language and OWL as its underlying semantic framework for the integration of its main reasoning tasks, i.e. consistency checking, classification and query answering.

Because the standard OWL syntaxes are largely impenetrable to a general wiki user, AceWiki offers an intuitive front-end in the form of a controlled natural language. The wiki articles are edited entirely in ACE and then automatically mapped to OWL in the background, without the user ever having to rely on explicit knowledge of OWL. This is an important difference with respect to other semantic wikis where the user must work in two different languages: the free-form natural language and the semantics modeling language.

The content of an AceWiki instance is written in a subset of ACE that is formally defined in a grammar notation called Codeco [Kuh13a]. The grammar targets an OWL-compatible fragment of ACE, i.e. ACE sentences that are semantically outside of the OWL expressivity cannot be expressed in the wiki. This guarantees that the complete AceWiki content can be automatically translated to OWL in the background. Additionally, the grammar is used to drive a look-ahead editor [SLH03] which guides the input of a new sentence by proposing only syntactically legal continuations of the sentence.

The content is structured into a set of articles, each article containing a sequence of entries. Each entry is either a declarative sentence, interrogative sentence (question), or an informal comment. The declarative sentences assert new information into the knowledge base and therefore influence the consistency and entailments of the knowledge base, the questions provide an access point to the entailed knowledge. The comments are not restricted by the look-ahead editor nor checked by the semantic reasoner. They are plain text with possible wikilinks to other articles. Upon every change in the wiki, an OWL reasoner determines its effect, and possibly flags inconsistencies or updates the dynamically generated parts of the wiki (e.g. concept hierarchies and answers to questions). The content words (proper names, nouns, transitive verbs, relational nouns and transitive adjectives) in the wiki sentences map one-to-one (i.e. link) to wiki articles. Semantically, content words correspond to OWL entities: proper names to OWL individuals, nouns to OWL classes, and the relational words to OWL properties.

Even though the AceWiki implementation is still in its alpha stage, it can be considered mature for academic purposes. AceWiki's code base has been used to implement various tools, such as the ACE Editor [Kuh13a] and the corpus query interface Coral [KH12].

In previous work, two usability experiments have been performed on AceWiki with altogether 26 participants [Kuh09]. The results showed that AceWiki and its editor are easy to learn and use. Another study confirmed that writing ACE sentences with the editor used in AceWiki is easier and faster than writing other formal languages [KH12]. It has also been demonstrated that ACE is more effective than the OWL Manchester Syntax in terms of understandability [Kuh13b].

From the viewpoint of a general CNL-based semantic wiki, the AceWiki user is restricted to using a single CNL with an unchangeable and unambiguous grammar and a predetermined reasoning mechanism.

[4] `http://attempto.ifi.uzh.ch/acewiki/`

4 AceWiki-GF

Our multilingual GF-implemented semantic wiki has been realized as an extension of AceWiki. We will refer to this extension here as AceWiki-GF and use "standard AceWiki" or simply "AceWiki" to refer to the AceWiki implementation before the work on the GF-based extension started.

Extending AceWiki has allowed us to reuse its infrastructure (look-ahead editor component, access to OWL reasoners and the presentation of reasoning results, document navigation system, etc.). In the following we only describe the main differences between AceWiki and AceWiki-GF. (See section 3.3 for the general discussion of the AceWiki engine.) Also, in the following we focus on the unique aspects that our proposed wiki system must manage, such as multilinguality and formal grammars, and we do not discuss the standard wiki features that make the collaborative editing possible such as "talk pages", community rules, revision history, user identification and access rights, reward system, etc. We consider such features general enough that they apply without major changes to our wiki system. This section discusses the main features and general goals of AceWiki-GF, while more details about its implementation are provided in section 7.

4.1 General principles and user interface

The standard AceWiki principles have been largely preserved in the extension. The wiki still follows the main principle of CNL-based wikis, i.e. that formal notations are hidden. In this case the user does not see the GF trees which actually define the content but only interacts with the natural language sentences. (Experienced users can still look at the GF parser output which provides information on syntax trees, translation alignment diagrams, etc.) This general principle is challenged somewhat if we allow grammar editing (or even simply viewing) in the wiki as this typically assumes more technical knowledge. Also, user understanding of the various forms of ambiguities induced by the underlying grammar can pose a problem for the end user. We are currently investigating how ambiguity is best presented to users and how we can help them to resolve it.

The user interface of AceWiki has largely been preserved, the main addition is an option for the user to set the content language (see the left sidebar of figure 1). Again, the possibility for grammar editing requires a new kind of user interface.

4.2 Structure and linking

In general, AceWiki-GF follows the AceWiki structure — the wiki is a set of interlinked articles, each article containing a sequence of sentences. New is the fact that also the grammar definition can be part of the wiki and can be referenced from the articles using wikilinks.

