Leveraging Incrementally Enriched Domain Knowledge to Enhance Service Categorization

Jia Zhang, Carnegie Mellon University, Silicon Valley, USA

Jian Wang, State Key Lab of Software Engineering, Computer School, Wuhan University, China

Patrick C.K. Hung, University of Ontario Institute of Technology, Canada

Zheng Li, State Key Lab of Software Engineering, Computer School, Wuhan University, China

Neng Zhang, State Key Lab of Software Engineering, Computer School, Wuhan University, China

Keqing He, State Key Lab of Software Engineering, Computer School, Wuhan University, China

ABSTRACT

This paper reports the authors' study over an open service and mashup repository, ProgrammableWeb, which groups stored services into predefined categories. Leveraging such a unique structural feature and hidden domain knowledge of the service repository, they extend the Support Vector Machine (SVM)-based text classification technique to enhance service-oriented categorization. An iterative approach is presented to automatically verify and adjust service categorization, which will incrementally enrich domain ontology and in turn enhance the accuracy of service categorization.

Keywords: *Enriched Domain Knowledge, Mashup Repository, Open Service, ProgrammableWeb, Service Categorization, Support Vector Machine*

INTRODUCTION

The ultimate goal of cloud computing is to enable everything as a service (XaaS) (NIST, 2011), where Software as a Service (SaaS) is one core objective. While software being published as universally accessible Web services, users can leverage existing services and quickly compose new value-added business processes and services. However, as cloud has become an unprecedented driving factor to encourage people to publish and share software as services, how to effectively and efficiently discover interested services from a "cloud" of resources remains a big challenge.

DOI: 10.4018/jwsr.2012070103

Copyright © 2012, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

One major technique is to establish service registries (Zhang et al., 2007) as centralized "service yellow pages" to help users find interested services. Earlier Universal Description, Discovery, and Integration (UDDI) registries are going out of date, however. Two major reasons are their tight binding to SOAP/WSDL services and their over standardization. In recent years, REpresentational State Transfer (REST) service, a light-weight HTTP Request/Response-based service style, has rapidly emerged and caught significant momentum (Pautasso et al., 2008). Thus, many non-UDDI service registries have been developed. Among them, The ProgrammableWeb (PW, http://www.programmableweb.com, acquired by Alcatel-Lucent in 2010) has become a popular one.

Without adopting the heavy UDDI standard, ProgrammableWeb provides a repository that allows people to publish reusable Web services in various formats (protocols including REST and SOAP), called Web APIs. Meanwhile, PW allows people to publish API-based applications, called mashups. A mashup represents a value-added business process leveraging one or more existing APIs published in PW. Such a light-weight service repository has attracted extensive attention. Since its inception in late 2005, the number of services published at PW has increased rapidly. Up to September 7, 2012, 7190 services and 6,763 mashups have been published at PW. Among the published services, 70% are REST services, 21% are SOAP services, 5% are JavaScript services, and 2% are XML-RPC services. Since APIs at PW represent reusable service components, throughout this paper, we will use the terms *API* and *service* interchangeably.

As the number of services accumulates at PW, it is important to facilitate users in querying and finding interested services (Gomadam et al., 2008). However, the current querying power at PW is limited. At publishing time, service providers are allowed to attach some user-defined name tags. Unlike UDDI that intends to regulate a comprehensive ontology system, ProgrammableWeb adopts a straightforward strategy. Every service is manually categorized into one of a preset list of domains (68 domains up to September 7, 2012) (Arabshian et al., 2012). The assigned domain name and provider-defined tags associated with the service are combined to support keyword-based search function.

Such an API search mechanism may cause confusion and decrease search accuracy. First, the manual process of service categorization may not be accurate. As a matter of fact, API "ShowMyIP" was originally classified in domain "Mapping"; and was moved to domain "Internet" later. In the metadata of the API, its description, summary and tags contain some representative keywords of domain "Internet" such as "IP" and "Internet." Second, it may be difficult to decide one single domain for some APIs, because some predefined domains overlap with each other conceptually. For example, domains Travel, Transportation, and Weather share many common concepts. For another example, the aforementioned API "ShowMyIP" does relate to the category "Mapping" in addition to the category of "Internet." Third, PW presets a special domain named "Other" and a significant number of services are found left in the category. Currently, 199 services are listed in the category of "Other," which is the top 14th category with the most number of services (over the entire 68 preset domains). Fourth, user-defined tags may be *ad hoc* and inconsistent, and sometimes lack of tag (Gomadam et al., 2008), cannot effectively help users find their interested services.

Table 1 shows some motivating examples. The second column shows the category name assigned to the Web API (whose name is in the first column) by ProgrammableWeb. However, as shown in the third column, our study indicates that these Web APIs should belong to several categories (domains). The names of the Web APIs even imply such cross-relationships. For example, users should be able to find the API "BestParking" from the Travel, Transportation, or Mapping categories. (The numbers represent the similarity between a Web API to a corresponding category. For another example, the API "StrikeIron Address Distance" is listed in

Copyright © 2012, IGI Global. Copying or distributing in print or electronic forms without written permission of IGI Global is prohibited.

22 more pages are available in the full version of this document, which may be purchased using the "Add to Cart" button on the product's webpage:
www.igi-global.com/article/leveraging-incrementally-enriched-domain-knowledge/74706?camid=4v1

This title is available in InfoSci-Journals, InfoSci-Journal Disciplines Computer Science, Security, and Information Technology. Recommend this product to your librarian:
www.igi-global.com/e-resources/library-recommendation/?id=2

Related Content

Service Portfolio Measurement: Evaluating Financial Performance of Service-Oriented Business Processes
Jan Vom Brocke (2007). *International Journal of Web Services Research (pp. 1-32).*
www.igi-global.com/article/service-portfolio-measurement/3097?camid=4v1a

Quality Models for Multimedia Delivery in a Services Oriented Architecture
Krishna Ratakonda (2009). *Managing Web Service Quality: Measuring Outcomes and Effectiveness (pp. 48-73).*
www.igi-global.com/chapter/quality-models-multimedia-delivery-services/26074?camid=4v1a

Advances in Privacy Preserving Record Linkage
Alexandros Karakasidis and Vassilios S. Verykios (2011). *E-Activity and Intelligent Web Construction: Effects of Social Design (pp. 22-34).*
www.igi-global.com/chapter/advances-privacy-preserving-record-linkage/53271?camid=4v1a

Enhancing the Testability of Web Services

Daniel Brenner, Barbara Paech, Matthias Merdes and Rainer Malaka (2009).
Managing Web Service Quality: Measuring Outcomes and Effectiveness (pp. 223-244).
www.igi-global.com/chapter/enhancing-testability-web-services/26081?camid=4v1a