

ReqWiki: A Semantic System for Collaborative Software Requirements Engineering

Guide for Users and Developers

René Witte
Bahar Sateli
Elia Angius
Srinivasan Rajivelu

Development Release
September 19, 2013

Semantic Software Lab
Concordia University
Montréal, Canada

<http://www.semanticsoftware.info>

Contents

1	Overview	1
2	Features	2
2.1	Top-Level Documents	3
2.2	Semantic Model	3
2.3	Semantic Forms	4
2.4	Automatic Traceability	5
2.5	NLP Services	5
3	Installation	8
3.1	Downloading Resources from GitHub	8
3.2	MediaWiki Installation	8
3.3	MediaWiki Extensions	8
3.3.1	Semantic MediaWiki	9
3.3.2	Semantic Forms	9
3.3.3	Semantic Assistants	9
3.3.4	ReqWiki Templates	9
4	Development Notes	11
4.1	System Architecture	11
4.2	Adding New NLP Services	12
	Bibliography	13

About this document

This document contains documentation for the *ReqWiki* project. You can obtain the latest version from <http://www.semanticsoftware.info/reqwiki>. The project resources are hosted on GitHub, <https://github.com/SemanticSoftwareLab/ReqWiki>.

Contents

Licenses

The ReqWiki project is published under the GNU Affero General Public License v3 (AGPL3).¹

¹AGPL3, <http://www.gnu.org/licenses/agpl-3.0.html>

1 Overview

The requirements engineering phase within a software project is a heavily knowledge-driven, collaborative process that typically involves the analysis and creation of a large number of textual artifacts. We know that requirements engineering has a large impact on the success of a project, yet sophisticated tool support, especially for small to mid-size enterprises, is still lacking.

Wikis, as an affordable and lightweight documentation and distributed collaboration platform, have demonstrated their capabilities in distributed requirements elicitation and documentation. While the integrity of software requirements specifications (SRS) documents inside a wiki can be enforced through the use of templates and wiki bots on a syntactic level, semantic SRS defects, such as ambiguity or inconsistency, require human revision. To this end, Natural Language Processing (NLP) techniques have been proposed as a means to (semi-)automatically improve requirements specifications, but so far have not been widely adopted.

In ReqWiki [Sateli et al. (2012)], we allow users to employ various state-of-the-art techniques from the NLP and Semantic Computing domains to improve the quality of SRS documents contained in a wiki platform, through using the NLP-provided guidance tips, as well as automatically detecting SRS defects.

The core of ReqWiki is a MediaWiki¹ engine, which provides a collaborative environment for SRS development. However, unadorned wikis can not fulfil all requirements, such as consistency management, semantic support, and natural language processing in the context of RE. For this, we added two substantial features to the wiki engine underlying ReqWiki, described in this user guide: a semantic data model and NLP web services.

The rest of this document is structured as follows: In Chapter 2 we present the ReqWiki features and how the wiki has been pre-filled with templates corresponding to the Unified Process methodology [Kruchten (2003)]. In Chapter 3 we will guide you step-by-step on how to install ReqWiki.

¹MediaWiki, <http://www.mediawiki.org>

2 Features

Based entirely on open source and open standards, ReqWiki is powered by MediaWiki - the powerful engine behind Wikipedia - and uses the Semantic MediaWiki extension to provide a technical platform with semantic support. The pre-defined semantic forms and templates in ReqWiki help users in creating various artifacts, while at the same time automatically creating the required semantic metadata. Figure 2.1 shows the user interface of ReqWiki.

