A Method for Context-aware Web Services Selection

Li CHEN, Zi-lin SONG, Ying ZHANG, Zhuang MIAO

Institute of Command Automation, PLA University of Science and Technology, Nanjing 210007, China, ivan.chen@163.com
doi: 10.4156/ijact.vol3.issue7.34

Abstract

Service-oriented computing (SOC) is emerging as a new, promising computing paradigm that centers on the notion of service as the fundamental element for developing software applications. Traditional service selection deals with applications which only consider functional matchmaking of services and do not put them into a context when selecting services. When recognizing the importance of contextual information, in this paper we mainly focus on the precise and effective context-aware service selection. At first we proposed a General Context Model (GCM) for service selection. After the construction of GCM, we gave a framework for context-aware service selection and discussed the method of service selection based on context and function. The experimental result shows that our method performs better than the methods without consideration of context.

Keywords: Service-oriented Computing, Web Service, Service Selection, Context-aware

1. Introduction

Service-oriented computing (SOC) is an emerging cross-disciplinary paradigm for distributed computing that is changing the way software applications are designed, architected, delivered, and consumed. Services are self-describing, open components that support rapid, low-cost composition of distributed applications [1]. Since services may be offered by different enterprises and communicate over the Internet, they provide a distributed computing infrastructure for both intra-and cross-enterprise application integration and collaboration. Web services are the current (but not only) most promising technology based on the idea of service-oriented computing.

The majority of existing approaches to service selection (also referred to as service discovery or service matchmaking) [2, 3] focus on finding the most relevant services to individual users according to function of services and do not take any contextual information into consideration, such as time, place and the interest of user. In other words, traditional service selection deals with applications which only consider functional matchmaking of services and do not put them into a context when selecting services.

The importance of contextual information has been recognized by researchers and practitioners in many disciplines, including e-commerce personalization, information retrieval, ubiquitous and mobile computing, data mining, marketing, management, and service selection which we will focus in this paper.

In the literature, context was initially defined as the location of the user, the identity of people near the user, the objects around, and the changes in these elements. Other factors have been added to this definition subsequently. For instance, Brown et al [4] included the date, the season, and the temperature. Dey et al [5] included the user's emotional status and broaden the definition to any information which can characterize and is relevant to the interaction between a user and an application. Some associated the context with the user, while others emphasized how context relates to the application. More recently, a number of other techniques for context-aware systems have been discussed in research literature, including hybrid techniques for mobile applications and graphical models for visual recommendation.

Dourish [6] introduced taxonomy of contexts, according to which contexts can be classified into the representational and the interactional views. In the representational view, context is defined with a predefined set of observable attributes, the structure (or schema, using database terminology) of which does not change significantly over time. In other words, the representational view assumes that the contextual attributes are identifiable and known a priori and, hence, can be captured and used within
the context-aware applications. In contrast, the interactional view assumes that the user behavior is
induced by an underlying context, but that the context itself is not necessarily observable. Furthermore,
Dourish assumes that different types of actions may give rise to and call for different types of relevant
contexts, thus assuming a bidirectional relationship between activities and underlying contexts:
contexts influence activities and also different activities giving rise to different contexts.

In this paper, on the foundation of formers we proposed a General Context Model (GCM) which
consists of Top Context Ontology (TCO) and several specific ontologies. The TCO is a high-level
ontology which captures general features of basic contextual entities. The specific ontologies represent
specific aspects of contextual information. After the construction of GCM, we take context-aware
service selection into consideration. The framework for context-aware service selection is proposed
firstly. Then we discussed the method of contextual matchmaking and functional matchmaking. After
the discussion, we took the experiment to evaluate our method. The experimental result shows that our
method could increase the precision of service discovery. In generally, our method can achieve better
performance than the methods without consideration of context.

The rest of this paper is organized as follows. Section 2 introduces related works about context-
aware systems and context-aware technology for Web services. In section 3, we construct the general
context model. The framework for context-aware service selection and method of service selection
based on context and function is discussed in section 4. Section 5 is the experiment to evaluate our
method. The conclusions and some opportunities for future work are presented in last section.

