Advanced Engineering Forum *Online: 2011-12-22*
ISSN: 2234-991X, Vols. 2-3, pp 579-582
doi:10.4028/www.scientific.net/AEF.2-3.579
© 2012 The Author(s). Published by Trans Tech Publications Ltd, Switzerland.

User-Oriented Web Service Search Technique for Mobile Mashup

Kazuki Hizen* and Takahiro Koita**

*Graduate School of Engineering, Doshisha University, Japan
**Faculty of Science and Engineering, Doshisha University, Japan

dtk0716@mail4.doshisha.ac.jp, tkoita@mail.doshisha.ac.jp

Keywords: Mashup, Mobile Phone, Web service

Abstract. This paper presents the basic design of a new web service search technique for mobile mashup. Web service search means searching web APIs to build a new mobile mashup service. The technique is based on user-oriented profiling and can be easily used on mobile devices. For web service search, a user must know the details of a web service in advance. However, conventional service search cannot provide such information on whether the web service can be used for the mashup. This study discusses a user-oriented web service search technique for mobile mashup. Its key feature is that users can easily search for web services that use mashup on mobile phones without knowing the details.

Introduction

In recent years, a wide variety of web services have been created on the Internet and those services are implemented by using web APIs. Many IT companies such as Google [1], Amazon [2] and mixi [3] started to publish web APIs that enable use of their web service from outside. By publishing APIs, mashup attracts developers' attention to create new services. Mashup is a technique that combines arbitrary web services mutually by using web APIs to make one new web service. The developer can create and provide a new value of the web service by mashup. Therefore, mashup is getting popular and is widely used all over the world. Additionally, mashup is changing not only for developers but also for users. Mashup needs to be easier for users, and user-oriented mashup has to be considered to support users' mashup. For the user-oriented mashup, searching which API a user or developer wants to use is important in composing a mashup service. Conventionally, search engines such as Google and Yahoo! [4] are used to search for APIs, and developer information of an API on the web site is obtained from the search results. However, the information is not specialized for mashup and users have to interpret the results to use them for mashup. On the other hand, Programmableweb [5] and mashupedia [6] provide only mashup information of existing web services and APIs, and are called mash up portal sites. Figure 1 shows the overview of search APIs using search engines versus mashup portal sites. This figure indicates that the conventional techniques are complicated for users to create new mashup service. Moreover, mashup services are used on mobile phones such as iPhone [7] and Android OS [8]. Mashup is expected to spread to more users in the near future. Searching for an API for the mashup on a mobile phone is also difficult due to the limitations of mobile phones.

Problem

In this section, we discuss the three problems of mashup techniques using Google or Programmableweb to achieve user-oriented mashup.

The first problem is that information of an API registered on the mashup portal sites such as Programmableweb or discovered by search engine is not the latest information, and its contents provided by the developer might be incomplete. Thus, the API registered in a mashup portal site might be different from the information on the site. Due to this mechanism, a mashup portal site cannot obtain the latest information of an API automatically. The specification of an API existing in a mashup portal site may be different from the predetermined one when users used the API, and sometimes the API will disappear.

The second problem is that information of an API registered on a mashup portal site or discovered by search engine employs a wide variety of data formats. For example, the data format shows “PHP” or “JavaScript” and category information shows “map” and “NEWS”. Therefore, a developer has to confirm and interpret the information.

The third problem is that, if a user accesses information of an API registered on a mashup portal site or discovered by search engine, the user has to move the website with the API. In this website, to search for the target API, the user must trace a web link.