GF grammars are naturally representable as sets of wiki articles. Each grammar module can be stored as an article and linked to the articles of the modules that it imports. Furthermore, grammar modules have internal structure — sets of categories and functions (which reference categories) — which can be linked to wiki content because the content is represented as a set of trees (i.e. structures of function names). One of the benefits of having a grammar definition as part of the CNL-based wiki is that it provides an integrated documentation of the language that the wiki users are required to use.

Note that a grammar can also reference modules which are part of the general RGL and are thus not editable and also not part of the wiki. These modules can be made accessible via external links, e.g. to the online RGL browser[5].

4.3 Sentence editing

The user interface for adding and modifying wiki entries is the same as in AceWiki, i.e. based on sentences and supporting the completion of a syntactically correct sentence by displaying a list of syntactically legal words that can follow the partially completed sentence. The language of the sentence depends on the chosen wiki language. In case a successful entry is ambiguous (i.e. parsing results in multiple trees) then

[5] `http://www.grammaticalframework.org/lib/doc/browse/`

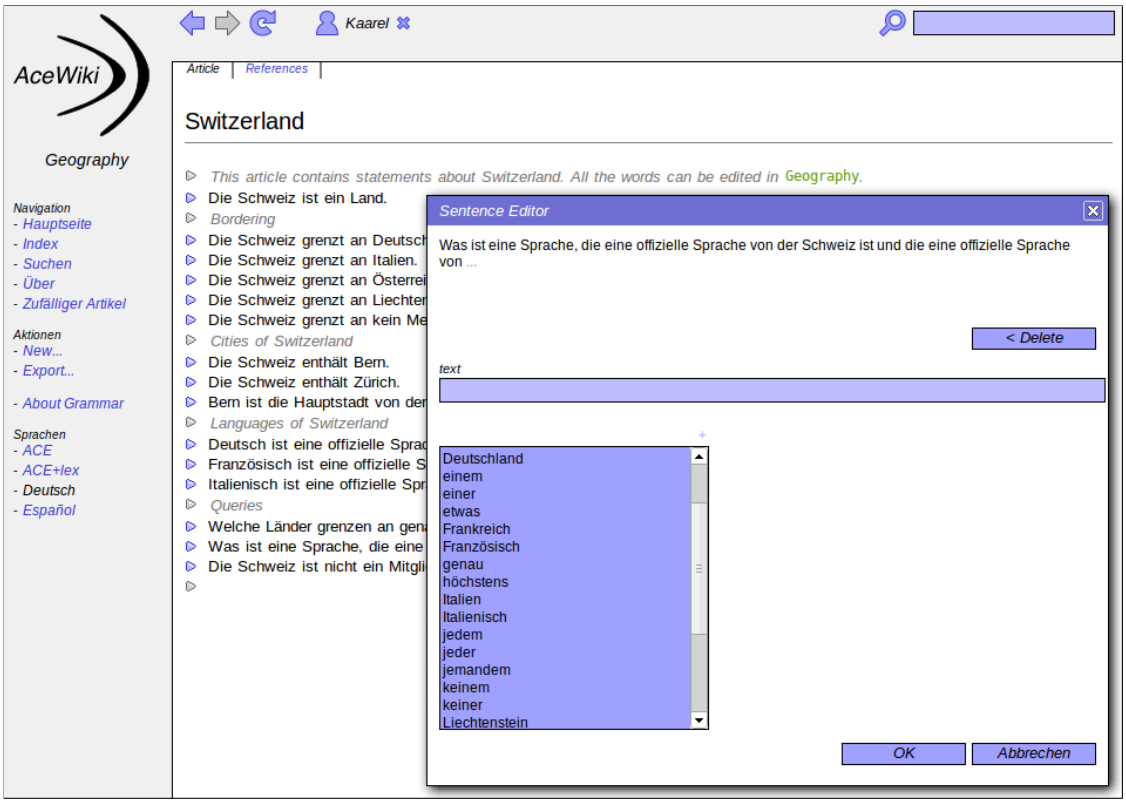

Figure 1: Multilingual geography article about Switzerland displayed in German. The user can change the language of the wiki (both the content and the user interface labels) using the language switching menu in the left sidebar. Otherwise the user interface is identical to the AceWiki user interface with the familiar look-ahead editor that helps users to input syntactically controlled sentences.

the ambiguity is preserved. If viewed in another language, multiple different sentences can then occur as linearizations of the ambiguity. This allows wiki users who work via the other language to resolve the ambiguity. A monolingual way to deal with ambiguity is to implement for every concrete syntax an additional "disambiguation syntax" [RED12] that overrides the linearizations of the ambiguous constructs to have an unambiguous, although possibly a more formal-looking notation. This syntax could be used to display the entry in the case of ambiguity.