2. Related Works

There exist a few context models and context-aware systems in literature. Strang and Limhoff-
Popien [7] survey existing context models and classify them into different types based on the data
structures. They classify the context models into 6 types: key-value models, markup scheme models,
graphical models, object-oriented models, logic-based models, and ontology-based models. They also
evaluate those context models using six requirements and showed that ontology is the most expressive
model and fulfill most of the requirements. Chen and Kotz [8] investigate the research on context-
aware mobile computing. They discuss the types of context used, the ways of using context, the system
level support on collecting context, and approaches to adapt to the changing context. Baldauf et al [9]
present a layered conceptual design framework to describe the common architecture principles of
context-aware systems. Based on their proposed design framework, they compare the context sensing,
context models, context processing, resource discovery, historical context data, security and privacy of
various existing context-aware system. The context ontology proposed by Wang [10] is representative.
The context is divided into Person, Activity, Computational Entity, and Location. Furthermore, the
domain ontology for Home is also proposed. However, this context ontology seems non-perfect and
lack of catholicity. The concepts in ontology are coarse, such as the location is just divided into
IndoorSpace and OutdoorSpace. Another important concept of context “Time” is just the supplement
of Activity.

Applying context-aware techniques to discover and select services has gained lots of attentions.
Balke and Wagner [11] propose an algorithm to select a Web service based on user’s preferences. The
algorithm starts with a general query. If there are too many results, it expands the service query using
user’s preferences. Mostefaouï et al [12] propose a formal definition of service contexts to model a
service’s contextual information. They use Composite Capabilities/Preferences Profiles (CC/PP) as
interoperable context representation to enable communication and negotiation of device capabilities
and user preferences in terms of a service invocation. Yang et al [13 14] design an event-driven rule
based system to recommend services according to people’s context changes. They provide an ontology-
based context model to represent context and utilize the context to assist service discovery. Yang
develops two types of context ontology for describing the circumstances of requesters and services
respectively: requester ontology and service ontology. However, the concepts in these two types of
context ontology are different. In other words, the context ontologies are heterogeneous which will
bring great challenge in context reasoning. Abbar et al [15] provide a service-oriented approach for
recommender systems. The module of recommender systems are encapsulated as services. A method to
recommender systems using the log files of a user and the current context of the user is discussed in
detail. The context information mainly comes from the log files of a user. The applicable scope is limited.

3. Modeling Context in Web Services

As we have mentioned above, different context has different properties and need different model to express the context type. At present, almost all systems have their own context model and make it is difficult to exchange information through different systems. It’s necessary to construct a general model of context to capture contextual information and adapt dynamic environment.

In realistic service selection, services are classified into several domains according to their function. In each domain, the context of services contains common context types and share common information. We could construct a general model of context of services. In different domains, there will be a few different features in detail. We will use this general context model for service selection.

The context models can mainly be divided into 6 types: key-value models, markup scheme models, graphical models, object oriented models, logic based models, and ontology based models. In this paper, we construct the General Context Model (GCM) based on ontology which is the most expressive model. The GCM is constructed by Top Context Ontology (TCO) and several specific ontologies such as Personal Context Ontology (PECO), Spatial Context Ontology (SPCO), Temporal Context Ontology (TECO), Environment Context Ontology (EUCO), Social Context Ontology (SOCO), and Quality Context Ontology (QUCO). Although some researchers treat the Quality of Services (QoS) separately, we consider the QoS as a kind of context which can influence the service selection. The application ontology is the ontology engineered for a specific use or application focus and whose scope is specified through testable use cases. The relationship between these ontologies is shown in figure 1 where the solid lines represent the father-child relation and the dotted lines represent the relation of citation. Figure 2 gives a segment of application ontology for rescue of Wenchuan earthquake in 2008.
The TCO is the start point of our works. The TCO is a high-level ontology which captures general features of basic contextual entities. The concept ENTITY is the most important and in the top level of TCO; much similar to the “owl: Thing” in OWL. ENTITY refers to all objects in the interaction between user and services, including user and services themselves. All other concepts in GCM are inherited from ENTITY.