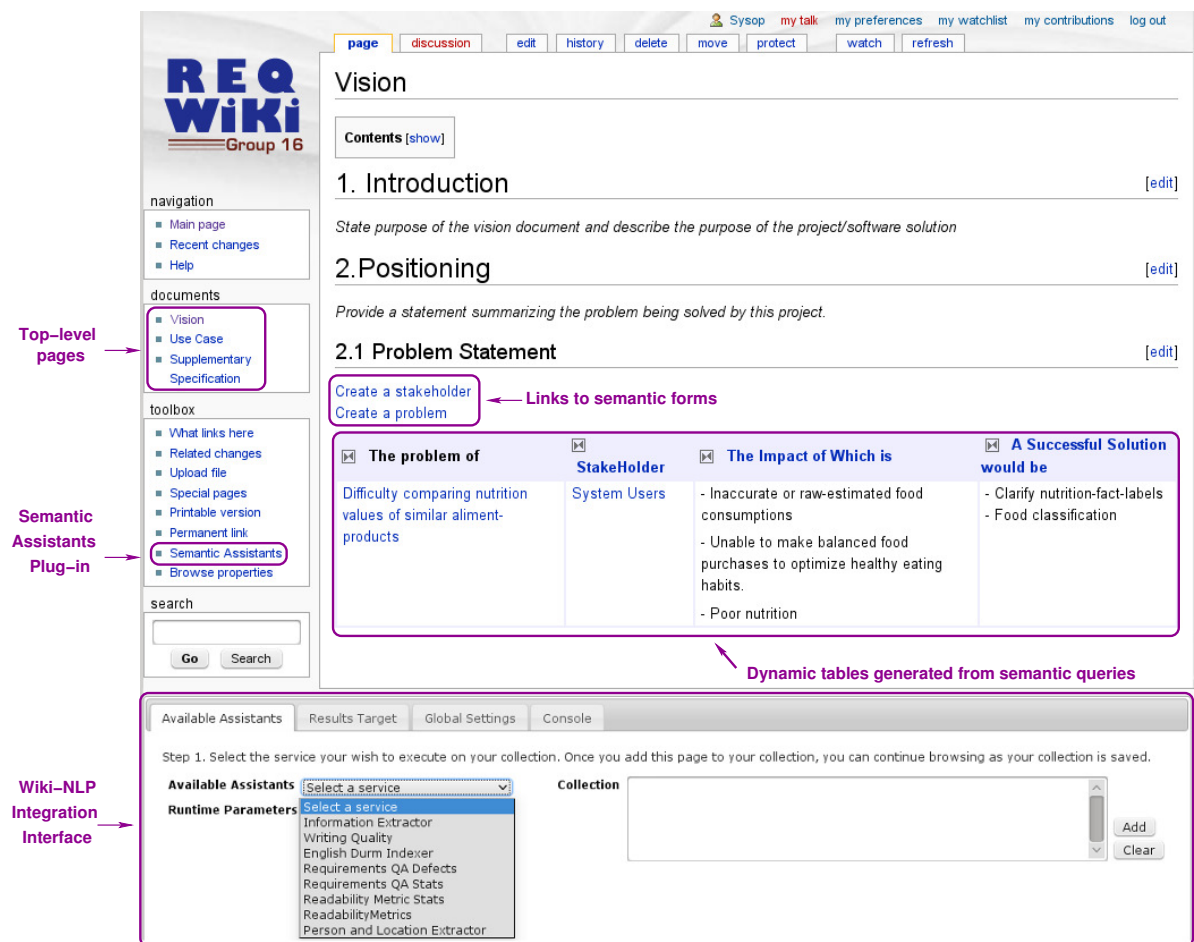


Figure 2.1: ReqWiki User Interface

ReqWiki makes use of pre-defined templates to allow its users author SRS documents in a uniformed manner based on the UP methodology. The templates in ReqWiki (see Chapter 3)

conform to the UP methodology and its artifacts, such as vision document, use case document and applicable supplementary specifications. However, by using the same approach, other software development methodology artifacts can be easily added to the ReqWiki system.

2.1 Top-Level Documents

ReqWiki features three top-level wiki pages that are considered the entry points to the SRS documentation platform and are designed to guide stakeholders through the RE process. For each top-level page, a corresponding “talk” page – provided by the MediaWiki engine – is also available for users to discuss contradictory ideas and leave notes for other stakeholders. The top-level pages are as follows:

- a **Vision** page to define the product position, stakeholders, assumptions, dependencies, needs and features,
- a **Use Case** page to define actors, goals and use cases,
- a **Supplementary Specification** page to define functional and non-functional requirements, standards, legal notes, test cases and traceability links.

Each of these top-level pages is divided into multiple sections. Users can manually edit the static sections content, such as the introduction to the SRS document. The dynamic sections, on the other hand, are populated by the pre-defined queries embedded in the page markup every time the page is requested by a user. This way, it is ensured that the top-level pages are always presenting the latest state of the ReqWiki content. In addition, at the bottom of all of three top-level pages, a list of user-defined terms is automatically generated and presented as a glossary to aid users in reflecting about the correct usage of the terms. The presence of this feature in ReqWiki helps to establish a common language among stakeholders and promotes consistency.

