Semantic Wiki Support for Intelligence Work

ANDREAS HORNDAL

KTH Computer Science and Communication

Master of Science Thesis
Stockholm, Sweden 2010
Semantic wiki support for intelligence work

Abstract

The semantic web is an extension of the World Wide Web where the semantics of information is defined. The formally defined semantics make it possible for machines to get a better understanding of how the information can be used, and what links between documents represent. By combining the wiki concept, a web site where anyone is able to create and edit content using a web browser, with ideas from the semantic web, something called a semantic wiki can be created. Incorporating ideas from the semantic web may improve the information structure, which makes it easier for a user to find relevant information and to find out how things relate to each other.

This thesis presents a semantic wiki prototype and describes how a semantic wiki could be used to support intelligence work with focus on gaining situation understanding. An experimental exercise showed that articles with dynamic content based on semantic search questions is a powerful new feature. The experiment also revealed that a semantic wiki is of no more use than an ordinary wiki unless the content is annotated with semantic information, which is a time-consuming activity. Future work includes investigating how the tagging process can be improved and automated.

Semantisk wiki som stöd vid underrättelsearbete

Sammanfattning


I rapporten presenteras en prototyp av en semantisk wiki och hur en semantisk wiki kan vara användbar vid beslutsfattande med fokus på situationsförståelse. En experimentell övning visade att möjligheten att skapa artiklar med dynamiskt innehåll baserat på en semantisk sökfråga var en ny kraftfull funktion. Experimentet visade också att en semantisk wiki inte är mer användbar än en vanlig wiki om inte innehållet är märkt med semantiska taggar, vilket är en tidskrävande uppgift. Fortsatt arbete innefattar att undersöka hur taggningsprocessen kan förbättras och automatiseras.
Preface

This report describes a master’s thesis carried out at the School of Computer Science and Communication (CSC) at the Royal Institute of Technology in Stockholm. The work was performed during the summer and autumn of 2007.

I would like to thank my supervisor at FOI, Christian Mårtenson, who helped me during the entire project with questions raised along the way. I would also like to thank my supervisor at KTH, Joel Brynielsson, for his detailed and valuable comments upon my report.

Andreas Horndahl, November 2008

"Make everything as simple as possible, but not simpler."

- Albert Einstein
Table of Contents

1 Introduction ...................................................................................................................... 1
  1.1 Network based defense and the knowledge support concept ................................ 1
  1.2 The semantic web ...................................................................................................... 1
  1.3 Idea: Semantic wiki for intelligence information management ............................ 2
  1.4 Thesis goals ............................................................................................................. 3
  1.5 Method ..................................................................................................................... 3
  1.6 Delimitations ............................................................................................................ 4
  1.7 Report structure ........................................................................................................ 4

2 Semantic web technology ................................................................................................ 5
  2.1 Semantic web vision ................................................................................................ 5
  2.2 Ontology ................................................................................................................... 6
  2.3 Ontology languages .................................................................................................. 8
    2.3.1 Resource description framework (RDF) .......................................................... 9
    2.3.2 RDF Schema (RDFS) ..................................................................................... 10
    2.3.3 Web ontology language (OWL) ..................................................................... 10
  2.4 Ontology reasoning ................................................................................................ 12
  2.5 SPARQL ................................................................................................................ 14

3 Semantic wiki for intelligence information management .............................................. 16
  3.1 Wiki ........................................................................................................................ 16
  3.2 Semantic wiki ......................................................................................................... 17
  3.3 Semantic wiki implementations ............................................................................. 18
  3.4 Using a semantic wiki to create situational understanding .................................... 19
  3.5 Related work .......................................................................................................... 20

4 Design ............................................................................................................................ 22
  4.1 Requirements ......................................................................................................... 22
    4.1.1 Introduction .................................................................................................... 22
    4.1.2 Search capabilities .......................................................................................... 22
    4.1.3 Articles with dynamic content ....................................................................... 23
    4.1.4 Context navigation ......................................................................................... 23
    4.1.5 Inference of new facts .................................................................................... 23
    4.1.6 Consistency checking ..................................................................................... 23
  4.2 Evaluation of existing software .............................................................................. 24
    4.2.1 Semantic MediaWiki ...................................................................................... 25
    4.2.2 Ikewiki ........................................................................................................... 26
    4.2.3 Makna ............................................................................................................. 27
    4.2.4 Systems for inference and storage ................................................................. 28
List of figures

Figure 1 – Theoretical intelligence workflow................................................................. 2
Figure 2 – Untyped / typed relations.................................................................................. 6
Figure 3 – Classification of ontologies according to their expressiveness.......................... 8
Figure 4 – Expressivity of the three OWL sublanguages.................................................... 11
Figure 5 – Example of an OWL ontology class hierarchy.................................................. 1
Figure 6 – Example of an OWL ontology property hierarchy............................................ 12
Figure 7 – An example of transitive and inverse properties................................................ 13
Figure 8 – Untyped versus typed relations including inferred statements......................... 14
Figure 9 – An example of a wiki article in Wikipedia describing Guinea............................ 17
Figure 10 – Screenshot from an SMW article describing California.................................... 26
Figure 11 – Screenshot from an Ikewiki article describing Bilberry................................. 27
Figure 12 – Screenshot from a Makna article describing Humphrey Bogart....................... 28
Figure 13 – Architectural overview..................................................................................... 31
Figure 14 – Semantic wiki components............................................................................. 31
Figure 15 – An example of a simplified rendering pipeline................................................. 32
Figure 16 – Command flow between the semantic bridge and the SAIL............................. 33
Figure 17 – Different approaches to dealing with external information sources.................. 34
Figure 18 – Flow of information between a wiki and content management systems............ 34
Figure 19 – Add statement dialog screenshot.................................................................... 36
Figure 20 – Screenshot from a wiki article about the city Conakry..................................... 37
Figure 21 – FCKeditor screenshot..................................................................................... 37
Figure 22 – Screenshot showing the ontology editor.......................................................... 38
Figure 24 – Workflow for creating a SPARQL query for important persons..................... 40
Figure 25 – SPARQL Wizard interface............................................................................. 40
Figure 26 – Query for important persons inserted into a wiki article.................................. 41
Figure 27 – SPARQL Wizard building blocks.................................................................. 41
Figure 28 – Screenshot showing a wiki article with dynamic content............................... 42
Figure 29 – Screenshot from the editing mode of an article containing dynamic content...... 42

Code-example 1 – Basic RDF example using XML.......................................................... 10
Code-example 2 – SPARQL query example................................................................. 15

Table 1 – RDF triple expressed in table format............................................................... 10
Table 2 – Inference and storage systems......................................................................... 29
Table 3 – SPARQL Wizard key features......................................................................... 39
1 Introduction

This chapter describes the context of this master’s project, the semantic wiki idea, the project’s goal, method, delimitations and report structure.

1.1 Network based defense and the knowledge support concept

The goal of every organization is to be as efficient as possible and to accomplish as much as possible using a limited amount of resources. The Swedish Armed Forces is no exception, seeking to meet the demands of a new era where information sharing plays an important role.

It all comes down to spending one’s resources on the right thing. Having access to relevant information is necessary in order to make good decisions and to establish situational awareness (SA). Situation awareness can be defined as “the perception of elements in the environment within a volume of time and space, the comprehension of their meaning, and the projection of their status in the near future” [END].

Network based defense is a key concept in the Swedish Armed Forces and concerns, among other things, how to share information in order to achieve information superiority. The goal is to share information so that a common operational picture can be created. Having a shared situation picture, that all involved parts agree upon, is necessary in order to coordinate actions.

Due to an increased degree of complexity in situations handled by the armed forces, improved concepts are needed to gain knowledge and share information. Involved actors can be of different types and whether they should be regarded as opponents, neutral or allied is not always clear. The interests, motivations and capabilities of each actor must be taken into account. Conflicts in the world of today are complex and often involve organizations, nations and people from all around the world. In order to establish SA all data, information and knowledge need to be coordinated between these parties. To handle the complexity, the Swedish Armed Forces are looking at adopting a concept denoted knowledge support (KS), which covers this coordination and describes how to share and enable knowledge between supporting functions. KS deals with knowledge acquisition, exchange and integration from different sources. Implementing a modern KS concept requires the use of new technology that can support the processes involved.

1.2 The semantic web

Since the birth of the Internet, the amount of available information has increased rapidly. Today, the problem is not about how to get hold of information, it is about how to filter the information and extract the parts that we need. This large amount of available information makes it hard to find the pieces of information that are actually meaningful. Regardless if you have an information overflow or have insufficient information the fundamental issue is how to deal with the available information in the most efficient way.

One might believe that computer software can help us navigate through the information jungle. However, a major problem is that much of the information available on the Internet is intended to be interpreted by humans. Thus, machines have a hard time understanding what the information really is about and, consequently, have a limited ability to help humans find the most useful information.
The semantic web is the term used for the vision of making Internet content interpretable by machines. Semantic web is about helping computers gaining a better understanding of what the information really means. Subsequently, when the information is understandable by computers, the computers would also be able to infer new facts. Hence, the semantic web concept, where information can be interpreted by machines, may not only improve the Internet, it may also be useful in other information sharing domains such as information sharing within the armed forces.

1.3 Idea: Semantic wiki for intelligence information management

A wiki is a web site where anyone is able to create and edit content using a web browser that has proven to be a useful tool for collaborative information management.

By combining a wiki with ideas from the semantic web concept, something called a semantic wiki could be created. Incorporating ideas from the semantic web may improve the information structure, which makes it easier for a user to find relevant information. Being able to find relevant information is central when it comes to gaining SA. The semantics could also be used to present information in new ways.

An analytical workflow can generally be divided into three phases: search, analyze and present. During the search phase, information is gathered from different kinds of sources. The sources of information may be of many kinds, such as databases, sensors and newspapers [FMUNDH]. In order to reduce the time spent in the search phase, a new phase can be introduced which deals with information input and structuring of information. Since information structuring is the focus of the semantic web, technology from this area may be useful.

![Figure 1 – Theoretical intelligence workflow. The solid line shows the amount of time spent in the different phases in a typical case of today, whereas the dotted line shows how much time may be spent in each phase if semantic techniques are used.](image-url)
If information is structured using semantic techniques, it might be possible to turn the solid
curve of Figure 1 above into the dotted one. Decreasing the amount spent on searching for
information enables the analyst to spend more time on the later phases: creative thinking and
analysis. Such benefits could prove especially useful in critical situations, when it is of
utmost importance to gain situational awareness in a short period of time.

Another benefit is that if information is structured, it is easier to see how new information
would affect a situation. If structured information is added, the new information might cause
a chain of reactions that may infer new statements that will drastically change the situation
picture. A list of potential threats, that is dynamically updated when new input is added, may
be very useful in time critical situations.

Our hypothesis is that analysts will benefit from a semantic wiki to make it easier to ‘connect
the dots’ and gain situation understanding by:

- improved search capabilities,
- automatically updated information objects,
- revealing implicit facts and complex relationships,
- supporting consistency checking.

1.4 Thesis goals

In order to examine whether or not a semantic wiki can be a useful tool for gaining
situational awareness in the context of military intelligence, a prototype of such a system
should be developed based on existing open source technology. The goal is to evaluate the
existing technology and propose, and to some extent implement, possible improvements that
are likely to be useful for the intended users.

Key questions:

- Is a semantic wiki a useful tool for the intended user group?
- What are the key components of a suitable semantic wiki?
- What wiki-platforms are available today and what kinds of improvements are needed in
  order for the wiki to be useful for the intended user group?

1.5 Method

The focus of this master’s project has been to examine if the semantic wiki concept would be
useful for analysts and decision-makers when trying to establish situation awareness. The
intent has not been to create a system that replaces existing software.

The work in this thesis project has been carried out as follows. To start with, a literature
review has been done. The literature covered the wiki concept, the semantic web and
information management within the armed forces with focus on intelligence information.
The literature review provided necessary background information in order to formulate key
features that hypothetically would be useful for establishing situation awareness. The key
features were broken down into system requirements. The next step was to test and evaluate
existing open source wikis and technologies for storing semantic information in order to find
out if and how existing software could be used to meet the requirements.