The PECO mainly contains the user’s physical and mental state, user’s behavior and target, etc. In our application ontology, the PECO of injured people includes the degree of injury; the mental state of injured people and so on. According to PECO of injured people, we will determine the medical resource we need. The SPCO mainly contains location, direction, speed, etc. The landform of Wenchuan is mountain, in rescue we need to take mountainous landform into account since the mountainous landform will influence the rescue greatly. The TECO is divided into two types: instantaneous context and continuous context. The instantaneous context uses an absolute dating system. This involves time stamping each event with an absolute real-time, say taken off the system clock on the machine, or some other coarser-grained system such as we use for dating in everyday life. For instance, in our application ontology, the dating scheme of earthquake time of Wenchuan consists of the year, month in the year, day in the month. On the contrary, the continuous context is a duration-based representation, for example we have participated in the rescue from 6 AM to 10 PM is a continuous context. The ENCO mainly contains environmental objects around user, such as weather, traffic status, noise level, etc. As we know, the weather condition is an important factor which will also influence the rescue. The SOCO describes the related people around user, such as relatives, friends and so on. For the injured people, the SOCO consists of police, PLA and doctors. The QUCO contains all aspects of services quality, such as execution cost, execution time, availability, successful execution rate, reputation, etc. After the rescue, we will evaluate the quality of rescue through QUCO.

4. The Context-aware Service Selection

4.1. The Framework for Context-aware Service Selection

With the increasing number of available services in the Internet, how to select desirable services fast and precisely becomes more and more important. Traditional methods of service discovery and selection couldn’t sense user’s requirement and context effectively. Therefore traditional methods can not discovery and select most desirable services for user intelligently and can not satisfy personalize requirement of users. Furthermore considering existing methods of context-aware service discovery, the contextual matchmaking is processed after the functional matchmaking. However, sometimes the validity of services depends on the context of user. For example, when the rescuers arrived to Wenchuan and need a weather report, obviously only the weather report services about Wenchuan is suitable for user. We could filter the irrelevant services according to user’s context such as location and time. In this paper, the contextual matchmaking is processed before the functional matchmaking. The advantage of this method is that the contextual matchmaking filters irrelevant services and constrains the number of services for functional matchmaking. It will improve the efficiency of service matchmaking since the functional matchmaking is time consuming.

In this paper, we propose a framework for context-aware service selection as shown in figure 3. The most important modules in this framework are context process module and service selection module. The context process module mainly obtains the raw contextual information, process and transform according to GCM, finally get the structured context for later usage. The service selection module focuses on matchmaking of service advertisement and service request, and mainly has two steps (contextual matchmaking and functional matchmaking).
 Generally, Web service selection is the process to find desirable services which can satisfy functional and non-functional requirements of user. Our method proposed in this paper also considers the context of user in service selection. Note that, the non-functional requirements such as quality of services, constrain of environment, etc have already been included in our GCM. Therefore in service selection module, we mainly consider contextual matchmaking and functional matchmaking.

The contextual matchmaking takes an offered context and a required context and it returns the degree of semantic compatibility between them. Each context is characterized by its properties in GCM. Let us note that offered context are individuals, i.e., specific instances of a class, whereas requested context is classes defined by restrictions, with restrictions on service properties determined by user-specified value preferences. The algorithm works on one context at a time. For each required context, it is able to recognize three possible subsumption relations with the offered context, namely: the offered context may be an instance of the requested context class (Exact), or an instance of a class that subsumes it (Subsumes) or an instance of a class that is subsumed by it (Plug-in). These semantic relations are determined by performing subsumption reasoning over the property values and class types of offered and required context. In case of exact match for all service contexts, the offered service is compatible with the user’s request. In case the matching is not exact, compatibility is evaluated depending on value preferences: if there exists a preference allowing relaxing the constraint over that property, a plug-in or subsumes case might be considered compatible.

The functional matchmaking between service advertisement and service request is usually based on bipartite graph matching which translates the service matchmaking to the problem of extended optimal matching for bipartite graph [16]. In this paper since we don’t focus on the functional matchmaking, we also use this method to calculate functional matchmaking for simply. After the contextual matchmaking and functional matchmaking, the services on the top of returned list are considered as the most desirable services for user.