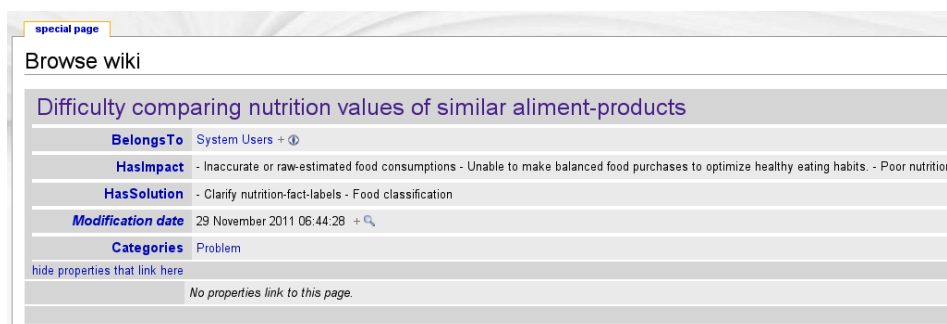
2.2 Semantic Model

In ReqWiki, to ensure consistency and compliance with a standard methodology, both within and between requirements artifacts, we formally model their concepts and relationships in a web ontology language. Using the ReqWiki underlying semantic wiki engine, we connect textual descriptions of artifacts with formal ontologies, providing further semantic reasoning and querying capabilities within the wiki environment. Our ReqWiki ontology defines the main concepts in the Unified Process and Use Case methodology for requirements engineering. This ontology provides for the means to semantically describe concepts stored in the Wiki (e.g., actors, goals, use cases, test cases, features) and relate them with each other. For example, we can now formally establish the relations between a Use Case and its Actors.

Using this ontology, once the wiki content is enriched with semantic metadata, we are able to semantically query the ReqWiki content, e.g., to retrieve a list of all the actors in a

2 Features

wiki. In addition, reasoning capabilities can be added to the system to automatically detect inconsistencies and conflicts between entities.

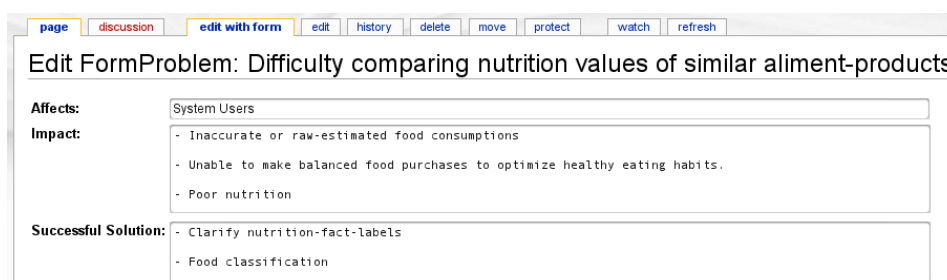


special page	
Browse wiki	
Difficulty comparing nutrition values of similar aliment-products	
BelongsTo	System Users + ⓘ
HasImpact	- Inaccurate or raw-estimated food consumptions - Unable to make balanced food purchases to optimize healthy eating habits. - Poor nutrition
HasSolution	- Clarify nutrition-fact-labels - Food classification
Modification date	29 November 2011 06:44:28 + ⓘ
Categories	Problem
hide properties that link here	
No properties link to this page.	

Figure 2.2: Embedded RDF data associated with a ReqWiki entity)

2.3 Semantic Forms

Every entity in ReqWiki, like a use case description, is uniquely identified by a URI in the system and can be retrieved via its title or the full text search feature that the MediaWiki engine offers. Users can start defining entities like actors or goals, by clicking on the provided links in the top-level pages. For example, in the *Use Case* page, a user can define a new use case description by clicking on the “Create a use case” link. The link takes the user to the corresponding *semantic form* to define a use case attributes, such as actors, main success scenario, etc. These semantic forms are provided by the *Semantic Form* extension installed on the ReqWiki engine and allow users to use user-friendly forms rather than directly dealing with wiki markup. Furthermore, the fields of each semantic form is related to a concept inside the ReqWiki semantic model, thus helps to simultaneously enrich the user provided data with semantic metadata.



page	discussion	edit with form	edit	history	delete	move	protect	watch	refresh
Edit FormProblem: Difficulty comparing nutrition values of similar aliment-products									
Affects:	System Users								
Impact:	<ul style="list-style-type: none">- Inaccurate or raw-estimated food consumptions- Unable to make balanced food purchases to optimize healthy eating habits.- Poor nutrition								
Successful Solution:	<ul style="list-style-type: none">- Clarify nutrition-fact-labels- Food classification								

Figure 2.3: A ReqWiki Semantic Form

Once the user saves the form, a new page with the use case description is created in the wiki and the user input data embedded in the pre-defined ReqWiki template for a use case. Since the new page is implicitly annotated with semantic metadata, when the user navigates back to the top-level *Use Case* page, he sees the new use case in a table that is generated by the in-line query embedded in the page, such as the one shown in Figure 2.1 (bottom).

