A Survey of Context Aware Web Service Discovery: From User’s Perspective

Wenge Rong and Kecheng Liu
Informatics Research Centre
University of Reading
United Kingdom, RG6 6WB
{w.rong, k.liu}@reading.ac.uk

Abstract—Web service is one of the fundamental technologies in implementing Service Oriented Architecture (SOA) based applications. One of the essential challenges in web service based applications is how to find a set of suitable web service candidates with regard to a user’s requirement. Currently most web service discovery systems have constrains on the content and format of submitted web service request. Therefore, during the information transformation from a user’s real web service need to formalized request, some useful information is lost implicitly or explicitly. To understand the user’s intention as much as possible, a lot of approaches have been proposed among them context aware methods have shown promising result. In this paper, we present an overview of the field of context aware web service discovery and try to classify current generation of those approaches into different categories. This paper also describes limitations of current context awareness in web service discovery and discusses possible applications that can enhance the overall discovery performance.

Keywords—web service; discovery; survey; context awareness; applications

I. INTRODUCTION

Since the concept of Service Oriented Architecture (SOA) was proposed in 1990s it has been lauded as a strategic mechanism for achieving dynamic business process. One of the key techniques in implementing SOA based applications is the web service, a software system designed to support interoperable machine-to-machine interaction over a network [1]. Through applying web service, the applications do not need have knowledge of expert over the technology or infrastructure details beneath the web service interfaces, which make it possible to seamlessly piece together applications spanning different technique boundaries. With the increasing number of web services available in public repositories, i.e. UDDI [2], web portals (e.g. xmethods ¹, webservicex², webservicelist³ and etc.) and Internet [3], how to locate a set of proper web services with regard to a user’s requirement is becoming one of the most significant tasks, which is normally called web service discovery.

Typically, a web service discovery process includes three major steps. Firstly, the providers advertise web services in public repositories by publicizing the programming interfaces via description files, e.g. files written in WSDL [4]; Secondly, a user submits a web service request and then a web service discovery model will be employed to find a set of web service candidates, which are normally sorted and displayed to the user in a certain order. Finally, the user invokes one of the retrieved web services. The second step, or the web service matchmaking process, is probably the most important component in the web service discovery protocol. In this step there are two fundamental concepts which deserve to be highlighted firstly, i.e. web service requirement and web service request.

The web service requirement, or requirement for short in this paper, is the user’s intentional web service need, which is usually not well organised and probably presented as several sentences in the user’s mind. In contrast, the web service request is abstracted from requirement and must be well formalised in the web service discovery system acceptable format. As such it is argued that the request is the objective representation of a user’s subjective requirement. Since it is the request that will be submitted to the web service discovery system, to guarantee the accuracy of the matchmaking process, the request must reflect as precisely as possible the user’s requirement.

However, to constitute a proper request is not easy for a user all the time as a web service discovery system usually has some specified constrains on the content and format of the submitted request. Hence there are always inevitable information distortions when a user tries to build the request from the real needs. Firstly, an acceptable request always contains smaller amount of information compared with requirement. Furthermore, the requirement itself might hardly be described properly, which will make the generated request varying a great deal from time to time. Secondly, to generate better requests a user might have to be familiar with certain knowledge. For example, when a user is using semantic based web service discovery system, the user will have to be familiar with ontology or other related techniques otherwise the user will not be able to generate even a valid request.

Due to the inconsistency between request and requirement, either the retrieved set of web services does not contain the expected ones, or it is too huge for the user to select suitable ones. To avoid the ambiguity of the request and then enhance the matchmaking performance, it is believed that the ability to obtain more information beneath the concise request to recover the original requirement is necessarily important and will help the matchmaking process become more accurate, which is one of the most challenging tasks in real web service discovery applications.

¹ http://www.xmethods.net
² http://www.webservicex.net
³ http://www.webservicelist.com

References


A large number of sophisticated approaches have been proposed to deal with such extra information. Among them the context aware web service discovery methods have been attained much importance by researchers from academic and industry sectors. In fact, context awareness has been widely studied and employed in different applications. However, the concept of context is too broad as context could be anything related to the user and applications. It is unfortunately too complicated to be modelled easily and then employed in the applications. For example, the context could be some factors affecting a user to extract the request, e.g. the user’s intention, suggestions, knowledge. It also could be the environment where the user asks a service request, e.g. location, time and etc. Therefore, researchers have introduced diverse definitions of context for different applications. Consequently different kinds of context awareness have been developed in the web service discovery domain.

