Annotating Software Documentation in Semantic Wikis

Klaas Andries de Graaf
VU University
Amsterdam
The Netherlands
ka.de.graaf@vu.nl

ABSTRACT

The use of software documentation in traditional file-based format has various issues. Storing and annotating the content of software documentation in semantic wiki pages addresses many of these issues and supports knowledge retrieval. This paper reports on the application of semantic annotation and knowledge retrieval using the ArchiMind semantic wiki system. Knowledge in documentation is annotated by indexing text with a lightweight software engineering ontology. The process and the use-cases of this semantic annotation of software documents are described.

Categories and Subject Descriptors

H.3.1 [**Information Storage and Retrieval**]: Content Analysis and Indexing - *Indexing methods*
D.2.7 [**Software Engineering**]: Distribution, Maintenance, and Enhancements – *Documentation*

General Terms

Documentation, Design

Keywords

Semantic Annotation, Software Documentation, Semantic Wikis, Software Engineering Knowledge.

1. INTRODUCTION

Software documentation is essential for the success of large and complex software projects. It makes the rationale of the system design explicit and it supports coordination between project stakeholders, as well as reasoning and learning about the system. Low documentation quality results in lower efficiency throughout software project phases, errors and lower software quality in general [1]. Much time and effort needs to be spent in the current practice of capturing, updating and searching knowledge in file-based documents. Wiki systems provide some advantages in versioning, responsibility, accessibility and traceability. However, issues with synonyms and homonyms [2] remain as well as spelling errors, abbreviations, ambiguity and context-dependent interpretation.

This paper describes how these issues can be addressed with in-page semantic annotation of text from software documentation. This mechanism is illustrated by examples and use cases in the ArchiMind semantic wiki. ArchiMind is an adaptation from the OntoWiki [3] semantic wiki. It uses, among other things, the in-page semantic annotation mechanism and a Software Engineering (SE) domain ontology to support the capture and retrieval of SE knowledge from software documentation.

ESAIR'11, October 28, 2011, Glasgow, Scotland, UK.
ACM 978-1-4503-0958-5/11/10.

In section 2.1, The SE domain ontology is described as well as the input and storage of software documentation. This serves as a basis for section 2.2 in which the in-page annotation mechanism is described and section 2.3. which contains use cases.

2. SEMANTIC ANNOTATION OF SE KNOWLEDGE IN DOCUMENTATION

2.1 SE ontology and input of documentation

ArchiMind uses a software engineering ontology to capture explicit SE knowledge in software documentation and implicit SE knowledge from experts. The ontology is adapted from the lightweight software engineering ontology by Tang et al. [2]. Setting and Behavior are added to suit the industry environment where it is currently being applied. Figure 1 depicts an overview of the main classes and relations of this ontology. This ontology is designed to be lightweight; adaptable and extendable for general software applications.

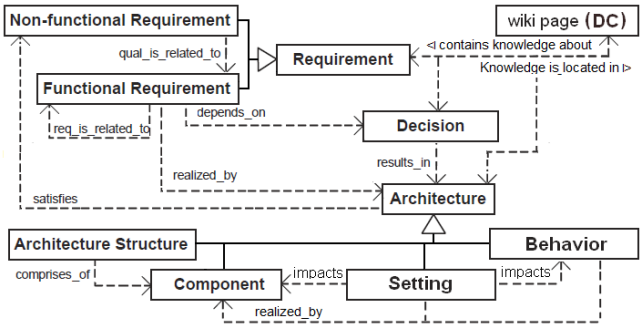

Figure 1 – Lightweight software engineering ontology

A WYSIWYG editor, with image upload functionality, was implemented to allow users to copy software documentation content in popular text editors and paste it in ArchiMind.

Software specifications are stored as HTML in the content section of wiki page instances of the ontology described above. The ontology contains Dublin Core [4] data properties to allow for specification of metadata (author, date, type, etc..) for many possible sources of SE knowledge such as official documents, meeting notes, code snippets, interface specifications and e-mails.

2.2 Semantic annotation of wiki pages

Software documentation wiki pages are annotated in ArchiMind by indexing text to ontology instances with the following actions:

1. Select the text fragment that can be indexed to a SE knowledge instance of the ontology. Figure 2 depicts an example where the text "*order_config_mngr*" is selected.

2. Select the (sub)class in the ontology which contains the SE knowledge instance that the selected text should be indexed to. Ontology class selection is done in the in-page annotation menu. In figure 2 this is class "*component*".
3. Select the SE knowledge instance, of the selected (sub)class, to which the selected text should be indexed. In the example in figure 2 this is the instance with label "*order configuration manager*". It is assumed that an SE knowledge instance already exists. Otherwise it should be created before indexing.

