A Grammar for Standardized Wiki Markup

Martin Junghans, Dirk Riehle, Rama Gurram, Matthias Kaiser, Mário Lopes, Umit Yalcinalp

SAP Research, SAP Labs LLC
3475 Deer Creek Rd
Palo Alto, CA, 94304 U.S.A.
+1 (650) 849 4087

martin.junghans@gmail.com, dirk@riehle.org, {firstname.lastname@sap.com}

ABSTRACT

Today's wiki engines are not interoperable. The rendering engine is tied to the processing tools which are tied to the wiki editors. This is an unfortunate consequence of the lack of rigorously specified standards. This paper discusses an EBNF-based grammar for Wiki Creole 1.0, a community standard for wiki markup, and demonstrates its benefits. Wiki Creole is being specified using prose, so our grammar revealed several categories of ambiguities, showing the value of a more formal approach to wiki markup specification. The formalization of Wiki Creole using a grammar shows performance problems that today's regular-expression-based wiki parsers might face when scaling up. We present an implementation of a wiki markup parser and demonstrate our test cases for validating Wiki Creole parsers. We view the work presented in this paper as an important step towards decoupling wiki rendering engines from processing tools and from editing tools by means of a precise and complete wiki markup specification. This decoupling layer will then allow innovation on these different parts to proceed independently and as is expected at a faster pace than before.

Categories and Subject Descriptors

D.3.1 Formal Definitions and Theory, F.4.2 Grammars and Other Rewriting Systems, H.4.1 Office Automation, H.5.2 User Interfaces, H.5.4 Hypertext/Hypermedia, I.2.4 Knowledge Representation Formalisms and Methods, I.3.6 Methodology and Techniques, I.7.2 Document Preparation, I.7.4 Electronic Publishing.

General Terms

Design, Human Factors, Standardization.

Keywords

Wikis, wiki markup, wiki markup grammar, wiki creole, wiki markup standard, wiki markup test cases, wiki markup parser, wiki engine performance.

WikiSym '08, September 8–10, 2008, Porto, Portugal.
Copyright 2008 ACM 978-1-60558-128-3/08/09...$5.00.

1 INTRODUCTION

Wikis were invented in 1995 [9]. They have become a widely used tool on the web and in the enterprise since then [4]. In the form of Wikipedia, for example, wikis are having a significant impact on society [21]. Many different wiki engines have been implemented since the first wiki was created. All of these wiki engines integrate the core page rendering engine, its storage backend, the processing tools, and the page editor in one software package.

Wiki pages are written in wiki markup. Almost all wiki engines define their own markup language. Different software components of the wiki engine like the page rendering part are tied to that particular markup language. In addition, users have to learn different wiki markup languages if they want to work with different wiki engines. Also, many organizations have to implement custom migration tools if they want to switch from one wiki engine to another.

Basically, each wiki engine is its own vertically integrated technology stack. A consequence of this vertical integration is that wiki engines are generally not interoperable, nor can components from one wiki engine be combined easily with components from another wiki engine.

Such vertical integration hinders wiki innovation significantly.

For example, if an innovator wanted to invent a new wiki editing paradigm, he or she would have to learn a particular markup and a particular rendering engine first before focusing on the wiki editor. However, the details of a markup parsing algorithm only get in the way of envisioning a new wiki editor.

To foster innovation in wiki technology, we need a decoupling mechanism between at least the following three major components of the wiki technology space:

- ***The core wiki engine*** (used for storage and retrieval of wiki pages)
- ***The wiki editing environment*** (used for collaboratively editing a wiki)
- ***Wiki processing tools*** (used, for example, in migration scenarios)

The missing decoupling mechanism is a precisely defined wiki markup standard that separates a wiki editor from a storage engine and from further processing tools. Figure 1 illustrates these parts and the role of a wiki markup standard. With such a stan-

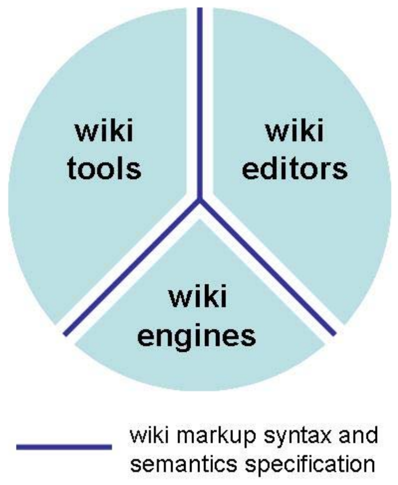

Figure 1. Decoupling of wiki software components by a markup syntax and semantics specification

dard, it becomes possible to develop wiki engines, processing tools, and wiki editors independently of each other. The resulting decoupling would make it easier to innovate in the wiki technology space because less moving pieces need to be juggled by the innovator.

