2013 IEEE Ninth World Congress on Services

Personalized Mobile Web Service Discovery

Khalid Elgazzar, Patrick Martin, Hossam S. Hassanein
School of Computing, Queen's University, Canada
{elgazzar, martin, hossam}@cs.queensu.ca

***Abstract*—Mobile devices with their various form factors have become the most convenient and pervasive computing platform, whether to carry out everyday business or to get online. Mobile users tend to adopt the fast food trend even in consuming online mobile services and functionalities. The Web service approach promises great flexibility in offering software functionality over the network, while maintaining interoperability between heterogeneous platforms. However, the diversity that exists in mobile devices and their platforms with variations in capabilities present unique challenges in developing services that can accommodate such diversity. Recent years have witnessed the rise of user-facing service developments that can be consumed on the go with standard interface, such as RESTful Web services. However, the discovery of such services does not match their growing popularity. In addition, existing discovery approaches lack supporting mechanisms that ensure the proper functioning of discovered services within the user context and failing to match personal preferences. This paper introduces personalized Web service discovery for mobile environments. Preliminary results show that incorporating user preferences and context significantly improves the overall precision of service discovery.**

***Index Terms*—Personal mobile services, Web services discovery, mobile computing, mobile devices.**

I. Introduction

According to latest *Mobile Factbook* released by PortioResearch [1], the global mobile customer base has exceeded 6.5 billion subscribers in the beginning of 2013, which represents 87% of the world's current population, 1.5 billion of subscribers have broadband subscriptions. These numbers are candidate to rise in the coming years, especially with the growing interest in broadband connectivity. These facts highlight the significant potential market for mobile services and applications. The revolution in broadband wireless access technologies with the deployment of 4G networks promises to offer data services 3-4 times faster than the current speed, offering a differential user experience. Both the proliferation of mobile devices and advances in wireless technologies have contributed to the growing interest in mobile services, expanding the horizon for a great potential for a multitude of services and opportunities.

In recent years, there have been a growing popularity of user-facing mobile services with the widespread adoption of RESTful Web services. RESTful Web services [2], which conform with the REST principals [3], have shown better performance and flexible provisioning in resource-constrained environments [4]–[6]. They are capable of communicating with both applications and users via dispatching the appropriate response format according to the consumer type. For example, XML or JSON-formatted responses are sent in response to requests originated from applications, while responses in HTML format are dispatched to the user's Web browser.

Mobile services make service personalization possible, where services are selected in the best interest of the user and that best fit the current situation. This is possible due to the fact that mobile devices are associated with users who have personal preferences, beside the ability to access various contextual information. However, existing discovery approaches lack robust techniques that can incorporate the user preferences and context while finding services that satisfy the user's objective. Consequently, relevant services might not perform properly due to device constraints or result in poor user experience due to lack of accommodating user preferences.

This paper leverages the use of context information and user preferences to personalize mobile service discovery and provide users with a differential experience. Our approach employs four types of context information to filter out discovered services that do not match the user context and rank the rest according to their aggregate relevancy to such a context.

The remainder of this paper is organized as follows. Section II highlights previous related work. Section III presents the proposed discovery approach, outlining the various context used and detailing our filtering and ranking methodology. Section IV demonstrates our experimental validation and Section V shows our preliminary performance evaluation results. Section VI concludes the paper and draws future directions.

II. Related Work

One of the chief benefits of mobile services is the ability to exploit various context to provide personalized behaviour. In addition, the objective of incorporating context information with Web service discovery extends to ensure proper functioning. For example, device-aware service discovery [7] ensures the compatibility of the discovered/selected service with the device constraints. User preferences is another dimension by which services can adapt to accommodate the consumer preferences. However, to date, there are no efficient models that can integrate user preferences with Web service descriptions and then be incorporated in the discovery process. A serious step towards this direction is proposed by García et al. [8], enabling users to express their preferences using an interactive interface. The authors demonstrate a use-case on how to integrate such preferences with semantic Web service discovery.