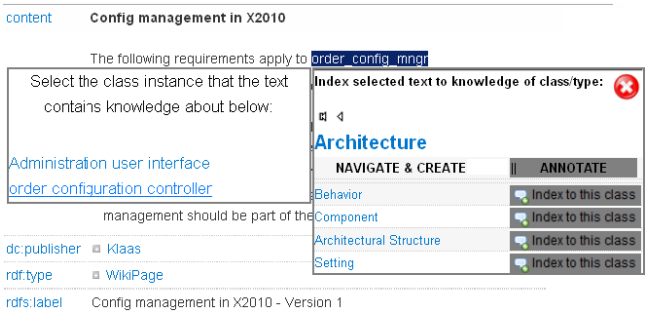

Figure 2 – In-page annotation

Data from the indexing action above is stored in a separate annotation database table as *{(URI: annotated wiki page), (string: indexed text), (URI: SE knowledge instance)}*. When viewing a wiki page, content of the page is checked against this database table and text that has been indexed is highlighted yellow. Clicking on the highlighted text will show the details of the SE knowledge instance(s) that the text is indexed to. This is further illustrated in the next section and depicted in figure 3.

The indexing actions above are also used to annotate the wiki page itself. A triple is stored in the knowledge base that captures the relationship between wiki page and the SE knowledge instance to which the text on the wiki page has been indexed to. The triple is stored as *{(URI: annotated wiki page), (URI: contains knowledge about <relation>), (URI: SE knowledge instance)}*. Another triple, creating a relation in the opposite direction, is stored as: *{(URI: SE knowledge instance), (URI: knowledge is located in <relation>), (URI: annotated wiki pag)}*. The rationale for storing this triple is increased usability. Inverse relations can be shown in ArchiMind, however, this requires an extra action and knowledge of this feature.

2.3 Use of annotations in knowledge retrieval

A software engineer is interested in the requirements realized by component "*order_config_mngr*" which s/he found in a printed diagram, remembers or was referenced to by a colleague.

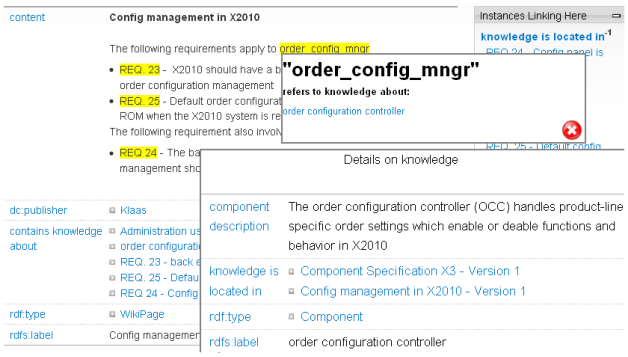

Figure 3 – In-page knowledge retrieval

A keyword search in ArchiMind on "*order_config_mngr*" returns the wiki page, annotated in the previous section, with "*order_config_mngr*" highlighted yellow in the wiki page content, as depicted in figure 3. When clicking on it, the indexed knowledge on a component with label "*order configuration controller*" and abbreviation "OCC", is shown. When properly defined, the SE knowledge instance contains relations to the requirements and behavior it realizes and settings that impact it. The SE knowledge instance may also contain relations to other indexed wiki pages that have knowledge about it, but use an official, misspelled, abbreviated or synonymous name.

Also consider searching for "*order"* or *"configuration*" when they are homonyms for classes, behavior, features or functions. After annotation, search results will include the correct SE knowledge instance and the wiki pages that have an index to the exact SE knowledge instance, e.g. a class. Wiki page instances listed in keyword search results contain expandable *<contains knowledge about>* relations to SE knowledge indexed in its content. These wiki pages relations and the relations between SE knowledge instances aid users in knowledge retrieval.

3. DISCUSSION AND EVALUATION

Traceability in software documentation is hard to create and maintain. SE knowledge is traceable to related knowledge in various topics, e.g. views, use cases, behavior, requirements and configurations, whereas document paragraphs and wiki pages are typically written on a single topic. Hyperlinks in traditional wikis provide traceability but do not show the exact meaning of relations, whereas in-page annotations do show the semantics of relations between SE knowledge.

An ongoing experiment in industry, aims to validate that ArchiMind is more efficient and effective in knowledge retrieval than the use of traditional file-based documentation. Subjects are asked to answer questions using file-based software documents and ArchiMind, containing the same documents with annotations.

4. CONCLUSION

The paper illustrated, by uses cases, how various issues with traditional software documentation can be addressed by using in-page semantic annotation and a SE domain ontology in semantic wikis. Correct annotation of documentation, using implicit and explicit knowledge, allows the use of ArchiMind as an expert system. It is expected that ArchiMind allows for more effective and efficient knowledge retrieval by software project members, resulting in increased overall software quality.

5. REFERENCES

[1] S. Nanz and D. L. Parnas, "Precise Documentation: The Key to Better Software," in *The Future of Software Engineering*: Springer Berlin Heidelberg, 2011, pp. 125-148.

[2] A. Tang, P. Liang, and H. van Vliet, "Software Architecture Documentation: The Road Ahead," in *proceedings of the 9th Working IEEE/IFIP Conference on Software Architecture (WICSA) 2011* Boulder, Colorado, USA, 2011.

[3] S. Auer, S. Dietzold, and T. Riechert, "OntoWiki - A Tool for Social, Semantic Collaboration," in *Proceedings of the 5th International Semantic Web Conference, ISWC 2006*, Berlin / Heidelberg, 2006, pp. 736-749.

[4] "Dublin Core Metadata Element Set, Version 1.1: Reference Description." vol. 2011: Dublin Core Metadata Initiative, 2004.