Determining similarity of model-based and descriptive requirements by combining different similarity measures

Katharina Wolter[1], Thorsten Krebs[1], and Lothar Hotz[1]

HITeC c/o Universität Hamburg, Department Informatik
Vogt-Kölln-Str. 30, 22527 Hamburg, Germany
{kwolter|krebs|hotz}@informatik.uni-hamburg.de

Abstract. RSL is a requirements specification language that was developed in the ReDSeeDS project. The language allows requirements specifications using both model-based and descriptive representations. In this paper we tackle the problem of determining similarity of requirements that use both types of representations. We argue that in this case a combination of similarity measures is needed. In order to confirm this claim we assess similarity measures from different research areas with respect to their suitability for comparing requirements specifications written in RSL.

1 Introduction

Reuse is still an open problem in current software development practice. Earlier approaches to solve this problem concentrated on the code level. However, the impact of reuse can be increased when integrated earlier in the development process. The work presented in this paper is part of the ReDSeeDS [1] project in which the participants develop a framework that supports reuse on the level of requirements. However, the reuse support is not limited to requirements specifications but covers all artefacts created in the software development process.

Starting with an initial requirements specification a repository is searched for similar specifications. This repository contains former software development projects stored in the form of software cases. A *software case* comprises a problem (requirements) and a solution (architecture, design and implementation). Each requirements specification is mapped to appropriate elements of the solution.[2] The retrieved case is then intended to be reused by modifying those pieces that need rework and keeping those pieces that can be reused without modification. Retrieval of similar requirements specifications from a repository is a key prerequisite for this approach.

[1] http://www.redseeds.eu, for a list of all participants we refer to the acknowledgements

[2] A complete description of a software case's internal structure is out of the scope for this paper. For a more detailed description of this approach we refer the interested reader to [1].

The requirements specification language (RSL) used for specifying requirements of a software case allows different kinds of representations, being more and less formal. Instead of relying on one similarity measure, in this paper we argue that a combination of different similarity measures is needed for determining similarity of requirements written in RSL. Therefore, we examine similarity measures from different research areas for their suitability with respect to different requirements representations.

The remainder of this paper is organised as follows. Section 2 briefly introduces the requirements specification language. In Section 3 we describe similarity measures from different research areas and assess their suitability for comparing requirements specifications written in RSL. We explain our concept for a combined similarity measure in Section 4. Finally, Section 5 concludes this paper with a discussion and summary.

2 The Requirements Specification Language (RSL)

The Requirements Specification Language (RSL) is a result of joint work from the ReDSeeDS project. In this section we only present those aspects that are relevant for the selection of similarity measures. RSL is defined using a metamodel approach. For more details we refer to [2, 3].

Typically, requirements specifications are written using natural language. Some approaches use constrained language in order to make the specification more precise and unambiguous.[3] RSL covers both approaches and allows to write requirements specifications in form of less formal *natural language hypertext* or more formal *constrained language sentences*. The constrained form of sentences has the advantage of being syntactically unambiguous and semantically rich and thus, provides a better basis for the retrieval. RSL provides two types of requirement representations: *descriptive* and *model-based* requirement representations. The descriptive type of representation offers scenarios in constrained language and sentence lists. The model-based representations provide activity and sequence diagrams similar to UML. However, both are based on the constrained language that is also used in scenarios.

A *sentence list* contains a list of sentences written in natural language hypertext or constrained language. Two sentence lists are equal if they contain the same sentences, independent from their order. A *scenario* contains a sequence of sentences written in constrained language. The order of sentences is important due to the fact that a scenario tells a story.

In RSL constrained language sentences contain links to a *domain vocabulary*, which is a software case-specific collection of notions that acts as a glossary and helps identifying all sentences in which the same notion appears. Links from natural language hypertext sentences to the domain vocabulary are also possible but not required.

In the ReDSeeDS framework, potentially reusable artefacts from former projects are identified based on the similarity of their requirements specifications.

[3] Two examples of constrained language text are *subject-verb-object (SVO)* sentences and the *Attempto Controlled English (ACE)*.