2.4 Automatic Traceability

Motivated by maintainability and liability project factors, *traceability* is the degree to which various software artifacts in a developing system are interrelated with each other. In a collaborative and volatile environment such as RE, traceability ensures the rationale for all artifacts are properly accounted for towards building “the right product”. Manually creating and updating these links, for example, with cross-references between documents, is highly time-consuming and error prone. Additionally, it must be possible to reference requirements artifacts from those produced in subsequent phases, such as design and code.

Relying on the populated ontology, ReqWiki supports the automatic creation of three forms of traceability links: semantic links inside the wiki, query-based links, and revision links.

- For the first type, semantic concepts defined in wiki pages are presented as hyperlinks, so that users can navigate to and retrieve their corresponding content. When defining these semantic concepts, the forms themselves already constrain field input to non-empty existing ontology instances of the expected field type. Thus, this form of traceability enforces link correctness and simplifies verification and validation phases, since dead links and unlinkable artifacts are detectable.
- For the second type, using the same semantic metadata in our ontology, we can insert in-line queries in top-level pages to create dynamic tables from the ontological meta-data. Figure 2.4 shows the Need-Feature table dynamically generated from a SMW in-line query.

User Needs versus Features															
<pre> 1 {{#ask: 2 [[Category:Features]] 3 ?BelongsTo=Need 4 ? = Feature 5 format= table 6 }}</pre>	<table> <tr> <th>Need</th><th>Feature</th></tr> <tr> <td>Modify policy detail information</td><td>Alter policy information</td></tr> <tr> <td>Modify policy detail information</td><td>Query the status of policy information</td></tr> <tr> <td>Alteration of unit link product</td><td>Input conversion to unit investment</td></tr> <tr> <td>Alteration of unit link product</td><td>Modify conversion to unit investment</td></tr> <tr> <td>Alteration of unit link product</td><td>Query unit investment</td></tr> <tr> <td>Alteration of unit link product</td><td>Query unit price</td></tr> </table>	Need	Feature	Modify policy detail information	Alter policy information	Modify policy detail information	Query the status of policy information	Alteration of unit link product	Input conversion to unit investment	Alteration of unit link product	Modify conversion to unit investment	Alteration of unit link product	Query unit investment	Alteration of unit link product	Query unit price
Need	Feature														
Modify policy detail information	Alter policy information														
Modify policy detail information	Query the status of policy information														
Alteration of unit link product	Input conversion to unit investment														
Alteration of unit link product	Modify conversion to unit investment														
Alteration of unit link product	Query unit investment														
Alteration of unit link product	Query unit price														

Figure 2.4: Embedded SMW query and resulting Feature-Need Table

- A third form of traceability available in the ReqWiki is revision control capabilities to when pages get changed. This not only indicates which author changed what content at which times, but also allows reverting to previous revisions in case of erroneous modifications.

2.5 NLP Services

The second major component of our ReqWiki system is the idea of semantic NLP *assistants* that support users in developing the requirements specification. Our goal is to offer these

2 Features

analysis capabilities directly within the wiki, to avoid forcing users to switch to a different text analysis application. This is achieved through our Wiki-NLP integration (See Chapter 4) that allows to setup an installation with custom NLP services, e.g., for domain-specific analysis pipelines or custom (company/organization-specific) quality rules. The integration is based on our Semantic Assistants framework, which can broker any deployed NLP pipeline as a standard web service [Witte and Gitzinger (2009)]. Within the current version of ReqWiki, we have made a number of both general and requirements-specific NLP services available to ReqWiki users:

Writing Quality Assessment is a service that integrates the After The Deadline [Mudge (2010)] tool and helps ReqWiki users find spelling and grammatical mistakes and provides suggestions for improvements, where possible.

Readability Metrics is a service that provides an overall readability score of the content that users have produced, indicating how hard to read their text is for other stakeholders. This service provides the users with an overall readability score of the wiki content. The result score indicates how hard to read and comprehend the artifacts are for other stakeholders.

Requirements Quality Assurance is a service developed based on the NASA requirements quality metrics [Laplante (2009)] that detects SRS defects like *Options*, *Directives* or *Weak Phrases* in a document. By using this service, users have the chance to automatically find these defects in their specifications and correct them, resulting in a higher quality SRS document.