We believe that future wiki engines will be less like today's simple collaborative editing tools and more like full-blown application platforms. Not only will users write natural language text, they will also program the wiki, creating simple one-off applications for their own needs [1]. We are not the only ones to make this conjecture. Several providers of wiki engines, most notably JotSpot [8], TWiki [15], and XWiki [3] are undertaking the necessary technical steps towards novel application platforms that follow the wiki philosophy [2].

At the 2006 International Symposium on Wikis [20], an effort was started to develop a common wiki markup standard called Wiki Creole [12]. The specification represents the effort of multiple involved parties, mostly wiki engine implementers and their corresponding sponsors.

Wiki Creole is being specified using prose. Unfortunately, the specification effort delivered no formal grammar to alleviate the shortcomings of a prose-only specification. Prose is highly susceptible to ambiguities, as this paper demonstrates. Nevertheless, Wiki Creole is the only available attempt at a wiki markup standard that has broad support in the community, and we expect it to enter a more formal standardization process. Thus, we decided to base our work on Wiki Creole.

To decouple the different technology components as outlined above, it is instrumental that we rely on a more precise syntax and semantics specification for wiki markup. This paper takes a step towards this goal by presenting the first complete grammar for a wiki markup standard [6]. A semantics definition is being worked on in parallel. To the best of our knowledge, no other complete grammar has been published before, not for Wiki Creole or for any other wiki markup specification.

The remainder of this paper is organized as follows. Section 2 discusses wiki markup and the challenges of formalizing it. Section 3 introduces a grammar for Wiki Creole using the EBNF notation (Extended Backus-Naur Form) [5]. It also discusses ambiguities we found in the current specification. Section 4 presents a parser implementation of our grammar done using the parser-generator ANTLR. Based on this parser implementation we describe performance issues that regular-expression-based parsers will be facing when trying to scale up. Section 5 validates the grammar and our work along the dimensions of usefulness, correctness, and efficiency. Section 6 discusses related work and Section 7 presents our conclusions.

2 WIKI MARKUP

Wiki markup is the text users write in a wiki page that constitutes the page's contents. Like in HTML, markup allows for formatting instructions like bold or italic. For example, the text

```
//EBNF grammar//
```

would be interpreted as the words "EBNF grammar" in italics, i.e.

```
<i>EBNF grammar</i>
```

and typically be displayed as

EBNF grammar

As can be seen from this example, like HTML, wiki markup is highly visual. While conceptually the notion of "italics" can be viewed as being independent of a specific visual display, practically it is not. Rather, the name of a specific syntactic element in wiki markup not only implies its meaning but also the way it is supposed to be rendered.

Wiki markup allows for more complex information. For example, a link may be written as

```
[[http://www.wikicreole.org|Wiki Creole Home]]
```

and be interpreted as the URL "Wiki Creole Home" linking to http://www.wikicreole.org, i.e.

Wiki Creole Home

In terms of usability, this can be viewed as an improvement over the HTML version of

```
<a href="http://www.wikicreole.org">
    Wiki Creole Home
</a>
```

Unfortunately, there is no commonly agreed-upon standard for wiki markup, and different wiki engines define different markup languages. For example, some wiki engines accept so-called "camel case" page links within a wiki. The wiki markup

```
WikiCreole
```

would be rendered as a relative link in the wiki to a page called "WikiCreole". Other wiki engines do not accept camel case but rely on the use of markup like squared brackets.

In 2006, several wiki engine developers defined a standard for wiki markup, called Wiki Creole [12]. The specification is an effort of multiple involved parties. It is being moved forward through work on a wiki as well as annual gatherings at the International Symposium on Wikis conference series.

Wiki Creole is being specified using prose. No grammar or semantics is provided. Wiki Creole, at the time of writing this paper, is at version 1.0. We expect that after a period of further feedback, Wiki Creole will be submitted to a more formal standards body for further evolution.