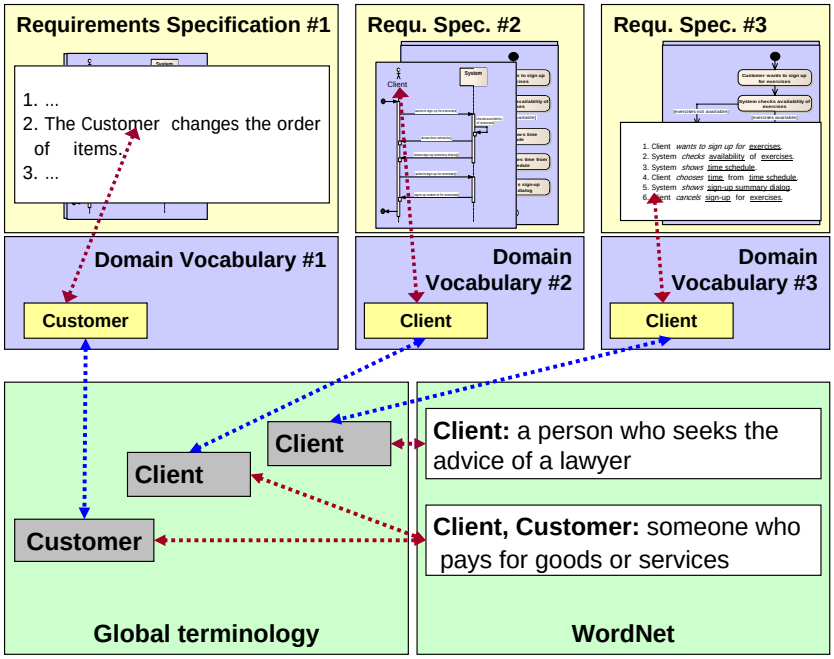

Fig. 1. Requirements specifications may use different requirements representations. Every specification has its own domain vocabulary from which there are links to the global terminology. The meaning of words in the global terminology is specified using WordNet.

However, in different domains the same word may be used with a different meaning. In order to unambiguously define the meaning of specifications and make them available for persons not integrated in the particular development project a case-independent *global terminology* is used (see Figure 1). Links from elements of the domain vocabulary to words of the global terminology allow unambiguously comparing software case requirements from different domains. We use the semantic lexicon WordNet[4] to specify the meaning of words (see Section 3.3).

3 Assessing Similarity Measures for Requirements written in RSL

One feature that distinguishes RSL form other approaches for requirements specification is that it offers different types of requirements representations, i.e. model-based and descriptive with the more formal constrained language and the less formal natural language hypertext sentences. A single requirements specification can contain different types of representation. Two requirements specifications can be conceptually equal, but have different representations.

[4] http://wordnet.princeton.edu/

While these different types of representations are a key feature of RSL, they are a special challenge for determining the similarity of requirements specifications. The measure must be able to handle the different types of representations. When the same requirement is represented differently in two specifications the similarity measure should still indicate the similarity in meaning.

Measures that capture the similarity of artefacts have been developed in many research communities for different types of artefacts. However, these similarity measures are typically developed for one specific type of artefact. Information Retrieval, e.g. addresses text-based documents while other approaches like SiDiff [4] compare model-based artefacts.

In order to support or disprove our argument that no single similarity measure is suitable for all representation types provided by RSL, in the following subsections we review measures from different research areas. The main focus is the evaluation of their applicability for determining the similarity of requirements specifications in RSL. For this goal we use following assessment criteria:

- The basic question of applicability is: *Which types of artefacts can be compared* (e.g. text document, UML model, etc.)?
- Different measures compute different types of results. Hence we want to know: *What is the meaning of a high similarity value*?
- Important for identifying similarity in meaning of requirements that are differently represented is the question: *Are the ambiguity and paraphrase problems solved* (see below)?

The *Ambiguity Problem* emerges when two artefact representations are the same but the actual meaning of the artefacts is different. The *Paraphrase Problem* describes the case where the artefact representations are different but the actual meaning of the artefacts is the same or at least similar [5].

These ambiguity and paraphrase problems are illustrated within Figure 1. Requirements specification #1 uses the word *customer* and Specifications #2 and #3 use the word *client*. But the actual meaning of the words *customer* and *client* in Specifications #1 and #2 is the same (synonyms cause the paraphrase problem); both words point to the same element in WordNet. Specifications #2 and #3 both use the word *client*, but with different meanings (homonym causes the ambiguity problem); the words link to different elements in WordNet.

3.1 Information Retrieval

Information Retrieval (*IR*) provides techniques for comparing text documents [6]. A major application domain of IR techniques is web search engines. Usually, large document collections are considered and the similarity measures are based on automatically generated document representations.

IR techniques can be applied to all artefact types that contain a reasonable amount of text. Traditional IR techniques consider artefacts equal if they contain the same words in the same frequency (stop words are not considered).

These techniques do not solve the ambiguity and paraphrase problem. Furthermore, structural information contained in the artefacts is not considered.

Two scenarios e.g., that use the same words but describe opposed procedures are considered to be equal by this approach.

3.2 Case-Based Reasoning

Case-based Reasoning (*CBR*) is the process of solving new problems based on the solution of similar past problems [7].CBR uses a predefined structure of attributes as case representations, which describe problem and solution of past cases. This structure can be one simple table or a relational data model with several tables. CBR has been applied in different domains such as medical diagnosis, fault diagnosis of technical systems or software reuse.