In this paper, we first give brief definition on context and then discuss the importance of context in web service discovery domain in section 2. Section 3 gives a classification of web service matchmaking model which is the core part in dealing with the contextual information. Afterwards, we present a comprehensive overview of the state-of-the-art on context aware web service discovery approaches in section 4. In section 5 we discuss some possible further application of contextual information to extend the web service discovery capabilities. Section 6 gives conclusion and future works.

II. CONTEXT DEFINITION AND IMPORTANCE IN WEB SERVICE DISCOVERY

A. What is Context

Context awareness has been becoming more and more important in SOA based systems. Particularly in the web service discovery application, the ability of web service matchmaking process to integrate context is essential for adaptability and personalization. However, like other context aware computing applications, to define the word “context” in web service discovery is the first challenging task.

![Contextual information example](Figure 1. Contextual information example)

To understand what the context is and how it works in the web service discovery, a brief example is given in Fig. 1 presenting a travel service request. When a customer asks for a travel service, there are several factors which might affect the user’s expectation of discovered web services. The root factors are described as $F = \{F_1, F_2\}$, where $F_1 = \text{“Business”}$ and $F_2 = \text{“Holiday”}$, which label the request as for a business trip or a personal entertainment journey. At the next level, the “Business” factor is further divided into $F_1 = \{F_{1,1}, F_{1,2}\}$ and the “Holiday” factor is further divided into $F_2 = \{F_{2,1}, F_{2,2}\}$, where $F_{1,1} = \text{“Time”}$, $F_{1,2} = \text{“Airline”}$, $F_{2,1} = \text{“Price”}$ and $F_{2,2} = \text{“Time”}$, which respectively indicate the user’s specified favourites on the expected service. Apparently all of these factors could change on different situation, but the finally generated requests could be similar, for example, \{“Ticket”, “From Paris”, “To London”, “Depart on 1\textsuperscript{st} of August”\}, if expressed in a vector of keywords. From this scenario it is learned that to better satisfy a user’s service requests in different situations, the web service discovery system need have to understand all of those factors, which could be called context in this example, beneath the objective request as much as possible when perform the matchmaking process if appropriate.

Different definitions of context can be found in the literature and these definitions have different impacts on context aware computing applications. Generally, the context definition can be divided into two types, i.e. formal definition and enumeration definition. Some scholars try to give formal definition on context, e.g. Dey formally gives following definition: Context is any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves [5]. From this definition it can be seen that the scope of the context is broad indeed. As such though this definition is clear, it is too general and then applicable when employed into real applications.

Some other scholars attempt to explain context from different angle by enumerating context attributes in the applications. [6] defines context from three aspects, namely where you are, who you are with, and what resources are nearby. They argue that the context should include location, lighting, noise level, network connectivity, communication costs and etc. [7] defines context as location, identities of the people around the user, time, date and etc. Lots of other similar enumeration methods or classifications can be found in the literature. This kind of approach is easier to be understood as the context is described as a set of attributes, which can be presented by different models. Though this definition is easy to be implemented compared to formal definition, the difficulty might arise in enumerating proper context attributes for each application as it is tedious and nearly impossible to elaborate all attributes and then the comprehensiveness cannot be guaranteed. Furthermore, the extracted attributes might differ from one application to another one, which will make the enumeration less adaptive among different applications. For example, in the scenario presented in Fig 1, one might hold a flying card with certain airline. In such case the membership would also yield some influence on the selection of web services. Therefore the system has to handle this extra factor. Similarly, if more aspects are considered and then more attributes are employed, the challenge of re-designing the system is becoming more complex.

From the above discussion it is found that due to the complexity of the context, it is infeasible to properly model context for all applications in one way. Therefore, it is believed that the context should be domain oriented or problem oriented. In other words, different applications should consider their own definition and corresponding employment of context concept. In this paper, context in web service discov-
ery is formally defined as any information that explicitly and implicitly affects the user’s web service request generation.