We note that some syntax-aware editors, e.g. the GF Syntax Editor[6] or the OWL Simplified English editor [Pow12] operate more on the abstract tree level and thus avoid the problem of ambiguous entries. These approaches also simplify smaller edits e.g. replacing a word in the beginning of the sentence. The fact that they abstract away from linguistic details like case and gender might make them preferable for users with only basic knowledge of the underlying language. It is therefore worth exploring these approaches as a complementary way to view and edit CNL statements in AceWiki-GF.

4.4 Lexicon and grammar editing

The standard AceWiki supports a single grammar and allows users to control only the set of (monolingual) content words. As a multilingual system with multiple editable grammars, AceWiki-GF needs to take a more general approach.

In AceWiki-GF, the controlled languages of each wiki are governed by a formal grammar. Examples of grammars are general purpose CNL grammars like ACE (ported to other natural languages) and grammars

[6] http://cloud.grammaticalframework.org/syntax-editor/editor.html

for specific domains, e.g. grammars for describing museum objects, mathematical expressions, or tourist phrases.

The wiki integrates the grammar definition allowing users to learn about the languages that it defines and even change it to extend/modify the languages. In principle, the grammar can be changed in multiple ways (translating an existing word, adding a new word, changing the phrase structure) each requiring a different grammar engineering experience from the users.

The AceWiki-GF extension has removed the ACE-specific lexical editor, and replaced it with a GF source editor. While general and powerful, this approach cannot be used to target all the above mentioned usage scenarios in a user-friendly way. For the case of a simple lexical editing, the editor user interface must focus on just the words and their forms (in multiple languages), ideally filling in automatically all the forms based on a few input forms using the so called smart paradigms [Ran09].

4.5 Multilinguality

In a wiki, language serves different purposes: expressing the wiki content, both user-edited and dynamically created; localizing the user interface (button labels etc.); etc. The languages fall into two types — natural (e.g. English) and formal (e.g. first-order logic). While natural language appears in the user interface via which users edit and view the content, the formal languages are typically used in the background as input to formal reasoners to reason about the content or automatically generate new content.

The content of AceWiki-GF articles has a language-neutral internal representation and can be presented in any of the languages that the underlying grammar supports, depending on the user's preference. In the case of a concrete wiki instance, a smaller number of languages might be initially preferred and more languages could be gradually added, providing translations of the wiki content. In addition to the wiki reader and the wiki editor, there is now a third role for a wiki user, namely the translator. The main task of the translators is to translate the existing words and to check if the automatically generated sentence translations are accurate with respect to the domain.

5 Multilingual ACE

In our main use case, a number of people speaking different languages are supposed to build and agree on a formal knowledge base. They do not share a common language and do not have enough technical knowledge to directly work with ontology languages such as OWL. For this situation, we want to provide a tool with which each user can contribute to the knowledge base in a restricted version of his/her own language, which gets immediately translated into a formal representation as well as a number of other natural languages at a high quality. In a sense, the different CNLs act as different lenses to see the same formal content of the knowledge base. This minimizes misunderstandings and feedback loops, as all users can at all times see the complete knowledge base in their language.

To offer the functionality of the standard AceWiki combined with the multilingual features made possible by GF, we reimplemented the ACE grammar in GF and ported it via the RGL API to multiple natural language fragments. (See the ACE-in-GF website[7] and the deliverable D11.1 [CFK12] for more details of this work.) With this grammar, the content of a wiki can be currently made available in 15 languages, which form a subset of the RGL that has been tested in the context of the multilingual ACE grammar. Most of these languages provide full coverage of the syntactic structures of ACE, for some languages a few structures (e.g. verb phrase coordination, some forms of questions) have not been implemented yet. Also, while the concrete syntax is designed to be unambiguous for ACE, i.e. every supported sentence generates just a single abstract tree, the grammar in general does not guarantee this property for the other implemented languages. In some cases it seems to be better to let a user work with an ambiguous representation if it offers a simpler syntax and if the ambiguity can be always explained (e.g. via the ACE representation) or removed in the actual usage scenario (e.g. in a collaborative wiki environment).

[7] `http://github.com/Attempto/ACE-in-GF`

5.1 Lexicon and grammar editing

The developed ACE-in-GF grammar comes only with small multilingual test vocabularies. Therefore, users are expected to build up the wiki vocabulary using the grammar editing feature of the wiki engine and choosing a subset of the 15 languages which they want to support. An initial setup of the wiki makes the ACE grammar available as a set of grammar modules falling into the following categories:

- ACE resource grammar (about 30 modules which are typically identical to their English resource grammar counterparts, sometimes overriding certain structures);
- ACE application grammar which reflects the AceWiki subset of ACE (one module);
- instantiation of this grammar into each supported language with additional modules that describe language-specific overriding of some of the functions;
- empty content word lexicon module(s) for each language.

