A Grid-Oriented Platform for Software Component Repository Based on Domain Ontology

Yutao Ma, Keqing He, Wei Liu, and Jingbai Tian
State Key Laboratory of Software Engineering, Wuhan University, Wuhan 430072, China
yutaom@acm.org, hekeqing@public.wh.hb.cn

Abstract

Reusable software component repository (SCR) and retrieval management of components have long been in active research for many years. However, building interoperable SCRs is still a challenge for software engineering. Now, domain ontologies have been employed to manage software components, and the method has been deemed as an effective way to achieve the interoperability among different SCRs. Recently, ISO/IEC initiated a new standard (No.19763) based on OMG MOF (Meta-Object Facility) and ISO/IEC 11179 (MDR, Metadata Registry) to specify a framework for metamodel interoperability. According to the 3rd part of the standard, viz. Metamodel for ontology registration, we developed a grid-oriented platform for ontology-based SCR. In this paper, we present the approach that implements the new standard, and introduce the design of architecture and core component of the platform, which has been successfully deployed in several software enterprises to facilitate producing software systems with software components. Eventually, we put forward the future work to solve the emergent problems when using the platform.

1. Introduction

We are witnessing an enormous expansion in the use of software in business, industry, administration, research, and even in everyday life. There continues to be a great deal of pressure to design and develop software systems within a short period of time. This urgency has reinvigorated the researches on software reuse, particularly in the area of component-based software development (CBSD), which could decrease the total cost and time of software development.

Component-based software engineering (CBSE) has become recognized as an effective solution to software crisis. Now, many heterogeneous and distributed software component repositories (SCRs) have been deployed to manage remarkable amounts of component resources of different software companies [1]. Because of physical isolation to each other and independent decisions on the classification and specification of components, these repositories have insufficient abilities to interoperate with each other so as to share and reuse component resources of other repositories [2]. In general, there are two feasible approaches to share component resources of different enterprises:

- A centralized repository that integrates various SCRs (at the level of single enterprise) with public and unified interfaces. However, there exist some problems such as performance and efficiency [2].
- Achievement of the interoperability among different domain ontologies that classify components and implement semantic component retrieval [3].

For the purpose of enabling knowledge sharing and reuse, an ontology is a specification of a conceptualization [4]. Recently, ontologies and the relevant domain models have been used to describe explicitly-shared application domain specific knowledge, which could be exploited in expressing more pertinent interoperability for various component resources [5]. Based on well-defined domain ontologies, a SCR can effectively classify, register, and manage domain-specific software components, and perform the semantic query with natural-like language. Moreover, the semantic interoperability between/among different SCRs can be achieved by virtue of ontology reuse and mapping. However, several efforts to establish standards associated with ontology are underway. To promote ontology-based interoperation, ISO/IEC initiated a new standard (No.19763) to specify a framework for metamodel interoperability. Our work is based mainly on the 3rd part of the standard, which provides a generic
framework for registering necessary information related to ontologies or metamodels [6].

Nowadays, Grid technologies are getting more popular and have been applied to various computational fields. The Grid infrastructure can support the sharing and coordinated use of diverse resources in dynamic and distributed virtual organizations [7]. This enables us to construct a grid-oriented platform that implements the interoperability among distributed SCRs, so different enterprises are able to share and reuse component resources of other enterprises. The objective of our research, therefore, is to develop a grid-oriented platform for SCR that utilizes the domain knowledge embedded in ontologies and domain models by applying a new ISO/IEC standard. In this paper, we mainly present how to implement the classification, registration, and query of software components according to the new standard, and introduce the design of architecture and core component of the platform, which has been deployed in several software companies to facilitate building software systems with software components.

The remainder of this paper is organized as follows. The related research section provides academic background of the platform. In the following sections, we introduce the design of architecture and core component of the platform and the method that implements semantic interoperability, customerized query, and intelligent management. Eventually, the final section summarizes and concludes the paper.