Some wikis, for example, "semantic wikis" allow for additional textual annotations that tie in with W3C technologies like RDF and OWL [14], [17], [22], [23]. Such markup allows wiki contents to be integrated with Semantic Web technologies. Features like semantic annotations represent natural future extensions of Wiki Creole.

3 WIKI CREOLE 1.0 GRAMMAR

The definition of Wiki Creole 1.0 can be found on its website [13]. We have developed a context-free grammar for Wiki Creole 1.0 that we discuss in this section. The full specification has been published as a technical report [6]. Wiki Creole 1.0 was released in July 2007 and is the most recent release at the time of writing. Appendix A shows an excerpt from the Wiki Creole specification, and Appendix B shows a matching excerpt from our grammar.

3.1 EBNF vs. prose

The benefits of using a grammar over using prose for the Wiki Creole specification are:

- ***(Almost) trivial parser construction using parser generators,*** ANTLR in our case.
- A formal language specification is needed for a subsequent semantics specification. This supports ***the integration of wikis into the Semantic Web*** [14].
- ***A clear wiki markup specification improves communication between wiki engine developers.*** There can be no different interpretation of the specification, because grammar-based parsers show uniform behavior. Such parsers have a precisely defined set of valid markup.
- ***Usability will also improve because users can rely on the same rendering behavior in different wikis.***
- ***Discovery of ambiguities in the prose specification through a more rigorous specification mechanism.***
- ***The base for a well-defined interchange of wiki page content between wiki engines.*** Without a precise markup definition, different markup interpretations result, and a well-working interchange becomes impossible.

3.2 ENBF vs. regular expressions

Today's wiki engines typically use regular expressions to define the markup syntax and create a parser. The benefits of using an EBNF-based grammar over a structured set of regular expressions, both for clarity and for parser implementation, are:

- ***An ENBF-based grammar is more complete than a structured set of regular expressions,*** because the ordering is explicit and defined using the same meta-notation as the rest of the grammar. The order and structure of a regular-expression-based parser is implicit in the program code.
- ***An EBNF-based grammar supports the creation and composition of an overall parse tree*** of a wiki from separately parsed structures. A regular expression based parser does not support such functionality.
- ***An EBNF-based grammar can be maintained and changed more easily than a structured set of regular expressions***, because the EBNF-based grammar again is more complete, more structured, and easier to read than a set of regular expressions.
- ***Simplified extension of standard markup with new features.*** Extensions can be added to the grammar in a straightforward way. In contrast to this, today's regular-expression-based parsers make it hard to extend the markup language because side-effects are hard to control.
- ***Regular expressions are not well-defined.*** In contrast to EBNF, regular expressions are not standardized and typically have only poorly defined semantics. Thus, different implementations are likely to behave differently, leading to inconsistent results of the parsing process.
- ***The ability to make performance predictions based on well-understood language theory.*** Today's regular-expression-based parsers are multi-pass parsers, leading to hard-to-predict performance behavior.

3.3 Wiki Creole ambiguities

The original Wiki Creole 1.0 specification contains many ambiguities. We attribute this to its use of prose. Our more formal approach using a grammar and a parser generator helped us to discover these ambiguities and also revealed the incompleteness and inconsistency of the prose specification.

We compiled a full list of ambiguities which we fed back into the specification process. Here, for illustration purposes, we discuss two categories of ambiguities that we discovered to show the usefulness of a grammar-based approach.

3.3.1 Nesting of markup

One category of ambiguities arises from nested markup tags. We use bold as the example. Starting and ending bold is signified by two stars.

The following wiki markup

```
1**2**3**4**5
```

could be interpreted as either

```
1<b>2</b>3<b>4</b>5
```

(shown as "1**2**3**4**5" where "2" and "4" are bold but not "3") or as

```
1<b>2<b>3</b>4</b>5
```

(shown as "1**234**5" where "2", "3", and "4" are set in bold). There is no notion of double-bold. The prose specification leaves open which of the two alternatives is the correct interpretation. The same applies to italics and related markup.

Common sense suggests the first alternative, and our grammar uses this variant. This is the behavior of most regular-expression-based parsers. From the prose specification we could not derive this as it remains silent on this category of ambiguities.

3.3.2 Double meaning of tokens

The markup to indicate a heading is the equal sign. The number of equal signs determines the depth of the heading. For example,

```
=== Double meaning of tokens ===
```

is to be interpreted as a third-level heading.