**B. The Importance of Context**

A high quality request is one of the fundamental bases for an effective web service matchmaking process. As discussed before, a request is usually well formalised and presented in a certain format acceptable by the web service matchmaking model. But the user’s original requirement is definitely not organised in the matchmaking model readable format. Therefore, when a user is going to use any web service discovery system, the first task is to transform the requirement to an acceptable request. In this paper, we will consider the request generation as a process of information transformation from the subjective meaning of the requirement to the well formalised and matchmaking model readable objective request.

Peirce [8] presents a triadic model for the general information transformation process, as shown in Fig. 2. He claims that each information transformation process can be defined as a three-tuple <Sign, Object, Interpretent>, where Sign is the signification that does not point to anything but itself, whilst the Object is linked to something with real objective meaning. These two parts are interrelated and mediated by Interpretent. From this picture, it is argued that the information distortion during the meaning making process is always like to happen. The reason is mostly due to the fact that there is no direct link between Sign and Object, which means that they have to be interpreted by a third party Interpretent, as presented by dotted line in Fig. 2.

According to Peirce’s triadic, the Sign is non-meaningful. It has to be interpreted by the Interpretent otherwise people cannot understand the real meaning. Obviously the Interpretent is the vital party and different way of interpreting will definitely cause different understanding. From this point of view, the process is naturally prone to be misunderstood because some important information would be lost implicitly or explicitly by the Interpretent. What is worse is that this triangle could be expanded recursively and infinitely [9]. Each Sign in one triangle can be further interpreted by another triangle as Object. Therefore, any misunderstanding from one triangle would be transmitted and propagated within the whole information transformation process.

From the above discussion, it is learned why misunderstanding during information transformation is prevalent and possibly unavoidable. Currently, most of the web service discovery methods mainly study how to match web service against the request, which is based on one general hypothesis, namely that the submitted request is the user’s real intentional requirement. Apparently this hypothesis is only partly true in the real world. From the empirical study it is commonly found that similar requests do not necessarily reflect similar requirement. For example, different users may submit similar keywords when using UDDI retrieving web services though the requirements are not same as the request generation process is highly depends on the users’ personal knowledge background and/or the methods and tools they are using. Since all of the involved information will be re-organised, analyzed and optimized, this process is then uncertain and the result is probably diverse from user to user.

Therefore, in the web service discovery domain, we can formally define the Sign as the generated web service request, the Object as the real intentional web service requirement. They are not linked directly but mapped via the Requirement Analysis. As such the web service request generation process can be described in the same triadic way, as shown in Fig. 3. It is an overall picture and certainly it could be also further divided into a series of triangles as the web request generation normally can be further divided into several steps such as requirement cleaning, analysis, optimization and etc. In this paper, context is studied from the point of view of request generation. It is clear that the requirement analysis process will be influenced by different factors, which are defined as context in this paper, information that explicitly and implicitly affects the user’s web service request generation.

**III. WEB SERVICE MATCHMAKING MODELS**

In the second step of web service discovery protocol, besides web service request, another concept also needs to be introduced in advance before we can go to discuss the context aware web service discovery in detail, which is the web service matchmaking model. The web service matchmaking model is in charge of matching request against all web service registered in the repository. The web service request and matchmaking model are highly interrelated and dependent on each other. On the one hand, the quality of a user’s request is essential to the effectiveness and performance of a matchmaking model. On the other hand, implementation of a certain matchmaking model will also have significant effect on the format of the request.

A lot of classification methods of web service matchmaking models can be found in the literature. For example, [10] defines three types of web service matchmaking models, i.e. Text document matching, Schema matching and Software component matching. [11] categorises web service matchmaking into Text document matching, Semi-structured document matching, Software component matching and
Schema matching. In this paper, we are going to classify web service matchmaking models from the Sign’s perspective, or the request format’s perspective.