2. Related work

Now, CBSE technologies have become popular so that component-based software systems emphasize appropriate reuse and composition of software components as a key concept. However, the reuse of component resources owned by other companies is often very challenging. Recently, using ontologies to classify and manage software components is deemed as a new attempt to build interoperable SCRs. Our prior work includes three aspects:

First, we investigated the method for acquisition of software component attribute ontology (SCAO) [8] (viz. a general and flexible classification mechanism for software components). With the help of domain experts, a domain-specific SCAO could be created with Protégé (an open-source tool) and will be available as an OWL (the Web Ontology Language) file. Then, we constructed the registry meta-model for domain-specific software components (see Fig. 4 in the paper), which is an ontology-based metamodel specifying a metadata registry similar to ISO/IEC 11179 (Metadata Registry, MDR).

Second, according to the 3rd and 4th parts of the new ISO/IEC standard, we proposed some feasible approaches to achieve semantic interoperability among complex information resources in terms of the reuse or mapping of domain ontologies [3]. To share and reuse the resource of software components, a special kind of complex information resources, we developed some prototype systems to realize these methods based mainly on ebXML [9] and UDDI [10] standards.

Third, service-oriented architecture (SOA), Web services, and grid technologies are dramatically changing the ways in which enterprises develop and deploy their Internet-facing applications. Many big companies such as IBM and HP are exploiting the benefits of Web services and grid computing to help them provide more fluid computerized business processes for customers. Hence, we tried to implement some important functions of a SCR with web services and grid technologies [10,11] according to the OGS (Open Grid Services Architecture) specification.

The previous work provides solid academic background and technical basis for the research presented here, which was supported by the Key Informationization Project of Manufacturing Industry of Wuhan City under Grant No. 20051001007. However, how to put the new ISO/IEC standard into effect and integrate our prototype tools into an industrial grid-oriented platform is still a hard task.

3. Architectural design

With the support of Municipal Bureau of Science and Technology, we now establish a SCR platform for manufacturing industry in collaboration with Kai Mu Company (KM), which is the biggest software company in Hubei Province. In Fig.1, SKLSE (State Key Lab of Software Engineering) and KM are two important nodes in grid environment such as P2P (peer to peer). For each grid node, there exists a DataStore used to store registration information of software components and local (or reference) SCAOs for application domains. Moreover, open source tools such as Protégé and self-developed management tools were integrated into the platform to provide services (such as ontology management, component classification and registration, and semantic query) that can be deployed in various operating systems or running environments for terminal users. The file format of data exchange is RDF/XML.

Reference ontologies (ROs) for application domains are published and maintained [12] in the light of
ISO/IEC 19763 standard and OMG Ontology Definition Meta-model (ODM); other companies search ROs provided by SKLSE via Internet, and manage local ontologies (LOs) that localize (or simply duplicate) the ROs using web services, so different LOS of a same RO could interoperate with each other at the level of concept [3,9]. Then, according to the registration template automatically generated in terms of a LO, software components can be registered and stored (submitted) into the DataStore, possibly indicating the broad resource sharing with other enterprises except for secret-keeping due to high business competition.

Figure 1. Architecture of the platform

4. Design of Core component

4.1. Structural design

The SCR based on domain ontology is the core component of the platform, which encapsulates many important functions (or services) introduced in Fig. 1. The intended users are developers in charge of designing new systems, who have some knowledge of the application domain.

In Fig. 2, it consists of web (or UI) based natural-like language front-end interfaces, modules of scenario refinement and execution, a domain model and domain-specific SCAO, and the repository of reusable software components. The scenario refinement module enhances the initial query by making use of the domain specific information contained in the domain model as well as the SCAO; the scenario execution module uses the repository when executing the scenario and retrieving the required artifacts (or registering new software components). Domain models are built through domain-specific analysis in order to gather and organize common concepts and relationships among them in the domain, while the ontology (LO) provides additional information that is used in conjunction with the domain model and SCR to retrieve and register the relevant information of software components.

4.2. Functional design

4.2.1. Classification and registration

Fig. 3 shows the UI of the application that manages software components without real data of commercial components. The system was built with Eclipse 3.1, and the domain model for manufacturing industry and the SCAO (shown in the top right part) were created with Protégé 3.1.