The conclusions that could be drawn from the evaluation formed a basis for the proposed
system design. Based on the system design, a prototype was developed in order to
demonstrate and evaluate if such a system will be of use for the intended user group.
To evaluate whether the system is useful in reality, the system was demonstrated for a group of people with relevant background experience. The demonstration was part of a workshop conducted by the Swedish Armed Forces. The tool was used to support a group of analysts given the task to make an economical assessment of the situation in a certain conflict region.

### 1.6 Delimitations

The focus of this thesis is to evaluate if and how a semantic wiki can be useful for the intended user group and point out issues to take into consideration when designing such a system. In order to do that a prototype has been developed where some of the key concepts were demonstrated and tested. In other words, the goal is to evaluate if and how a semantic wiki is a useful tool for gaining situation awareness and not to present a complete software design solution. All wiki systems and semantic web tools that were used to create the prototype are open source software.

Semantic information systems often rely on an ontology, which is a formal description of the world to which information should be related. The ontology defines and describes what kind of things and relations that are available and is of great importance to the system. However, constructing and designing an ontology is beyond the scope of this master’s thesis.

### 1.7 Report structure

Chapter 2, *Semantic web technology*, describes the vision of the semantic web, the ontology concept and how ontologies can be used to support inference of new facts and hypotheses.

Chapter 3, *Semantic wiki for intelligence information management*, describes the wiki concept, possible benefits of extending a wiki with semantic technology and how a semantic wiki can be used in order to achieve a shared situation picture. The last part of the chapter describes related work.

Chapter 4, *Design*, begins by stating system requirements. Thereafter follows a section that describes how existing software was evaluated. The last part of Chapter 4 describes the proposed architecture.

Chapter 5, *Implementation*, gives an overview of how the prototype was implemented and what open source software it is based upon.

Chapter 6, *Experiments*, describes how the prototype was tested. It also contains a discussion section that points out problems with the experiment setup and describes how an experiment can be constructed that copes with these problems.

Chapter 7, *Results*, summarizes the results from the software evaluation, the implementation and the experiment.

Chapter 8, *Discussion*, discusses the results.

Chapter 9, *Conclusions*, summarizes the project findings.

Chapter 10, *Future work*, gives an overview of how the Semantic wiki concept can be further developed.
2 Semantic web technology

This chapter describes the semantic web vision, the building blocks of the semantic web, available technology and standards.

2.1 Semantic web vision

The World Wide Web (WWW) is made up of many web documents, distributed over a large number of servers across the globe. The key characteristic of a web document is that it contains hypertext markup tags. From a computer’s perspective, hypertext markup tags provide information about how to present information, not what the information really means and implies. Since a computer has no information regarding what the information means, it has a limited ability to know how the information can be used unless provided with such information.

To create a more intelligent web, the key question is if the web can be reconstructed by adding extra information to documents that makes it possible for machines to understand what the information really is about. If data and links on the web could be defined in such a way that a computer could get a better understanding of what the information is and can be used for, tools like search engines and other information-providing applications such as a wiki could be improved.

The attempt to construct a web that makes it possible for machines to interpret and understand web documents is known as the semantic web. The inventor of the World Wide Web, Berners-Lee, describes the semantic web like this:

“The Semantic Web is an extension of the current Web in which information is given well-defined meaning, better enabling computers and people to work in cooperation... a web of data that can be processed directly and indirectly by machines” [BER01]

Since the amount of available information on the Internet is vast, Internet search engines like Google have become very popular tools for finding information. It is not possible for a human being to browse through millions of documents manually. Search engines are of great use but the technology evolution does not stop here. There are still problems to solve. Today, if a search is made on a popular search word, the search result may end up in thousands or more hits1. Search engines often use techniques such as text analysis to extract meta-information from available Internet documents. If Internet documents included well-defined machine-readable information describing the information itself, search engines could make use of this information to improve the quality of the search result even more.

Tagging information with semantic information is in one perspective similar to defining links between information objects, but with one important difference. The difference is that the link itself represents a term described in an information model. Wikipedia defines an information model as follows:

“An information model is, within the field of data modeling, an abstract but formal representation of entities including their properties, relationships and the operations that can be performed on them.”[WPI]

The semantic web attempts to express and enable semantic relations among represented entities. In this context, entity is a catch all word for something that has a distinct existence

1 The resulting set may be ranked but the top ranked result is not guaranteed to be the most relevant document.
but does not necessary have a material existence. Semantic relations are meaningful associations between two or more entities, or sets of entities. For example, it is possible to express knowledge like: “every person is a mammal”.

Figure 2 describes links extracted from a typical homepage for a person named John. John’s homepage may consist of a personal presentation and links to the homepages of members of his family. The links on John’s homepage describes, from a machine’s point of view, nothing more than that there exists a relationship between the pages. It does not say anything about the actual meaning of the relationship. Assigning types, taken from an information model, to the links enriches the links with valuable information that helps a machine to understand what the relationships stand for. In summary, the idea is to make information within Internet documents understandable and accessible to computers so that it can be reused across various applications in an efficient way.

Adding semantic annotations to Internet documents does not only improve the ability of computers to help us find relevant information, it also makes it possible for computers to reason and infer new facts. Examples of how reasoning can be used can be found in Section 2.4.

2.2 Ontology

Ontologies describe knowledge about a certain domain by specifying concepts, relations between concepts and axioms and can be used to build information models. The term ontology has its origin in philosophy and is defined as “A branch of metaphysics concerned with identifying, in the most general terms, the kinds of things that actually exists. Thus, the ontological commitment of a philosophical position includes both its explicit assertions and its implicit presupposition about the existence of entities, substances or beings of particular kinds.” [KAB07]

Based on other well-known definitions, [KAB07] has defined an ontology for use in data and knowledge modeling as:

An ontology is an explicit formal conceptualization of a shared understanding of the domain of interest inducing the vocabulary of term, semantic as well as their pragmatics.

In [STU98] the terms used in the definition are explained as follows:
Explicit: The type of concepts used, and the constraints on their use are explicitly defined.

Formal: The ontology should be machine readable.

Shared: Reflects the notion that an ontology captures consensual knowledge, that is, it is not private to some individual but accepted by a group.

Conceptualization: Abstract model of some phenomenon in the world by having identified the relevant concepts of that phenomenon.

According to [MCG03], the following requirements must be fulfilled in order for something to be considered an ontology:

- Finite and controlled (extensible) vocabulary.
- Unambiguous interpretation of classes and term relationships.
- Strict hierarchical subclass relationships between classes.

A controlled vocabulary means that there is a finite set of terms and that every term has an explicit definition. Formal definitions of terms make it possible for machines to interpret the information in a correct manner. The relationship between terms should also have explicit and unambiguous definitions. In practice, this means that relationships must be defined using a formal language interpretable by a machine.

Strict hierarchical subclass relationships are necessary for type deduction. If A is a superclass of B then the relationship between A and B is said to be strictly hierarchical if an instance x of type B entails that instance x is also of type A. A hierarchy where elements do not follow this kind of “is-a” structure can consequently not be said to have a strict hierarchical subclass relationship between classes.

The level of expressiveness may vary between ontologies. In [MCG03] the following list of ontology features is presented. The list can be used to classify ontologies with respect to expressiveness.

- Controlled vocabulary: list of terms.
- Thesaurus: information describing relationships between terms such as synonyms.
- Informal taxonomy: a hierarchy exists but inheritance is not strict.
- Formal taxonomy: hierarchy with strict inheritance.
- Frames (classes): classes with sets of properties that can be inherited is supported.
- Value restrictions: property values can be restricted.
- General logic constraints: values of properties can be restricted by logic rules.
- First-order logic constraints: extended support for logic rules. First-order logic with detailed restrictions such as disjoint classes or transitive relationships is supported.

Figure 3 provides a graphical representation of the list above. Classification of some well-known ontologies are included in the figure. Ontologies with a high level of expressiveness fall under the category “heavy-weight ontologies” whereas less expressive ontologies fall under the category “light-weight ontologies”.

Chapter 2 – Semantic web technology

The set of statements that describe the vocabulary in domains are referred to as the terminology box (TBox). For instance, a statement that specifies that a class is a subclass of another class is a TBox statement. The set of statements that describe statements about instances are referred to as the assertion box (ABox). Examples of typical ABox statements are “A is an instance of class B” and “John isMarriedTo Lisa”. Together ABox and TBox statements make up what is known as the knowledge base.

The benefits of using ontologies can be summarized as follows:

- It provides a shared understanding about concepts in a domain by formal definitions.
- It provides a way to reuse knowledge about the domain.
- It provides a way to encode knowledge in a way that is understandable by machines.

A WWW document containing information about the country Guinea could make use of an ontology in numerous ways. The document may include a sentence like “Guinea is a nation located in West Africa”. Assume that the terms “is a” and “located in” are defined in an ontology and that the text has been tagged using these terms. A search engine may use this explicit and machine-readable information to answer string like “Nations in west Africa” with a list of nations instead of links to sources that may contain this information. If you add ‘populationSize’ to the ontology and in the text annotate the sentence “population size: 9,402,000” a search engine may answer a question like “List the biggest nations in the word” in a similar way. The result could be presented as a table with two columns containing the name of the nations and population size in descending order.

2.3 Ontology languages

Ontology languages are formal languages used to construct ontologies. This chapter describes RDF, RDFS and OWL, the recommended W3C standards for creating ontologies and sharing semantic information.

Figure 3 – Classification of ontologies according to their expressiveness (adopted from [ALE06]).
2.3.1 Resource description framework (RDF)

Resource Description Framework is a metadata model specification developed by W3C, but it can also be used as a general framework for information modeling [WPRDF]. In essence, RDF provides a basis for coding, exchanging, and reusing structured metadata. RDF has several serialization formats. The two most common are XML and Notation 3 (N3). RDF is not limited to describing metadata about web pages, it can also be used to describe objects and concepts of the real world.

RDF provides two kinds of entities, resources and literals. A resource is an abstract concept or an object. A literal is a string that can be used to express basic properties such as names, numbers, etc. Resources and literals are used to compose statements. These are known as triples since they consist of three parts. The three parts are:

1. a subject: the resource that the statement is about.
2. a predicate: a resource that represents the relationship between the subject and the object.
3. an object: a resource or literal.

A statement links the subject to the property via the predicate. The predicate can be seen as a named relation between the subject and the property value. A set of RDF statements (triples) represents a labeled, directed graph.

Every resource described using RDF has a global unique name represented by a uniform resource identifier (URI). The URI does not necessarily have to exist on the WWW, it is simply used to uniquely identify the resource. The URI is conceptually analogous to an address or a personal identity number\(^2\). If two RDF documents use the same URI to describe a resource, then the documents describe the same thing.

Below is an example of what a resource URI for the country Guinea may look like.

http://www.andreashorndahl.se/country#Guinea

The URL (http://www.andreashorndahl.se/country) is used as a namespace and the last part of the URI (right-hand side of #) defines the local resource name. A resource defining the term population may have the following URI.

http://www.andreashorndahl.se/country#population

A full-length statement that the resource Guinea has a population of 9,402,000 can be expressed as

http://www.andreashorndahl.se/country#Guinea http://www.andreashorndahl.se/country#population 9,402,000

Namespaces can be replaced by aliases. For instance, a namespace alias called country can be defined by

xmlns:country="http://www.andreashorndahl.se/country#"

Namespace aliases make it possible to rewrite statements in a shorter form. A short version of the above statement can now be written as:

country:Guinea country:population 9,402,000

A table version of the triple is provided in Table 1.

---

\(^2\) A personal identity number is the Swedish equivalent of a social security number in the US.
### Table 1 – RDF triple expressed in table format.