978-0-7695-5024-4/13 $26.00 © 2013 IEEE
DOI 10.1109/SERVICES.2013.18

Al-Masri et al. [7] developed a device-aware service discovery mechanism that is capable of selecting Web services that function properly within the mobile device constraints. The mechanism takes advantage of HTTP sessions to collect device information and store it at the server side. This information is compared later with the requirements that are set by the providers to rank services according to the compatibility with the device features. The authors propose an extension called "WSDL-M" to the standard WSDL definition [9] in order to include the required device-specific features.

Mobile environments pose unique challenges to service discovery due to two inherent characteristics: mobility and frequent context changes. Mobile services that can adapt their behavior in accordance with dynamic context changes are highly desired. Towards that, Maamar et al. [10] discuss the development of capacity-driven Web services starting from the description, discovery, composition, to the invocation of proper desired functionality. Similar research on services with different qualities is presented in [11]. However, the execution of services with multiple capacities requires runtime environments to support the appropriate mechanisms for assessing the environment status, so that services can promptly adapt their behaviour.

III. Context-aware Service Discovery

The development of mobile services that proactively provide information to users when and where they need it are on the rise. Such services support a variety of adaptation techniques to accommodate possible heterogeneity of consumer devices, platforms, and preferences. The primary concern of mobile users is finding services that best fit their current situation and preferences. Traditional service discovery techniques lack robustness to support such flexibility, i.e. incorporating context information in finding relevant services to a user objective.

The primary goal of our proposed approach is to effectively employ various context information to personalize mobile service discovery, while ensuring the compatibility of discovered services with user devices. Our approach encompasses two main components, *context management* and *service ranking*.

1) Context Management: The context management component relies on four types of context in order to provide personalized mobile service discovery: user preferences, device profile, environment context and user ratings.

`User preferences`: Web service descriptions do not support integrating user preferences. This implies that discovery mechanisms may retrieve services that satisfy the user objective, but not necessarily accommodate the user preferences. Such preferences could be dynamic and might change according to the user context and required service. The user also might be strict about some preferences, while being flexible with some others, for example, in finding a nearby food catering service the user may specify *"I request vegetarian restaurants, however, I prefer those that accept credit card payments"*.

`Device profile`: Incorporating the device profile information with the Web service discovery process ensures the compatibility of the discovered service with the device features and constraints. For example, a payment service that features Near field Communication(NFC) support, requires an NFC-enabled device to function.

`Environment Context`: Providing mobile users with services that fit their current context, such as location, is a distinguishable user experience. In mobile environments, frequent context changes such as depredation in bandwidth or connectivity interruption might substantially impact the performance of services. Based on current conditions the user might be able to execute limited functionalities of particular services. The objective of integrating the environment context into service discovery is two-fold: select services that fit a certain context and ensure that services run properly given specific environment conditions.

In order to effectively implement this dimension, providers are required to specify the minimum and preferred conditions for reliable operation of their services. Listing 1 outlines possible parameters that a service provider might be interested to specify. These conditions can be associated with service description files. While at the customer side, a dedicated *Context Manager* collects the various environment parameters to better assess the customer's environment status. The discovery mechanism then matches the current user context with the provider's recommended conditions. Relevant services are ranked based on aggregated suitability to various context.