<table>
<thead>
<tr>
<th>Human information functions</th>
</tr>
</thead>
<tbody>
<tr>
<td>SOCIAL WORLD: beliefs, expectations, functions</td>
</tr>
<tr>
<td>Commitments, contracts, law, culture,…</td>
</tr>
<tr>
<td>PRAGMATICS: intentions, communications,</td>
</tr>
<tr>
<td>Conversation, negotiations,…</td>
</tr>
<tr>
<td>SEMANTICS: meaning, propositions,</td>
</tr>
<tr>
<td>Validity, truth, signification, denotations,…</td>
</tr>
<tr>
<td>The IT platform</td>
</tr>
<tr>
<td>SYNTACTICS: formal structure, language, logic,</td>
</tr>
<tr>
<td>Data, records, deduction, software, files,…</td>
</tr>
<tr>
<td>EMPIRICS: pattern, variety, noise, entropy,</td>
</tr>
<tr>
<td>Channel capacity, redundancy, efficiency, codes,…</td>
</tr>
<tr>
<td>PHYSICAL WORLD: signals, traces, physical distinctions,</td>
</tr>
<tr>
<td>Hardware, component density, speed, economics,…</td>
</tr>
</tbody>
</table>

![Figure 4. Semiotics framework](image)

Morris carries on Peirce’s work and further investigates three complementary levels on Sign, i.e., syntactics level, semantics level and pragmatics level [12]. Syntactics studies the relation of Sign with Sign system. Semantics studies the relation between Sign and the object it presents. Pragmatic studies the relationship between the Sign and the Interpreter. Stamper develops Morris’s work and proposes a six layer semiotics framework [13], as shown in Fig. 4. This framework is broadly divided into two parts, i.e. IT Platform and Human Information Functions and covers Sign from a multi-discipline perspective. Inspired by their work, in this paper, the web service matchmaking models are divided into the following four categories: 1) keywords based matchmaking; 2) Syntactics based matchmaking; 3) Semantics based matchmaking and 4) Pragmatics based matchmaking.

Our approach of classifying matchmaking models does not aim to put the classification into hierarchical order and claim that the upper layer methods are definitely better than those in lower layer. It is only trying to provide a different insight of difference between approaches in different layers.

### A. Keywords Based Matchmaking

The intuitive thought is to employ the keywords based similarity mechanism finding suitable web services. Actually, current UDDI based web service searching service provides for web service discovery in this way. Most of web portals and Internet based web service search engines also employ this model. A lot of keywords based similarity methods have been proposed in IR applications and most solutions are based on Vector Space Model (VSM) [14]. The similarity between keywords and documents is based on term analysis. The most popular term analysis method is TF-IDF (Term Frequency-Inverse Document Frequency) [15] in which the weight of a term is defined as:

\[
W_{Di} = TF_{DI} \times IDF_i
\]

where \(W_{Di}\) represents the weight of word \(T_i\) in the Document \(D\), \(TF_{DI}\) is the occurrence count of \(T_i\) in the Document \(D\), \(IDF_i\) is the distribution of word \(T_i\) in the whole document set and can be defined as:

\[
IDF_i = \log \frac{N}{n_i}
\]

where \(N\) is the whole number of documents in the set and \(n_i\) is the number of documents which contains the word \(T_i\). The higher the \(IDF_i\) is, the more indicative the word \(T_i\) is. Finally, the similarity of a document \(D\) and a search request \(Q\) can be calculated by the widely used cosine distance:

\[
sim(D, Q) = \frac{\sum_{i=1}^{n} w_{D,i} \cdot w_{Q,i}}{\sqrt{\sum_{i=1}^{n} w_{D,i}^2} \cdot \sqrt{\sum_{i=1}^{n} w_{Q,i}^2}}
\]

From the user’ perspective, this kind of matchmaking is convenient since the user only need provide several keywords. From the matchmaking model’s perspective, this type of approaches is easy to be implemented. However, keywords based matchmaking suffers two problems: 1) It cannot capture the underlying semantics of web services advertised in repositories and 2) It cannot represent the user’s real intention [10]. In fact, a web service description file is not just plain document consisting of a collection of words or sentences. It is designed to deal with certain logics and functions advocated by the web service provider. Apparently, keywords based approaches is not capable of tackling with such information.

### B. Syntactics Based Matchmaking

Relying on textual description to match request against web services is apparently insufficient. To overcome the incapability of keywords based approaches, other methods have been proposed from different perspective. In these methods, importance has been attained on the web service’s function, interface, operation, structure or the combination of any of them. Since these aspects are fall into the scope of syntactics in semiotics framework [13], in this paper, we call them syntactics based matchmaking models.