<table>
<thead>
<tr>
<th>Subject</th>
<th>Predicate</th>
<th>Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>Guinea</td>
<td>population</td>
<td>9,402,000</td>
</tr>
</tbody>
</table>

A first glance of what an RDF document in XML format may look like is provided in Code-example 1.

```xml
<?xml version="1.0"?>
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns="http://www.andreashorndahl.se/country #">
  <Country rdf:ID="Guinea">
    <population>9,402,000</population>
  </Country>
</rdf:RDF>
```

**Code-example 1 – Basic RDF example using XML.**

### 2.3.2 RDF Schema (RDFS)

RDF can in a structured way define relations between resources. However, the capability of RDF is limited when it comes to defining how terms used in statements relate to other terms and if the terms have any restrictions. Moreover, RDF does not support class-subclass relationships. RDF Schema (RDFS) is a language that can be used to describe the vocabulary used in RDF triples. RDFS extends RDF by introducing ways to add information about how terms relate to each other.

RDFS introduces subclassing. A class Man can be defined to be a subclass of Person. Likewise, a property (predicate) can also be subclassed. A property fatherTo can be defined to be a subproperty of ancestorTo.

Another important concept introduced by RDFS is the ability to define property restrictions such as domain and range. Property restrictions and how they can be used to infer new facts are described in further detail in Section 2.4.

### 2.3.3 Web ontology language (OWL)

RDF and RDFS are not expressive enough to describe all types of relationships when constructing a heavy-weight ontology. For this a more expressive language is needed that to a further extent is able to express complicated relationships. The web ontology language, OWL, is under development by the World Wide Web Consortium and aims to meet the requirements of the semantic web. OWL is based on RDF/RDFS but provides new constructs that can be used to express more complex relationships that enhance reasoning abilities. For example, OWL allows transitive properties. How this feature can be used to infer new facts is described in further detail in Section 2.4.

OWL provides three sublanguages [W3OWL2]:

- **OWL Lite** is the least expressive of the three sublanguages and should therefore be the easiest to develop tools for. One example of its limitations is that although it supports cardinality constraints, the constraints are restricted to cardinality values of 0 or 1.

- **OWL DL** provides more expressive constructs than OWL Lite and is closely related to description logics. The intended users are those who want advanced expressiveness, computational completeness (all conclusions are guaranteed to be computable) and decidability (all computations will finish in finite time). This sublanguage includes all
OWL constructs, but under certain restrictions. An example of a restriction is that a class may be a subclass of many classes but a class cannot be an instance of another class.

- OWL Full supports all constructs of OWL but without computational guarantees. OWL Full supports a class being treated as an individual and a collection of individuals at the same time.

Even though OWL Full is the most expressive of the three sublanguages, applications that use OWL focus on supporting OWL DL because it is unlikely that any reasoning software will be able to support all constructs of OWL Full.

Each of the sublanguages is an extension of its simpler predecessor as shown in Figure 4. The following set of relations hold. Their inverses do not.

- Every legal OWL Lite ontology is a legal OWL DL ontology.
- Every legal OWL DL ontology is a legal OWL Full ontology.
- Every valid OWL Lite conclusion is a valid OWL DL conclusion.
- Every valid OWL DL conclusion is a valid OWL Full conclusion.

An OWL ontology was developed within the project that this master’s project is part of. Figures 5 and 6 give graphical views of the class and property hierarchies. An example OWL ontology serialized as XML is provided in appendix B.

Since OWL ontologies are meant to be interpreted by machines, tools such as Protégé [PROTÉGÉ] are often used to create ontologies. Figure 5 and 6 provide screenshots of what an ontology can look like when viewed in an ontology editor.
2.4 Ontology reasoning

Ontology languages such as OWL support and encourage development of software that uses algorithms to discover inconsistent statements and to infer logical consequences from a set of facts. This kind of software is commonly known as semantic reasoners [WPSR].

The strict rules that can be defined in an ontology make it possible for algorithms to discover inconsistent statements. This is useful when constructing ontologies and supports the creation of high quality ontologies since contradictory statements can be detected automatically. Discovering potential contradictions is not only useful when creating ontologies, it may also be useful when a user tries to add a fact to the knowledgebase.

OWL constructs let us define classes and properties. These can be used to make statements about objects (resources) in order to describe, in a precise and formal way, objects and how they relate to other objects.

A useful feature is to deduce facts based on type relationships. In the ontology example in appendix B, Guinea is stated to be of type Nation. The ontology specifies that Nation is a subclass of Civil control feature. Like in many object-oriented languages, this means that every instance of type Nation is also an instance of Civil control feature. Hence, a search for instances of type Civil control feature will include Guinea.

What type a certain individual belongs to can also be inferred. It is possible to define a class to include those individuals that fulfill specific conditions. It is possible to define a “Parent” class which holds all instances of the class “Person” that have at least one child.

Properties may have restrictions such as range and domain. If a property hasChild is defined to have the class Person as both domain and range, an inference engine may infer that both y and x is instances of class Person if an “x hasChild y” statement exists. The specification of range and domain in OWL is inclusive. This means that specifying the domain and range to be class Person for the hasChild property implies that both x and y can be inferred to be instances of class Person. It does not mean that the property is limited for instances of type Person. The meaning of range and domain should not be interpreted as an exclusive type like an argument in a Java method.

OWL also comes with a set of binary relations that can be used to infer new facts. One such relation is inverse of. In the ontology example, there is a property named partOf which is defined to be the inverse of hasPart. Thus, the statement “Guinea hasPart Boké Region” also implies that “Boké Region isPartOf Guinea”. This type of relation can be useful in
many applications, for example when describing geographical relationships. By defining partOf to be the inverse of hasPart the fact will be visible from two different viewpoints. The fact will be visible both in a list of facts about Guinea and in a list of facts about Boké Region. By supporting inverse properties you only need to add a fact at one place, the inverse fact will be inferred and added automatically.

A feature that facilitates the creation of geographical relationships is the possibility to define transitive properties. Consider the following situation:

1. object x is connected to object y by property p,
2. object y is connected to object z by property p.

Specifying that property p is transitive makes the following statement true:

3. object x is connected to object z by property p.

In the example ontology, the “part of” property is defined to be a transitive property. That means that if we add the statement “Boffa Prefecture isPartOf Boké Region”, an inference engine can infer that “Boffa Prefecture isPartOf Guinea”. Figure 7 illustrates the use of transitive properties. The dashed line represents inferred facts.

![Figure 7 – An example of transitive and inverse properties.](image)

The isPartOf property which was defined to be both transitive and to have an inverse counterpart makes it possible to answer complex queries. Since events that occur at Boffa Prefecture also can be inferred to have occurred in Guinea it is possible to answer a search question like “List all organizations that have been involved in political events in Guinea”.

Returning to the example mentioned in Section 2.1, the statement “John is a sibling of Lisa” can be inferred if the ontology defines the relationship sibling as:

If “x sonOf y” AND “y fatherTo z” THEN “x isSiblingOf z”.

Figure 8 shows a graphical comparison between untyped and typed relations including inferred facts.
Supporting class-subclass relationships does not only make it possible to answer complex queries; it also increases the possibility to construct an intuitive GUI. Classes and properties can be displayed as hierarchies. Displaying available classes and properties in a hierarchy reveals information about how classes relate to each other. If the user does not find a suitable class for the object he/she wants to classify, the hierarchy may give a hint on what a suitable, less precise, class could be. If a user wants to classify an event as a political scandal and no such class exists there is a possibility to specify the event to a less specific type. In this case, “political event” may be a valid choice.

Specifying domain and range provides means for creating a user interface with a high level of usability. This can be achieved by filtering the possible options available when the end user wants to make a statement about a certain object. For instance, if the end user wants to add a fact for a Person, only the properties that have the corresponding domain will be accessible.

### 2.5 SPARQL

SPARQL (pronounced sparkle) is a recursive acronym for SPARQL Protocol And RDF Query Language. It is standardized by the RDF Data Access Working Group (DAWG) of the World Wide Web Consortium (W3C).

SPARQL queries are used for querying RDF graphs via pattern matching. Since SPARQL is developed as a query language for RDF, queries are constructed by using RDF triples. The language supports conjunctive patterns, value filters, optional patterns and pattern disjunction. The current version\(^3\) of SPARQL only provides methods for retrieving selected data. No equivalent of SQL insert is available yet.

The result of a SPARQL query is a set of triples or RDF graphs. Resulting sets can be serialized as XML or JSON (JavaScript Object Notation).

SPARQL Protocol for RDF specifies an interface that is supported via HTTP or SOAP. A client can use this to pass SPARQL queries to a remote query processor.

---

\(^3\) W3C Recommendation, 15 January 2008.
SPARQL allows OWL/RDF sources to be queried in a way similar to SQL. In comparison to search queries based on natural text, questions formulated using SPARQL can be very precise. Code-example 2 provides an example of a SPARQL query. The example query asks for events and actors that are involved in these events. The query also includes geographical information.

```
  OPTIONAL{ ? KS:event_has_participant ?Actor. }
  ?Event KS:has_start_date ?date.
}
```

Code-example 2 – SPARQL query example.
3 Semantic wiki for intelligence information management

This section describes the wiki concept, the semantic wiki concept and how a semantic wiki can be used to achieve situational awareness. The last subsection describes related work.

3.1 Wiki

A wiki is a web site where everyone is able to create and edit content. Regardless of context, there is a need for companies and organizations to be able to handle the rapid information exchange that takes place in the world today. Since 1995 when Ward Cunningham released the first wiki software, the wiki has proved to be a successful tool for collaborative information management. The reasons for this are several. Primarily, the wiki has a low entry barrier. The only software needed is a web browser and it is simple to contribute with information. Although some of the advanced features require training, it is easy to use basic features like adding text.

The structure of a wiki is similar to how web pages are organized. Web pages use links to refer to other documents, which makes it easy to access them. This feature is included in the wiki and makes it intuitive to use for anyone that is familiar with the World Wide Web.

In addition to being a fast tool for information exchange, wiki software has also proven to be capable of producing high quality information. The quality of the articles in the well-known wiki-based encyclopedia Wikipedia has been compared to their counterparts in the respected Encyclopedia Britannica [NAT05]. The average number of errors in a Wikipedia article was almost as low as for an article in Britannica. Britannica had on average three errors compared to four in the Wikipedia articles. In other words, the quality of articles collaboratively written by thousands of users can in some cases be as good as articles written by a small number of experts. Figure 9 shows a screenshot of a Wikipedia article describing Guinea.

As the web evolves and new standards emerge, web applications may include features such as drag-and-drop and auto-completion which can make web applications, including wiki software, look and be as easy to use as an ordinary desktop application.
3.2 Semantic wiki

A semantic wiki tries to combine the wiki concept with ideas from the semantic web, creating an environment where information can be used more efficiently. The semantic web vision is about making information understandable by machines. If a machine would be able to understand more of the meaning of the information, by making it explicit and formal, the machine could help us more efficiently. The same ideas are applicable to a semantic wiki.

A semantic wiki is a wiki that has an underlying model of the knowledge described on its pages. The model allows the ability to capture or identify further information (metadata) about pages and their relations [WPSW].

Articles in regular wikis, such as Wikipedia, consist of regular text and untyped hyperlinks. In contrast to a regular wiki, a semantic wiki contains structured knowledge where links are typed. See Section 2.1 for an example of typed and untyped links.

By adding information about what a link between two articles represents it becomes clearer how articles relate to other things. From the viewpoint of a machine, the relation between two articles is no longer an anonymous link; it describes what the link represents and the direction of the link. Directed typed links, which are often represented as RDF triples, can be seen as information objects associated with a wiki article. Having access to triples associated with documents is of great importance when searching for information. A search question like “How many people live in Sweden” may be answered directly with 9,142,817 instead of a list of pages containing natural text that needs to be read in order to extract the information. A semantic enabled wiki can in theory be able to answer complex queries like “List countries and population size for all countries in Europe” and present the result as tables instead of pages that might contain the information searched for. Accessing information objects within a wiki article requires the text to be semantically tagged and is consequently not possible in an ordinary wiki. The information within an article is accessible
in a completely new way in a semantic wiki where pieces of information can be gathered across articles to produce the result of a search.

As described in Section 2.4 new facts can be derived from relationship statements and class memberships. Type/class inference can be helpful to structure the information in a semantic wiki. For example, if a user created a link fatherTo between articles, a reasoner may deduce that both documents describe persons. Even though no one has explicitly said that the documents describe persons, the documents will automatically be included in a search for persons.

