Semantics-Based Component Repository: 
Current State of Arts and a Calculation Rating Factor-based Framework

Chengpu Li, Xiaodong Liu and Jessie Kennedy
School of Computing
Napier University
Edinburgh, UK
{c.li, x.liu, j.kennedy}@napier.ac.uk

Abstract

Nowadays, Component-Based Development (CBD) has been widely used in software engineering. As a core activity in the CBD process, component retrieval has attracted extensive research attention. Despite the research efforts so far, one major problem still exists in component retrieval, that is, the mismatch between user’s query and identified components. This paper proposes a semantics-base approach to tackle the mismatch issue in component retrieval process effectively. A review is first presented on the current state of relevant technologies involved in semantics-based component retrieval, and on the current development in this field. Component retrieval rating factors and their calculating method are proposed. The rating factors are used to illustrate and improve the precision of component retrieval in quantitatively.

1. Introduction

At present, software is widely used in everyday life of almost every sector. To satisfy the demand of various users in terms of efficiency, flexibility, reliability and security, to name a few, software has become more powerful in function, and more complicated in structure. As a result, software programs have become larger and more complex, and thus more time-consuming for the programmers to deliver. Having been able to help deliver a more user-friendly system within a tight timeframe, Component-based Software Engineering (CBSE) started to develop in the late 1980s. As a core activity in the component-based development process [3], component retrieval has been researched extensively for years. However, the mismatch between user query and identified components still remains as a major hurdle of component reuse. To improve the precision of component retrieval, a semantics-based component retrieval method is proposed in this paper.

Based on a critical analysis of the existing work of semantics-based component retrieval, we proposed a rating factor method to calculate the precision degree of the retrieval result.

The remainder of the paper is organized as follows: section 2, 3 and 4 present a critical analysis of current state of arts of the related research areas, including component retrieval, ontology language and domain knowledge. Section 5 analyzes existing ontology-based component retrieval approaches. Section 6 proposes an ontology-based rating factor method to calculate the level of component retrieval precision. Section 7 presents the conclusion and recommends the further work.

2. Component retrieval

As mentioned above, one of the great hurdles in component-based development is how to retrieve the suitable components. Among the bulk of solutions proposed so far, none of them surpassed satisfactorily the known hurdles in searching and retrieving relevant components from a repository.

The existing approaches can be classified into four different types, i.e.:

a) Simple keyword and string search

Simple keyword searching [8] is most widely used by search engines, whereby a user can locate the target object via specifying a set of keywords. The search engine will compare them with the labels of the objects and retrieves the matched. This is the most simplified approach without giving consideration to any additional information, such as relationships among objects or synonymous labels.

b) Faceted classification and retrieval

The faceted classification [9] [10] approach attempts to classify the objects in the repository based on predefined taxonomies, e.g. subjects.
Although this approach is useful for objects that can clearly fall into such categories, it is less useful for those without explicit classification.

c) Signature matching
Signature matching [14] [15] focuses on the type and number of arguments defined for methods and in essence, it approaches matching indirectly by identifying whether an object is relevant.

d) Behavioural matching
Behavioural matching [4] is the most involved approach with consideration of the functional behaviour of objects, in which objects are provided with input vectors, and the outputs generated are compared to the expected outputs. Objects that exhibit certain behaviour are retrieved and presented to the user.

All the above search methods lack accurate semantics of components and query requirements and fail to reach the best precision. To resolve these drawbacks, many researchers have developed valuable solutions so far. Their work is detailed in section 4.

3. Ontology-based languages and methods for component description and retrieval

In the last section, we discussed some main methods of component retrieval such as simple keyword and string search, faceted classification and retrieval. Their limitation in the accuracy of retrieval motivated researchers to identify more effective methods for component retrieval. One of the directions is to use ontology-based languages for component description and retrieval. Various ontology languages have been developed for this purpose, from RDF to RDFS, DAML+OIL to OWL.

3.1 OWL

The OWL language is a research-based revision of the DAML+OIL web ontology language, which is designed for use by applications that need to process the content of information rather than just present the information [5].

OWL can be used to explicitly represent the meaning of terms in vocabularies and the relationships between those terms. This representation of terms and their interrelationships is called an ontology. OWL has more facilities for expressing meaning and other semantics than XML, RDF and RDFS, and thus OWL goes beyond these languages in its ability to represent machine interpretable content on the Web.

OWL DL is one increasingly expressive sublanguages of OWL, it was designed to provide the maximum expressiveness possible while retaining computational completeness, i.e., all conclusions are guaranteed to be computed, decidability, i.e., all computations will finish in finite time, and the availability of practical reasoning algorithms. OWL DL includes all OWL language constructs, but they can be used only under certain restrictions. For example, number restrictions may not be placed upon properties which are declared to be transitive. OWL DL is so named due to its correspondence with description logic, a field of research that has studied the logics that form the formal foundation of OWL.

Any ontology description language will be featured with its semantic capacity, accuracy in expressiveness and effectiveness in computability. Based on the knowledge of DAML, OIL, DAML+OIL and OWL extended the semantic capacity of RDFS and improved its accuracy in description [6]. However, with the increase of complexity in computation and reasoning, it becomes less effective in computability. The Descriptive Logic serves an important role in the balance of expressiveness and computability of OWL.