CBR is able to compare all types of artefacts. However, the information contained in the artefacts needs to be reduced to the fixed structure of the case representations. Due to RSL's flexible structure of requirements specifications (defined in the RSL metamodel), loss of information can not be avoided.

In CBR two cases are considered to be equal if they share the same case representation. The above cited CBR applications each consider one specific domain while our requirements specifications potentially span a variety of different application domains. Thus, the ambiguity problem becomes more important. Basic CBR techniques use traditional IR measures for string comparison. These techniques do not address the paraphrase and ambiguity problem. However, more sophisticated methods include taxonomies to solve these problems [8].

3.3 Using Taxonomies

Semantic lexica and taxonomies can be used to determine the meaning of words used in an artefact. One example for a semantic lexicon is WordNet[5], which was developed at the Cognitive Science Laboratory at Princeton University [9]. It is based on the concept of synonym sets (called *synsets*) that group synonymic nouns, verbs, adjectives and adverbs and defines, among others, the semantic relations *hypernyms / hyponyms* (generalisation) and *holonym / meronym* (composition) between synsets.

WordNet has been applied in combination with different approaches in order to improve retrieval results. A matter of current research in IR is the application of WordNet, or more generally semantic lexica, in order to solve the ambiguity and paraphrase problem [10, 11]. [12] used WordNet in combination with CBR approaches for retrieval of UML class diagrams.

Several similarity measures have been published based on WordNet (see [13] for an overview). These measures provide a similarity value for synset pairs, sometimes specifically for noun pairs or verb pairs. Since these measures cannot compare sentences or whole paragraphs they need to be integrated in other techniques. Most measures are based on path-lengths between synsets defined by semantic relations. Two synsets are considered similar when there is a short distance between them. Because the meaning of words is given the paraphrase and ambiguity problem do not emerge.

[5] http://wordnet.princeton.edu/

3.4 Structure-based Similarity

Structure-based similarity measures consider the structure of the artefacts to be compared. *Graph-based* and *Description Logics (DL)* approaches measure similarity of two artefacts by comparing the vertices and arcs or concepts and roles, respectively. Both approaches focus on the artefact's structure and basically compare subgraphs using both taxonomic comparison of elements and their relations to other elements. In contrast to CBR, the structure-based similarity measures can also handle flexible, i.e. only partly known, structures.

Structure-based similarity measures are well suited for comparing artefacts with a flexible structure like the RSL requirements specifications. The measures are not suitable for unstructured artefacts like plain text documents. However, RSL specifications can contain such plain text elements.

When the structure of artefacts is known, not all elements are compared blindly, but only matching elements. For RSL's restricted English sentences this means that nouns are compared to nouns only, verbs to verbs, etc. Additionally, not all nouns need to be compared to one another but subjects and objects can be distinguished. These approaches consider two artefacts equal when the same elements are represented with the same relations to other elements.

Finally, structure-based similarity measures can evaluate RSL hyperlinks from sentences to the domain vocabulary and global terminology in order to solve the ambiguity and paraphrase problem.

4 Concept for a combined similarity measure

The results described in the previous section show that no single similarity measure is able to compare all types of RSL elements. In the following we exemplarily describe a combination of different similarity measures to compare a query with the requirements specification of software cases stored in the repository. For a detailed description for comparing all RSL elements we refer to [14].

When comparing two constrained language sentences the structure of the sentences should be taken into account. Comparing the structure can be achieved, for example, by using a structure-based similarity measure. Due to the fact that all major elements of a constrained language sentence are linked with the global terminology, a WordNet-based measure is used to compare the single words. Similarity of sentence's structure and similarity of contained words are integrated into one similarity measure for constrained language sentences. For a combination of values from different measures a weighting of the values is needed. Reasonable values for such a weighting are to be determined in the upcoming experiments (see next section).

In contrast to constrained language sentences the natural language hypertext sentences are not highly structured. For their comparison only the text representation and the potentially contained links to the global terminology are evaluated. Information Retrieval approaches are used to compare the sentence's text and a WordNet-based similarity measure is applied for comparing linked

words. This combined measure is used when comparing two natural language hypertext sentences but also when comparing a natural language hypertext sentence with a constrained language sentence.

The comparison of sentence lists is based on the measures for sentences. The sentences of both lists are compared pairwise. For each sentence of the one sentence list that is contained in the query the maximum similarity to any sentence in the other sentence list identifies the best match. The similarity between the two sentence lists is the normalised sum of the maximum similarities. The same measure is used when comparing a sentence list with a constrained language scenario. However, when two constrained language scenarios are compared to one another, the order of sentences are taken into account. Moreover, constrained language scenarios only contain the well-structured constrained language sentences. Thus, they are compared using a combination of Structure-based measure and WordNet-based measure.