Document Indexer creates a back-of-the-book style index of the wiki content and stores it in the wiki as a page. This service builds on MuNPEx¹, an open-source tool that groups words into noun phrases, to generate an inverted index, with entries automatically linked to wiki pages. Users can compare the result of this service to the glossary section of the SRS documents to check its completeness.

Information Extractor automatically extracts named entities, such as persons, organizations or locations, from wiki pages, based on the ANNIE system [Cunningham et al. (2011)]. This service is especially useful in analysing existing domain documents during the requirements analysis phase.

These NLP pipelines can be executed on the ReqWiki content using the Wiki-NLP user interface shown in Figure 2.5. Users can ask for this interface by clicking on the “Semantic Assistants” link inside the wiki navigational menu (Figure 2.5 - left). The NLP interface allows users to see “Available Assistants” in the Semantic Assistants server to help them with various NLP analyses. Hovering over each item in the list shows a brief description of what each service does. Once the user selects a service, he can then gather a so-called “collection” of wiki pages for analysis using the “Add” button.

¹Multi-Lingual Noun Phrase Extractor <http://www.semanticsoftware.info/munpex>

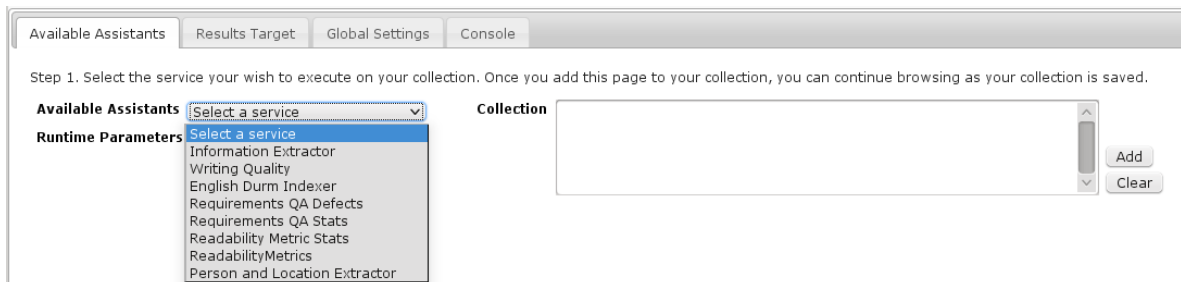


Figure 2.5: NLP services invocation interface in ReqWiki

Next step is for users to specify where they would like the NLP analysis results be written to. The current options available the “Target Options” are:

“Same Page” , i.e., the results will be written in the same page as the resource. If the user has multiple pages in the collection, each results will be written into its corresponding source page. The current options for this item are the wiki page’s main “body”, or its associated “talk”² section.

“Another Page” , i.e., writing the results to a different page in the wiki. If the user has multiple pages in the collection, the results of all the wiki pages will be aggregated into one output and written into the specified wiki page. In order to precisely choose a destination for this option, users have to select a namespace from a list of available namespaces in the wiki and provide a unique page name. If the specified wiki page exists in the wiki, the results will be appended to the end of the page, otherwise a new wiki page will be automatically created.

“Another Wiki” i.e., writing the results of the analysis into a different wiki than the source, provided that its engine is supported by the integration. Needless to say, users must provide the integration with (1) the address of the destination wiki, (2) its underlying wiki engine and version, (3) a valid username and password, and (4) a valid page name. Similar to the previous option, the results of the analysis will be aggregated and made persistent in the destination wiki page.

A service execution request can then be made by clicking on the “Run” button in the second tab. The “Console” tab of the Wiki-NLP interface shows user-friendly logs of the NLP service execution and indicates where the results are written if the execution is successfully done.

Where applicable, users can dynamically change which Semantic Assistants server they want their wiki to connect to. The “Global Settings” tab in the Wiki-NLP interface shown in Figure 2.5, allows users to select a Semantic Assistants server from a list of pre-defined addresses, as well as defining a custom end point. Once the user clicks the “connect” button, the selected server address is stored in the browser cookies and will take effect as soon as the user refreshes the browser.

²In some MediaWiki skins, a talk page is also known as the “discussion” page.

3 Installation

ReqWiki is an open source software released under APGL3 license and its resources can be downloaded freely from the Semantic Software Lab public repository on GitHub website accessible from <https://github.com/SemanticSoftwareLab/ReqWiki>.