Wiki Creole allows for fewer closing equal signs than opening equal signs. They don't need to balance out. Also, an equal sign in the heading text is not prohibited either.

It is therefore not possible to distinguish between equal signs at the end of the heading text and the optional closing equal signs. For example, the following heading has multiple interpretations and it is undefined how to parse it.

```
== C programming: A comparison of == with = =
```

There are at least four different interpretations of when the heading closes and what its content is (after the first two conjoint equal signs, after each of the two single equal signs, and at the end of the paragraph).

Common sense suggests that the final equal sign closes the heading, and in fact this is what we find as the commonly implemented behavior. The primary reason for this is that almost all wiki engines today use line-oriented regular expression matching as the core of their parser.

3.3.3 More ambiguities

Using a grammar-based parser has significant benefits over using a regular-expression-based parser. For example, the pipe symbol "|" indicates table cells as well as alternative text of an image. A grammar-based parser easily distinguishes between both, because in the context of an image table cells are not allowed. Table cells and images are clearly separated by enclosing markup. Regular expressions in contrast cannot maintain context easily.

However, and more importantly, as we discuss in the performance section below, there are good reasons not to use regular-expression-based parsers when scaling up a wiki engine for performance.

We have found many more ambiguities that we fed back into the prose-based specification process. We discuss some of them in more detail in the technical report that presents the full grammar [6]. For the purposes of this paper we hope that the above demonstration is sufficient to illustrate the usefulness of introducing a grammar for a wiki markup standard.

4 WIKI CREOLE 1.0 PARSER

We created a parser (and related tools, see [6], [7]) for Wiki Creole to not only theoretically but also experimentally validate the grammar. We created the parser by subjecting it to the parser-generator ANTLR 3.0.1 (released on August 13, 2007).

ANTLR creates an LL(*)-parser from the grammar [10], [11]. This is a top-down parser for a subset of all context-free languages. The input stream is parsed from left to right and thereby the leftmost derivation will be triggered. In contrast to other recursive-descent parsers, ANTLR's parsers are not table-based. Thus, the generated code is human-readable and can be changed or improved easily. The LL(*) algorithm uses as much look-ahead as it needs to decide about the application of parser rules.

4.1 Implementation

The grammar definition for ANTLR consists of two parts, the scanner and the parser rules [6]. The scanner rules allow no ambiguities. The parser applies the widely used convention that the first specification wins: The order of alternatives in parser rules is important. The parser tries to match the first rule first, then the second, etc. Since only terminal symbols are allowed for exceptions in the rules, it is necessary to use terminal symbols mixed up with non-terminal symbols in some parser rules.

It was our goal to define a grammar without any ANTLR-specific options or modifications so that it works out of the box with any parser generator (besides a different notation for grammar rules).

Unfortunately, ANTLR is not able to handle our more complex context-free grammar properly. An example is the single star "*" in normal text, which is not intended to represent wiki markup but to be interpreted literally (cf. Appendix B, non-terminal symbol "onestar"). The generated parser wants to match each single star as bold markup "**" and throws an exception, because there is no second star after the first one. Thus, we had to introduce ANTLR-specific code into the grammar to utilize the look-ahead when a star appears. This unexpected behavior of ANTLR is caused by a variant implementation of the LL(*) algorithm in ANTLR. With the help of our ANTLR-specific work-around, the Wiki Creole grammar remained context-free.

4.2 Performance

All wiki engines that we know use regular expression matching for their parser implementation.

We can achieve better performance of a parser by reducing the complexity of the grammar definition. This is independent of whether the grammar is explicit (our EBNF-based grammar) or implicit (hand-coded regular expression matching). This is because the parser performance depends on the number of necessary derivations for a given input. From this point of view, a wiki markup specification that takes grammar-based parsers into account will speed up parser implementation and parser performance.

Here, we face two opposing forces: The usability of the markup language and its parsing performance. Wiki Creole tries to be as easily usable as possible, even for inexperienced users. This requirement leads to a more complex and hence less readable grammar. The number of non-terminal symbols explodes and makes the grammar unwieldy. This in turn hurts performance.

The parser we generate does not use backtracking. While useful for dealing with ambiguities in the Wiki Creole specification, it is computationally expensive.