The value of the information is increased since it can be used and presented in new ways that improve the accessibility for both man and machine. In summary, a wiki will benefit from semantic techniques in the following ways

- Semantic tagging improves search capabilities and introduces new ways to present search results.
- New facts can be deduced.
- Improved context awareness – the relations between articles are clearer since links are typed.

### 3.3 Semantic wiki implementations

Several prototypes of semantic wikis have been developed [SWSA]. It is important to clarify that there are two different approaches regarding the usage of a semantic wiki. In [KRÖ06] these are described as follows:

“Semantic data for wikis” The goal of this approach is to extend a regular wiki with the possibility to add semantic annotation to improve search capabilities and to add context navigation. The focus of this type of a semantic wiki is on people who already use a wiki but want to add advanced features. Adding machine-readable information may also open up for the use of inference engines. Text content can be extended with semantic information so that at least some parts of the information can be processed by machines.

“Wikis for semantic data” This approach considers the wiki as a collaborative tool for ontology editing. The text in each article can be seen as an informal description of an ontology concept intended for human eyes. The described concepts can in parallel be converted into formal ontology statements by knowledge engineers. Krötzch et al. points out how this approach can be used using the words “The use of wikis for such task facilitates the dynamic, evolutionary development of ontologism, and supports the gradual lifting of informal textual descriptions to formal conceptualizations.” [KRÖ06]

The two approaches suit two somewhat different audiences. However, the boundaries between them are not strict and it would be possible to create a wiki that is capable of editing ontological information and be a user-friendly tool for textual information as well.

The main strength that made the wiki a popular tool is that it is easy to use and this should be taken into consideration when constructing a semantic wiki. Adding semantic capabilities may improve the wiki but it may come with the cost of a decreased level of user-friendliness. Wikis that focus on ontology development have an even stronger need for a user-friendly interface. The ontology editing possibilities can otherwise become an obstacle and make ordinary users feel that the system is difficult to use [KRÖ06].

Semantic MediaWiki (SMW) is an extension of MediaWiki, one of the most popular and widespread open source wiki softwares [SMW]. The intended users are those who already have a MediaWiki installation but want to integrate semantic features. SMW does not come with any inference engine yet and hence cannot infer implicit statements. According to the developers of SMW, “The internal knowledge model of SMW is closest related to OWL
DL, although just a small fragment of the expressive means of this language is actually available within the wiki” [KRÖ06]. SMW is described further in Section 4.2.1.

In contrast to SMW, Ikewiki is not an extension of an existing wiki software [KRÖ06]. Ikewiki is a Java web application that makes use of the Jena semantic web framework. Ikewiki supports different reasoning mechanisms such as OWL Lite and RDFS for basic class inference and Pellet which enhances the wiki with OWL DL support. An important difference between Ikewiki and other semantic enabled wiki software, such as SMW, is that it is not possible to annotate an article with an arbitrary predicate on the fly. The predicate must be defined before it can be used. SMW, on the other hand allows users to annotate articles with arbitrary predicates that may be defined later on. Ikewiki was originally developed to support collaborative knowledge formalization and can be used as an ontology editor. Ikewiki is described further in Section 4.2.2.

In addition to IkeWiki and SMW, there are several other open source projects aiming at creating a wiki with support for semantic annotation [SWSA]. However, most of these semantic wiki implementations have not reached a state of maturity beyond the prototype phase.

### 3.4 Using a semantic wiki to create situational understanding

The semantic web in combination with a wiki shows great potential when it comes to providing a basis for asking knowledge-oriented questions and to presenting the results in a suitable way. As stated in the previous section, typed links make it possible to construct a user interface where it is easier to understand how documents relate to each other. A wiki article about an organization may have typed links to both partners and rivals. Since statements such as “x hasPartner y” can be used when searching, it would be easy to list all partners. Statements (triples) can also be combined to create more advanced search queries such as “list all competitors with more than 1000 employees that my partner has”.

By inserting search results in form of dynamically updated tables into a wiki article, it is possible to create a wiki article that summarizes and displays knowledge in a compact, yet appealing, form.

For instance, in order to get an understanding of the potential threats against a specific refugee camp the following questions could be inserted into a wiki article:

1. Which actors have a motive to perform threatening activities against the refugee camp?
2. Which actors have the resources needed to perform threatening activities against the refugee camp?
3. What events involving the refugee camp has taken place the last 3 years?
4. Are there any internal conflicts between the refugees?

The first and second questions may be hard to answer directly since “motive” and “threatening activities” are abstract concepts. It would probably be necessary to break down the questions into related questions that can give concrete hints about what the motives might be. A starting point may be to list all actors that have been involved in violent activities in the area where the camp is located. Given an expressive enough ontology, it is possible to answer the third question directly. The fourth question might be too abstract to answer directly, but a starting point may be to list the ethnic and religious groups present at the camp. Since the resulting tables are updated every time the article is accessed, the article presents relevant and up-to-date information about the current situation.

The questions listed above reveal different kinds of requirements to consider when designing a semantic wiki focused on situational understanding. Key questions are:
Chapter 3 – Semantic wiki for intelligence information management

1. What kind of tools can be developed to support the creation of queries like those mentioned above in a format that is interpretable by machines?
2. How should an ontology be constructed in order to be as expressive as possible?
3. What software exists that could be used to answer semantic queries?
4. What is the state of the art of combining wiki and semantic technologies?
5. Are there any semantic wiki implementations available as open source that can be extended with additional capabilities?

3.5 Related work

SmartCOP is a research project at Northrop Grumman that tries to use the wiki concept to create a collaborative environment for information sharing in a military context [WHI05]. SmartCOP links wiki pages, known as SmartPages, to each track in the Common Operational Picture (COP). A COP is a display of relevant information, such as enemy troops shared by more than one command. By using wiki principles where everyone (based on permissions) can add or edit content, a powerful organization framework for knowledge management can be offered. According to the developers of SmartCOP, the wiki concept is suitable when it comes to creating an environment that promotes user interaction and collaboration across the enterprise. They point out that chat, e-mail and shared file servers do not support discovery and search across the enterprise as well as a system based on wiki principles. The developers describe SmartCOP as:

“SmartCOP is self-organizing, self-sustaining, and self-repairing. SmartCOP is the battlespace equivalent of Wikipedia (www.wikipedia.org), a profoundly successful effort to create a free, web-based, community-driven encyclopedia of everything – ambitious in purpose and unique in design. The marquee trait of Wikipedia (and SmartCOP) is the decentralized and self-sustaining nature of the capability.”

If the system reaches a critical mass of users, the developers believe that information achieves a high level of quality since errors quickly can be detected. SmartCOP supports templates, a feature available in other wiki implementations as well. Templates may help a user to quickly create a wiki page and give directions regarding what kind of information that should be added to the page. A template for a hospital page may include a section where the hospital’s capacity is described, whereas a template for hostile warships may include sections where information on weaponry can be entered. Another important feature that SmartCOP supports is alerts. The user can be notified when information is added, modified or deleted. The developers argue that feedback mechanisms are especially useful in discussion sessions: “In real-world operations, it is not unusual for a decision-maker or Subject Matter Expert to be engaged in numerous chat sessions that must be constantly monitored for new information, forcing the user to work hard for the system. In a SmartPage, alerts can be established for the full range of activity and interaction with the page, empowering the system to work hard for the user.”

In [MÅR06] it is described how a wiki can be utilized for enhancing situational awareness in both civil and military command and control situations. First, the authors point out that a wiki could be useful in large-scale civil crisis scenarios such as natural disasters or terrorist attacks. Their first example on how to use a wiki in these types of scenarios is a non-public wiki where a group of trusted users scan media for news and update the wiki with relevant information. Another example is a public wiki where anyone can add content. The latter example requires a quality marker approach to ensure information quality. According to the authors, a wiki would also be a useful tool for information sharing in military operations other than war (MOOTW). In this kind of operation, it is common that armed forces arrive late to the conflict area compared to other organizations. Non-governmental organizations often reach conflict areas before large military organizations have mobilized. If the
organizations that reach the area first collect knowledge about the situation in a wiki, the information could easily be shared with the armed forces using inter wiki links\textsuperscript{4}. In addition to providing examples of how wikis can be used in a military context, Mårtenson et al. also discuss how a wiki can be improved. They point out that in command and control situations, wiki add-ons that extend the wiki with the ability to include maps and geographic locations are of great use. Semantic capabilities can also be of great value in this context since it may enhance situational awareness. A semantic wiki will be able to answer questions like “list all friendly units participating in a certain mission in a certain region”.

\textsuperscript{4} Inter wiki links are links between related wikis.
4 Design

This chapter presents a design solution that addresses the questions and conditions presented in Chapter one. To begin with, the requirements are described. Thereafter an evaluation of existing semantic wiki implementations follows. The last section describes the proposed architecture. The following chapter, Chapter 5, will describe how the design was implemented.

4.1 Requirements

4.1.1 Introduction

Our hypothesis is that an analyst may benefit from a semantic wiki in three ways:

1. Improved search capabilities.
2. Automatically updated information objects – implemented as wiki articles with dynamic content.
3. Context navigation – improved possibilities to see how documents relate to each other.

We also believe that it would be beneficial, to some extent, to utilize ontologies for supporting:

4. Inference of new facts based on rules and constraints.
5. Consistency checking.

The five features described above represent the key features that we believe should be supported in a semantic wiki. In Section 4.2 three semantic wiki implementations are evaluated based on how well they integrate semantic technologies to support the above-mentioned features. The basic features are described in further detail in the following subsections.

4.1.2 Search capabilities

By utilizing semantic web techniques, search can be improved in many ways. An ontology can be used to expand terms in a search query to more general terms and include synonyms [McG03].

By supporting semantic tags with terms defined in an ontology it is also possible to formulate precise questions and get a precise answer. A search question like “How many people live in Sweden” may be answered directly with 9,142,817 instead of a list of pages containing natural text that has to be read in order to extract the information. The system needs to translate the question into a form that a semantic database can understand, for instance SPARQL, or provide an interface that helps formulate the equivalent semantic search question. Regardless of how the system handles searches, it must not be hard to formulate a search question that makes use of semantic tags.

Requirements

- The system must provide a search method that makes use of semantically tagged information.
- The system must be able to present the result of a search question in a useful way.
• The system must provide an (graphical) interface in order to support the creation of advanced queries.

### 4.1.3 Articles with dynamic content

As described in Section 3.3, a semantic wiki makes use of semantically tagged information to provide a way to define a dynamically updated part in a wiki article. By supporting dynamic content, it is be possible to insert a dynamic list of actors active in a certain region. Without the use of semantic tags, such lists would be more difficult to create. The proposed wiki should be able to handle wiki articles with parts that are dynamically updated. The wiki should also provide an interface that makes it easy to insert the dynamic objects.

**Requirements**

• The system shall support articles with dynamic content based on a semantic search query.

• The system GUI should support insertion of dynamic content.

### 4.1.4 Context navigation

An important aspect when enhancing a wiki with semantic technology is how semantic information is presented and how it is used to connect wiki articles in meaningful ways. The navigation between articles could be improved by using semantic information to describe what kind of relation the link represents in contrast to just stating that there is a link. This aspect is commonly known as *context-based navigation* [DEL06].

**Requirement**

• The system shall provide GUI components that make use of semantic content in order to improve the navigation between articles.

### 4.1.5 Inference of new facts

Supporting inference of new facts is an important feature. New facts can be deduced using a reasoner.

Even though the ontology might include complex restrictions, it is not certain that a reasoner is able to deduce all statements that are possible in theory. As described in Section 2.3.3 the web ontology language (OWL) has three sublanguages with different level of expressivity. An ontology with a high degree of expressivity requires a powerful reasoner that supports all constructs used. A powerful reasoner requires a lot of computing resources in terms of memory and computational power. For this reason, the requirements regarding inference capability cannot be determined in advance. Being able to infer facts is a requirement but to what extent is an open question.

**Requirement**

• The system shall, to some degree, be able to infer new facts based on restrictions and rules defined by an ontology.