3.1 Downloading Resources from GitHub

In order to download the ReqWiki resources from GitHub, there are two options:

1. **Downloading a .zip file.** All the ReqWiki resources can be downloaded a .zip file directly from the GitHub website. To do so, browse to <https://github.com/SemanticSoftwareLab/ReqWiki> on your web browser and click on the “Zip” button on top of the page. Then, save and unzip the file to your local workstation.
2. **Cloning the repository.** You can also acquire a read-only version of the ReqWiki resources from a command line terminal of your choice. Note that you need to have Git¹ installed on your system. Provided, browse to a desired directory in your local workstation and type the following line in your command line terminal:

```
1 git clone https://github.com/SemanticSoftwareLab/ReqWiki.git
```

3.2 MediaWiki Installation

Once you have the resources, you need to install a MediaWiki instance on a web server. ReqWiki required MediaWiki version 1.16 to function properly. Installation details of MediaWiki instances can be found on the MediaWiki website².

3.3 MediaWiki Extensions

In addition the MediaWiki default installation, ReqWiki requires a number of extensions to be installed on the wiki.

¹Git, <http://git-scm.com/>

²MediaWiki Installation, <http://www.mediawiki.org/wiki/Help:Installation>

3.3.1 Semantic MediaWiki

To provide semantic enrichment and query of the wiki content, ReqWiki requires Semantic MediaWiki extension version 1.5 or higher. First download the SMW extension from the Semantic MediaWiki website³. Unzip and copy the extension to the `extensions` folder in the root of your wiki installation. Next, paste the following lines at the end of the `LocalProperties.php` file in the root of your wiki installation.

```
1 include_once("$IP/extensions/SemanticMediaWiki/SemanticMediaWiki.php");
2 enableSemantics('localhost');
```

These two lines *install* the SMW extension on your wiki, but we still need to initialize the SMW tables in the wiki database. To do so, browse the special pages of your wiki and click on the “Admin Admin functions for Semantic MediaWiki” link under the “Semantic MediaWiki” category. On the next page click on the “Initialise or upgrade tables” button to start the database initialization process.

3.3.2 Semantic Forms

Similar to the SMW extension, ReqWiki requires the Semantic Forms extension to be installed on its MediaWiki engine. First download the extension from the MediaWiki website⁴. Then copy the file to the `extensions` folder of your wiki and add the following line to the end of the `LocalProperties.php` file in the root of your wiki installation.

```
1 include_once("$IP/extensions/SemanticForms/SemanticForms.php");
```

3.3.3 Semantic Assistants

One last extension to install on your MediaWiki instance in the Semantic Assistants extensions. This extension provides the Wiki-NLP user interface shown above in the wiki. To do so, copy the Semantic Assistants extension inside the `extension` folder in your ReqWiki check-out over to the `extensions` folder of your MediaWiki installation. Finally, add the following line to the end of the `LocalProperties.php` file of your wiki.

```
1 include_once("$IP/extensions/SemanticAssistants/SemanticAssistants.php");
```

3.3.4 ReqWiki Templates

After installing the extensions, we need to install ReqWiki customized templates to pre-fill the wiki with top-level pages, forms and semantic properties pertaining to the wiki semantic model. In order to import the templates to your wiki, browse to the special pages of your wiki and click on the “Import pages” link under the “Page tools” category⁵. From the “Import XML

³SMW Website, <http://www.semantic-mediawiki.org/wiki/SemanticMediaWiki>

⁴Semantic Form extension, http://www.mediawiki.org/wiki/Extension:Semantic_Forms

⁵In the default installation of MediaWiki, permission of importing pages to the wiki is limited to users with the “admin” role.

3 Installation

data” box click on the “Browse” button and upload the [all_templates.xml](#) file located in the [templates](#) folder of your ReqWiki check-out, as shown in Figure 3.1. Alternatively, you can import the templates one by one, by choosing the corresponding XML file from the list.