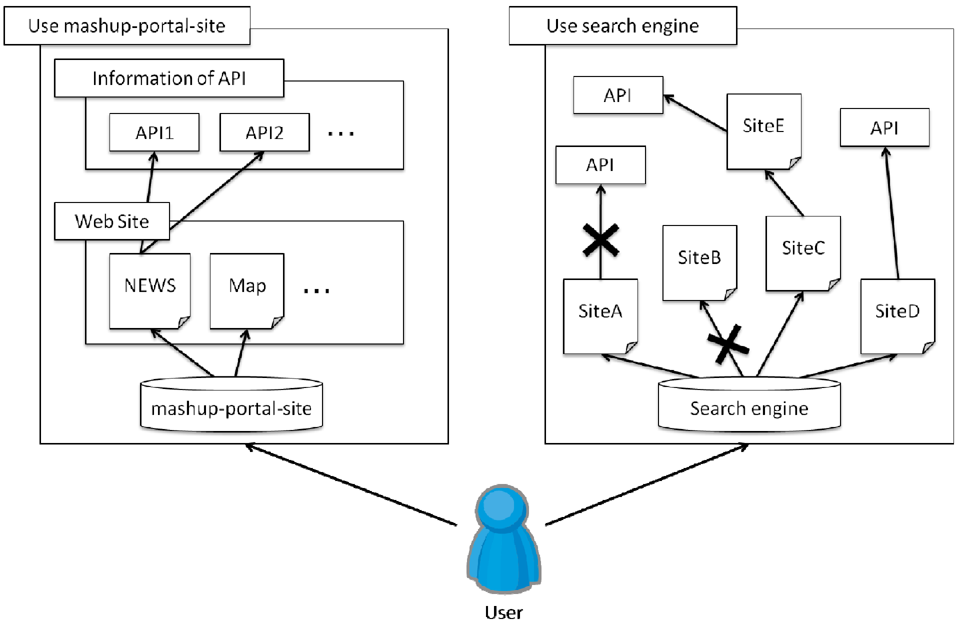

Fig. 1. Searching APIs using search engine and mashup portal site

Approach

As described in section 2, a mashup portal site or search engine cannot provide enough information to users for easy mashup. Furthermore, a mashup portal site is not suitable for supporting a user’s mashup. To solve these problems, we introduce a new user-oriented web service search technique to support a user’s mashup. We aim to achieve user-oriented mashup without using conventional techniques in the present study. For that, we focus on three approaches as follows.

1) The latest API information should be able to be acquired by automatic operation at any time.
2) Information of the API obtained must be classified automatically as to usage and kind.
3) The API should be able to be obtained directly only by accessing information of the API obtained.

These three approaches address the above three problems respectively. By achieving these three purposes, we can offer a user-oriented web service search technique for the support of mashup.

Related Works

Web service search on a large scale [9] is one important related work. Steinmetz presents their approach for web service discovery on web scale, targeted to support flexible and on-demand web service usage on the web. They search the Internet by using the technology of data mining to find an API that exists on the Internet. The technology of data mining that uses the model is called the support vector machine model [10]. The search for API has been facilitated by using this model. The metadata obtained by using the technology of service-finder ontology [11] and seek crawl ontology [12] for all obtained information is stored in the database. All obtained information is arranged as this stored metadata by applying the interpretation to semantics.

The targeted information of an API differs between the related work and our research. In related work, information of an API is collected for WSDL. We think that data forms other than WSDL are important. So, all the data forms are targets in our research. Moreover, Steinmetz applies meaning to all obtained information by manual work. However, it is better to apply meaning to all obtained information automatically. The Steinmetz study can solve certain problems; however, new problems of meaning arise. Thus, we introduce profile information without the meaning of web service, and this approach is different from related works.

Solution Design

In this section, our solution design is discussed, and our solution consists of two parts: profile information and I/O data interface. The first part is profile information for users who want to use API for mashup, and that is the information of using APIs. The second part is I/O data interface. This interface is used to accumulate and classify I/O data in APIs.

Profile Information. In the part of using profile information of users, the profile information used for mashup is shown in Table 1. First of all, profile information that is received from an arbitrary developer who uses mashup is assumed to be a standard development time. Next, information of the API used from the web service created is extracted. At the end, the information is classified further based on information on the web service used for mashup. As a result, API used for mashup can sequentially be automatically classified from the latest one when the developer designs the mashup. By using the profile information, we enable classification of the API from the latest one sequentially. In addition, a user is enabled to understand the development setting intuitively using the development language.

Table 1 Profile information

Profile information	Advantage
Development language	Whether API operates under what environment is understood.
API used to develop	When other developers want to use the API, it becomes a reference as a balance with another.
Information on the web service made by the mashup	The developer understands for what purpose it was made.
Development time	The developer understands the time of information.

I/O Data Interface. We introduce an I/O data interface to adapt various APIs. When a user wants to mashup using an API, the API is exchanging of a lot of data. Data exchanged here are not only location information and weather information, but also numerical values and character strings with meaning. These data used when a developer does mashup are collected in the database once. In the database, a lot of I/O data are collected and classified with the common part automatically. For example, certain data used for doing mashup have significant information such as “figure” and “location information”. Thereby, the API is classified by I/O data. The composition is shown in Figure 2. By using this method, we will enable classification of an API using I/O data automatically. This method differs from mashup portal sites because of automatic classification.