### 4.1.6 Consistency checking

Ontologies can contain information defining under which circumstances a property can be used. For example, a property named ‘population size’ may be restricted to only having values of type integer. Another example is that an individual can be restricted to be of one type only. Stating that an object is of multiple types can be very useful in some situations but can also be a source of contradictions. An obviously false fact that can easily be detected by a human might be that a certain thing in the knowledgebase is said to be a person while at the
same time is said to be a country. By supporting consistency checking, it is possible to inform users that the knowledge base contains such contradictory facts.

**Requirement**

- The system shall, to some degree, be able to find inconsistencies.

### 4.2 Evaluation of existing software

The goal of the evaluation is to examine how well the currently available open source wiki systems meet the specified requirements. We examine how well the systems provide user interfaces to support use of semantic technologies and how the semantic technologies are integrated. In addition, we examine the possibilities to extend the systems by looking at the level of documentation and add-on support.

At the moment there are about 30 semantically enabled system prototypes [SWSA]. The ones that passed the basic requirements (accessible for download, running demo, etc.) have been studied in further detail. Out of 30 candidates, the three most promising were chosen to be evaluated and have been installed on a local test server.

**User interface**

Given that the intended user is an intelligence analyst and not a knowledge engineer familiar with ontologies and related concepts, an intuitive user interface is a requirement. Many of the state of the art wiki systems make use of the wiki-syntax [KRÖ06] [SHA06]. The wiki-syntax is used to format the content. Using wiki-syntax to format text does not require any special tools in contrast to HTML that is often hard to read. However, in some respects wiki-syntax struggles with the same problems as HTML. It is useful for text formatting, but for creating more advanced content such as tables, the wiki-syntax is hard to read and maintain[DEL06]. For this reason, evaluating the level of usability is important.

**Use of semantic technologies**

At present, there are no uniform thoughts on how to integrate wikis and semantic technologies. A common approach is to enhance a wiki with semantic technologies in order to improve the capabilities of structuring and relating wiki articles to each other. The goal is to extend already existing wiki systems to make use of the new and powerful features introduced by semantic technologies. Another approach is to view the wiki as a visualization and collaborative editing tool for an ontology. These two rather different points of view result in implementations that are quite different. From our perspective, the implementation approach does not matter as long as the wiki can be used for the intended purposes or be modified to suit our needs.

A concrete implementation issue that different implementations solve in different ways is how to deal with the new concepts introduced by the semantic web. For instance, every object tagged with semantic information using the OWL language is a so-called resource. Should this concept be introduced in the wiki or is it just a technical term that should be hidden from the end user?

How the wikis integrate semantic technologies can be divided into several parts. One part consists of how the wiki visualizes the new features and another part consists of the storage and inference layer. These parts can be evaluated separately.

An important feature of the semantic web is the ability to ask complex queries. What kind of queries that the system is able to answer depends on the inference capability. Another aspect to consider is what kind of support in the form of GUI tools that the end user has access to in order to create the queries.
Customization support

A standard wiki system, such as MediaWiki, must often be customized in many ways in order to suit the needs. Support for customization is therefore important to consider when evaluating existing wiki implementations. It is also important that the system is well documented.

Customization may also be supported in different ways. The visual part of the GUI can often be customized using CSS if the wiki makes use of modern web standards. However, changing graphical details such as colors is not the hard part when it comes to customization. Changing or adding functionality such as map integration requires a system that has a structured and well-documented support for add-ons.

4.2.1 Semantic MediaWiki

Semantic MediaWiki (SMW) is an extension of the popular MediaWiki system [KRÖ06]. In theory, SMW could be used in combination with another MediaWiki extension for WYSIWYG\(^5\) editing such as the FCKeditor. However, at present, the extensions are not compatible. In order to support navigation between wiki articles in SMW there is a fact box at the bottom of every article. The box contains all incoming and outgoing relations. Figure 10 shows a screenshot of what an article may look like.

SMW supports the creation of articles with dynamic content. The syntax for dynamic queries is not based on any existing standard such as SPARQL, but it is similar to wiki-syntax. SMW itself does not come with any tool to support creation of such queries. However, the HALO extension comes with a tool that supports the creation of semantic search queries\(^6\).

SMW supports import of OWL DL ontologies in RDF-serialization. However, because SMW does not include a reasoner an imported ontology might not work as expected since the use of domain and range restrictions require such a capability.

The level of documentation available for SMW is acceptable. The biggest advantage of SMW is not the documentation provided by the developers themselves. More important is the fact that SMW is a MediaWiki extension, which means that it follows the structure of such an extension. Documentation describing how to build extensions for MediaWiki has reached an acceptable level with regard to usability. The documentation is contained in a wiki format, which encourages users to contribute. This makes the documentation alive and up-to-date.

---

\(^5\) What You See Is What You Get, content displayed during editing appears similar to the final output.

\(^6\) The HALO extension was not released at the time when the evaluation was done.
4.2.2 Ikewiki

Ikewiki [SHA06] is, in contrast to SMW, not based on any existing wiki software. The developers motivate this by stating that in order to create a useable semantic wiki some key concepts must be changed and these changes affect the wiki in a way that makes a complete rewrite necessary.

Ikewiki offers a modern GUI through heavy use of AJAX\(^7\) techniques. This makes the system feel more like an ordinary application. However, using the latest technology does not necessarily imply a high level of usability. At current date\(^8\), the user interface is dependent on features only available in the latest versions of modern web-browsers due to the heavy use of AJAX techniques. Even if Ikewiki is used with the latest browser-software available, there are some bugs in the GUI\(^9\).

In order to support navigation between wiki articles there is a statement box on the right-hand side of every article. The box contains all incoming and outgoing relations, including inferred ones. Figure 11 shows a screenshot from the Ikewiki user interface. Notice that relations can be hidden by clicking on the “-” symbol. This is useful when an article has many outgoing or incoming relations. Ikewiki supports inline SPARQL queries that can be used to create articles with dynamic content but does not come with any GUI tool to support the creation of such queries.

---

\(^7\) AJAX (Asynchronous JavaScript and XML) is a group of web development techniques used for creating interactive web applications.

\(^8\) 2007-08-12.

\(^9\) Ikewiki has been updated since the evaluation was done. Today, it supports all modern well-established browsers.
Ikewiki makes use of the semantic web framework Jena as a backend. Jena offers a storage and inference engine that supports different kinds of ontology languages such as RDFS and OWL\cite{SHA06}. Ikewiki can also be used with the Pellet reasoner, which has full DL support. By integrating Jena, Ikewiki is able to infer new facts as well as doing consistency checks.

The documentation available for Ikewiki is at the moment sparse. Even though there are guidelines that describe how different parts of the system works, there is need for more information on a detailed level. The system can be extended with new features since the source code is available. However, Ikewiki has no architectural support for extensions at the moment\footnote{By architectural support we mean a well-documented approach that supports extensions without changing source code.}.

\begin{figure}
\centering
\includegraphics[width=\textwidth]{ikewiki_article.png}
\caption{Screenshot from an Ikewiki article describing Bilberry.}
\end{figure}

### 4.2.3 Makna

Makna is a semantic wiki based on the JSPWiki\cite{DEL06}. Makna should be regarded as a stand-alone application, even though JSPWiki supports plug-ins. Makna uses Jena as a backend for inference and storage. By integrating Jena, Makna is able to infer new facts as well as doing consistency checks. The user can choose to switch off inference. This is a valuable feature since reasoning may require a lot of computing power and is not always necessary in order to get what you look for. Jena itself has support for SPARQL queries but Makna has not integrated this feature yet.

Being able to ask complex semantic questions is an important feature of semantic wikis. Makna provides a tool called \textit{Semantic search}, for asking such queries. The tool lets the user select an article or a class and then lets the user choose properties from a list. The list of properties is filtered based on the previous selections. This can be seen as a first step towards creating a more advanced and powerful tool for creating semantic queries.
Even though Makna has some promising support for the creation of semantic search queries, it offers no support for inserting dynamic content into a wiki article.

Makna has a statement box on the right-hand side of every article to support context navigation. The box contains all incoming and outgoing relations, including inferred ones. Figure 12 shows a screenshot of the user interface. Notice that, in contrast to Ikewiki, it is not possible to filter the factbox on the right-hand side.

![Figure 12 – Screenshot from a Makna article describing Humphrey Bogart.](image)

### 4.2.4 Systems for inference and storage

In order to infer new facts and make consistency checks, the wiki must be extended with a reasoner. Having the semantic wiki in mind, this section describes the strengths and weaknesses for some of the available systems for storage and inference of semantic information. The ontology in combination with the inference engine determines to which degree new facts can be inferred. From one perspective, determining how semantic information should be stored and to what degree inference should be supported is the most important decision to make when developing a semantic wiki. The storage backend is the most important part of the system if reasoning capabilities are a requirement. Consequently, choosing subcomponents for a storage and inference backend is important and perhaps more important than choosing the wiki to extend. The focus differs between different systems and shift from scalability to computational completeness. One significant difference is if the focus is on ABox or TBox statements, as described in Section 2.2.

Although two out of the three evaluated wikis make use of the Jena semantic toolkit, Jena may not be the best-suited storage and inference layer. Other candidates are Pellet [PELLET], FaCT++ [FACT], KAON2 [KAON] and RACER [RACER]. Yet another framework for storage and inference of semantic data is Sesame [SESAME], which focuses on storage but can be used with various reasoners such as OWLIM [OWLIM].
KAON2 differs in some respects from the other mentioned systems. It uses a different type of algorithm compared to other reasoners such as Pellet\textsuperscript{11}. The algorithms used by KAON2 focus on knowledgebases with large amounts of ABox statements.

Different inference engines support different levels of reasoning. The Jena framework is bundled with several types of reasoners that can handle ontologies up to OWL Lite. It can also be used in concert with the Pellet reasoner in order to support OWL DL [JENA].

KAON2 supports OWL Lite and a subset of OWL DL. FaCT++ and RACER support OWL DL under some restrictions. OWLIM supports most of OWL Lite [OWLIM].

In order to formulate queries that can be answered by the storage backend, systems need a query language. Different systems support different kinds of query languages. The systems mentioned in this subsection have support for SPARQL except for FaCT++ where the supported query language is unknown. However, neither of them supports all features in the SPARQL specification. KAON2 lacks support for using variables in the predicate position and does not support the OPTIONAL pattern. Ontology and query support for the above mentioned systems are summarized in Table 2.

<table>
<thead>
<tr>
<th>System</th>
<th>OWL Lite</th>
<th>OWL DL</th>
<th>Query support</th>
</tr>
</thead>
<tbody>
<tr>
<td>Jena</td>
<td>X</td>
<td></td>
<td>SPARQL*</td>
</tr>
<tr>
<td>Pellet</td>
<td>X</td>
<td>Subset</td>
<td>SPARQL*</td>
</tr>
<tr>
<td>KAON2</td>
<td>X</td>
<td>Subset</td>
<td>SPARQL*</td>
</tr>
<tr>
<td>FaCT++</td>
<td>X</td>
<td>Subset</td>
<td>?\textsuperscript{12}</td>
</tr>
<tr>
<td>RACER</td>
<td>X</td>
<td>Subset</td>
<td>SPARQL*</td>
</tr>
<tr>
<td>OWLIM/Sesame</td>
<td>Subset</td>
<td></td>
<td>SPARQL*</td>
</tr>
</tbody>
</table>

\textsuperscript{*} Does not support all features in the SPARQL specification.

4.2.5 Conclusions

One of the initial requirements was that the wiki should be able to provide tools for performing searches that utilize semantic information. In theory, questions could be formulated using natural language [KATO]. However, transforming natural language into a formal language that the semantic repository understands is not an easy task [KATO]. An alternative to supporting queries in natural language is to allow users to enter SPARQL queries directly via an interface. Creating SPARQL queries requires knowledge about the SPARQL syntax and basic knowledge about RDF. One cannot expect that the user in mind is familiar with this. Users need tools to guide them through the process of creating semantic search queries. Even though some of the mentioned wiki systems support inline queries and some have interfaces that can guide the user, none of them has all required components. In order to meet the requirements the wiki needs to support a query language, support the user in the creation process of such queries and provide a way to reuse search queries so that they can be used as inline elements in a dynamic article.