In order to add a new word to the wiki a line needs to be added to the lexicon wiki page, i.e. the page that corresponds to the lexicon module. Although editing the lexicon technically means editing the GF grammar, the lexicon module of the grammar is conceptually much simpler than the grammar in general and maps one-to-one to the respective ACE lexicon structure (for English). The structure of lexicons in all the supported languages is roughly the same even if some languages are morphologically more complex (e.g. have more case endings). The language-specific lexical structures are hidden from the user behind language-neutral categories like `N` and `V2` and constructed by functions like `mkN` and `mkV2` which are capable of guessing the full word paradigm on the basis of only one or two input forms. Thus, support for multilinguality does not increase the conceptual complexity of the wiki. However, a user who edits the lexicon must be familiar with the GF RGL Lexical Paradigms API[8].

Wiki users experienced in GF are also able to modify the full grammar, although we do not see many compelling use cases for that as ACE itself is pre-defined and thus changing its grammar should not be allowed (e.g. it would break the functioning of the mapping to OWL). Its verbalization to other languages, however, is sometimes a matter of taste, and could be therefore made changeable by the wiki users, e.g. users can add an alternative formulation of an ACE sentence in some language by using a GF variant. Also, the possibility to define arbitrary GF operators can make certain lexicon entry tasks more convenient.

5.2 Underlying semantic representation

In order to map the wiki content to first-order logic or OWL, each wiki entry is linearized in ACE and an external ACE parser (APE) is used to generate the DRS that corresponds to the ACE form. The grammar currently represents ACE by two concrete languages “Ace” and “Ape”, where the first corresponds to an ACE sentence assuming a suitable background lexicon, and the second includes the lexicon in the sentence string and thus explicitly provides all the necessary information to the ACE parser. The wiki user is viewing and editing only the “Ace”-formatted articles.

6 Other use cases

Our main use case for AceWiki-GF is about general knowledge engineering, as described above. Below, we present some additional use cases, which we might explore in more depth in the future.

6.1 Wikis based on other grammars

The grammar of AceWiki-GF is exchangeable and allows for GF grammars that have no mapping to a logic language like OWL. The focus of such a wiki would be on the multilinguality features, and not on formal reasoning and querying. For instance, such wikis could be used for applications such as a tourist phrase

[8] `http://www.grammaticalframework.org/lib/doc/synopsis.html`

book, a museum catalog, a technical manual, or a collection of mathematics exercises. To that aim, some grammars developed in the MOLTO project are described below that could be used within AceWiki-GF, namely the Phrasebook grammar[RED12] and the museum object descriptions grammar[DRE12].

6.1.1 Phrasebook

To serve as a phrasebook based on the existing Phrasebook grammar, a wiki should ideally have the following features that make it different from an ACE-based wiki:

- by definition a (tourist) phrasebook must contain at least two languages, and most users of the wiki would like to view the wiki content in at least two languages, possibly presented in parallel;
- the wiki should be structured more like a book, with chapters, sections and a table of contents, and there is less use for traditional wiki links;
- more sentences are likely to be ambiguous, and the wiki should effectively explain the ambiguities;
- the domain is simpler (i.e. it is not based on any formal logic), and therefore the users are more likely to want to contribute to the language that underlies the wiki.

6.1.2 Museum object descriptions

A possible wiki for describing museum objects would be highly structured, with many interlinked wiki pages for paintings, painters, etc. There would be need for a large number of external links, and possibly embedded multimedia content. This domain is also more formal and would benefit from automatic answering of user queries like "Which painter lived in Paris?", or "Which Dutch painter painted which French painter?".

6.2 Language design

Another possible use case for AceWiki-GF is collaborative language design. The wiki users could take e.g. the ACE grammar as the starting point and customize it for a specific domain, possibly changing some of its original features and design decisions. Alternatively, they could start by cloning an existing resource grammar and then port it to a different language.

In this case most of the activity would happen in the grammar pages and the wiki sentences would serve only as a way to test the currently effective grammar implementation.

7 Implementation

AceWiki is a web-based application implemented in Java and built with the Echo Web Framework[9]. AceWiki-GF extends AceWiki by integrating the GF toolset into the wiki engine. Both systems include the ACE parser APE[10] in order to map ACE sentences to the OWL form and provide other ACE-specific analysis (paraphrasing, syntax trees, etc.).

