Finding Reusable Units of Modelling – an Ontology Approach

[1] Marcus Zinn, [2] Klaus P. Fischer-Hellmann, [1] Andy D. Phippen, [2] Alois Schütte

[1] Centre for Security, Communications and Network Research
University of Plymouth, United Kingdom
[2] h_da – University of Applied Sciences Darmstadt, Germany
mail@marcuszinn.de

Abstract: Today's software units (classes, components and services) have a huge number of information that is needed or produced during the development and use of these units. In fact, a single piece of information can have different values depending on the point of time in the entire lifecycle. The availability of certain information, as for example documentation, determines among other things the capabilities of a unit. Again, other information is necessary and critical for the success of the entire development process when applying certain procedure models. Retrieval of these units and their contents is important for re-use. There are no suitable models that consider the different units and their contents. Also the current searching behaviour of software developers and architects has not been covered yet. Due to this fact, the benefits in performing reusing software units and the development of software processes are decreasing.

This paper discusses an ontology approach that can be used as a foundation for the search of such units. Moreover, this part of the ontology is focused on the actual searching behaviour of software developers and the finding of units.

1 Introduction

In the object-oriented software development, different units of modelling are used. Every type of unit provides a different amount of information that can be used differently [ZFP09]. Typical units are classes, components and services [WF04]. In the scope of this paper, a component has the meaning of a deployed component. There are two problems: development issues related to a common view of these different units [WF04] and the search for these units [WJS09]. The search for units as a research subject has already been studied for some time. [Pr91] and [MBC91] proposed first approaches. [Ga06] and [LAP04] show a list of the different attempts that have been developed until now. Among other problems, the following problem has been identified:

"*Efficient search and retrieval is needed, to assure that the developer is capable of finding previously built reusable assets.*" [Ga06]

For this reason, the question arises what an efficient way for a search could be. The current research focuses on the use of semantics in form of ontologies as a foundation of a search (see [TSB09] and [BSW08]). Some of these new attempts focus on the representation of the technical circumstances, as described in [HNK09]. Other studies concentrate on the grammatical structure in such a search [WJS09]. These approaches assume a complicated predefined input behaviour. [He94] showed already in 1994 that there is a significant gap between the description of the problem and that of the solution. Therefore, components are described functionally whereas the searcher actually describes the problem.

In the following paragraphs, the results of the analysis of the present "searching behaviour" of software engineers are presented. Based on these results, an existing ontology is extended. This work is part of a research on a service-based software construction process (SSCP) incorporated the field of Software Reuse Environments. The paper contributes to the research area with the enhancement of an ontology for supporting the search of units of modelling. Aim of this paper is to define the extension of an ontology in order to reflect today's searching behaviour of software engineers. This can be used in a semantic model to find units of modelling. Therefore, the input behaviour must be determined and modelled. Furthermore, an ontology defined by the authors within the scope of the basic research should be extended. This paper concludes with the fact that the input behaviour does not have to be changed when searching for reusable units in order to achieve exact results.

2 Analysis of the searching behaviour for units of modelling

To get a first impression how software developers tend to search for reusable pieces of software, a questionnaire was given to a group of software development experts. Conducting a representative survey is left as future work in this research. The 15 participants of this questioning were software developers, software architects, and technical project managers who had at least a three-year experience in software development. When the survey was performed, the test persons were active in different software development projects in one of the following areas: CAD, automation, power, or in general software development. 93% of the feedback indicated the use of a general search engine (in most cases www.google.de) in order to search for units. A relationship can be seen in the examples given by the test persons (i.e., "Class C# Device Discover") and the given search criteria (e.g., manufacturer and technology). Therefore, the analysis shows that there is important and optional information in this relationship (see Layer 1 in Figure 1). The result of the questioning is presented in the following paragraphs as "actual searching behaviour". Besides, it constitutes a hypothesis of the authors. From the examples of the search enquiries, the analysis of the given search information is displayed in Figure 1. The important information from Layer 1 refers to the functional application object (or content purpose) for that the functionality of the unit (Layer 2) is searched. Layer 2 corresponds to the following structure:

Searched technical contents (application object) + optional describing information (for the technical contents and/or the technical unit).