Conclusion

In this study, we introduced two methods as a user-oriented web service search technique. In the method using profile information, we search information of an API to use profile information. By using profile information, we enable classification of an API from the latest one sequentially. Also, by using I/O data for an API, we enable classification of the API automatically. However, neither method is sufficient; thus, we will combine the two methods in the future so that they complement each other. Concretely, I/O data classified on the server side is combined with information obtained by the method of using profile information. In doing so, we will use the two methods to make a better user-oriented web service search technique. As mobile phones develop, mashup is expected to spread to more users in the near future. To correspond to the future of mashup, we should create a user-oriented web service search technique that works on mobile phones.

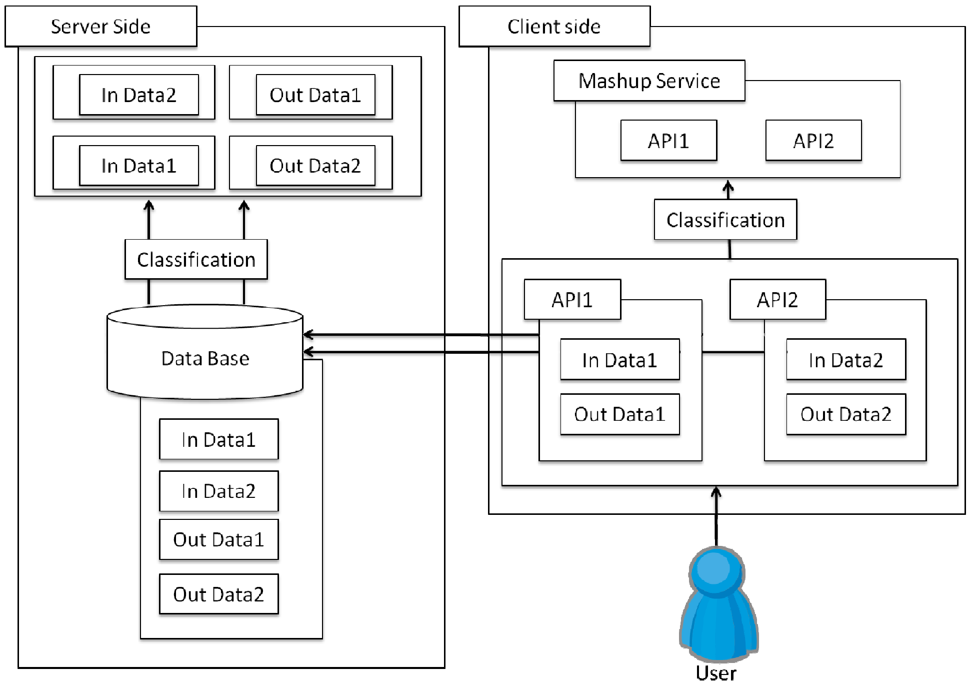

Fig. 2. I/O data interface

References

[1] Google.com, http://www.google.com/
[2] Amazon.com, http://www.amazon.com/
[3] mixi.jp, http://mixi.jp/
[4] yahoo.com, http://www.yahoo.com
[5] Programmableweb, http://www.programmableweb.com/
[6] Mashupedia.jp, http://www.mashupedia.jp/
[7] Apple.com, http://www.apple.com/
[8] Android OS, http://www.android.com/
[9] Web service search on large scale, Nathalie Steinmetz, Proc. of the International Conference on Service Oriented Computing (ICSOC2009), pp.437-444, November 24-27, 2009.
[10] Information Extraction: Algorithms and Prospects in a Retrieval Context, Marie-Francine Moens, Springer, 2006.
[11] Machine learning in automated text categorisation, Fabrizio Sebastiani, Journal of ACM Computing Surveys (CSUR), Volume 34, Issue 1, pp.1-60, 1999.
[12] Ontology-based feature aggregation for multi-valued ranking, Nathalie Steinmetz, Holger Lausen, Manuel Brunner, Iva ́n Martinez, and Alex Simov, Proc. of the ICSOC (Int'l Conf. on ServiceWaveWorkshop2009), pp.258-268, 2009.