7.1 Differences from AceWiki

Because AceWiki is a monolingual engine, several modifications needed to be done to accommodate multilinguality, i.e. to support the wiki viewing/editing in multiple languages depending on the user's preference:

- the Codeco grammar/parser for ACE was replaced by the a GF parser using any GF grammar for parsing, linearization and other operations;

[9] `http://echo.nextapp.com/`
[10] `http://github.com/Attempto/APE`

- the English-specific lexicon editor was replaced by a simple GF source editor which can be used to edit any GF grammar modules, among them lexicon modules;
- the atomic wiki entry, which for the monolingual AceWiki was an ACE sentence, was changed to a GF abstract tree set. The new representation is language-neutral and can furthermore represent ambiguity;
- the notion of wiki article/page was extended to also include arbitrarily named pages (in AceWiki all pages are named by their corresponding OWL entity) and pages that represent editable grammar modules.

7.2 GF Webservice and GF-Java

AceWiki-GF communicates with the GF technology via the GF Webservice [BAR09] which provides the GF parsing and linearization (and the related look-ahead and translation) services for grammars in the PGF format [ABR10]. The GF Webservice has been recently extended to provide a GF Cloud Service API[11] which additionally allows for modifications to the grammar.

GF-Java[12] is a small Java library that AceWiki-GF uses to interact with the REST/JSON API of GF Webservice. The general goal of GF-Java is to

- provide a convenient tool for Java programmers for accessing the GF tools, e.g. by offering enumerated constants, checked exceptions, GF structures (e.g. trees) as Java objects, etc.;
- provide a general API to the GF tools, where the implementation can either be the GF Webservice or possibly JPGF[13] (which currently does not provide grammar editing);
- offer custom extensions to the GF tools, e.g. an iterator over all possible sentence completions with growing length.

GF-Java conforms to the GF Webservice REST API, i.e. it provides the same access methods and default parameter values.

7.3 Grammar integration

The wiki makes the grammar available as a set of interlinked grammar modules (e.g. abstract, concrete, resource). The users can link to the grammar modules from content articles and navigate the import hierarchy of the modules.

A simple GF source editor (figure 2) is currently provided for the users to update the grammar. It can be used to edit any grammar module that is available as part of the wiki. This excludes RGL modules which are only available as precompiled static modules as part of the GF Webservice. Upon update a grammar page is saved via the GF Webservice resulting in a compilation of a new underlying PGF file.

The user-friendliness of the current editor can be improved by integrating some of the features (e.g. syntax highlighting, auto-completion) of existing GF grammar editors (GF online editor for simple multilingual grammars[14]; the GF Eclipse Plugin[15]). More importantly, a dedicated front-end should be provided to the lexicon modules, which benefits from their simpler structure (where each function has a simple category and is often linearized using a well-defined RGL operator). Such an editor can offer a more structured (e.g. tabular) interface and can hide most of the complexities of GF source, see for example the "row" and "matrix" views of the GF online editor for simple multilingual grammars.

[11] http://cloud.grammaticalframework.org/gf-cloud-api.html
[12] https://github.com/Kaljurand/GF-Java
[13] https://github.com/GrammaticalFramework/JPGF
[14] http://cloud.grammaticalframework.org/gfse/
[15] http://www.grammaticalframework.org/eclipse/index.html