Listing 1: A sample of a provender's recommended description of Web service `X`.

```
<?xml version="1.0" encoding="UTF-8"?>
<rdf:RDF
 xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
 xmlns:env="http://www.example.sample/env#"
 xmlns:dev="http://www.example.sample/dev#">

<rdf:Description
 rdf:about="http://www.ServiceDomainName/X.name?wsdl">
   <env:serviceName>X.name</env:serviceName>
</rdf:Description>

<rdf:Description
 rdf:about="http://www.example.sample/user/device">
   <dev:SoftwarePlatform>OS.Vendor</dev:SoftwarePlatform>
   <dev:NFCSupport env:type="required">yes</dev:NFCSupport>
   <dev:SupportCSS>yes</dev:SupportCSS>
   <dev:GPSEnabled>yes</dev:GPSEnabled>
</rdf:Description>

<rdf:Description
 rdf:about="http://www.example.sample/env/network">
   <env:signalStrength>good</env:signalStrength>
   <env:bandwidthMB env:type="required">2</env:bandwidthMB>
   <env:dataRate>20</env:dataRate>
   <env:errorRate>10</env:errorRate>
</rdf:Description>

<rdf:Description
 rdf:about="http://www.example.sample/env/proximity">
   <env:location>
     <env:name>city_center<env:name>
     <env:longitude>44.242102</env:longitude>
     <env:latitude>-76.516589</env:latitude>
   </env:location>
</rdf:Description>
.
.
.
</rdf:RDF>
```

`User ratings`: Current Internet models encourage users to leave feedback and rate services based on their actual experience. User ratings reflect their level of satisfaction. These ratings can be used as an effective tool to show the user-perceived quality of service operation rather than quality claimed by providers. Our approach takes advantage of user ratings to rank relevant Web services, assuming that proper handling mechanisms for ratings and complaints are applied.

2) Service Ranking: The service ranking component receives a set of relevant Web services $S = \{s_1, s_2, ..., s_l\}$ and ranks them based on the four types of context assuming that: user preferences $P = \{p_1, p_2, ...p_i\}$, device profile $D = \{d_1, d_2, ..., d_j\}$, environment context $E = \{e_1, e_2, ..., e_k\}$, and user ratings R_{s_l}, where R_{s_l} represents the normalized average rating for a service s_l.

To show how ranking works, suppose that the matchmaking retrieves a set of relevant services S to a service request SR. These services perform similar functions, but vary in their requirements for proper execution. During the request communication session, the discovery mechanism collects the features of the customer's device $Dc = \{dc_1, dc_2, ..., dc_j\}$ corresponding to the provider's recommended device profile D. The context manager on the mobile device gathers various ambient conditions to assess the environment status $Ec = \{ec_1, ec_2, ..., ec_k\}$. For each service s_l, there are features ps corresponding to user preferences P as expressed by the matrix in Eq.(1). These features can be extracted from the service description files.

$$S_P = \begin{bmatrix} ps_{1,1} & ps_{1,2} & ... & ps_{1,i} \\ ps_{2,1} & ps_{2,2} & ... & ps_{2,i} \\ \vdots & \vdots & \vdots & \vdots \\ ps_{l,1} & ps_{l,2} & ... & ps_{l,i} \end{bmatrix} \quad (1)$$

where rows represent the set of service features for service s_l that are related to the user preferences and columns represent a single feature across the entire set of relevant services. A similar matrix can be obtained for the other three context domains, D,E, and R.

The rank of each service s_l then is represented by the following formula in Eq. (2).

$$\begin{aligned} Rank_{s_l} = & \sum_i w_i \times f(p_i, ps_i) + \\ & \sum_j w_j \times f(d_j, dc_j) + \\ & \sum_k w_k \times f(e_k, ec_k) + \\ & R_{s_1} \end{aligned} \quad (2)$$

where w represents the weight of each corresponding feature. This weight indicates the level of importance of such a feature to either the service customer or the service provider. The function f computes the relation between two objects as shown in Eq. (3).

$$f(a,b) = \begin{cases} \frac{b}{a+b} & \text{if } a, b \text{ are numbers} \\ sim(a,b) & \text{if } a, b \text{ are strings} \end{cases} \quad (3)$$

where $sim(a, b)$ is a featureless similarity factor computed between objects a and b using Normalized Google Distance (NGD) [12]. $sim(a, b)$ is calculated by Eq. (4).

$$sim(a,b) = 1 - NGD(a,b) \quad (4)$$

$f(a, b)$ yields values $\in$ [0...1]. In the case where a and b are numbers, values greater than or equal 0.5 means that b satisfies a. The closer the value to 1, the higher satisfaction the condition b achieves to the condition a. In the case where a and b are text (keywords), the function value close to 1 indicates that the terms a and b are semantically related, where values close to 0 indicate that they are not related. Algorithm 1 symbolically explains the ranking procedure.