The SCAO was designed as a concept (viz. a node in the tree) of the domain model and could be localized for manufacturing industry according to a RO (OWL file) provided by SKLSE. Then, all contents of the SCAO were parsed by Jena (a free tool) to automatically generate the registration template (shown in the area with blue dashed line around it) comprising attribute categories, subcategories, and items, which could help users to easily fill out the registration information even if he knows little about the domain. Finally, registered software components (viz. individuals or instances of the SCAO) will be submitted to the repository and stored in the rational database such as MySQL by APIs of Jena too.
Three meta-classes (SCAO, Attribute Ontology, and Concept) that inherit Ontology, Ontology Component, and Ontology Atomic Construct from Metamodel for ontology registration (ISO/IEC 19763-3), respectively, were created to denote software component, attribute category, and attribute (or slot), respectively. To implement these meta-classes in object-oriented languages such as Java, we also imported Packageable Element, Class, and Attribute from UML 2.0 Metamodel to be the superclass of RegistryModel, RegistryClass, and RegistryAttribute, respectively. According to our meta-model, a (logic) SCR that inherits Package from UML 2.0 Metamodel is actually
the set of software components. Then, at the level of model, these classes will be realized by OntModel, OntClass, OntProperty, and APIs of Jena.

Table 1 lists a part of registration information about the attribute ontology (ManagementAttribute) of SCAO (see Fig. 5) according to ISO/IEC 19763-3.

Table 1. Part of registration information about SCAO

<table>
<thead>
<tr>
<th>Attribute or Reference</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>SCAO ManagementAttribute</td>
</tr>
<tr>
<td>URI</td>
<td>N/A N/A</td>
</tr>
<tr>
<td>Administration_Record</td>
<td># #</td>
</tr>
<tr>
<td>modelType</td>
<td>OWL OWL</td>
</tr>
<tr>
<td>consistsOf</td>
<td>Registry_Attr: Component Price: Concept</td>
</tr>
<tr>
<td></td>
<td>Management_Attr: Component Version: Concept</td>
</tr>
<tr>
<td></td>
<td>Functional_Attr: Component License: Concept</td>
</tr>
<tr>
<td></td>
<td>Technical_Attr: Component</td>
</tr>
</tbody>
</table>

Then, an example (see Table 2) is given to show the registration information of a real registered software component based on the registry meta-model.

Table 2. Part of registration information of a registered software component

<table>
<thead>
<tr>
<th>Name</th>
<th>Category</th>
<th>Attribute</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Registry_Attr</td>
<td>Lifecycle status</td>
<td>Approved</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Extrinsic Object</td>
<td>File (such as dll)</td>
<td>1,500 (RMB)</td>
</tr>
<tr>
<td>Management_Attr</td>
<td>Price</td>
<td></td>
<td>1.0.0.2</td>
</tr>
<tr>
<td>DpUI Functional_Attr</td>
<td>Description</td>
<td>It provides UI interfaces for users.</td>
<td></td>
</tr>
<tr>
<td>Technical_Attr</td>
<td>Operating System</td>
<td>Windows 2K/XP</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Programming</td>
<td>C++</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Language</td>
<td>Standard</td>
<td>COM</td>
</tr>
</tbody>
</table>

Figure 5. A part of SCAO described in OWL format

Table 1. Part of registration information about SCAO

<table>
<thead>
<tr>
<th>Attribute or Reference</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>SCAO ManagementAttribute</td>
</tr>
<tr>
<td>URI</td>
<td>N/A N/A</td>
</tr>
<tr>
<td>Administration_Record</td>
<td># #</td>
</tr>
<tr>
<td>modelType</td>
<td>OWL OWL</td>
</tr>
<tr>
<td>consistsOf</td>
<td>Registry_Attr: Component Price: Concept</td>
</tr>
<tr>
<td></td>
<td>Management_Attr: Component Version: Concept</td>
</tr>
<tr>
<td></td>
<td>Functional_Attr: Component License: Concept</td>
</tr>
<tr>
<td></td>
<td>Technical_Attr: Component</td>
</tr>
</tbody>
</table>

4.2.2. Simple inference

```sql
SELECT ?name ?id
WHERE {?x sc:license ?license.
FILTER regex (?license, "W3C").
?x sc:isFree "true".
FILTER (?isFree = true).
}
```

Figure 7. An example of component search

Because the SCAO provides a means of understanding what the components in both the domain model and SCR mean, it serves as a surrogate for the meaning of traditional search items (such as keyword or facet) so that a more complete response to a user’s query will result in greater query flexibility, preciseness and user satisfaction even if the user knows little of the domain and SCR. Furthermore, this facilitates building user interfaces when query conditions always change.