The Semantics for OWL are given through translation to a particular Description Logic. Therefore OWL is both syntax for describing and exchanging ontologies and has a formally defined semantics that give the meaning.

3.2 Description Logic

To date, Description Logic (DL) has become a cornerstone of the Semantic Web for its use in the design of ontologies. The OWL-DL and OWL-Lite sub-languages of the W3C-endorsed Web Ontology Language (OWL) are all based on Description Logic.


Description logics are a family of knowledge representation languages which can be used to represent the terminological knowledge of an application domain in a structured and formally well-understood way. The name description logic refers to:
i). concept descriptions used to describe a domain; ii) the logic-based semantics which can be given by a translation into first-order predicate logic. Description Logic was designed as an extension to frames and semantic networks, which were not equipped with formal logic-based semantics. DL is thus sound and complete.

The ontology languages have been used in several semantic-based, i.e. ontology-based in this context approaches.

4. Ontology-based component retrieval approaches

Semantics-based (ontology-based) approaches have been applied for component representation and retrieval. Several process architectures from user queries to component retrieval have been built as well. The major approaches are summarized as follows:

**Case One [13]**

Step 1: Initial Query Generation
Accept a user query in natural language, and apply natural language processing technology to analyze a user query based on semantics, and translate the query into conceptual graphs, then into WSDL/RDF.

Step 2: Query Refinement
A program understanding tool is applied to analyze downloaded software packages; software services and features associated with the package will be translated into conceptual graphs also, then into WSDL/RDF.

Step 3: Component Retrieval
Finally a semantic matchmaker will search for desired software components by matching the user query conceptual graphs to components’ conceptual graphs. Then WSDL/RDF representations of the user query and components will be matched; a comparison will be made between conceptual graph matching and WSDL/RDF matching.

**Case Two [7]**

Step 1: Initial Query Generation
User can specify the requirements for component in natural language. System converts it to the web ontology language to ensure the correct terms are used in the query.

Step 2: Query Refinement
Query reasoning and retrieval expand. In this part the initial query is expanded and refined. The request of retrieval is mapping against the ontology model to ensure that correct terms are used in the query. The context of the retrieval is established through the ontology model.

Step 3: Component Refinement
According to the semantic description of retrieval request, the semantics-based component retrieval approach carries out semantic retrieval and matching in component repository through search engines. If the results of retrieval are not satisfied with user’s demand, the reasoning engine will carry out retrieval by combining latent semantics with derivation semantics, and returns the collection of results finally.

**Case Three [12]**

Step 1: Initial Query Generation
The user employs imperative or nominal sentences for specifying the requirements for components. A heuristic-based approach is used to identify keywords and concepts expressed by the user and to generate an initial query.

Examples of the mechanisms used consist of: i) a set of terms or phrases that indicate the initiation of a query; ii) a set of pronouns (ignored during query processing); iii) articles that can be eliminated through parsing; and iv) heuristics for creating SQL statements.

Step 2: Query Refinement
The keywords and concepts identified in the previous step are mapped against the ontology to ensure that correct terms are used in the query. Related terms based on the context of the retrieval are also identified for query expansion.

The context of the retrieval is established through the domain model. For example, if the user is interested in implementing a particular feature for an auction site, the corresponding objective within the auction domain model is used as the anchor point to determine the necessary processes and actions that must be part of the system and the appropriate components to support them.

Step 3: Component Retrieval and Feedback
The functional requirement specified by the user is decomposed into specific processes and actions using the domain model. They are then compared to the
methods of objects. The percentage of actions supported by a particular object indicates how relevant it is to that requirement. Objects that meet a certain threshold value (e.g., a percentage) are automatically retrieved. The user can determine the usefulness of these objects and provide feedback, based on which the threshold value could be adjusted.

From the foregoing cases, it is concluded that all these approaches have established in their studies the fundamental scheme of: i) Initial Query Generation, ii) Query Refinement, iii) Semantic Match; and have incorporated the domain knowledge. The retrieval was improved by semantically matching between a user query semantic representation and software component semantic descriptions against a domain ontology.

The differences in these approaches underlay with the initial language process, the ontology-based languages in use and the validation methods.

5. Application domain knowledge

To achieve high level of retrieval precision, it is important to describe the user’s query requirements and the components formally in ontology languages in the process of ontology-based component retrieval so as to create a viable map. The relationship between component description and domain ontology carries equal significance.

In this regard, the domain knowledge of the components is discussed with focus on Information Visualization [2], which involves the development and application of novel information visualization techniques in a range of application areas for various types of data. Given the access to their vast experience and existing application codes, it is proposed to choose Information Visualization as the application domain for process of component retrieval.

6. Ontology-based component retrieval rating factor and calculation

In view of the foregoing works, it is clear that the ontology-based approach can be successfully introduced into the component retrieval process. Regarding the issue to increase the retrieval precision, we need to define the degree of precision in the result. We thus propose to set up a rating factor to calculate the degree of similarity between the user’s query and the components. This method is to establish a calculation mechanism, which is composed of two parts: the multi-layered user’s query description and the multi-layered component description. This section first defines the above two parts of the mechanism respectively. Then it depicts how to build the rating factor into the calculation mechanism.