Algorithm 1: Ranking Algorithm

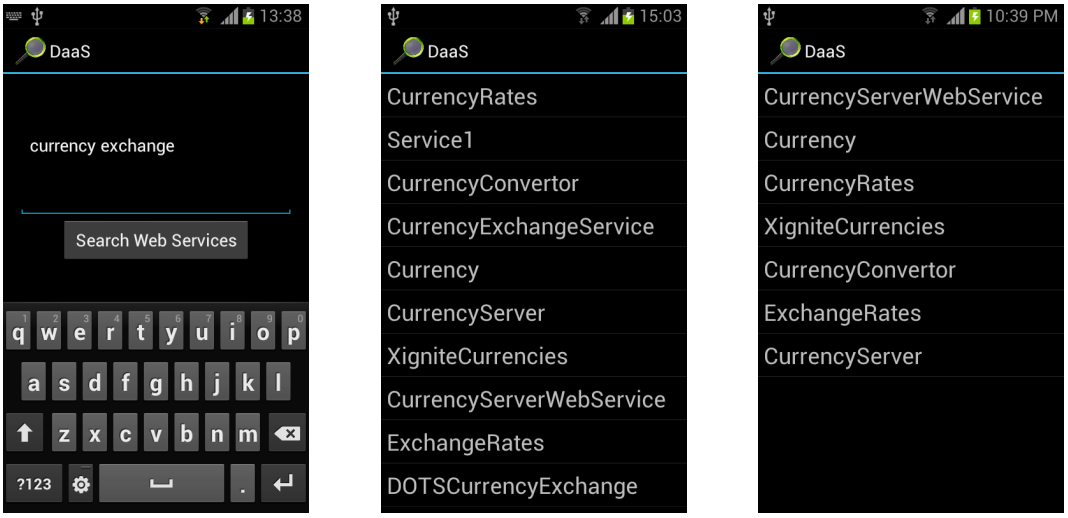
Input: set of relevant services S,
various context P,D,E, and R,
and corresponding weights W
Output: ranked list of services $RankS$

```
Function Rank(S, P, D, E, R)
Initialize RankS
// calculate the rank of each s ∈ S
// based on all context domains, i.e. P, D, E,R, Eq. (2)
foreach s in S do
    Rank_s= Calculate Eq. (2)
    RankS=RankS + Rank_s
end
//sort the results
Sort(RankS)
return RanS
```

The result is a list of services ordered based on aggregate relevancy to the user's request, preferences and current context. This list is typically shorter than the retrieved list of services. The service requester selects one of those services to run.

IV. Experimental Results & Discussions

We have developed a preliminary prototype to validate the functionality of the proposed approach. The prototype has two parts, a user interface to submit service requests and a keyword-based service matching algorithm. Experiments are conducted with 1200 WSDL documents of valid online Web services that serve various domains. These service description files are collected from real world Web service providers and online service directories, such as WebserviceList, WebserviceX and xMethods. Our focus is to evaluate the context-aware filtering and ranking technique, not to evaluate the precision and recall of the information retrieval technique. The user interface is deployed on a smartphone with Dual-core 1.2 GHz Cortex-A9, 1 GB RAM, and running Android 4.0.4 platform [13]. The device does not feature NFC support.



(a) A simple mobile-based service request interface.

(b) DaaS rank relevant services according to current context.

(c) DaaS re-rank relevant services to cope with changes in context.

Fig. 1: Prototype screenshoots of the mobile side.