The major thought of this kind matchmaking model is to use keywords based methods to firstly filter a set of web service candidates. After that some algorithms focusing on web services’ function, interface, operation or structure are employed to further refine and re-assess the candidate trying to find a final ordered list. Therefore, the relevance of the request and web service can be generally defined as:

\[
sim(D, Q) = \alpha \times \text{sim}_{keywords}(D, Q) + \beta \times \text{sim}_{structure}(D, Q)
\]

where \(\text{sim}_{keywords}\) is the keywords based similarity between request and web service registered in the repository, \(\text{sim}_{structure}\) is the structure based similarity score, which is the most important indicator in syntactics matching model. \(\alpha\) and \(\beta\) are empirical weights of each similarity score respectively and satisfy \(\alpha + \beta = 1\).

[16] propose an interface matchmaking model. Firstly, a rough list of candidates is retrieved by using VSM method. Secondly, a structure matchmaking algorithm is used in which the data type, message structure and operation is calculated respectively. [17] propose similar matchmaking model in which attributes similarity, interface similarity and QoS similarity are combined with lexical similarity.
As this kind of approaches further studies the syntactic structure of web service description files to find possibly matched web services, compared with keywords based approaches, they might be more effective since the information used for web service matchmaking is more indicative and meaningful. However, from the user’s perspective, the user experience would be impeded since the user probably have to provide the request in the format of desirable WSDL file, though potential partial [16], which apparently need the users have some knowledge background on web service description file and its structure. Furthermore, another disadvantage of this kind of discovery models is also obvious. Lack of semantic expression is an important inhibitor because similar structure does not always guarantee similar functions or behaviours and also the other way around [18].

C. Semantics Based Matchmaking

Semantics based models are consequently brought forward in the community trying to add semantic information into web service discovery process. Different standards describing semantics have been developed, e.g. DAML-S [19], OWL-S [20], WSDL-S [21], WSML [22] and the like. These approaches try to understand the meaning of web service description files and the users’ request by taking advantage of ontology. Semantics based web service matchmaking models try to annotate web service description files, either manually or automatically to concepts based on certain ontology. Afterwards it is the concepts that will be involved in matchmaking. A typical process is presented in an example in [18], where the proposed framework maps operations, messages, preconditions, effects to corresponding ontology concepts. Semantics based matchmaking develops rapidly and a lot of different approaches have been proposed to annotate different parts of the interface files as such it is too diverse to be summarized in a uniform way.

The benefits are prominent as the web service interfaces are then machine-readable. Thus the discovery model would be more effective when matchmaking web services. Despite all of these advances, understanding of ontology or other related knowledge might be a must to the user when using this kind of approaches since the submitted request must be in similar format based on certain ontology. That will probably place more burdens on the user. Furthermore, there is pragmatic heterogeneity since the web service providers and customers might have different way to implement domain specific process even they hold similar terminology [23]. Different providers could also use different ontology to express the capabilities of their web services, which will cause higher computation complexity to cover possible ontology inconsistency.

D. Pragmatics Based Matchmaking

Roughly speaking, the pragmatics matching model is not a brand new model like the previous three ones. It does not advocate more revolutionary matching and similarity calculation mechanism. The focus of this kind of model will not only on the structure or meaning within the requirement and description file. It mainly lays emphasis on how the existing service requirement is transformed to service request, in what way, and then helps a discovery system to grasp the process of requirement filtering, which is the core part of a pragmatics perspective on information processing [24]. In fact, this is the main reason why the term “pragmatics” is borrowed in this research.

![Pragmatic Matching Model Infrastructure](image)

Therefore, the pragmatics matching model is defined as a matching model with the ability of taking into account of context to discover web services. The basic infrastructure is depicted as Fig. 5, where the matching part is anyone of the three matching model or a hybrid combining them together.

IV. CONTEXT AWARE WEB SERVICE MATCHMAKING

From above discussion it is learned that one of the utmost challenges in current web service discovery problems is to balance the performance and easy for use. The matchmaking model always expects that the imputed request carries information as much as possible and also well formatted as required. Unfortunately, the request is inevitably incapable of meeting such requirement. This dilemma calls for more sophisticated methods, and context awareness is one of the probable solutions to meet the gap.