The level of inference needed is still an open question but we assume that inference is needed to some degree. Jena offers a flexible and well-documented framework that supports inference in various degrees and has support for SPARQL. From our perspective, Jena with an optional connection to Pellet in order to support OWL DL offers the most stable platform on which a semantic wiki can be developed.

\textsuperscript{11} In contrast to implementing a tableaux based algorithm, KAON2 uses an algorithm that reduces a DL knowledge base to a disjunctive datalog program.

\textsuperscript{12} Information about whether or not FaCT++ supports SPARQL could not be found.
It is well-known that the user interface affects the overall experience for the end user. A wiki that is not intuitive and easy to use might be regarded as useless, no matter how much reasoning-power it carries under the surface. MediaWiki, upon which SMW is based, offers many useful extensions that can add new features and improve the user interface. Extensions available for MediaWiki, such as a WYSIWYG editor and extensions that allow users to display resulting sets on a timeline, are of great help when trying to maximize the user experience.

We believe that a semantic wiki implemented as an extension to MediaWiki that uses Jena as a backend for storage and inference ought to be the most promising approach. Unfortunately, no such extension exists. Our intention is to develop such an extension. In order to develop a semantic wiki that integrates tools for reasoning provided by the Jena semantic web Framework a system architecture needs to be formulated. In the following section, we propose such an architecture.

## 4.3 Proposed architecture

This section describes an overall system architecture for a semantic wiki based on the requirements listed in Section 4.1. The purpose is to present an architecture that addresses the problems that arise when a wiki is extended with semantic technologies. The architecture will later act as a guideline for implementing a prototype.

### 4.3.1 Overview

The core problems addressed are:

1. What building blocks should a semantic wiki consist of?
2. How and where should the semantic information be stored?
3. How should the inference engine be integrated into the wiki?

Semantic information can be stored internally in many ways. The standard way of representing such information is to use triples. How to store these triples is currently an open issue and it is not self-evident that the most efficient storage method is a relational database such as MySQL. A wiki that has the ability to infer new statements, which is one of the key benefits of adding semantics, also requires more computer power. Storage of wiki articles and semantic information in a close manner may not be the optimal solution. Separating pure text data from semantic information in the form of triples may be a promising alternative. Even though the goal is to tag every article semantically within the wiki there are reasons to believe that a significant amount of articles will be left without semantic tagging since tagging is time-consuming. Tools for automated classification and tagging of articles can be integrated to improve the process, but as a first step towards creating a semantic wiki such tools should not be assumed to exist. A loose coupling between the wiki and the storage and inference layer opens up for the possibility to place the two systems on separate servers with different configurations. A subsystem, a semantic bridge, within the wiki that handles communication between the wiki and the semantic storage and inference layer is our proposition to address this problem. A graphical description of the semantic bridge can be found in Figure 13.
Another important decision to make when choosing system architecture for a semantic wiki is to decide if the wiki should consist only of semantically tagged information or if the wiki should support articles with no semantic information at all. The latter perspective is preferable if semantic support is added to an already existing wiki with a large amount of articles.

The wiki GUI should be improved in order to support the new semantic annotation possibilities. The GUI subcomponents access information via the semantic bridge.

In summary, the basic building blocks that form a semantic wiki are:

1. A standard wiki system.\(^\text{13}\)
2. A set of GUI components that utilize and support semantic annotation.
3. A subsystem that is responsible for the storage of the semantic information.

A graphical description of the building blocks can be found in Figure 14.

\(^{13}\) Standard wiki functionality includes version management, support for multiple users, etc.


4.3.2 User interface

Since the wiki prototype is a web application it is necessary to specify a strategy for development of a user interface suitable for the user. The interface should make use of AJAX techniques, which can reduce loading times and make the user feel more productive. Accessing, especially inferred, semantic information may not be as fast as accessing standard wiki information. By using AJAX techniques such information can be accessed easily on demand by the user when needed. A system can support AJAX techniques by utilizing software hooks. This is accomplished through HTTP requests with a parameter that disables the HTML code for the design framework and only returns the actual text content. MediaWiki has successfully implemented support for AJAX-hooks [MW].

Rendering pipelines is an important concept when creating a flexible and extendible wiki. Many wikis make use of wiki-syntax for formatting and for adding content other than text into articles. During the rendering process of an article, the wiki-syntax is replaced by the corresponding HTML code. Wiki extensions can use this process to intervene at an appropriate place in the rendering process pipeline. A rendering pipeline that is accessible for extensions is therefore helpful when creating an extendible system architecture. For example, it makes it possible to insert result tables generated from dynamic queries. A graphical example of a rendering pipeline is available in Figure 15.

![Figure 15 – An example of a simplified rendering pipeline.](image)

Rendering pipelines can also be used to create different data views. Examples where different views of the same data can be useful are dynamic queries that can be presented as both tables and timelines.

In summary, the prototype should follow the following architectural principles:

- **AJAX-hooks** should be used to create a dynamic interface.
- **Rendering pipeline hooks** should be used to extend the wiki with new functionality.
4.3.3 Semantic storage and inference layer

The semantic storage and inference layer (SAIL) is responsible for storing and inference of semantic information. The SAIL supports commands for retrieving and adding semantic information. The wiki is integrated with the SAIL through the semantic bridge component. The semantic bridge should be constructed so that it can be modified or extended to support different types of SAIL systems. Since the support for standard languages for retrieving and updating semantic information varies between third party SAIL systems, commands might need to be translated. Figure 16 gives a graphical overview of the command mapping process.

The SAIL may be implemented as a separate application. The SAIL should be constructed in such a way that it can both be tightly and loosely connected to the wiki software. A wiki with an integrated SAIL may have less overhead, whereas a wiki with a separate SAIL server may have more communication overhead but can be placed on more powerful hardware dedicated for inference and storage.

The wiki and the SAIL might be written in different programming languages such as PHP and Java. Many programming languages support tight integration to other programming languages. Whether or not the wiki and the SAIL can be tightly connected depends on the chosen implementations. For this reason, it is recommended that the communication between the SAIL and the wiki is based on XML.

4.3.4 External sources

In the military intelligence context, there are many sources of information [FMUNDH]. Given this, it is important to determine how the architecture should support external sources. This section describes a general approach to integrating external sources.

Important aspects when dealing with external information sources are interoperability and change management. An important question is how metadata can be exchanged between systems and what happens if the information structure is altered solely at the external source or vice versa.
One approach is to provide external systems access to semantic info via the wiki system. Another approach is to enable direct access to the SAIL. Figure 17 gives a graphical representation of the two approaches.

Regardless of approach, one must deal with how semantic information should be handled in the SAIL. If new information is added via the semantic wiki and thereafter processed and inserted into the SAIL, questions arise on how to transform any existing meta or semantic information originating from the external source. For this reason, every external source may have a special transformation component that takes care of the transformation in terms of converting any existing metadata into semantic information interpretable by the wiki. Figure 18 gives a graphical representation of the transformation process.

In a military context, it is likely that a wiki system is part of a bigger system where the wiki provides one way out of many to access information. In these cases, when semantic information is shared between systems, the second approach is more suitable.

Which approach to choose depends on how the external information system structures information. If the external system makes use of semantic data, the second approach would be preferable.
5 Implementation

This chapter describes the prototype that was implemented based on the system design described in Chapter 4. The prototype is named SMWE, which is an abbreviation for Semantic MediaWiki Extension. The installation of MediaWiki plus the SMWE extension was named Semantic MilWiki.

5.1 Overview

MediaWiki was chosen as the wiki platform to be extended in order to meet the requirements described in Section 4.1. The wiki platform will be extended with the FCKeditor in order to support WYSIWYG editing of wiki articles. The storage and inference subcomponent of SMWE will be based on Jena semantic web framework.

The SMWE extension will extend the MediaWiki software to support:

- Ontology import/export.
- Creation of semantic links between articles.
- Easy creation of semantic search queries through the SPARQL wizard.
- Article classification.
- Dynamic articles, i.e., the ability to insert a search query into an article.
- Inference of new facts based on rules defined in imported ontologies.
- Ontology editing, i.e., new classes and restrictions can be added to imported ontologies.
- Display of semantic information, i.e., semantic information will be visible in a box on the right-hand side of the article.

5.2 Using ontological information

Each article in the semantic wiki can be assigned to a specific class defined in an imported ontology. When specifying a class, semantic information is added to the article. Assigning a class to an article often results in new implicit information. For example, assigning the class President to an article describing a Person may result in new implicit information. In this case, the article would also be tagged with type “Person” since President is a sub-class of Person.

Users are free to add semantic information such as relations and classes to articles, but altering of ontological information such as class definitions is restricted to administrators. This approach implies that the SMWE is the type of semantic wiki that makes use of ontologies in order to extend the wiki with valuable new functions in contrast to being a pure ontology editor (see Section 3.4 for more information). In contrast to other semantic wiki implementations, SMWE has a closed approach. Only properties and classes defined in the imported ontology can be assigned to an article.

In order to prevent inconsistency, available properties that can be assigned to an article are filtered in a way that only relevant properties are displayed. In addition, only articles that match the requirements of the property will be available to choose as an object. For example, if a user wants to make a statement that a certain Person has a role in a specific Event, only
5.3 Extending MediaWiki

As mentioned in Section 4.3.2, MediaWiki supports software hooks that allow extensions to intervene in various processes in the system. This feature is one of the cornerstones that the SMWE makes use of to integrate new functionality in MediaWiki. Functions, known as event handlers, assigned to hooks will be called at predefined places in the main MediaWiki code enabling extension developers to add new functionality. Hooks are used by SMWE to add functionality such as dynamic content based on a semantic query.

MediaWiki extensions, including SMWE, also utilize the special pages feature to extend the software. A special page is a page within the wiki that contains more advanced content than just plain text. Special pages often include system management functionality such as modifying user permissions. The SPARQL Wizard (described in Section 5.7) and the ontology editor (described in Section 5.5) are implemented as special pages.

5.4 GUI

In order to see semantic statements related to a specific article, a new GUI component has to be added to MediaWiki. A common way to show this kind of information is through a box, commonly known as the fact box. SMWE also makes use of a fact box. It is placed on the right-hand side within an article. The fact box displays both incoming and outgoing relations.

Figure 20 shows a screenshot where the fact box can be seen. The fact box improves the wiki by explicitly displaying the relationships between the articles, which gives a better contextual overview.
The free open source editor FCKeditor was integrated with MediaWiki and SMWE. The FCKeditor replaces the standard MediaWiki editor with a WYSIWYG tool, which has an interface similar to standard word-processing applications. By extending the wiki with this editor, the entry barriers decrease since a user does not need to be familiar with wiki-syntax. A screenshot showing a wiki article in edit mode using FCKeditor is shown in Figure 21.
5.5 Ontology editor

Our semantic wiki prototype has a restrictive perspective when it comes to altering the ontology by adding TBox statements. However, it is still possible to alter the ontology using the wiki ontology editor. This editor lets a user add new classes and properties. Altering the ontology, for example changing the class hierarchy, should normally be restricted to advanced users. However, no such limitations are implemented in the current version of SMWE. Figure 22 shows a screenshot from the ontology editor interface when adding a new property.

Like other features that SMWE adds to MediaWiki, the ontology editor is implemented as a special page.

![Figure 22 – Screenshot showing the ontology editor.](image)

5.6 Storage and inference of semantic information

As stated in the design section, Jena was chosen as storage and inference layer. SMWE is implemented using the web application script language PHP. Jena, on the other hand, is a Java application. Hence, the question of how to integrate a Java and a PHP application had to be solved, even though it was not the focus of this master’s thesis.

PHP and Java can be tightly integrated in several ways. One approach is to use a Java based web application server such as Resin [RESIN], which enables use of Java commands directly from within PHP. A similar approach is to use the Java/PHP Apache extension, which also supports the use of Java commands [PHPJ].

Every time you access an URL a web server takes the HTTP request parameters and produces an output. HTTP is a stateless protocol, which means that the host does not need to
keep information between requests. PHP produces an ordinary web page and follows the same principles. Even though it would be possible to run Java commands from within PHP, the stateless characteristics of the PHP script would imply that the Java based server would have to be started every time the page is accessed. This would be an inconvenient solution. It would be better if the storage server was online all the time.