We conducted some experiments, comparing our generated parser with the hand-implemented and optimized MediaWiki parser. In general, our parser did not perform better. This is not surprising, given that we are comparing a non-optimized generated parser with a highly optimized mature and widely-used parser. We did not mind, since for the purposes of this work, we did not intend to provide a competitive implementation of a parser (and renderer). Rather the emphasis is on validating the grammar as well as giving interested parties a chance to create a basic parser very quickly using a parser generator.

5 VALIDATION

We validate the quality and relevance of the grammar and its related work along the dimensions of usefulness, correctness, and efficiency.

5.1 Usefulness

In Section 3 we discuss how the EBNF-based grammar forced us to rigorously work through the prose specification. This led to the discovery (and resolution) of many ambiguities and inconsistencies in the original specification. We argue that this clearly demonstrates the usefulness of a more formal approach to markup specification than is currently the state of the art.

Table 1. Overview of grammar test cases

Bold and italics formatting

- bold and italics in a text paragraph
- bold and italics that spans multiple lines in one paragraph
- bold and italics that ends automatically at paragraph end
- no text in bold, italic markup enclosed or EOF
- bold and italics combined

Lists

- at least five levels of nesting
- list elements that are empty, contain bold, italics, links, nowiki-inline, forced line breaks, EOF
- automatic closing at end of list item line
- optional blanks before and after the bullet items
- a list end is any line that does not start with the current list indicator (optional blanks omitted)

Nowiki (escape from wiki markup to native format)

- no markup interpreted
- distinguish between inline nowiki and a block
- ensuring defined nowiki-block structure

Links

- can appear in paragraphs, in bold, italics, list items and table cells
- images are allowed in link description
- the double slash from the URI is no italic markup
- pipes occur in link description

Headings

- at least three different sized levels
- closing equal signs at the end are optional
- no blanks before left side opening equal signs
- no markup has to be parsed inside

Horizontal rules

- blanks before and after the markup optional
- nothing else on this line is allowed

Tables

- cells separated by single pipes
- cells can contain bold, italics, links, images, inline nowiki, forced line breaks, EOF
- header cells

5.2 Correctness

We developed a set of test cases for the grammar and the parser implementation. We created test cases for each markup element as well as for interesting combinations. Table 1 lists these test cases.

In addition, we used the wiki page for the Wiki Creole 1.0 specification [13] as a more complex and not artificially created test case. This page is not entirely written in Creole 1.0. It uses bold and italic markup of a former version of Wiki Creole. After fixing these issues by updating the markup of these elements to the according equivalent of version 1.0, the parser confirmed the compliance of this page to Wiki Creole 1.0.

Our parser further confirmed the compliance of all created test cases with the grammar and hence the Wiki Creole specification as we interpreted it. In addition, several non-compliant test cases we tried threw the expected parser exception properly. We argue this demonstrates the likely correctness of the EBNF-based grammar.

5.3 Efficiency

We expect a better performance from a hand-optimized parser developed using an EBNF-based grammar than from a parser based on regular-expression-matching. This advantage is based on performance optimizations around avoiding ambiguities that a more precise grammar promises and that are not available to a structured set of regular expressions.

In addition, we expect wiki-related software to scale along at least two dimensions. One dimension is the size of the artifact to parse. In our vision of future wiki software, a wiki is not just a loosely connected collection of wiki pages but rather one consistent document that we may want to parse as a whole. For example, wiki processing tools may want to parse the whole wiki, starting with the entry page as a root node, and appending linked wiki pages into the tree representing the overall structure of a wiki, all in one process. An example of such a processing tool is an import/export tool for wiki migration. A regular-expression-based parser will never be able to parse and comprehend a wiki as a whole.

A second dimension of scalability is handling a large number of concurrent user requests. Any performance advantage that a proper grammar-based parser affords will pay back handsomely in terms of overall runtime performance and resource consumption.

A final advantage is that wiki engines using our grammar can maintain a representation of the wiki based on the parser's abstract syntax tree (AST). This avoids the need to parse the same wiki page over and over again when loading the page from disk or cache for processing. Specifically, the rendering process can transform the existing AST into the output format, be it XML, HTML, or PDF, without having to parse wiki markup in the first place.

6 RELATED WORK

As already discussed, our work is based on Wiki Creole [12]. We use the prose specification of Wiki Creole 1.0 [13] as a starting point for a grammar for a standardized wiki markup. We are not aware of any finished or published work on wiki grammars, for Wiki Creole specifically or for any other wiki markup.