In this section, we are going to summarize context awareness in web service discovery from the source of context collection’s perspective. As such context awareness in this paper is divided into four categories, i.e. Personal profile oriented context, Usage history oriented context, Process oriented context, and other context. Before we discuss these different types of context awareness in detail, an important concept deserves to be elaborated in depth. Generally, context can be provided by the user explicitly or collected by the system implicitly. Hence the context can be firstly divided into explicit context and implicit context.

A. Explicit vs. Implicit

Explicit context is provided directly by the user during the matchmaking process. At the beginning a user’s requirement is often ambiguous. As a result the generated request does not always reflect the original thought well. But the user can be asked to submit extra information to support the request clarification. Obviously as it is the user who know the best the real intention, all the extra information provided by the user is the most important and indicative. For example, Q&A is a major means to get user’s purpose in most applica-
tions. Through providing several questions and checking the user’s answers step by step, the system can accumulate useful background information such as demographic data, purpose, preference and etc. Thus it is becoming easier and clearer to understand user’s real intention. However, due to the natural character of complexity in defining context, it is nearly impossible to propose a meta-model for explicit context collection for all applications. From a user’s perspective, always being asked to submit extra information besides the request will probably become gradually unacceptable. Furthermore, there also exists information that users cannot explicitly express [25].

Compared with explicit context, the implicit context is accumulated by the system in automatic or semi-automatic way. A lot of systems are using this kind of context to analyse the user’s potential needs. For example, a location based service provision will normally check the location where the user submits the request but the location detection process is not felt by the user. Because the user is not explicitly involved in the context constitution process, this method is apparently more applicable than explicit context. Recently, much effort has been put in this area and it is believed automatically or semi-automatically recognition of context is going to be the major direction in the future.

B. Personal profile oriented context

The basic idea of this kind of context is to constitute a user’s personal profile recording the personal data, preference and other information which can be used for personalisation. [26] provides an example showing how to build personal profile and point out its importance in decomposing the discovery goal, setting selection criteria and supplying parameters, thereby achieving personalisation in web service discovery. [27] proposes a similar model to represent user profile and then use it to support personalisation in web service discovery. Currently some approaches have been proposed to building up the web service user’s personal profile by modelling different attributes, e.g. location, time [28][29], the user’s situation [30][31]. Though these methods demonstrated promising performance to some extent, there are also some limitations related to such methods. For example, the system architecture would become complicated when more new constraints or attributes are introduced. Furthermore, some application domains have their own constraints, which make a proposed method for that domain less adaptive to other application domains.

C. Usage history oriented context

The impulse of building up context from user’s usage history is based on the finding that the user’s data consumption pattern can be employed for predicting the user’s next behaviour. The assumption of this kind of context is that the user’s web services requested during a certain period is more or less similar to some extent. As such through collecting related usage data from a long term observation, it is possible to find some useful information from the user’s past experience. This kind of context can be further divided into two types, i.e. Personal usage history oriented context and Group usage history oriented context.

1) Personal usage history oriented context: A user’s previous experience in web service request is very strong indicative information for user’s preference and domain background. Therotically, the accumulated personal usage information can make it possible to achieve personalisation in the matchmaking process. [32] proposed a user’s usage model by keeping system log information for user-system interaction. By studying the recorded log the proposed model can utilising this kind of context and help recommend the service selection decision. But this approach does suffer the fact that only relying on the user’s own past experience would make the information distortion amplified, which is probably because the user’s previous selection might not be a real proper behaviour. Actually, most usage history based web service discovery systems refer to other user’s behaviours in similar situation, which can be called group usage history oriented context.

2) Group usage history oriented context: Recently the concept of social group have been attached much importance with the development of web 2.0, where users wish to share their experiences with other people in virtual communities. This idea is employed by several web service discovery systems which make recommendation based on a group of users’ similar behaviour to help the matchmaking process. Since the system performance relies on other users’ decision, this kind of context is so called group oriented.

One of the most popular approaches used in building up group oriented context is collaborative filtering (CF) method based solution and a lot of solutions have been developed [33]. The CF method is divided into memory based approach and content based approach. Most web service discovery system adopt memory based approach in which all of the users are firstly asked to rate part of all web services in the repository based on his/her usage experience. After that a set of similar users, or called neighbourhood, is found by using user similarity algorithm. The most widely used similarity measure methods in memory based approach include Pearson correlation [34] or cosine similarity. Finally, when a user is asking for a web service, the system will recommend a set of candidates based on the user’s neighbours’ previous decisions.