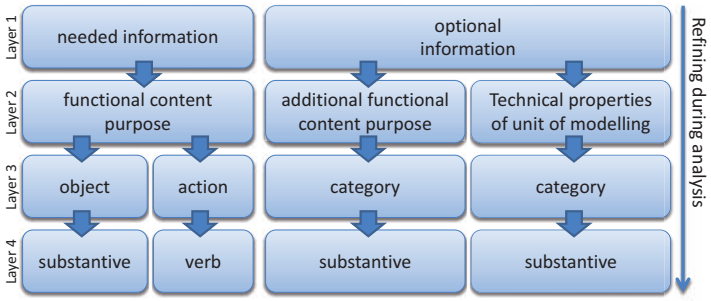

Figure 1: Structure of the search input

In addition, the given examples of the search show that the functionality is described in most cases by an action and an object connected with this action, for instance, "printer" (object) "search" (operation or action) (Layer 3). This example shows the simple substantive-verb relationship in the grammatical area (Layer 4). [WJS09] groups such relationships within the scope of the parsing for search algorithms and refers to it as "Advanced Similarity Word Pair". Optional information (Layer 1) is divided into two areas. On the one hand, the application object (functional content purpose) is further described. On the other hand, the technical properties of the unit itself can be described (Layer 2). In both cases, categories are used, for example, "WebService C# Device Discover", and it can be assumed that a web service based on the C# technology is searched. In the grammar of the search, it only concerns a few substantives. From the pattern shown above, the following grammatical construction can be derived:

Functional content purpose (Substantive + Verb) + additional functional content (Substantive) + technical content (* Substantive).*

Beside the grammatical construction and the contents of the search, another important point reveals itself in the analysis: The "problem-solution" relationship. All test persons described the solution in their search (e.g., a class carries out a function for the Gaussian algorithm), but not the problem. Therefore, this factor is interesting because a component may solve different problems (perhaps also in a different way). Furthermore, a problem can refer to several solutions. When questioning the participants why they do not search for the other position (in this case the problem), the answers were quite different. Two types of responses were mentioned remarkably frequently:

1) During the search for the problem, solutions can be hardly found.
2) During the search for the problem, the problem must be described precisely in order to find a precise solution.

This stands in contrast to the statement from [He94] that the searching person describes the problem, not the solution.

3 Problems of finding units of modelling

Nevertheless, the approach described in Chapter 2 contains some problems:

Full-text search: The use of the search engines making text comparisons can lead to false or not usable results [TSB09].

Substantive verb description: A simple substantive-verb structure in a relationship-based search engine faces following problems: On the one hand, the substantive can be selected unspecific ally (i.e., device), although a printer can be the searched object. This entails the generation of unsuitable hits during the search. On the other hand, verbs and substantives can have synonyms (e.g., graphics and display graphics) or they can be wrongly associated [WJS09]. The last example also shows that in some cases it may be a matter of interpretation. From the point of view of the automation, "machine is computable" seems illogical because a machine does not change. However, this statement makes sense from the CAD point of view because a machine must be recalculated by the change of knowledge-based properties [Cl06]. This includes the reconsideration of engine space due to the update of cubic capacity size. This instance can lead to a change of the whole vehicle. In addition, a problem arises concerning the existence or non-existence of a word in another language. Thus, a search launched with the German expression "Gerät suchen" will not be able to find a component described as "Device discover".

Consistency of the statement: The shown example "web service C# Device Discover" does not state to which "web services" and "C#" they really refer. Hence, a search formulated such can lead to false results:

1) A web service written in C # is searched that performs "Device Discovery".
2) A component is searched that contains a web service or rather uses one and performs "Device Discovery". This component should have been developed in C#.

For an exact allocation of the given information, other details are missing. In this example, the information is clearly allocated to "C#". The problem in this case is that such an input does not specify whether the information is optional or mandatory.

Problem-solution relation: A search for the solution as described above presents all the solutions that fit to given keywords and their relations. By means of ratings (evaluations, frequency of the choice, etc.), statistical probabilities can be determined for the best result [Ga06]. It is, however, an open question whether information about the problem is missing or has already been considered satisfactorily in the solution description. In spite of these significant problems and open questions involved, the survey shows that this searching behaviour nevertheless is actually used. Therefore, an attempt was made to cover the existing searching behaviour in an ontology. On this basis, it can be investigated how to improve the search result while using the same input behaviour.

4 A search ontology for reusable units of modelling

4.1 Structure

In context of the current research, an ontology to the subject “service-based software construction process” was developed by the authors in order to counteract the problems explained in Chapter 3. This ontology serves only the search of units of modelling. Here, certain modelled properties were incorporated by other ontologies (e.g., technical component properties from [Ga05]) since these have already been edited. This also includes the description of technological facts (components, services, etc.). Figure 2 shows the distribution within this ontology.

