An Approach to Engineer Communities of Web Services: Concepts, Architecture, Operation, and Deployment

Zakaria Maamar, Zayed University, UAE
Sattanathan Subramanian, INRIA Saclay-Île-de-France, France
Philippe Thiran, University of Namur, Belgium
Djamal Benslimane, Claude Bernard Lyon 1 University, France
Jamal Bentahar, Concordia University, Canada

ABSTRACT

This article presents an approach that provides the necessary assistance to those who are in charge of engineering communities of Web services. Current practices indicate that Web services providing the same functionality are gathered into one community, independently of their origins and the way they carry out this functionality. The provided assistance manifests itself with the concepts to use, the architecture to select, the operations to script, and the deployment to track. Two protocols frame the interactions in an environment of communities of Web services namely the Web Services Community Development Protocol and the Contract-Net Protocol. The former manages a community in terms of Web services attraction/registration/withdrawal to/with/from this community. The latter satisfies users’ needs in terms of Web services selection/contracting/triggering. Finally, the article presents a prototype illustrating the engineering approach with focus on Web services attraction.

Keywords: Community, Engineering, Web Service

INTRODUCTION

For the World Wide Web Consortium, a Web service “is a software application identified by a URI, whose interfaces and binding are capable of being defined, described, and discovered by XML artifacts and supports direct interactions with other software applications using XML-based messages via Internet-based applications”. For the last few years, the development pace of Web services has been spectacular (Benslimane, 2007, DPD; Daniel, 2005; Dustdar, 2005). On the one hand, several standards have been developed to deal with for example
Web services definition, discovery, and security (Andrews, 2003; Curbera, 2002). On the other hand, several projects have been initiated such as Web services composition, personalization, and contextualization (Baresi, 2007; Medjahed, 2007). These standards and projects have usually a common concern: Web services composition. Composition addresses the situation of a user’s request that cannot be satisfied by any single, available Web service, whereas a composite Web service obtained by combining available Web services may be used.

Nowadays, competition between businesses does not stop at goods, services, or software products, but includes as well systems that offer the most recent and accurate information. For example, Google and Yahoo are both search engines. The common practice is to bind to one of the engines according to various factors like reliability, efficiency, previous experiences, financial charges, etc. Web services are definitely not excluded from this competition. Independent providers develop several Web services that could offer the same functionality such as currency exchange. It is reported in (Bui, 2005) that although Web services are heterogeneous, the functionalities these Web services offer are sufficiently well defined and homogeneous enough to allow for market competition to happen. To ease and improve the process of Web services discovery in an open environment like the Internet, we suggested in (Benslimane, 2007; Maamar, 2007; Subramanian, 2007) along with other researchers in (Benatallah, 2003; Medjahed, 2007; Medjahed, 2005) to gather similar Web services1 into groups known as communities. The notion of group/community/cluster highlights the importance of developing guidelines that would permit the management of Web services to be now parts of communities. Although Web services are investigated in various research projects (Anderson, 2006; Foster, 2006; Mrissa, 2008; Younas, 2006) these guidelines still lack and hence, examining the following elements would be deemed appropriate: (1) how to initiate, set up, and specify a community, (2) how to specify and manage the Web services in a community, and (3) how to reconcile conflicts within a community and between communities?

A community of Web services is dynamic by nature: new Web services join, other Web services leave, some Web services become temporarily unavailable, some Web services resume operation after suspension, just to name some. All these events need to be closely monitored and followed up, otherwise conflicts arise. For example, if a Web service left a community without prior notice, its peers would continue to assume it is still in this community. Moreover, Web services do not always exhibit a cooperative attitude when they become members of a community. First, they can compete on common computing resources, which may affect their performance scheduling. Second, they can announce misleading information (e.g., non-functional details) to enhance their participation opportunities in composite Web services. Last but not least, they can become malicious when they try to alter other Web services’ data or behaviors.

Designing, developing, and managing communities of Web services seem to be a cumbersome process on designers/developers, who would definitely benefit from an approach that would assist them engineer such communities. For this purpose, this assistance needs to shed the light on 4 elements: concepts to use, architecture to select, operation to script, and deployment to track. The rest of this article proceeds as follows. Section 2 consists of three parts dedicated to concept definition, architecture of a community environment, and functioning of this architecture, respectively. Section 3 details the internal structure of the two types of Web services that populate a community. A prototype simulating community functioning is presented in Section 4. Sections 5 and 6 are about related and future work, respectively. Conclusions are drawn in Section 7.
The Measurement of Electronic Service Quality: Improvements and Application
www.igi-global.com/article/measurement-electronic-service-quality/1909?camid=4v1a