The critical part of CF based approaches is the user’s rate on each web service. Different approaches have their own rate explanation. For example, [35] propose an extended concept-based collaborative evaluation system for web service selection by using agent instead of user to collect the rating for each web service. [36] provides a recommendation method in which predictions for web services by a user is calculated and presented in order of their ratings. [37] uses memory-based CF in QoS aware web service discovery. It uses QoS value as rating and tries to find similar users via observing similar QoS experience. [38] develops a CF based web service selection mechanism using goal and preference together but not single rating to predict and recommend web services.

Except CF based group context, there are other kinds of group contexts. [39] propose a method to record the connection between the requests and the services’ invocation and execution in a particular community. They use the observa-
tion data to identify which a service is likely to be relevant for a request. Generally, the usage based context is only effective when the empirical information is sufficient enough otherwise the recommendation could be worse as the improper experience will be pulled into the discovery process.

3) Process oriented context: Compared with the usage based context, the process oriented context is defined as the information captured from the current session of discovery process. This kind of context aims to study the reaction of the retrieved set of web service candidates and then optimize the discovery. [40] propose a method in which a set of candidate will be retrieved based on the user’s request. After that, Probabilistic Latent Semantic Analysis (PLSA) is employed to cluster the retrieved candidates into different groups. Finally, the request will be compared against each group rather than each web service.

Besides the feedback based process oriented context, there is another important context which is studied from viewpoint of service composition process. In real applications it is often when a business calls for a certain service but no single web service can fulfill the requirement, which leads to the need of services composition. It is believed that most retrieved web services will always join certain composition processes to fulfill users’ requirement. From this viewpoint, web service discovery is not an isolated action. It should be discussed in the context of composition. As indicated in [41], the web service discovery should be highly dependent to the intended composition process. Actually, it is important to accommodate this kind of inter-service dependency [42]. This kind of context is then proposed from this perspective. [43] proposes a mechanism to automate web service dependency management in SOA model. [44] discuss the importance of service pattern and develop a mechanism to capture the patterns.

4) Other context: Since context cannot be defined clearly in a uniform way, the classification in this paper is not thorough and comprehensive. In the literature, there are still a lot of web service discovery sysyem which integrate different attributes in the discovery process, e.g. trust and reputation [45], Goal [46], Domain [47] and etc, which we call other context.

V. FURTHER APPLICATION OF CONTEXT IN WEB SERVICE DISCOVERY

Context awareness provides adequate information in different aspects for the web service discovery. The application of context in the discovery process can be roughly divided into three categories, i.e. request optimisation, result optimisation and personalisation.

Request optimisation is the major application of contextual information. As elaborated in previous sections, the request normally contains relatively small amount of useful information and is not much precise. Nevertheless, as the main input of discovery process, the quality of request is of much importance. Therefore, the context can be applied in optimising request, which could include request rewrite [48], request expansion [49] and etc.

Also it is anticipated that some web services that are displayed with lower ranking in the retrieved set are more relevant to the user’s expected real need. This situation is certainly inevitable because the request is always likely to be distorted from the original intention. The result re-ranking is one of the most significant applications with contextual information.

Personalisation is another important application of contextual information as traditional web service discovery provides search service in the uniform way, or called “one-fit-all”. Different users will get same results if the inputted requests are same. From this perspective, context awareness provides an ideal mechanism to solve this problem.

VI. CONCLUSIONS

Context awareness has made significant progress in various applications. Over last few years, a lot of context aware web service discovery models, methods, frameworks have been proposed to improve user’s experience in web service matchmaking. However, as the context covers a broad scope it is very difficult to give a standard definition on context. Different approaches employ different attributes (“context”) to their specified problem. In this paper, an overview of context awareness is presented from request’s perspective trying to clarify the difference between request and requirement. The importance of context awareness has been elaborated and a classification of context aware discovery approaches is introduced and discussed. Finally some possible challenges of applying contextual information are pointed out. It is hoped that this work can help advance the discussion of context awareness in web service discovery technologies.

REFERENCES