5. Experiments and Results Analyze

In order to evaluate the validity and efficiency of our method proposed in this paper, we give the experiments in this section. Hardware condition: Intel Pentium(R) Dual-Core 2.8GH, 2G Memory. Software condition: Eclipse as development environment, context data is from three context-aware research projects that are SOCAM [17], MIDAS [18] and CoBrA [19]. OWL-S TC4 [20] is used as benchmark of our experiment and transformed to context-aware service description.

In this paper, we cover the two most useful evaluation measures: recall, precision. Consider an example service request (of a test reference collection) and its set $R$ of relevant services. Let $|R|$ be the number of service in this set. In our experiment, the relevant services set of test service request has been given in the OWLS-TC4. Assumed that a given service matchmaking approach (which is being evaluated) processes the service request $SR$ and generates a service answer set $A$. Let $|A|$ be the
number of service in this set. Further, let \( R_o \) be the intersection of the sets \( R \) and \( A \); \( |R_o| \) be the number of service in \( R_o \). The recall and precision measures are defined as follows.

Recall is the fraction of the relevant service (the set \( R \)) which has been retrieved i.e.,

\[
Recall = \frac{|R_o|}{|R|}
\]  

(1)

Precision is the fraction of the retrieved service (the set \( A \)) which is relevant i.e.,

\[
Precision = \frac{|R_o|}{|A|}
\]  

(2)

In the experiment, we randomly select 7 service requests to calculate recall and precision separately. We mainly compare our proposed method to the method which is based on functional matchmaking without context. At first we only use functional matchmaking in service selection module of our framework. And then we use both contextual matchmaking and functional matchmaking. The contextual matchmaking used in our experiment is Plug-in and Exact. If the contextual matchmaking between service and service request is Subsumes, we think that the service is irrelevant. The threshold of functional matchmaking is 0.9. The results of precision and recall are shown in figure 4.

![Figure 4a. Precision of service request](image1)

![Figure 4b. Recall of service request](image2)

Through the experiment, we could find that when we take context into consideration the precision of service discovery increases obviously whereas the recall decreases inconspicuously. In context-aware and pervasive computing environment, the recall isn’t the crucial evaluation because the capacity of user’s terminal is restricted and maybe can’t accept and process too many results. User will merely select services which are in the top of returned service list. The precision is more important. It is worthwhile to exchange a little recall for more efficient and precise service selection.
In order to evaluate the efficiency of service selection, we measure average processing time with different quantity of services. The result of average processing time is shown in figure 5. When the quantity of services increases from 100 to 500, the average processing time of service selection increases with different degree. Our method performs better than the method which doesn’t take context into consideration, because before the functional matchmaking, the irrelevant services are filtered by context at first. The different strength of contextual matchmaking can influence the processing time either. In the experiment, with the strength of contextual matchmaking increases (from Plug-in to Exact), the processing time decreases. At the same time, the recall of services also decreases. As we have discussed above, a little loss in recall is acceptable since we can provide fast and precise service selection. The efficiency of service discovery will improve if the user doesn’t require services as many as possible.

6. Conclusions and Future works

Service-oriented computing (SOC) is emerging as a new, promising computing paradigm that centers on the notion of service as the fundamental element for developing software applications. Traditional service selection deals with applications which only consider functional matchmaking of services and do not put them into a context when selecting services. In this paper, we first introduced the definition of context and proposed a general context model for service selection. Then we gave a framework for context-aware service selection and discussed the method of service selection based on context and function. In the end, we gave experiments to evaluate our method and compared to the method without any context. The experimental result shows that our method has better performance both in precision and efficiency.

Context-aware service selection is the first step of our research. When there is no single service can satisfy the user’s demand, we need to compose several services to fulfill the requirement. How to accomplish context-aware service composition in pervasive environment is an attractive problem encouraging us for deeper research.

7. Reference

A Method for Context-aware Web Services Selection
Li CHEN, Zi-lin SONG, Ying ZHANG, Zhuang MIAO
International Journal of Advancements in Computing Technology Volume 3, Number 7, August 2011


- 298 -