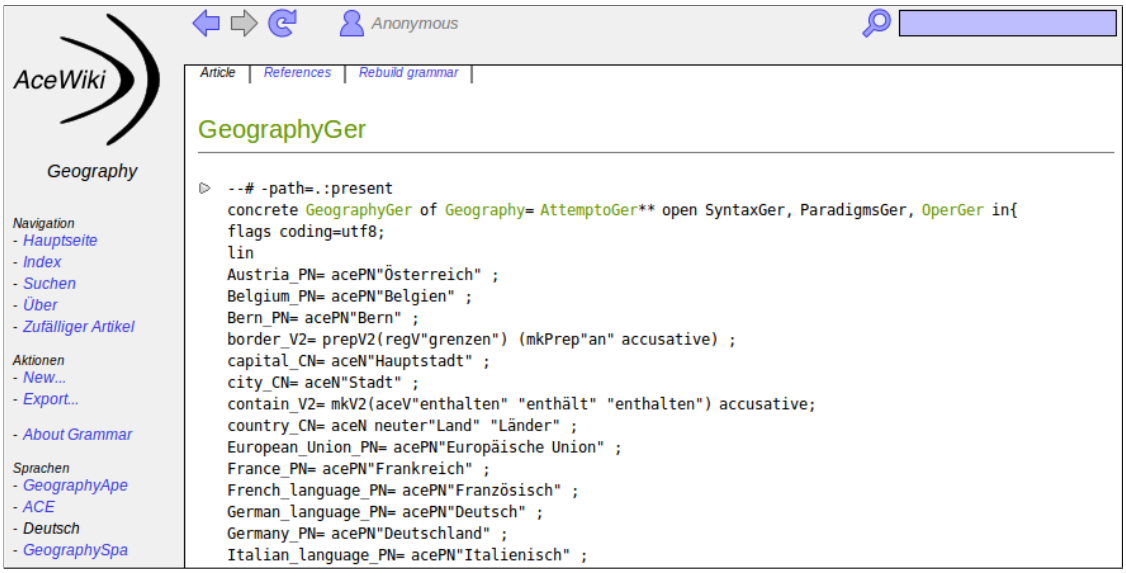

Figure 2: The lexicon can be edited in bulk using a simple GF source editor. In most cases the user can rely on GF smart paradigms, i.e. provide just the lemma form of the word for the automatic generation of the full lexicon entry. The lexicon of each language is in a separate module. The multilingual lexicon entries are aligned via the abstract functions.

7.4 Multilinguality

The standard AceWiki user interface (labels etc.) is based on English, and since the user is expected to know English anyway (to be able to effectively work with ACE) the localization of the user interface has not been a high priority. However, in the multilingual content setting the user interface must also be multilingual.

We have refactored the AceWiki codebase to simplify the localization of the AceWiki user interface. The language of the user interface is now determined by the user's locale setting, which in case of AceWiki-GF is guessed on the basis of the name of the concrete language (e.g. 'GeographyGer' is mapped to the German locale), unless the locale is explicitly specified in the grammar. We note that the GF technology could also be used to generate multilingual user interface labels as done in [MMB08].

The multilinguality of the content is achieved by keeping the wiki content as GF trees and presenting it in any of the languages that the grammar supports, depending on the preference of the user.

Currently, the names of the articles and the content of the free-form comments are not multilingual, i.e. they exist only in the language in which they were written.

7.5 Configuration and storage

The AceWiki-GF configuration file (see an example in appendix A) specifies for each wiki at compile time:

- the URL of the used GF webservice;
- the location of the PGF-formatted grammar, as a directory and filename in the webservice directory;
- the start category of the grammar (optional);
- the default concrete language;
- whether registration and login is required to view and/or edit the wiki;
- the local directory for the wiki data.

In addition to the configuration, the wiki is defined by a set of data files (which holds the tree structures of the wiki sentences and the GF source of the grammar modules) and the external GF webservice. The RGL modules are also part of the webservice, i.e. they are currently not stored as part of the wiki data files.

Every existing PGF grammar can be used as the basis of a wiki, assuming that it is (can be) made available via the GF Webservice. In order to edit the grammar in the wiki, its source code must be available. We have provided a set of Python scripts[16] that automate the setting up of a wiki based on an existing grammar.

7.6 Performance evaluation

We looked at the various performance properties of AceWiki-GF resulting from the introduction of GF tools, specifically tools that are accessible over HTTP and possibly running on a remote server. In our preliminary tests a demo wiki was running on the Attempto project server `http://attempto.ifi.uzh.ch` and using a GF webservice running on the GF server `http://cloud.grammaticalframework.org/`. The wiki was used by a single user.

The parsing (and look-ahead) performance is comparable to the standard AceWiki, although using the external GF webservice adds some delay. In case of the standard AceWiki the token predictor is implemented in Java and built into the wiki engine. In case of AceWiki-GF, every look-ahead (pressing of the tab-key in the look-ahead editor user interface) triggers a query to the GF server. We have tested the look-ahead feature with grammars that have a large number of terminals that can occur in the same syntactic position (e.g. approx. 5000 place names in an address grammar[17]) and found the performance satisfactory.

Multilingual viewing of a sentence or a set of sentences involves the linearization of their trees. Compared to parsing, linearization is a much faster operation in GF. Translating a complete article into another language (which happens only the first time an article is opened by any user, as at this point the linearizations are cached) is reasonably fast assuming a medium sized article. Similarly, translating a sentence into all the languages of the wiki is a fast operation.

Grammar editing (such as introducing a new word) triggers the recompilation of the grammar by the GF server. Recompilation of the grammar is currently relatively slow, the runtime depending on the amount of included languages and the complexity of their (resource) grammars. Any change to the grammar can invalidate some or all of the wiki content. Currently an automatic refresh is not triggered in the wiki because naively re-linearizing every tree would involve a serious performance penalty. Ideally the GF server should respond to every grammar update with a list of affected functions, categories and languages, so that only the affected parts of the wiki could be refreshed.

7.7 Availability

The current implementation of AceWiki-GF is publicly available on GitHub[18] and can be used via some demo wikis[19]. The demo wikis are based on a wide variety of GF grammars, including the grammars available via the existing GF demos (such as Minibar[20]) and an editable version of the multilingual ACE grammar.