5 Summary and Discussion

This paper presents a novel approach to comparing requirements specifications that use different representations. The requirements specification language RSL allows to specify requirements using both model-based and descriptive elements. In order to compare requirements specifications written in RSL we assessed similarity measures from different research areas according to their suitability and proposed a combination of similarity measures for this task.

In the following we present some topics for discussion and future work that will be addressed when evaluating the approach.

A topic of current research is the question whether two requirements with the same meaning but different representations should be treated equal or not. Apparently, two such requirements specify the same thing but we may want to consider their representation type because requirements engineers may like to work with specific requirements representations better than with other ones and thus react differently on the retrieval of different representation types.

A matter of evaluation is also in how far the ambiguity and paraphrase problems appear in real projects and how well our combined similarity measure solves them. It may happen that requirements engineers use only informal descriptive requirements representations without terminology hyperlinks, in which case IR techniques without terminology would not solve the two problems.

Finally, fine tuning will be needed for the selected similarity measures. When numeric values are computed that denote similarity between two elements, then threshold values or weights are used. During the evaluation of the approach, our combined similarity measure will be iteratively improved.

The evaluation will be done by using industrial application areas with a reasonable number of requirements specifications written in RSL and a tool prototype, which is currently developed in the ReDSeeDS project.

Acknowledgments

This research has been supported by the European Community under the grant IST-2006-33596 (ReDSeeDS). The project is coordinated by Infovide, Poland with technical lead of Wasaw University of Technology and with University of Koblenz-Landau, Vienna University of Technology, Fraunhofer IESE, University of Latvia, HITeC e.V. c/o University of Hamburg, Heriot-Watt University, PRO DV, Cybersoft and Algoritmu Sistemos.

References

1. Śmiałek, M.: Mechanisms for requirements based model reuse. In: International Workshop on Model Reuse Strategies - MoRSe. (2006) 17–20
2. Kaindl, H., Śmiałek, M., Svetinovic, D., Ambroziewicz, A., Bojarski, J., Nowakowski, W., Straszak, T., Schwarz, H., Bildhauer, D., Brogan, J.P., Mukasa, K.S., Wolter, K., Krebs, T.: Requirements specification language definition. Project Deliverable D2.4.1, ReDSeeDS Project (2007) www.redseeds.eu.
3. Kaindl, H., Smialek, M., Mukasa, K.S., Wolter, K., Bildhauer, D., Pooley, R.J.: A comprehensible requirements specification language based on a metamodel. submitted to: 16th IEEE Requirements Engineering Conference (RE'08) (2008)
4. Kelter, U., Wehren, J., Niere, J.: A generic difference algorithm for UML models. In: Proceedings of the SE 2005, Essen, Germany, Essen, Germany (2005)
5. Lenz, M., Bartsch-Spörl, B., Burkhard, H.D., Wess, S., eds.: Textual CBR. In: Case-Based Reasoning Technology - From Foundations to Applications. Springer (1998) 115–137
6. Grossman, D.A., Frieder, O.: Information retrieval: algorithms and heuristics. Springer (2004)
7. Aamodt, A., Plaza, E.: Case-based reasoning: Foundational issues, methodological variations, and system approaches. Artificial Intelligence Communications **7**(1) (1994) 39–59
8. Bergmann, R.: On the use of taxonomies for representing case features and local similarity measures. In: 6th German Workshop on CBR. (1998)
9. Fellbaum, C., ed.: WordNet: An Electronic Lexical Database. MIT Press (1998)
10. Varelas, G., Voutsakis, E., Raftopoulou, P., Petrakis, E.G.M., Milios, E.E.: Semantic similarity methods in WordNet and their application to information retrieval on the web. In: WIDM '05: Proceedings of the 7th annual ACM international workshop on Web information and data management, New York, NY, USA, ACM Press (2005) 10–16
11. Liu, S., Yu, C., Meng, W.: Word sense disambiguation in queries. In: CIKM '05: Proceedings of the 14th ACM international conference on Information and knowledge management, New York, NY, USA, ACM Press (2005) 525–532
12. Gomes, P., Pereira, F.C., Paiva, P., Seco, N., Carreiro, P., Ferreira, J.L., Bento, C.: Using wordnet for case-based retrieval of uml models. AI Commun. **17** (2004) 13–23
13. Pedersen, T., Patwardhan, S., Michelizzi, J.: WordNet::similarity - measuring the relatedness of concepts. In: Proc. 19th National Conference on Artificial Intelligence (AAAI-04). (2004) 3p
14. Wolter, K., Krebs, T., Bildhauer, D., Nick, M., Hotz, L.: Software case similarity measure. Project Deliverable D4.2, ReDSeeDS Project (2007) www.redseeds.eu.