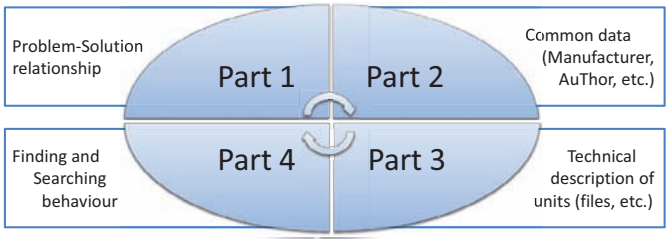

Figure 2: Structure of the reuse ontology

Part 1 shows the access to the ontology: “the problem-solution approach”. This is still the untreated part of the whole research. Part 2 contains general “business information” about the solution as, for example, manufacturer, name, and author. In Part 3, the solution is described as a technical unit; that is, type of unit, technology, file format, files, etc. In the fourth part, the technical contents are described. Possible descriptions are made, for example, in form of a substantive-verb combination and they also contain some optional information. This part of the ontology will be described in this publication.

If an instance of the ontology is generated (e.g., by the registration of a newly developed unit), the user must specify various information that is stored in the suitable areas of the ontology. Furthermore, the data can be entered automatically into Part 3 of the ontology, for instance. This is possible because the technical data is automatically detectable such as file size, file type, file name, and technology. Nevertheless, the data from the other sections of the ontology is not automatically detectable.

In the following, the modelling of the searching behaviour displayed in Chapter 2 will be described in more detail. This corresponds to Part 4 of the ontology. Moreover, it is focused on the problems indicated in Chapter 3.

4.2 Description of the technical and professional contents

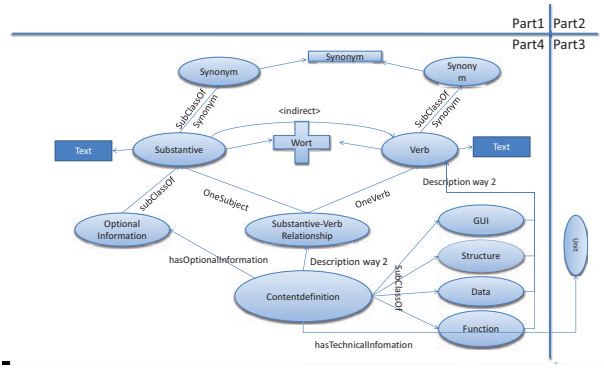

Figure 3: Model of the finding based upon substantive-verb relation

A unit has a so-called “content definition” describing the technical contents. It also uses two different ways of description. These ways are related to each other. The modelling of the optional information for the technical properties is made in Part 3 of the ontology and will not be described in this publication.

4.2.1 Way of description 1

The first way of description is the substantive verb combination explained in Chapter 2 and represents a technical and domain specific description of the contents. In detail, each element (substantive and verb node) of this tuple has a text field and each substantive and verb can have a translation. Within this ontology, this corresponds to a text. At this point, however, an ontology shortcut to a language ontology allowing translations is planned. For this reason, Figure 3 is simplified and as a result the element “word” (+ icon) is displayed as a shortcut to a word ontology. Based on it, cross-language searching and finding elements are possible. Therefore, “device discover” corresponds to “device search”. Similarly, the translations will also proceed with synonyms. A “printer” is a special “device”. Thus, the search for a device may deliver “printer” if appropriate. At this point, an ontology can also be used (“- icon” synonym).

The optional information for the application object is also displayed and modelled as substantives. In order to create the description of the technical properties, Part 3 of the ontology is related to the content definitions object.

4.2.2 Way of description 2

The second part of the description of the solution defines the technical contents from the point of view of its intended purpose. This definition is based on the fact that a unit of modelling may be seen from three different perspectives: Functional contents, technical properties, and technical contents. As previously presented [ZTP08], a component carries only one certain technical content type. Therefore, a component offers either func-

tions such as simple data, user interfaces or it provides structure information to the solution of a problem.

4.2.3 Search variations on the basis of the ways of description

Because of the mentioned features, an ontology-based search can be simply expressed: for instance, “function device discovery”. In this example, “function” represents optional information. Hence, an attempt was made to generate an indirect relationship between the technical and the professional contents. Two variations have evolved that will be explained in the following.

Variant 1: Verbs as synonyms for technical content types

In this case, four contents types (Data, Function, UI and Structure) are associated with certain verbs. These verbs fit to the content type (e.g., a function is a content type of something that executes something; UI is a content type of something that illustrates). Thus, a function can be “executed” or graphics can be “illustrated”, for instance. Table 1 shows some examples of a possible assignment.