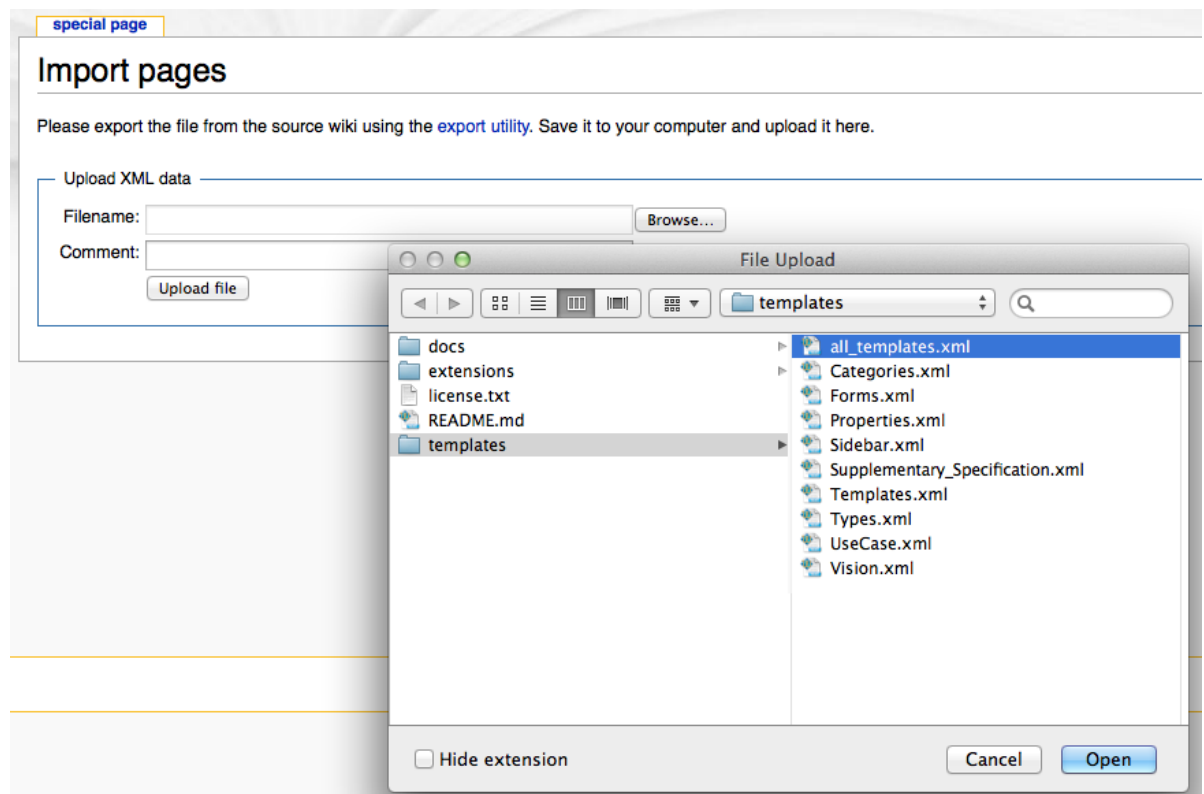


Figure 3.1: Importing ReqWiki templates into MediaWiki engine

4 Development Notes

Although the templates in ReqWiki have been designed to conform to the UP methodology, by using the same approach, other software development methodology artifacts can be easily added to the ReqWiki system. Also, since ReqWiki is empowered by an open service-oriented architecture, additional NLP services can be added to ReqWiki with little effort. In this chapter, we describe the system architecture behind the ReqWiki and how new NLP services can be added to the architecture.

4.1 System Architecture

The text mining capabilities in ReqWiki are provided by a novel integration of NLP within wiki systems. The Wiki-NLP integration [Sateli and Witte (2012)] is a cohesive architecture that provides a seamless integration of NLP techniques within wiki environments. The ReqWiki architecture, shown in Figure 4.1, is mainly composed on two components; A server-side component delegates the NLP services requests to the Semantic Assistants framework – an open source framework that publishes concrete NLP pipelines as standard web services [Witte and Gitzinger (2009)]. And a client-side, light-weight plug-in that needs to be installed on the wiki engine. The communication between the two end points are performed over standard HTTP protocols and is realized through custom JavaScript code that is injected into the user's browser.