The service matching algorithm and the data set reside on a desktop machine with a Pentium(R) Dual-core CPU E6300 @ 2.8GHZ, 4 GB RAM and running Windows 7 Professional 32 bits. The phone connected to the desktop machine through a WiFi link with an average Round Trip Time (RTT) = $8ms$.

For each of these services, we have generated a context file that includes a set of required and recommended device features such as NFC support, environment parameters such as served locations, recommended network links and conditions such as WiFi, 3G and bandwidth. A user profile is also generated, including preference parameters such as payment method, food, restaurants, shopping places, travel interest, etc. Some of these preferences are marked required and others are marked preferred. Once the objective matching algorithm retrieves relevant services, our ranking component rules out services the do not satisfy the required preferences or accommodate device features, despite matching the request objective. The rest of services are ranked according to their aggregate relevancy to the user preferences and context.

The matching module measures the similarity between the request and Web service description based on a non-repeated keyword match scheme. First, it extracts the keywords from the service description, wherever the tag `<wsdl:documentation>` is found in the description file, whether the tag belongs to the service or to individual elements. Then, it performs a keyword pair-matching to measure the similarity between keywords obtained from the service request and those obtained from the service description. Further details on keyword/feature extraction can be found in [14].

Figure 1b shows relevant services that match a service request searching for *"currency exchange"*. The objective matching algorithm retrieves 18 services that match the request from our data set. Figure 1c shows the shortlisted services

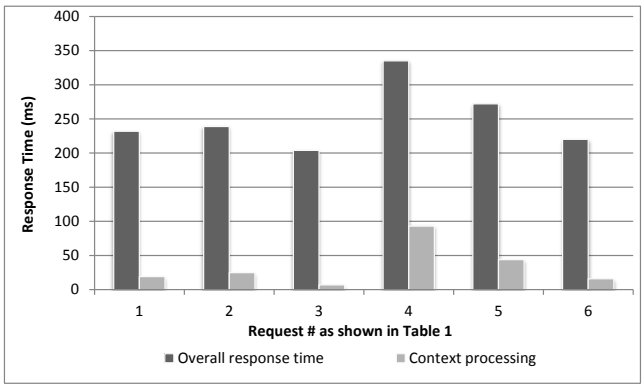


Fig. 2: The overall response time in contract with contribution of context matching.

that match the user preferences and accommodate the device features, sorted by the relevancy degree. A manual inspection of the retrieved services and their generated context files shows that the Web service *CurrencyServerWebService* offers services that are in the closest proximity to the user and accepts mobile payments, whereas all ruled out services are not within 20 miles of the user location.

Table I shows further results of other queries and their respective retrieved and shortlisted number of services. The *context overhead* indicates the percentage of context matching time to the overall time.

V. Performance Evaluation

In order to evaluate the performance of the proposed approach we measure the overall response time of the different queries shown in Table I, indicating the contribution of the context matching. The end-to-end response time includes both

#	Request(keywords)	# of Retrieved Services	# of Shortlisted Services	Rule Out Parameters	Overall Response Time	Context matching	Context Overhead(%)
1	Food Order	14	4	user location, NFC support	232	19	8.2
2	Currency Exchange	18	7	user location	239	25	10.5
3	E-mail Verification	5	5	NA	204	7	3.4
4	Weather Forecast	85	20	user location	335	93	27.8
5	Stock Quote	35	3	graphic support	272	44	16.2
6	Shipment Service	10	3	user location, payment method	220	16	7.3

TABLE I: Performance parameters of various service requests, demonstrating retrieved and shortlisted services and ruling out parameters.

communication, service matching and context matching time. Figure 2 illustrates the overall response time in contrast with the contribution of context matching for various service requests. We observe that the response time is dominated by the service matching while the context matching time is proportional to the number of retrieved services. For example, in the case of request # 4, the context matching time represents 27.8% of the overall response time, while in the case of request # 3 it only represents 3.4%, given that context matching is performed on the resource-constrained mobile device. With this little overhead, the relative quality of service discovery for requests 3 and 4 is improved by 76.4% and 91.4%, respectively. We define the relative quality of service as the percentage of ruled out services to the total number of retrieved services. Therefore, we conclude that incorporating the context into the service discovery process brings significant benefits into the quality of service recommendations, with a little processing overhead.