Content type	assigned verbs
Function	calculate, execute, accomplish, bear, manage
Data	offer, suggest
UI	show, present, demonstrate
Structure	structure, align, regulate, arrange, classify

Table 1: Technical content type-verb relation

For each instance of the ontology, this allocation would be firmly “wired”. Moreover, only a few verbs are associated to the content types. With the help of this assignment, the search could be “execute device discover”. In addition, synonyms and translations are available for this search.

Variant 2: Direct links of the substantive verb tuple with the content types.

In contrast to variant-1, the verbs from the substantive-verb tuple are now connected directly with the content types. Although the allocation from Table 1 can be maintained, every entered verb, however, must receive an allocation. A result would be that the search enquiry “device discover” will search for units offering a function that searches for devices. In comparison to the first variant, only two words are required instead of three.

4.2.4 Realisation of a search

To launch the search, a search query must occur at first such as “device discover C#”. Part 4 of the ontology can be used for the identification of the substantive-verb tuples. All the other terms (in this case “C#”) are understood as optional terms and are searched

for within the remaining parts of the ontology. C# is a technology whose relationship with the component is modelled in Part 3 (technical information). From the perspective of the ontology, the search is called “unit has content definition with tuple (device-discover) and has a relationship with C#”. Moreover, with the perspective of the search in Variant 2, it is obvious that the user searches for a function and not for simple data, a user interface, or structure information.

4.3 Problem solution

In Sections 4.1 and 4.2, the ontology does not solve every problem mentioned in Chapter 3. The use of ontologies in order to avoid the problems of text-based search is not new [TSB09]. It is already known that because of their logical structure, ontologies are suited to perform inheritance hierarchies. An example that could be expressed is “device is a printer” [St09]. The use of simple substantive-verb tuples describing the technical contents with fixed verb-content mapping presents a novelty. In comparison to a 100% semantic search with input methods considered complicated [WJS09]; this approach can lead to more “wrong” search results. However, a “substantive-verb content type” triple can arise from a search enquiry. This is the result of the semantic assignment of substantive-verb-tuple to a unit as well as the allocation of the verb-content type “substantive verb”. As a result of such a search, only the units owning this triple are performed. In contrast to a text-based search that seeks words in all texts of a data record, the input words are analysed in their relationship and are only searched if they are related to that relationship. The optional information is used to improve the search result. Variants 1 and 2 from Chapter 4.2.3 indicate that there are different possibilities to model the relations between professional and technical contents. Variant 2 is identical to the searched input behaviour but provides more exact results. Table 2 shows the search results of the given input “device discover C#”:

Dataset	**Text-based search**	**Variant 1**	**Variant 2**
Some component with the description “device Microsoft discover c#”	hit	no hit	no hit
Some component with a description triple “device-discover-function” and optional description “Microsoft c#”	hit	no hit	hit
Some component with a description triple “device-discover-data” and optional description “Microsoft c#”	hit	no hit	no hit

Table 2: Example of search results

The text-based search in Table 2 provides a hit for each data record because the searched data is available. Variant 1 delivers no hit because the search enquiry does not display the substantive-verb-content type triple. Variant 2, however, delivers exactly one hit. Although only the tuple “device-discover” was entered, the triple “device discover function” was implicitly also searched. This reduces the number of possible hits in contrast to the entered tuple.

5 Conclusion and future work

The ontology approach shown in this paper contains a semantic modelling of the following (search) input pattern for the search of reusable units:

Functional content purpose (Substantive + Verb) + additional functional content (Substantive) + technical content (* Substantive).*

This allows the searching behaviour that appears to be broadly applied nowadays to text-based search engines also to be applied to semantic search engines. Thereby, it is possible to make use of the usual advantages of an ontology as, for example, using a short-cut to other ontologies and the advantages of a semantic search (see [St09]). This leads to a better result in contrast to a text-based search (see [TSB09]) because a text-based search only compares the searched words with the dataset. In order to have an exact result, the searched words must be in a certain semantic relationship (*Substantive + Verb)* and must be implicitly combined with the technical content type (Structure, UI, Data or Function) of the searched unit. This simple approach combined with the typical information about reusable units of modelling (i.e., manufacturer and technical information) represents an innovation to the area. This paper shows in an example that this approach works. The search pattern can be grasped completely in an ontology without changing the effort or the input for the user. In addition, this publication shows that the input behaviour of software engineers identified as typical does not have to be changed and a better search result can be achieved in comparison to a text-based search.