Proponents of WYSIWYG editors sometimes suggest that such editors will completely replace wiki markup. This argument does not apply here, as wiki markup and any visual representation of it should be equally powerful and just an alternative representation of the same contents; users will still need a precise specification of what can be expressed and how, if not for textual editing purposes, then for wiki interchange purposes.

Some work has been done on wiki interchange formats. In [19] Voss proposes a MediaWiki-based interchange format for page content. Voss uses XML, however, no underlying wiki markup grammar is presented. Voss' proposal is specific to MediaWiki and hence can't bridge between different wiki engines. In return for this limitation, the DTD can address all the features of MediaWiki that are not available in other wiki engines.

Völkel proposes a general wiki interchange format in [16], [18]. Again, no formalization in terms of a grammar specification is provided. In contrast to the approach in [19], the work by Völkel is not tied to a specific wiki engine. The advantage of this approach is that interchange can happen between different wiki engines. A drawback is that it is hard to cater to the specifics of individual wiki engines. Thus, Völkel's work addresses the least common denominator between wiki engines only.

7 CONCLUSIONS

In this paper, we present and discuss the benefits of a grammar for Wiki Creole, a community standard for wiki markup. We use the Wiki Creole 1.0 prose specification as our markup definition. The grammar shows the benefits of a formal syntax definition as it allowed us to discover many ambiguities in the original Wiki Creole specification. The grammar has been published and made available to the public as a technical report [6]. We validate the grammar by implementing a wiki markup scanner and parser using the parser generator ANTLR. We discuss our wiki markup test cases and how they validate the parser implementation. The grammar and the test cases are being made available to the public to foster innovation in the wiki community. Our goal for this work is to decouple the different technology components in the wiki space, which in turn lets us develop these components at their own speed, enabling faster and better innovation of wiki technology.

REFERENCES

[1] Craig Anslow and Dirk Riehle. "Towards End-User Programming with Wikis." In Proceedings of the Fourth Workshop on End-User Software Engineering (WEUSE IV). IEEE Press, 2008. Page 61-65.

[2] Ward Cunningham. "Design Principles of Wiki." Keynote given at the 2006 International Symposium on Wikis. See http://c2.com/doc/wikisym/WikiSym2006.pdf.

[3] Ludovic Dubost. XWiki. See http://www.xwiki.org.

[4] Google Inc. Trends on Wiki Keyword Search. See http://www.google.com/trends?q=wiki.

[5] ISO/IEC. EBNF Grammar Specification. ISO/IEC 14977 : 1996 (E). ISO, 1996.

[6] Martin Junghans, Dirk Riehle, Rama Gurram, Matthias Kaiser, Mario Lopes, and Umit Yalcinalp. "An EBNF grammar for Wiki Creole 1.0." In ACM SIGWEB Newsletter, Volume 2007, Issue Winter (Winter 2007), Article No. 4.

[7] Martin Junghans, Dirk Riehle, and Umit Yalcinalp. An XML Interchange Format for Wiki Creole 1.0 In ACM SIGWEB Newsletter, Volume 2007, Issue Winter (Winter 2007), Article No. 5. ACM Press, 2008.

[8] Joe Kraus. "The Long Tail of Software: Millions of Markets of Dozens." See http://bnoopy.typepad.com/bnoopy/2005/03/the_long_tail_o.html.

[9] Bo Leuf and Ward Cunningham. The Wiki Way: Quick Collaboration on the Web. Addison-Wesley, 1999.

[10] Terrence Parr. ANTLR. See http://www.antlr.org. Web-published, 2007.

[11] Terrence Parr. The Definitive ANTLR Reference. The Pragmatic Programmers LLC, 2007.

[12] Christoph Sauer and Chuck Smith (editors). Wiki Creole. See http://wikicreole.org. Web-published, 2007.

[13] Christoph Sauer and Chuck Smith. (editors). Wiki Creole 1.0. See http://wikicreole.org/wiki/Creole1.0. Web-published, 2007.

[14] Sebastian Schaffert, Max Völkel, Stefan Decker. Proceedings of the First Workshop on Semantic Wikis. Web-published, 2006.

[15] Peter Thoeny. TWiki. See http://www.twiki.org.

[16] Max Völkel. WS3: Wiki Interchange Format. See http://www.wikisym.org/wiki/index.php/WSR_3. Web-published, 2006.