We chose the approach to regard the wiki and the SAIL as two separate applications, which do not require a tight connection. A simple socket communication was an easy and straightforward approach to solve the problem. The socket connection, where commands and information are encapsulated using XML, can easily be substituted for something more advanced if needed.

### 5.7 SPARQL Wizard – A tool for creating dynamic queries

As mentioned in the requirements section, a tool for creating dynamic queries is needed. In contrast to other semantic wikis, the end user of the prototype presented in this report should not have to be familiar with semantic technologies in order to use the system. For that reason, development of an intuitive user interface for creating semantic search queries has been a goal. The dynamic queries are based on SPARQL.

The goal has not only been to create a user-friendly tool to help users create SPARQL queries with valid syntax. The goal has also been to help users in the creation process by hiding non-valid options and using color codes to indicate what to do next. Only options that “make sense” are available. Technically, the options are filtered based on domain and range restrictions (described in Section 2.4) defined in the ontology. The user interface has green and red buttons to guide the user. Buttons that can be used to create the query has a green color. Buttons for deletion has a red color.

The key features of the SPARQL Wizard are summarized in Table 3.

<table>
<thead>
<tr>
<th>Key features</th>
</tr>
</thead>
<tbody>
<tr>
<td>Construction of queries by</td>
</tr>
<tr>
<td>Selecting class/type for variables</td>
</tr>
<tr>
<td>Selecting properties/relations</td>
</tr>
<tr>
<td>Load and save queries</td>
</tr>
<tr>
<td>Active filters</td>
</tr>
<tr>
<td>Generating raw SPARQL queries</td>
</tr>
<tr>
<td>Displaying results</td>
</tr>
</tbody>
</table>

Table 3 – SPARQL Wizard key features.

Figure 23 describes a typical workflow when creating a semantic search query.
1. **Create variables:** Create a new variable and assign a type to it. The type is selected from a hierarchical list, which makes it easy to find an appropriate type.

2. **Add relations:** For each variable, select a relation to another variable. The available properties are filtered so that only those that make sense are displayed. The filter is based on the domain and range restrictions stated in the ontology.

3. **Generate query:** The semantic query is now visible in SPARQL format. If the user has adequate knowledge about SPARQL further adjustments can be made directly to the generated query.

4. **Execute query:** The query result can be viewed directly in the wizard tool. The result table will display the same result as if the query had been inserted into a wiki article.

5. **Save query:** The query can be saved for later use, typically for being inserted into a wiki article.

Figure 23 – SPARQL Wizard workflow.

Based on the workflow described in Figure 23, a search query resulting in a list of important persons and their religion, ethnicity and organizational belonging can be formulated by following the steps described in Figure 24. Figure 25 provides a screenshot based on these steps. Figure 26 shows what the result may look like.

1. **Specify, by creating variables and assigning types, what the result table should contain.** In this case, the list should contain a person, a religion, an ethnicity and an organization.

2. **Relations for variables can be defined in the corresponding section below the variable input section.** In this case, we are specifying relations that the person has. Each person we are looking for is typed as ‘important person’ in the database. Given this, we use the relation `has_importance` to specify that the person must be of type `Big_man`. We also add the relations `has_ethnicity`, `has_religion` and `is_leader_of`. The three latter relations are specified to be optional since this kind of information might not always be available.

Figure 24 – Workflow for creating a SPARQL query for important persons.

Figure 25 – SPARQL Wizard interface.
5.7.1 Component architecture

The SPARQL Wizard component is built using AJAX techniques. The architecture is based on a Model-View-Controller design pattern. Figure 27 shows the components of the SPARQL Wizard and how they are connected to each other. The queries are stored in a MySQL database in an XML format.
5.7.2 Integration with wiki

Saved search queries can be inserted into wiki articles as dynamic information objects. When an article is accessed, the system will execute the search query and display an updated version of the result. An example of an article where a query is inserted is shown in Figure 28. The dynamic part of the content is represented by a box labeled with the same name as the query had in edit mode, see Figure 29.

![Figure 28 – Screenshot showing a wiki article with dynamic content.](image)

![Figure 29 – Screenshot from the editing mode of an article containing dynamic content.](image)

5.7.3 Possible SPARQL Wizard improvements

The SPARQL Wizard is an attempt to provide a tool that makes it easy for the user to create advanced semantic search queries. The wizard can still be improved in numerous ways, for instance by providing support for different types of result visualizations, by supporting all constructions of the SPARQL specification, and by providing a graph based query visualization. A graph based visualization of the search query may give a better understanding of how the query elements are related to each other.
6 Experiments

This chapter describes how our prototype was tested. It also includes a discussion about the experiment setup.

6.1 Experiment setup

The prototype was demonstrated and evaluated within the Knowledge Support Limited Objective Experiment 5 (KS LOE). The KS LOE was a one-week national workshop setup with multinational participation, where the different parts of the KS concepts were discussed and worked on. Using KS concepts and techniques, the experiment participants were supposed to analyze the economical situation within a specific country with respect to a certain industry. The information about the case was available in form of Power Point-slides and Word-documents, stored in a document management system.

Even though the KS LOE was about evaluating the whole KS concept, we focus here on the role of the semantic wiki. The semantic wiki, and the information within, was available for the participants as an alternative source of information. Since the semantic wiki was only a complementary part, it was hard to influence the experiment setup as a whole.

The wiki did not share document database with the primary document management system used. Instead, information was manually extracted from Word documents and Power Point slides, and converted into semantically tagged wiki articles. Support for information sharing between the wiki and the primary document management system is possible in theory but was not implemented. Access to the document system used in the KS experiment was not granted within reasonable time before the workshop started.

The wiki could be used as a tool for information searching or for presenting information. 2 to 3 persons had access to the semantic wiki and were available as a support team, which the participants could ask for information. The team is referred to as “semantic support team”.

Potential benefits of the semantic wiki were evaluated in two ways:

1. Relevant examples based on the workshop background material that showed the potential of a semantic wiki was demonstrated in order to get feedback.
2. The semantic wiki was used as a complementary alternative information source during the task-solving part of the experiment, to examine if the wiki could present or provide an alternative situation picture based on the same background information.

Whether or not a semantic wiki is useful for enhancement of SA depends heavily on how and to what extent the information is semantically tagged. A semantic wiki without information tagged in an adequate way is of no more use than an ordinary wiki. For this reason it was necessary to do some preparations beforehand. The case material was manually analyzed in order to extract semantic information about persons, organizations and different types of events. At the time when the preparation step begun, there was a lack of detailed information about the kinds of tasks the participants were supposed to solve. Hence, it was hard to know in advance what information that would be beneficial to tag. Since tagging information semantically is time-consuming it was not possible to tag everything within the given time frame. The information extraction resulted in a relatively small number of tagged articles and structured excel documents. Each line in the excel documents represented a statement, which easily could be added to the semantic wiki later if needed.
6.2 Demonstrations

Before the actual task-solving part of the experiment begun, the semantic wiki was demonstrated. The purpose of the demonstration was to show some features that might be useful later in the task-solving part of the experiment. The demonstration included:

- Creation of a semantic search query.
- Creation of a wiki article where static and dynamic content was mixed. The query was inserted as a part of the article.
- A short demonstration of how changes of data affect the dynamic part of the article.

The result of the semantic search query that was created in the demonstration was a table of actors and their respective areas of responsibility.

The comments from the audience can be summarized as:

- Dynamically updated lists is a very powerful feature.
- The wiki is a great tool for visualizing information because people are used to the Internet way of accessing information and using hyperlinks.
- Adding advanced meta-tags in form of semantic information is powerful but depends on an ontology. Construction and maintaining issues regarding ontologies must be taken into consideration before systems of this kind can be used in real scenarios.

6.3 Supporting system analysis

The semantic support team tried to support the participants by

- Providing an alternative search method based on semantic information.
- Providing an alternative way of presenting information. Information could be presented as dynamic lists or timelines.

An example of an incoming question was “Which important persons have a connection to an area X”. If all background material had been semantically tagged, the question could have been answered directly. As this was not the case, some requirements had to be fulfilled before the search query could be executed. In this case the requirements were:

1. Every event, person and region in the background material must be available in the wiki in the form of an article.
2. Every article must be classified in a correct manner. For example, every event must be assigned to the event class.
3. Every connection between persons, events and regions must be tagged.

The tagging procedure described above was performed several times during the experiment since pieces of information that were necessary to answer the incoming requests were not tagged from the beginning.

Many of the incoming requests were on an abstract level and had to be broken down into a set of concrete questions. An example of such a question is “which actors affect the mining industry?”. Even though this question was too abstract to be answered directly, a semantic search query that lists all actors, which in some way are connected to a specific type of event in a specific geographical area that contains mining facilities, can be answered.

During the task-solving part of the experiment, the semantic support team made a short demonstration of how the wiki could be used to create an overview of the available information.
information. In the demonstration, a wiki page that contained a table based on a semantic search query was created. The table listed actors participating in a specific type of event in a certain area. This type of queries can be used to list actors participating in events connected to the mining industry in the country that was analyzed. A comment from the audience was that dynamically updated lists is a powerful feature.

6.4 Discussion

The background information in the experiment was only available as natural text and images. Even though documents of this kind represent a common way to spread information, a semantic wiki is of no use if the document is not semantically tagged. The documents need to be classified and information within the document needs to be linked to other documents in order to exploit the full potential of a semantic wiki.

When the information extraction of the background material was performed, the focus area of the task was not given. In theory, there was no limit for what background information that could be useful. If more detailed information about the task had been accessible in advance, the information extraction could have been more focused. The lack of information resulted in that information was added to the wiki during the experiment. This was expected to some degree. However, it was not expected that such a big portion of information would have to be tagged during the experiment in order for the semantic wiki to be useful. Structuring information, in this case classifying and making semantic links between objects, takes time and is one of the biggest challenges when trying to make use of semantic technologies. The initial goal, to examine the potentials of the wiki when tagging had been made, could more easily have been reached if better preparation had been possible.

In order for any information system to be of use, intelligence questions must be broken down into concrete sub questions that can be transformed into well-defined queries, which can be answered by the information system. The intent of the experiment was to reach a level of detail where such queries could be formulated, which would promote the use of a semantic wiki. However, due to lack of time questions raised by the experiment participants were most often not concrete enough to be translated into queries that the semantic wiki could answer.

In summary, the experiment setup was not optimal because of two reasons:

1. A vague case definition made it impossible to semantically tag all background material in advance, given the time restriction. For this reason, information needed to be tagged during the experiment in order to answer the incoming requests.

2. Information requests from the experiment participants were too abstract. There was not enough time to break down high-level questions into concrete questions that could be answered by semantic search queries.

The non-optimal setup environment affected the experiment in two ways:

1. Too much time was spent on adding semantic information in order to answer requests, instead of actually answering them.

2. Few questions were on an appropriate abstraction level to be answered directly.

Based on the experience gained during the described experiment, a proper experiment for evaluating if a semantic wiki is useful should fulfill the following requirements:

1. A significant part of the background information should be semantically tagged before the experiment starts.

2. The analysts must have time to break down abstract questions into concrete questions. Alternatively, the problem to solve must be on a more detailed level.
Chapter 7 – Results

The evaluation of existing wikis that make use of semantic technologies showed that the area is still young. None of the wikis suited our needs out of the box. Even though tools for ontology reasoning, wikis and WYSIWYG editors have been around for some time, few systems integrate these tools fully. Further, the evaluation showed that there is still a lot of work to be done in order to improve the user experience. This is mainly due to the various technical prerequisites needed in order to make use of the new features. The prototype presented in this thesis tries to combine these tools to create a powerful, yet intuitive, wiki with special focus on situational awareness.

The experiment confirmed that the tagging of information is a critical factor. Improper tagging increases the risk of rendering incomplete search results, which in many cases are of no use and could in some cases even become dangerous if users rely on them in critical situations.

The experiment involved tagging of information under time restraints so some conclusions can be made about this topic. The experiment showed that filters that limit the available options when creating new statements are significantly time saving. A hierarchical structure of classes and properties also speed up the tagging process. That hierarchical structures and filters had a great impact on the speed of tagging is an important project finding. Although foreseen, it was an important confirmation.