However, not every problem is solved by the solution presented in this publication. On the one hand, it is not finally clarified whether it is better to describe the solution only with a search or whether the problem should be described as well. On the other hand, the possibility to recognise whether the optional information describes the application object, the technical unit, or the technical contents is missing. Within the scope of the further research of "service-based software construction", these two open problem formulations, in particular, will be analysed in more detail. However, the searching behaviour still is supposed not to change.

References

[BSW08] Bast, H.; Suchanek, F.; Weber, I.: Semantic Full-Text Search with ESTER: Scalable, Easy, Fast. In: International Conference on Data Mining Workshops (ICDMW '08)-Proceedings, Pisa Italy, 2008; pp. 959-962

[Cl06] Claassen, E.: Protection of Intellectual Property in the Product Development Process. In: Proceedings of the 11th Seminário Internacional de Alta Technologia, Universidade Metodista de Piracicaba, Brasil, 2006

[Ga06] Garcia, V. C.; De Almeida, E. S.; Lisboa, L.B.; Martins, C. A.; Meira, A. R. L.; Lucredio, D.; De M. Fortes, R. P.: Toward a Code Search Engine Based on the State-of-Art and Practice. In: Asia Pacific Software Engineering Conference-Proceedings, Bangalore, India, 2006; pp. 61-70

[Ga05] Gangemi, A.; Grimm, S.; Mika, P.; Oberle, D.; Lamparter, S.; Sabou, M.; Staab, S.; Vrandecic, D.: Core Software Ontology - Core Ontology of Software Components - Core Ontology of Services, 2005, online available at http://cos.ontoware.org, (Accessed 14.1.2009)

[He94] Henninger, S.: Using Iterative Refinement to Find Reusable Software. In: IEEE Software Journal, Vol. 11(5), 1994; IEEE; pp. 48–59

[HNK09] Hewett, R.; Nguyen, B.; Kijsanayothin, P.; Efficient Optimized Composition of Semantic Web Service. In: IEEE International Conference on Systems, Man, and Cybernetics (SMC 2009) Proceedings. San Antonio, USA, 2009; IEEE; pp. 4065 - 4066

[LAP04] Lucrédio, D.; Almeida, E. S.; Prado, A. F.: A Survey on Software Components Search and Retrieval. In: Proceedings of the 30th EUROMICRO Conference, Rennes, France, 2004; IEEE/CS Press; pp. 152–159

[MBC91] Maarek, Y. S.; Berry, D. M.; Kaiser, G. E.: An Information Retrieval Approach for Automatically Constructing Software Libraries. In: IEEE Transactions on Software Engineering Journal, Vol. 17(8), 1991; IEEE; pp. 800-813

[Pr91] Prieto-Díaz, R:, Implementing faceted classification for software reuse. In: Communications of the ACM, Vol. 34(5), 1991; ACM; pp. 88–97

[St09] Stuckenschmidt, H.: Ontologien – Konzepte, Technologien und Anwendungen, Springer Verlag, Heidelberg Germany, 2009, ISBN 978-3-540-79330-4

[TSB09] Tümer, D.; Shah, M. A.; Bitirim, Y.: An Empirical Evaluation on Semantic Search Performance of Keyword-Based and Semantic Search Engines: Google, Yahoo, Msn and Hakia. In: 4th International Conference on Internet Monitoring and Protection, Venice/Mestre, Italy, 2009; IEEE, pp. 51-55

[WF04] Wang, G., Fung, C. K.; Architecture Paradigms and Their Influences and Impacts on Component-Based Software Systems. In: Proceedings of the 37th Hawaii International Conference on System Sciences (HICSS'04), Big Island, Hawaii, 2004;IEEE ; pp. 1-10

[WJS09] Wang, H., Jing, L., Shao, H.: Research on Method of Sentence Similarity Based on Ontology. In: Proceedings of the First WRI Global Congress on. Intelligent Systems (GCIS 2009), Xiamen, China, 2009; IEEE; pp. 465-469

[ZFP09] Zinn, M.; Fischer-Hellmann, K. P.; Phippen, A. D.: Development of a CASE tool for the service based software construction. In: Proceedings of the 5th Collaborative Research Symposium on Security, E-learning, Internet, and Networking (SEIN'2009), Darmstadt, Germany, 2009; Centre for Security, Communications and Network Research; pp. 134-144

[ZTP08] Zinn, M.; Turetschek, G.; Phippen, A. D.: Definition of software construction artefacts for software construction. In: Proceedings of the 4th Collaborative Research Symposium on Security, E-learning, Internet, and Networking (SEIN'2008), Wrexham, UK, 2009; Centre for Security, Communications and Network Research; pp. 79-91