8 Conclusion

The main contribution of this work is a system that combines two existing technologies: GF and AceWiki. With the help of GF, an existing monolingual semantic wiki (i.e. AceWiki) has been extended by multilinguality features. This means that writing an article in one language makes it immediately available in all other languages, with precise translations offered by GF.

[16] `https://github.com/Kaljurand/GF-Utils`
[17] `http://kaljurand.github.com/Grammars/`
[18] `http://github.com/AceWiki/AceWiki`
[19] `http://attempto.ifi.uzh.ch/acewiki-gf/`
[20] `http://cloud.grammaticalframework.org/minibar/minibar.html`

So far, our approach has not led to any major problems, and the resulting implementation is stable enough to serve as a prototype and as a basis for evaluation. We believe that the resulting user interface is intuitive and useful, and we will test these assumptions in user experiments within the next months.

The main focus of our future work is providing a user-friendly lexical editor as a user interface component of the wiki.

A Configuration example

Example of a *web.xml* configuration of a MOLTO Phrasebook wiki instance. The configuration specifies a start category for the wiki sentences, the URL of the GF webservice (in this case the public GF webservice `http://cloud.grammaticalframework.org` is used), and the default concrete language, among some technical details.

```
<servlet>
  <servlet-name>/grammars/Phrasebook.pgf</servlet-name>
  <servlet-class>ch.uzh.ifi.attempto.acewiki.BackendServlet</servlet-class>
  <load-on-startup>1</load-on-startup>
  <init-param>
    <param-name>engine_class</param-name>
    <param-value>ch.uzh.ifi.attempto.acewiki.gfservice.GFEngine</param-value>
  </init-param>
  <init-param>
    <param-name>pgf_name</param-name>
    <param-value>/grammars/Phrasebook.pgf</param-value>
  </init-param>
  <init-param>
    <param-name>start_cat</param-name>
    <param-value>Question</param-value>
  </init-param>
  <init-param>
    <param-name>service_uri</param-name>
    <param-value>http://cloud.grammaticalframework.org:80</param-value>
  </init-param>
  <init-param>
    <param-name>ontology</param-name>
    <param-value>grammars__Phrasebook</param-value>
  </init-param>
</servlet>
<servlet-mapping>
  <servlet-name>/grammars/Phrasebook.pgf</servlet-name>
  <url-pattern>/grammars/Phrasebook.pgf</url-pattern>
</servlet-mapping>

<servlet>
  <servlet-name>/grammars/Phrasebook.pgf/Phrasebook</servlet-name>
  <servlet-class>ch.uzh.ifi.attempto.acewiki.AceWikiServlet</servlet-class>
  <init-param>
    <param-name>backend</param-name>
    <param-value>/grammars/Phrasebook.pgf</param-value>
  </init-param>
  <init-param>
    <param-name>language</param-name>
    <param-value>PhrasebookEng</param-value>
  </init-param>
  <init-param>
    <param-name>title</param-name>
    <param-value>Phrasebook questions</param-value>
  </init-param>
</servlet>
<servlet-mapping>
  <servlet-name>/grammars/Phrasebook.pgf/Phrasebook</servlet-name>
  <url-pattern>/grammars/Phrasebook.pgf/Phrasebook/</url-pattern>
</servlet-mapping>
```

References

[ABR10] Krasimir Angelov, Björn Bringert, and Aarne Ranta. PGF: A Portable Run-time Format for Type-theoretical Grammars. *Journal of Logic, Language and Information*, 19(2):201–228, 2010. 10.1007/s10849-009-9112-y.

[BAR09] Björn Bringert, Krasimir Angelov, and Aarne Ranta. Grammatical Framework Web Service. In *Proceedings of EACL-2009*, 2009.

[BSS+09] Jie Bao, Paul R. Smart, Nigel Shadbolt, Dave Braines, and Gareth Jones. A Controlled Natural Language Interface for Semantic Media Wiki. In *3rd Annual Conference of the International Technology Alliance (ACITA'09)*, September 2009.

[BSVW12] F. Bry, S. Schaffert, D. Vrandečić, and K. Weiand. Semantic wikis: Approaches, applications, and perspectives. *Reasoning Web. Semantic Technologies for Advanced Query Answering*, pages 329–369, 2012.

[CFK12] John J. Camilleri, Norbert E. Fuchs, and Kaarel Kaljurand. Deliverable D11.1. ACE Grammar Library. Technical report, MOLTO project, June 2012. `http://www.molto-project.eu/biblio/deliverable/ace-grammar-library`.

[DRE12] Dana Dannélls, Aarne Ranta, and Ramona Enache. Deliverable D8.2. Multilingual grammar for museum object descriptions. Technical report, MOLTO project, March 2012. `http://www.molto-project.eu/biblio/deliverable/multilingual-grammar-museum-object-descriptions`.