[17] Max Völkel, Markus Krötzsch, Denny Vrandecic, Heiko Haller, Rudi Studer. "Semantic Wikipedia." In Proceedings of the 15th International Conference on World Wide Web. ACM Press, 2006. Page 585-594.

[18] Max Völkel, Eyal Oren. "Towards a Wiki Interchange Format." In Proceedings of the First Workshop on Semantic Wikis. Web-published, 2006.

[19] Jacob Voss. Wikipedia DTD. See http://meta.wikimedia.org/w/index.php?title=Wikipedia_DTD. Web-published, 2007.

[20] Wiki Symposium. See http://www.wikisym.org.

[21] Wikimedia Foundation. English Wikipedia. See http://en.wikipedia.org.

[22] World Wide Web Consortium. OWL Specification. See http://www.w3.org/2004/OWL. W3C, 2004.

[23] World Wide Web Consortium. RDF Specification. See http://www.w3.org/RDF. W3C, 2004.

APPENDIX A: EXCERPT FROM THE WIKI CREOLE 1.0 SPECIFICATION

A.1. Bold and Italics

Bold and italic text can be used inside paragraphs, list items and table cells. Links appearing inside bold and/or italic text should also become bold and/or italic. The bold/italic text will end at the end of paragraphs, list items and table cells -- thus it cannot span several of them.

A.1.1. Bold. A star (*) is the most used symbol to emphasize text online. Double symbols are generally used in Creole to avoid accidentally parsing text not meant to be parsed. [...]

Creole:

```
**bold**
```

Recommended XHTML:

```
<strong>bold</strong>
```

Sample Output:

bold

A.1.2. Italics. A slash (/) looks like slanted italics, so it is intuitive and thus easier to remember.

Ignore // for italics processing if immediately following http: or ftp:

Creole:

```
//italics//
```

Recommended XHTML:

```
<em>italics</em>
```

Sample Output:

italics

Creole:

```
Bold and italics should //be
able// to cross lines.
But, should //not be...
...able// to cross paragraphs.
```

Recommended XHTML:

```
<p>
Bold and italics should <em>be
able</em> to cross lines.
</p>
<p>
But, should //not be...
</p>
<p>
...able// to cross paragraphs.
</p>
```

Sample output:

Bold and italics should *be able* to cross lines.
But, should *not be...*
...able *to cross paragraphs.*

A.1.3. Bold Italics.

Creole:

```
**//bold italics//**
//**bold italics**//
//This is **also** good.//
```

Recommended XHTML:

```
<strong><em>bold italics</em></strong>
<em><strong>bold italics</strong></em>
<em>This is <strong>also</strong> good.</em>
```

Sample Output:

Bold italics
Bold italics
*This is **also** good*

Unacceptable:

```
**//bold italics**//
//**bold italics//**
```

[…]

APPENDIX B: EXCERPT FROM A GRAMMAR FOR WIKICREOLE 1.0

```
text_formattedelement
    :   [...]
    |   bold_markup  text_boldcontent
            ( ( NEWLINE )?  bold_markup )?
    ;

text_boldcontent
    :   ( NEWLINE )?  ( text_boldcontentpart )*
    |   EOF
    ;

text_boldcontentpart
    :   ital_markup  [...]  ( ital_markup )?
    |   text_formattedcontent
    ;
```

```
text_formattedcontent
    :   onestar  ( text_unformattedelement
            onestar  ( text_linebreak )? )+
    ;

text_unformattedelement
    :   text_unformatted
    |   text_inlineelement
    ;

text_inlineelement
    :   text_first_inlineelement
    |   nowiki_inline
    ;
```

```
text_first_inlineelement
    :   link
    |   image
    ;

text_unformatted
    :   (   ~(  ITAL
            |   STAR
            |   LINK_OPEN
            |   IMAGE_OPEN
            |   NOWIKI_OPEN
            |   FORCED_LINEBREAK
            |   ESCAPE
            |   NEWLINE
            |   EOF )
        |   forced_linebreak
        |   escaped )+
    ;
```

```
onestar
    :   ( { input.LA(2) != STAR }?  ( STAR )?)
    |
    ;

bold_markup
    :   STAR  STAR
    ;

ital_markup
    :   ITAL
    ;

STAR    : '*';

ITAL    : '//';
```