6.1 The user’s query description layers

As mentioned in Case 3, to retrieve a component via the ontology-based approach, the first job is to use the semantic information to process user queries. It assumes the availability of a domain ontology, containing the terms, relationships, and constraints that normally occur in a certain application domain. The user specifies the requirements for the components that one has in mind in a natural language. An initial query is generated and augmented with domain specific information derived from domain models and takes into account of the context within which the user is performing the search. Keywords from the user’s query are mapped to the domain ontology and appropriate terms derived are used in the search query. The ontology is also used to expand the query with synonyms to broaden the search, if needed. The user’s query description layers which are supported by beer domain ontology are built up as shown in figure 1. Parts of the shadow are the example of user’s query description for “Bitter”.

![Figure 1: User’s query description layer of beer ontology example](image)

With the support of application domain knowledge, multi-layered user’s query description will be established as the first part of architecture. In these multi-layers, an initial user’s query can be decomposed step by step into descriptions with more specified
details. And this decomposition process will be supported by the application domain ontologies. All those descriptions involved will be expressed in OWL.

6.2 Component description schema

A component description schema is needed to realize the mapping between user’s query and components. The component description schema may describe the component through the following three facets: Component Form Facet, Application Functionality Facet and Semantic Information Facet. The details of which are summarized in table 1.

<table>
<thead>
<tr>
<th>Component Form</th>
<th>Application Functionality</th>
<th>Semantic Information</th>
</tr>
</thead>
<tbody>
<tr>
<td>· BasicInfo</td>
<td>· FuncDescription</td>
<td>· ComMetam</td>
</tr>
<tr>
<td>· Presentation</td>
<td>· FuncClassify</td>
<td>· ComSimu</td>
</tr>
<tr>
<td>· Kind</td>
<td>· KeyWords</td>
<td>· ComReq</td>
</tr>
<tr>
<td>· Hierarchy</td>
<td>· AppDomain</td>
<td>· Interface</td>
</tr>
</tbody>
</table>

Table 1. Component description sub-facets

6.3 Component retrieval rating factor calculation mechanism

Having the above two parts in place, the component retrieval is thus regarded as the matching between the user’s query description multi-layers and component description schema. While both of these two parts were described in the OWL-DL, the relationship from the matching can be obtained automatically. In fact, each relationship is paired with a description from one layer in the user’s query description multi-layers, and a facet in the component description schema. It is proposed to assign a parameter to every layer of the user’s query description multi-layers. The lower the layer is, the more accurate the description will be in relation to the user’s query, and the greater the value of the parameter. Similarly, a parameter will also be assigned to every sub-facet in the component description schema. This parameter will be defined according to the impact of each facet on the precision of component retrieval. As such, every relationship will be described and weighted by two parameters. In turn, a rating factor for each retrieval between the user’s query and components will then be calculated to get a specific percentage to reflect the degree of match of the component with the user’s query.

7. Discussion and Conclusion

We have found the following major issues that need to be addressed to achieve wider acceptance of component-based retrieval:

1) Availability of an evolvable well-maintained repository of components for reuses, of which the structure, extent and content may change along with technological development. It is proposed to build a component repository manager to tackle this issue. Such a repository may range from in-house (local) component repository, to open commercial component repository, and to global web-base component market place.

2) Availability of precise, easy, and widely accepted approaches to the retrieval of specific components matching user requirements. At present, such approaches are limited for the following reasons:
   - Lack of sophisticated and precise search methods and techniques
   - Lack of comprehensive, rigorous and machine-recognizable component description
   - Lack of support from domain knowledge of application domains.

To address the above problems and needs, an evolvable ontology-based component repository is to be developed to facilitate the component retrieval process. In the meantime a rating factor calculating method is to be established to denote the precision of component retrieval in quantitative terms. However, we also observed that the viability of the proposed thought may be affected by the following concerns:
   - The completeness and suitability of existing domain models and ontologies. In reality, they often appear as incomplete and not suitable for an evolvable component repository. In particular, the existing ontology structure and metadata lack evolvability.
   - Evolvable component repository. An evolvable ontology framework and an associated repository management mechanism are needed to support such a repository.
   - Improving search precision via multiple ontologies and weighted relationships (relevance) between components and these
ontologies. A match rating factor will be calculated according to the weighted links. Such a rating factor indicates the confidence of the search results to the user; that is, how well the components meet the desired functionalities and quality features expressed by the user.

The further work will focus on the following actions:

- To investigate the domain model and existing ontologies; to improve and detail the ontologies with the support of the domain model;

- To investigate the appropriateness of OWL-DL, including whether it provides the expected description capacity, and whether the inference power satisfies multiple ontologies search, i.e., able to handle the similarities among ontologies and the relevance between components and ontologies.

- To identify a model to calculate the rating factor of the confidence of the component search results.

- To investigate the possibility of an evolvable ontology-based component repository, and propose an outline approach.

Reference