[FKK08] Norbert E. Fuchs, Kaarel Kaljurand, and Tobias Kuhn. Attempto Controlled English for Knowledge Representation. In *Reasoning Web, 4th International Summer School 2008, Tutorial Lectures*, 2008.

[GRS12] Chiara Ghidini, Marco Rospocher, and Luciano Serafini. Modeling in a Wiki with MoKi: Reference Architecture, Implementation, and Usages. *International Journal On Advances in Life Sciences*, 4, 2012.

[Kal07] Kaarel Kaljurand. *Attempto Controlled English as a Semantic Web Language*. PhD thesis, Faculty of Mathematics and Computer Science, University of Tartu, 2007.

[Kal08] Kaarel Kaljurand. ACE View — an ontology and rule editor based on Attempto Controlled English. In *5th OWL Experiences and Directions Workshop (OWLED 2008)*, Karlsruhe, Germany, 26–27 October 2008. 12 pages.

[KH12] Tobias Kuhn and Stefan Höfler. Coral: Corpus access in controlled language. *Corpora*, 7(2):187–206, 2012.

[KR93] Hans Kamp and Uwe Reyle. *From Discourse to Logic. Introduction to Modeltheoretic Semantics of Natural Language, Formal Logic and Discourse Representation Theory*. Kluwer Academic Publishers, Dordrecht/Boston/London, 1993.

[Kuh08] Tobias Kuhn. AceWiki: A Natural and Expressive Semantic Wiki. In *Semantic Web User Interaction at CHI 2008: Exploring HCI Challenges*, 2008.

[Kuh09] Tobias Kuhn. How Controlled English can Improve Semantic Wikis. In Christoph Lange, Sebastian Schaffert, Hala Skaf-Molli, and Max Völkel, editors, *Proceedings of the Fourth Workshop on Semantic Wikis, European Semantic Web Conference 2009*, volume 464 of *CEUR Workshop Proceedings*. CEUR-WS, June 2009.

[Kuh10] Tobias Kuhn. *Controlled English for Knowledge Representation*. PhD thesis, Faculty of Economics, Business Administration and Information Technology of the University of Zurich, 2010.

[Kuh13a] Tobias Kuhn. A principled approach to grammars for controlled natural languages and predictive editors. *Journal of Logic, Language and Information*, 2013.

[Kuh13b] Tobias Kuhn. The understandability of OWL statements in controlled English. *Semantic Web*, 4(1):101–115, 2013.

[LMKR09] R.A. Lotufo, R.C. Machado, A. Körbes, and R.G. Ramos. Adessowiki on-line collaborative scientific programming platform. In *Proceedings of the 5th International Symposium on Wikis and Open Collaboration*, page 10. ACM, 2009.

[MMB08] Moisés S. Meza-Moreno and Björn Bringert. Interactive Multilingual Web Applications with Grammatical Framework. In Bengt Nordström and Aarne Ranta, editors, *Advances in Natural Language Processing, 6th International Conference, GoTAL 2008, Gothenburg, Sweden*, volume 5221 of *LNAI*, pages 336–347, Heidelberg, August 2008. Springer.

[Pow12] Richard Power. OWL Simplified English: A Finite-State Language for Ontology Editing. In Tobias Kuhn and Norbert E. Fuchs, editors, *Proceedings of the Third International Workshop on Controlled Natural Language (CNL 2012)*, volume 7427 of *Lecture Notes in Computer Science*, pages 44–60, Berlin / Heidelberg, Germany, 2012. Springer.

[Ran09] Aarne Ranta. The GF Resource Grammar Library. *Linguistic Issues in Language Technology*, 2(2), 2009.

[Ran11] Aarne Ranta. *Grammatical Framework: Programming with Multilingual Grammars*. CSLI Publications, Stanford, 2011. ISBN-10: 1-57586-626-9 (Paper), 1-57586-627-7 (Cloth).

[RED12] Aarne Ranta, Ramona Enache, and Grégoire Détrez. Controlled Language for Everyday Use: the MOLTO Phrasebook. In *Proceedings of the Second Workshop on Controlled Natural Language (CNL 2010)*, Lecture Notes in Computer Science. Springer, 2012.

[SLH03] Rolf Schwitter, Anna Ljungberg, and David Hood. ECOLE — A Look-ahead Editor for a Controlled Language. In *Controlled Translation, Proceedings of EAMT-CLAW03, Joint Conference combining the 8th International Workshop of the European Association for Machine Translation and the 4th Controlled Language Application Workshop*, pages 141–150, Dublin City University, Ireland, May 15–17th 2003.