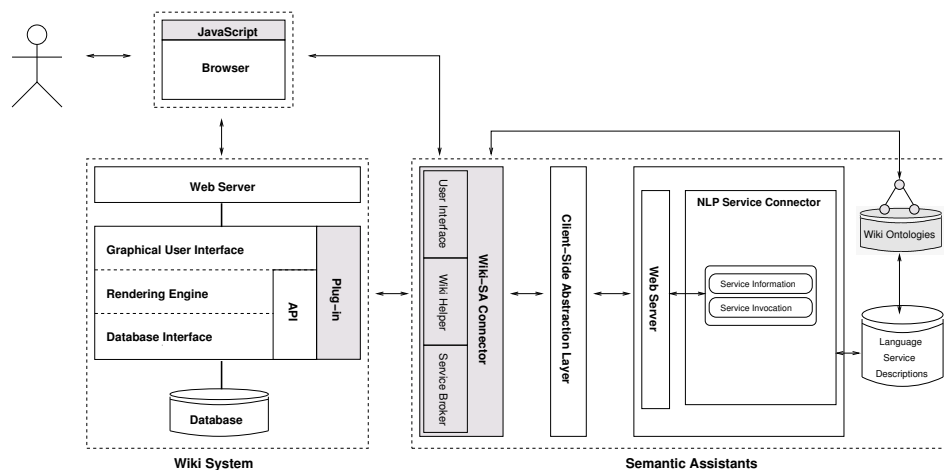


Figure 4.1: Wiki-NLP integration architecture in ReqWiki

4.2 Adding New NLP Services

By default the ReqWiki NLP services are provided by a pre-defined Semantic Assistants server configured in its architecture. If you would like to offer additional NLP services within your ReqWiki environment, you have to set up a Semantic Assistants server with your custom NLP services deployed in it. To do so, please consult the Semantic Assistants user guide¹ on how to set up a custom Semantic Assistants server.

Once the new server is up and running, you can configure your ReqWiki to connect to the your Semantic Assistants server rather than its default one. You can do so through the NLP interface in the wiki. In the “Global Settings” tab, shown in Figure 4.2, select the “Custom Server” option and fill in the host name and port number fields and click “Add Server”. Then select the “Pre-defined Servers” option, choose your new server from the list and click on “Connect”. The new changes will take effect next time you refresh your browser.

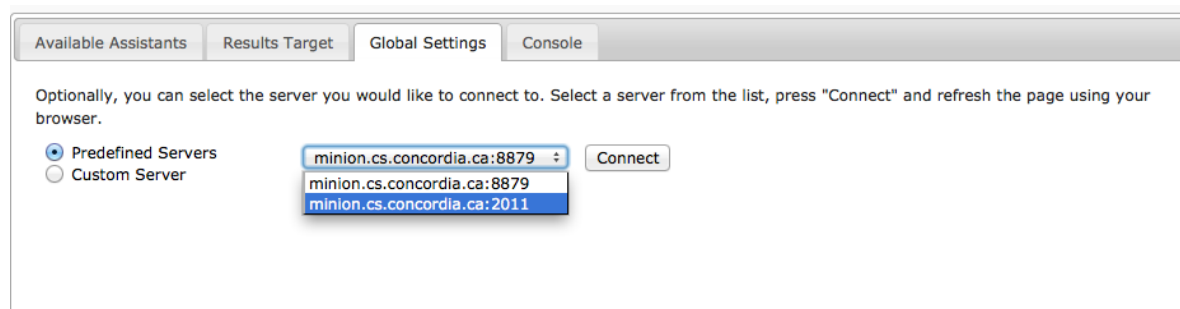


Figure 4.2: Wiki-NLP integration architecture in ReqWiki

¹Semantic Assistants, url<http://www.semanticsoftware.info/semantic-assistants>

Bibliography

- Hamish Cunningham et al. *Text Processing with GATE (Version 6)*. University of Sheffield, Department of Computer Science, 2011. ISBN 978-0956599315. URL <http://tinyurl.com/gatebook>.
- Philippe Kruchten. *The Rational Unified Process: An Introduction*. Addison-Wesley, 3rd edition, 2003.
- Phillip A. Laplante. *Requirements Engineering for Software and Systems*. Auerbach Publications, 1st edition, 2009. ISBN 1420064673, 9781420064674.
- Raphael Mudge. The Design of a Proofreading Software Service. In *Workshop on Computational Linguistics and Writing: Writing Processes and Authoring Aids, CL&W*, 2010.
- Bahar Sateli and René Witte. Natural language processing for mediawiki: The semantic assistants approach. In *The 8th International Symposium on Wikis and Open Collaboration (WikiSym 2012)*, Linz, Austria, 08/2012 2012. ACM, ACM. URL <http://www.wikisym.org/ws2012/bin/download/Main/Program/p20wikisym2012.pdf>.
- Bahar Sateli, Srinivasan Sembakkam Rajivelu, Elian Angius, and René Witte. Reqwiki: A semantic system for collaborative software requirements engineering. In *The 8th International Symposium on Wikis and Open Collaboration (WikiSym 2012)*, Linz, Austria, 08/2012 2012. ACM, ACM.
- René Witte and Thomas Gitzinger. Semantic Assistants – User-Centric Natural Language Processing Services for Desktop Clients. In *3rd Asian Semantic Web Conference (ASWC 2008)*, volume 5367 of *LNCS*, pages 360–374, Bangkok, Thailand, February 2–5 2009. Springer. doi: 10.1007/978-3-540-89704-0_25. URL <http://rene-witte.net/semantic-assistants-aswc08>.