ABSTRACT

Due to the increase of published Web Services (WSs), finding the suitable WS that satisfies the user goals among discovered WSs still needs deep investigations. Certainly, QoS requirements represent a more appropriate and decisive factor to distinguish similar WSs. A lot of research efforts in this direction have been made but are still limited due to the complexity and diversity of QoS constraints. The novelty of our approach lies in its simplicity since it is based on WS Popularity Score (WSPS). This score is computed using an algorithm based on both user’s requirements and quality measures of each discovered WSs such as pertinence, age, frequency, etc. The paper reports a validation of the proposed algorithm, its implementation and evaluation through Information Extraction (IE), in order to illustrate, and assess the convenience of our approach.

Keywords: QoS, WS Discovering, WS Popularity, WS Popularity Score, WS Ranking, WS Selection

INTRODUCTION

WSs became the main source for constructing software systems over the Internet. Because of the increasing number of published WSs, several WSs may share similar functionalities. However, if many functionally-equivalent WSs exist, it becomes a fastidious task to discover and thereafter select the appropriate one. In fact, this issue has gained a great research attention in literature, and motivated researchers to propose several mechanisms that will select the appropriate WS for the requester based on his needs.

To distinguish between similar WSs, it is mandatory to consider non functional parameters, i.e. Quality of Service (QoS), to find out the efficient WS (Chen, Zheng, Liu, Huang, & Sun, 2013). However, using QoS parameters raises other challenges such as how to specify, store and rank WSs?

A number of research efforts have studied this open issue by suggesting some solutions (Li et al., 2012; Wang et al., 2010; Zheng et al., 2011; Manikrao, & Prabhakar, 2005). However, most...
of these solutions did not address the problem seamlessly without extending WSs registry or WSs languages such as WSDL for instance, did not provide a WS ranking algorithm, and/or not integrate user feedback into WS selection process, etc.

Quality of Service (QoS) determines the service availability, usability, pertinence and utility, which influence the popularity of the service. This ascertainment motivates us to look, in this paper, at various WS QoS metrics and requirements, and how these metrics can may influence the published WSs consumption. As a result, we will propose a new flexible approach using an efficient algorithm based on WS Popularity Score (WSPS). This score consists of ranking WS, including simple, composite and semantic WSs, by using metrics such as usability, age, pertinence, etc. In addition we opt to assign weight to each metric as a filter to select the more adequate one since one metric may have high importance over the other according to user importance.

Recall that the popularity notion is reminiscent of the PageRank (Page, Brin, Motwani, & Winograd, 1999; Tyagi, & Sharma, 2012) popularity score for pages of the World Wide Web or, more generally, for nodes in a graph. Popularity has gained more attention in several domains such as web page search, Information Extraction, web mining, etc. In the case of web search domain, popularity focuses on external and internal link popularity (Pandia Search Central, 1998). Internal link popularity refers to the number of links to a website from web pages that belong to it. While external link popularity represents the number of links from outside sources that lead back to the particular website. Therefore, a website with high link popularity has what is called “link cardinality” or “link superiority”. The website then is considered an informative one, and has a high rank on web search engines.

Based on the best results attained in web popularity, and in order to propose an efficient approach for WS popularity, we should efficiently study and experiment this potential solution. More specifically, we must start by defining quality criteria or quality metrics representing parameters to compute such a score. Consequently our main contribution in this paper is to define fundamental QoS metrics related to WSs, elaborate a refined QoS WS taxonomy, and propose the required logical formula using these metrics to calculate WS popularity score of each discovered WS.

An interesting research effort presented in Maamar, Santos, Wives, Badr Faci, and de Oliveira (2011) consists of facilitating WS discovery and composition by integrating social networking at WS level. In their paper, the authors tried to model WS relations by acquiring inter-actions between services in a social network. This model provided a grouping WSs mechanism that establishes the degree of similarity between WSs. Consequently this will be helpful to organize and to extract sequences of anterior successful interactions for future needs of the user requester.

Recently Maamar et al. (2014) proposed to realize a social ecosystem of WSs by the actors (WS providers, WS actors) and the interactions that occur between these actors during this ecosystem management. Those parameters (actors, interactions) create and maintain relationships with others by managing different types of interactions, for example making WSs in social networks of WSs; supporting users seeks advices from existing members in a social network of consumers, and combining social networks of consumers and of WSs to achieve users’ requests. This paper defines how to identify the actors to build the ecosystem that’s will be used to describe and discover WSs.

This article is organized as follows. The next section outlines related work referring to WS selection by using QoS. In the third section we define and explain in more details WS popularity metrics, in addition we define for each QoS metric a formula that evaluates such metric. In the fourth section we present how we calculate the WS Popularity Score for simple, composite and semantic WS. Before concluding we describe the implementation of our approach in DIVISE framework and we evaluate it using IE WSs in the fifth and sixth sections.
Web Effort Estimation Using Case-Based Reasoning
www.igi-global.com/chapter/web-effort-estimation-using-case/7165?camid=4v1a