The experiment also confirmed our hypothesis that dynamic information objects within a wiki article is a powerful and useful feature. Inferred statements, such that a certain city is part of a country, deduced from the fact that the city is part of a region that is part of a country, proved to be useful. This kind of inference, using transitive logic rules, made it possible to list persons, events and other things related to regions and sub-regions.

Summary of project findings:

• The core components of a semantic wiki are a standard wiki system, a storage and inference engine and a user interface extension that exploits the semantic information.

• Dynamic information objects within a wiki article is a powerful and useful feature.

• Filters, limiting the available options when making new statements, are helpful and time saving when adding new information.

• Inferred facts, especially related to regions and things within regions, were useful in the experiment.
8 Discussion

The benefits of tagging information with semantic information are clear. It enables the information to be processed and understood by machines in a new way that opens up for advanced searches, context awareness and improved navigation. On the other hand, it is unclear under what circumstances it is worth the effort. To which extent text has to be tagged to make semantic queries and reasoning useful, is still an open question.

Semantic technology does not only bring about new features, but also new requirements. For example, a search for events occurring at a specific region requires that every article describing an event is properly tagged as an event. Otherwise, it will not show up in the search result as the user expects. If the resulting set is incomplete, it can in many cases be regarded as useless, and sometimes even as dangerous.

Since tagging information is time-consuming, it is important to consider how the wiki is going to be used and to develop an ontology in advance that matches the requirements. It is better to have a wiki where a limited set of relations are used but all articles are tagged, rather than a wiki with a large set of relations where not all information is correctly tagged. The latter case will provide incomplete search results.

In the performed experiment, background information was delivered late and was handed over in various non-suitable formats. Hence, there was not enough time to process the material in a proper way. The experiment turned more into a study of how fast new information can be tagged semantically than how to make use of it. A more structured study, where background information is semantically tagged beforehand, would be necessary in order to draw any well-founded conclusions about to what extent semantic information is beneficial. The study should include both the tagging process and the use of the tagged information in order to measure the overall benefits. However, the performed experiment has given valuable feedback favoring our initial hypotheses concerning the advantages of the semantic wiki.
9 Conclusions

The goal of this master’s thesis was to examine if a semantic wiki is a valuable tool to gain situational awareness in a military intelligence context. The hypothesis was that analysts would benefit from a semantic wiki by:

• improved search capabilities,
• automatically updated information objects,
• revealing implicit facts and complex relationships,
• supporting consistency checking.

In order to try the hypothesis a prototype semantic wiki with these features had to be developed. A step on the way was to evaluate existing software.

After the evaluation of existing semantic wikis, a general architecture was proposed, a prototype semantic wiki was implemented and finally the prototype was tested in an experimental exercise. The prototype proves that it is possible to create a wiki that makes use of semantic information in a way that supports creating articles with dynamic content, searching based on semantic information, and improved contextual navigation.

Even though the performed experiment was not optimal for measuring potential benefits, the results indicate that our hypothesis was correct. The semantic wiki presented in this thesis might be a valuable tool for an intelligence analyst in order to improve his or her situational awareness. However, the potential benefit of a semantic wiki is in essence limited by how and to what extent the information within it is tagged. Our prototype fully relies on humans performing the tagging. When creating new articles, the user must specify the document type and the kind of relationship a reference to another article represents. Nonetheless, given that suitable procedures can be implemented, a semantic wiki can be of great potential value to an analyst that quickly has to gain situational awareness.
10 Future work

Extending a wiki with semantic technology increases the possibilities to make advanced searches and helps the user to understand how things connect to each other. However, this requires information to be properly tagged. A semantic relationship can be seen as a link between two objects with additional type information. To also specify the relation type is of course more time-consuming than just specifying that a relation exists. Further work is needed on how to speed up and automate the tagging process. It would be helpful if the articles could be automatically classified by a machine. One possible way is to search the document for certain words that indicate that the document is of a certain type. Another way is to compare unclassified documents to documents that already have been classified.

Our semantic wiki implementation integrates the Jena semantic framework, which supports consistency checking. However, this feature was not integrated in the wiki user interface due to time constraints. Making the user aware of inconsistent statements could improve information quality.

Inference of new facts is a powerful feature but gives rise to new problems. In a military context, information quality is central. It is important to know to what degree a piece of information can be trusted. Methods and techniques that guarantee traceability are needed. It is especially important to clarify what information inferred facts are based on. In order to use the wiki as a tool for information sharing in a military context, it is also important to be able to restrict access to certain information. To infer facts from restricted information is an even more difficult question since the inferred fact might makes it easy for the user to reveal the restricted information that the inferred fact was based upon.

The benefits of supporting articles with dynamic content ought to be most apparent when the wiki is used in a scenario where the situation changes. Testing the semantic wiki under such circumstances would be interesting.

So far, we have not investigated how the system scales. There are reasons to believe that a semantic wiki needs significantly more computing power\(^{14}\) than a standard wiki, due to the inference capabilities. A useful feature can be to provide inference only when it is explicitly asked for.

---

\(^{14}\) Both memory and processor capacity should be taken into account.
References


**Web resources**

[FACT] D. TSARKOV. FaCT ++.  
http://owl.man.ac.uk/factplusplus/  
Last visited September 9, 2008

[FCK] FCK. FCKeditor.  
http://www.fckeditor.net/  
Last visited September 9, 2008

http://jena.sourceforge.net  
Last visited September 9, 2008

http://kaon2.semanticweb.org/  
Last visited September 9, 2008

http://www.MediaWiki.org/  
Last visited September 9, 2008

Last visited September 9, 2008

[OWLIM] ONTOTEXT. OQLIM.  
http://www.ontotext.com/owlim/  
Last visited September 9, 2008

[PELLET] CLARK & PARSIA, LLC. Pellet OWL Reasoner.  
http://pellet.owldl.com  
Last visited September 9, 2008

[PHPJ] PHP.NET. PHP/Java integration.  
http://se2.php.net/manual/sv/ref.java.php  
Last visited September 9, 2008

[PROTÉGÉ] BMIR. Protégé.  
http://protege.stanford.edu  
Last visited September 9, 2008

[RACER] RACER-SYSTEMS. Racer.  
http://www.racer-systems.com  
Last visited September 9, 2008

[RESIN] CAUCHO. Resin application server.
http://caucho.com/resin/doc/overview.xtp
Last visited September 9, 2008

[SESAME] OPENRDF. Sesame.
http://www.openrdf.org
Last visited September 9, 2008

[SPARQL] W3C. SPARQL Query Language for RDF.
http://www.w3.org/TR/rdf-sparql-query/
Last visited September 9, 2008

http://ontoworld.org/wiki/Semantic_Wiki_State_Of_The_Art
Last visited September 9, 2008

http://www.w3.org/TR/2004/REC-owl-features-20040210/#s1.3
Last visited September 9, 2008

[WPI] WIKIPEDIA. Information model.
Last visited September 9, 2008

[WPRDF] WIKIPEDIA. RDF.
Last visited September 9, 2008

[WPSR] WIKIPEDIA. Semantic Reasoner.
Last visited September 9, 2008

[WPSW] WIKIPEDIA. Semantic Wiki.
Last visited September 9, 2008
## Appendix A: List of acronyms

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AJAX</td>
<td>Asynchronous JavaScript and XML</td>
</tr>
<tr>
<td>FOI</td>
<td>Swedish Defence Research Agency</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical user interface</td>
</tr>
<tr>
<td>HTML</td>
<td>Hypertext Markup Language</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol</td>
</tr>
<tr>
<td>KS</td>
<td>Knowledge Support</td>
</tr>
<tr>
<td>LOE</td>
<td>Limited Objective Experiment</td>
</tr>
<tr>
<td>OWL</td>
<td>Web Ontology Language</td>
</tr>
<tr>
<td>RDF</td>
<td>Resource Description Framework</td>
</tr>
<tr>
<td>RDFS</td>
<td>RDF Schema</td>
</tr>
<tr>
<td>SA</td>
<td>Situation awareness</td>
</tr>
<tr>
<td>SAIL</td>
<td>Storage and Inference Layer</td>
</tr>
<tr>
<td>SPARQL</td>
<td>SPARQL Protocol And RDF Query Language</td>
</tr>
<tr>
<td>SU</td>
<td>Situation understanding</td>
</tr>
<tr>
<td>WWW</td>
<td>World Wide Web</td>
</tr>
<tr>
<td>WYSIWYG</td>
<td>What You See Is What You Get</td>
</tr>
<tr>
<td>XML</td>
<td>eXtensible Markup Language</td>
</tr>
</tbody>
</table>
Appendix B: Example ontology

<?xml version="1.0"?>
<!DOCTYPE rdf:RDF [  
<!ENTITY owl "http://www.w3.org/2002/07/owl#"  
<!ENTITY xsd "http://www.w3.org/2001/XMLSchema#"  
<!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#"  
<!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#"  
]>  
<rdf:RDF xmlns="http://www.foi.se/2007/KSOne.owl#"  
xml:base="http://www.foi.se/2007/KSOne.owl"  
xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"  
xmlns:xsd="http://www.w3.org/2001/XMLSchema#"  
xmlns:owl="http://www.w3.org/2002/07/owl#"  
xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">  
<owl:Ontology rdf:about=""/>  
<owl:AllDifferent>  
<owl:distinctMembers rdf:parseType="Collection"/>  
<owl:AllDifferent>  
<Prefecture rdf:ID="Boffa_Prefecture"/>  
<Prefecture rdf:ID="Boké_Prefecture"/>  
<Region rdf:ID="Boké_Prefecture"/>  
<Region rdf:ID="Guinea">  
<hasPart rdf:resource="#Boké_Prefecture"/>  
</Region>  
<owl:Class rdf:ID="Civil_control_feature">  
<rdfs:subClassOf rdf:resource="#Feature"/>  
</owl:Class>  
<owl:Class rdf:ID="Event"/>  
<owl:Class rdf:ID="Feature">  
<rdfs:subClassOf rdf:resource="#Object"/>  
</owl:Class>  
<owl:Class rdf:ID="Nation">  
<hasPart rdf:resource="#Boké_Prefecture"/>  
</Nation>  
<owl:DatatypeProperty rdf:ID="has_population_size">  
<rdfs:domain rdf:resource="#Feature"/>  
<rdfs:range rdf:resource="&xsd;string"/>  
</owl:DatatypeProperty>  
<owl:ObjectProperty rdf:ID="hasPart">  
<rdfs:domain rdf:resource="#Object"/>  
<rdfs:range rdf:resource="#Object"/>  
<owl:inverseOf rdf:resource="#isPartOf"/>  
</owl:ObjectProperty>  
<owl:ObjectProperty rdf:ID="isPartOf">  
<rdfs:domain rdf:resource="#Object"/>  
<rdfs:range rdf:resource="#Object"/>  
<owl:inverseOf rdf:resource="#hasPart"/>  
</owl:ObjectProperty>  
<owl:ObjectProperty rdf:ID="isPlaceOfEvent">  
<rdfs:domain rdf:resource="#Event"/>  
<rdfs:range rdf:resource="#Object"/>  
<owl:inverseOf rdf:resource="#occursAt"/>  
</owl:ObjectProperty>  
<owl:Class rdf:ID="Nation">  
<rdfs:subClassOf rdf:resource="#Civil_control_feature"/>  
</owl:Class>  
<owl:Class rdf:ID="Object">  
<owl:ObjectProperty rdf:ID="occursAt">  
<rdfs:domain rdf:resource="#Event"/>  
<rdfs:range rdf:resource="#Object"/>  
<owl:inverseOf rdf:resource="#isPlaceOfEvent"/>  
</owl:ObjectProperty>  
<owl:Class rdf:ID="Prefecture">  
<rdfs:subClassOf rdf:resource="#Civil_control_feature"/>  
</owl:Class>  
<owl:Class rdf:ID="Region">  
<rdfs:subClassOf rdf:resource="#Civil_control_feature"/>  
</owl:Class>  
</rdf:RDF>