In Fig. 6, all search conditions created according to the SCAO are converted to SPARQL query language [13] by codes and then executed by Jena. For example, if we want to seek components that are free and hold a license of W3C (the World Wide Web Consortium), the query will be the text shown in Fig. 7. Eventually, the result including name and id (the unique key) of a software component will return back to users.
Besides ontology-based (semantic) component search, we also provide some simple inference functions such as “requiring” (the explanation is positioned in the rectangular area of Fig. 6) for developers to seek components that they need. Additional information on what components are related when using a selected component might be of interest to the user, who wants to reuse it and composite with other components. Because of the inference function of ontologies, ontology-based SCRs enable us to easily accomplish the “requiring”. Fig. 8 shows the structure of our ontology model based on the rational database and OWL file.

There are two kinds of statements in our ontology model. On the one hand, a software component has many attributes, namely, there is an ObjectProperty (has_attributes) between the concepts of software component and SC attributes; on the other hand, an attribute has one or many values, that is to say, there is a DatatypeProperty, ObjectProperty, or other kinds of Property between the concepts of SC attributes and named classes. To implement the inference, the property between the attribute (“Requiring”) and its value (one or many software components) was designed as a TransitiveProperty. So, the inference of dependency relationships (a kind of association between components) among components can be realized according to a transitive rule (shown in Fig. 9) defined in Jena. Finally, the inference results, the
complete transitive-closure of a selected component, will return back to users.

Figure 9. Transitive rule defined in Jena

4.2.3. Data sharing and management

Along with more and more popular research of semantic web, many semantic repositories or systems have been developed. In general, all of them can be divided into 3 categories according to their persistent strategies: relational database-based, file system-based, and memory-based. In our platform, all ontologies (ROs and LOs) represented as the OWL format are stored in file system and managed by Protégé; the information of registered software components (viz. individuals of the SCAO) is stored in rational databases such as MySQL using APIs of Jena. All of these data can be easily accessed and shared with web services or applications of the grid environment.

On the other hand, to reduce the time of distributed inference and network delay, an ontology model is populated from existing data (from file and database) at startup, and then will be loaded into memory in the local computer when performing component search and inference. However, a disadvantage of using an in-memory model in this situation is that the model would have to be repopulated from scratch each time the application was launched. Additionally, if the ontology model is too large for an in-memory model to be practical, this will make the performance of the platform deprived dramatically.

5. Problems to be solved

Sugumaran et al. [5] summarized the limitations of traditional component retrieval schemes such as keyword search and faceted classification. Because of considering the user’s intensions and relevant domain information, the method based on domain ontology provides more precise results, easier usability, and better semantic inference for users, resulting in greater query flexibility, augmentation and user satisfaction.

Intelligent component search (and inference) plays an important role in using a SCR. There is always a trade-off between the performance and the cost of an implementation, i.e., precision rates versus query processing. So, we encountered two chief problems proposed by users, namely, the complementary fuzzy method for component search (so users would have more choices) and the performance (or efficiency) of component inference based on Jena (viz. the insufficient ability to process ultra-large in-memory model and the rigorous requirement on computer hardware), which are to be solved when using the SCR.

6. Conclusions

With the support of Municipal Bureau of Science and Technology of Wuhan City, the platform has been successfully deployed in KM and other 160 companies related to manufacturing informationization since 2006. Until now, they have reduced development costs about 4 million RMB per year, and made an accumulative profit of more than 15 million RMB when using it to promote software component industrialization. Hence, the work as an application case of ISO/IEC 19763 standard was highly praised by ISO/IEC JTC 1 SC 32 officials.

To tackle the emerging problems, the future work includes:

- Similarity-based fuzzy search;
- Tools to support ontology evolution and management;
- Extending Jena to improve the performance of component inference (such as adding new rules and improving existing algorithms).

We expect that the platform will attract more enterprises to join the project and contribute to the development of software industry in Hubei Province.

Acknowledgement

This work is supported by National Basic Research Program (973) of China under grant No. 2006CB708302, National High Technology Research and Development Program (863) of China under grant No.2006AA04Z156, National Natural Science Foundation of China under grant No. 90604005, and the Open Research Foundation of State Key Laboratory of Software Engineering of China under grant No. SKLSE 05-03.

References