VI. Conclusion and Future Work

In this paper we propose a robust service discovery approach that makes use of user preferences and context information that mobile devices can use to personalize the discovery of mobile Web services. The advantage of personalized service discovery is two-fold: ensures that discovered services fits the user's device constraints and provides mobile users with a pleasant experience by accommodating their preferences. The experimental validation and performance evaluation of our prototype show promising results. The overhead that context processing adds to the discovery process is negligible in contrast with the improvements of the service discovery quality. We plan to further investigate indicators that reflects the user-perceived satisfaction of personalized service discovery. We also plan to study the tradeoff between performing the context matching on the mobile device and offloading it to the server side, where the request objective matching occurs.

Acknowledgement

This research is supported by Natural Science and Engineering Research Council (NSERC) of Canada and CA Technologies.

References

[1] Portio Research Mobile Factbook 2013, http://www.portioresearch.com/media/3986/Portio%20Research%20Mobile%20Factbook%202013.pdf.

[2] J. Meng, S. Mei, and Z. Yan, "Restful web services: A solution for distributed data integration," in *International Conference on Computational Intelligence and Software Engineering, 2009. CiSE 2009.*, pp. 1–4, 2009.

[3] R. T. Fielding, *Architectural styles and the design of network-based software architectures*. PhD thesis, University of California, Irvine, USA, 2000.

[4] H. Hamad, M. Saad, and R. Abed, "Performance evaluation of restful web services for mobile devices," *International Arab Journal of e-Technology*, vol. 1, no. 3, pp. 72 –78, 2010.

[5] K. Wagh and R. Thool, "A comparative study of soap vs rest web services provisioning techniques for mobile host," *Journal of Information Engineering and Applications*, vol. 2, pp. 12–17, 2012.

[6] F. Belqasmi, R. Glitho, and C. Fu, "Restful web services for service provisioning in next-generation networks: a survey," *IEEE Communications Magazine*, vol. 49, pp. 66 –73, December 2011.

[7] E. Al-Masri and Q. H. Mahmoud, "Mobieureka: an approach for enhancing the discovery of mobile web services," *Personal Ubiquitous Computing*, vol. 14, pp. 609–620, October 2010.

[8] J. M. García, D. Ruiz, and A. Ruiz-Cortés, "A model of user preferences for semantic services discovery and ranking," in *ESWC 2010, Part II*, vol. 6089, pp. 1–14, 2010.

[9] R. Chinnici, J.-J. Moreau, A. Ryman, and S. Weerawarana, "Web services description language (wsdl) version 2.0 part 1: Core language," June 26 2007. http://www.w3.org/TR/wsdl20.

[10] Z. Maamar, S. Tata, D. Belaid, and K. Boukadi, "Towards an approach to defining capacity-driven web service," *International Conference on Advanced Information Networking and Applications (AINA'09)*, pp. 403–410, 2009.

[11] A. Tao and J. Yang, "Context aware differentiated services development with configurable business processes," in *Proceedings of the 11th IEEE International Enterprise Distributed Object Computing Conference*, pp. 241–252, November 2007.

[12] R. L. Cilibrasi and P. M. B. Vitanyi, "The google similarity distance," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 3, pp. 370–383, 2007.

[13] Android 4.0 Platform, http://www.android.com/about/ice-cream-sandwich/.

[14] K. Elgazzar, A. E. Hassan, and P. Martin, "Clustering wsdl documents to bootstrap the discovery of web services," in *Proceedings of the 2010 IEEE International Conference on Web Services*, pp. 147–154, 2010.