Procedia Computer Science

Volume 88, 2016, Pages 272–276

7th Annual International Conference on Biologically Inspired Cognitive Architectures, BICA 2016

Intelligence Search Engine and Automatic Integration System for Web-Services and Cloud-Based Data Providers Based on Semantics

Artyom Chernyshov, Anita Balandina, Anastasiya Kostkina and Valentin Klimov

National research Nuclear University "MEPhI", Moscow, Russian Federation

a-chernyshov@protonmail.com, anita.balandina@gmail.com, anakost@bk.ru, vvklimov@mephi.ru

Abstract

During the last years the concept of publicity accessible services and application interacting to one another has become quite popular. In this connection the majority of specialists and especially business integrators and B2B (business to business) application users often experience some diffi-culties while running necessary services and application. So that we are going to introduce a new approach based on RESTful web-services composition and a program system which would au-tomatically search and unify semantic web-services. The basis of this method is usage WSMO-lite (Web Service Modeling Ontology) for semantic descriptions and different methods of web-services composition depending on client demands. We also are going to compare the developing system to its analogues.

Keywords: Web-services, Integration, Web-service composition, SaaS.

1 Introduction

According to numerous studies in the IT area, the number of different software systems, which use web technologies, such as web-services is constantly growing. According to W3C defini-tion, web-service is a software system designed to support interoperable machine-to-machine in-teraction over a network [1]. In general, there are three main instances that interact within a web service: service requestor, service provider, service broker. As the service is developed, performer (service provider) registers it in the directory (service broker) where it can be found by the potential customers (service requestor). As the customer found the appropriate service in the directory, he can import the WSDL specification for further usage in the process of development his software. WSDL describes the format of requests and responses exchanged between the customer and per-former in the process of work. The following standards are used to ensure interoperability:

Selection and peer-review under responsibility of the Scientific Programme Committee of BICA 2016

doi:10.1016/j.procs.2016.07.434

1. XML: Extensible Markup Language, is used to storage and transfer structured data;
2. SOAP: a messaging protocol based on XML;
3. WSDL: language which describes the external web service interface based on XML;
4. UDDI (Universal Discovery, Description and Integration) is the directory of web-services and information about the companies that provide web-services into the public domain, or specific companies. By the moment, UDDI in not very common, it is mostly spread in small corporate networks.

In the 2010 it was offered a new concept of the Internet technologies development which was called web 3.0 or also known as semantic web. The technical part of the Semantic Web consists of a family of standards for description languages, including XML, XML Schema, RDF, RDF Schema, OWL, as well as some others. We consider only the last three of mentioned above.

1. RDF (Resource Description Framework) is a simple way to describe the instance data in the format of the subject-object relation, where any member of the trio uses only resource identifiers.
2. RDF Schema defines a set of attributes, such as rdfs: Class, to define new types of RDF-data. The language also supports an inheritance relationship type rdfs: subClas-sOf.
3. OWL (Web Ontology Language) expands the description of new types (in particular, the addition of transfers), and also allows us to describe new types of RDF Schema data in terms of the existing ones (for example, to determine the type, which is the in-tersection or the union of the two existing).

The important part of the semantic web is the concept of the semantic web services. In gen-eral, semantic web services could be defined as finished program elements with uniquely de-scribed semantics, which could be accessed via the Internet and are suitable for automated search, composition and performance based on their semantics. The major difference between semantic web service and common web-service if that user can access not only web interface description (normally WSDL description) in terms of data transmitted service, return values and generated errors, but also its semantic description, i.e. what this service does, its domain, destina-tion and so on. There are a number of semantic web services description language: SAWSDL, OWL-S, WSMO. All of these languages are oriented to interact with the WSDL. In this paper we consider only WSMO-lite [2], which will be used in our system.

2 Related Works

The most part of projects in this field is connected with developing search engine for SOAP web services. However, there are some projects for RESTful web services composition (also named RESTful service composition or RSC) and they are based on cloud computing.

The CHOReOS Enactment Engine is a novel, extensible, open source middleware system, de-veloped in the context of the CHOReOS project, that provides a fully automated deploy-ment process for service compositions. The CHOReOS Enactment Engine offers developers ac-cess to service deployment using the Platform as a Service (PaaS) model [3], relying on an Infrastructure as a Service (IaaS) [3] provider to support the virtualized target environment. In the cloud PaaS model, application developers do not need to care about the virtualized in-frastructure and can focus on application development. The architecture of The CHOReOS En-actment Engine concludes the next components or major parts: Infrastructure Provider, Config-uration Agent, EE client and Enactment Engine. Infrastructure Provider (IaaS) plays the im-portant role in creating and destroying virtual machines (also called nodes) in cloud computing envinroment; it's worth to notice, that Enactment Engine depends on it. In its turn, EE (PaaS) installs Configuration Agent in each cloud node and deploys choreography services according to the specification (EE client) sent by the deployer. EE client is a script, written by deploy-ers, that specifies the choreography deployment and invokes the EE to trigger the deploy-ment process. Configuration agent manages and runs scripts that

implement the process of configuring operating systems, installing required middleware, and finally deploying the ser-vices [4].

The EE enables deployment automation by providing a RESTful API that receives the declarative description of the service composition to be deployed and returns information describing the deployment outcome. The use of a declarative description of composition have been used previously in the context of component-based system deployment. A service composition may encompass services belonging to multiple organizations. The EE has two main mechanisms to cope with this situation. The first uses a service specification attribute to define under which "cloud account" the service will be deployed (and billed). The second mecha-nism is used when an organization performs the deployment of a composition encompass-ing both services belonging to the organization itself and also existing third-party services. This situation can be modeled in the composition specification, so the EE will deploy only the services belonging to the organization and bind them to the existing third-party services [4].

ubiREST specifically defines a layered communication middleware supporting RESTful Ser-vices while exploiting nowadays ubiquitous connectivity. ubiREST aims at providing a communi-cation layer enabling RESTful services within ubiquitous networking environments . To this ex-tent, ubiREST aims at effectively exploiting all the diverse network technologies at once to create an integrated multi-radio networking environment, hence offering network-agnostic connectivity to services. On the other hand, ubiREST aims to provide proper programming abstractions ena-bling service-oriented applications to adapt and evolve at run time [5]. Taking into account the principles of REST architectural style, as well as the difficulties arising in the framework of ubiq-uitous networking environments, developers have created own style named Pervasive REST and based on an existing style REST, which refines REST to specifically address ubiquitous network-ing environments, while keeping REST principles unaltered.

iServe is an open source platform that unifies the publication, discovery, and use of Web Ser-vices (e.g., WSDL services), Web APIs (e.g., Twitter API, Flickr API, etc), and Things available on the Web. iServe is the platform to provide this level of support homogeneously across types of devices and software interfaces, providing a convenient one-stop-shop for the location and use of distributed software building blocks as necessary for creating novel Web and IoT application. [6].

3 System Description

Analyzing the up-to-date requirements applicable to B2B applications by the experts in the area, we highlighted the most important ones, which are listed below.

1. Search and discovery of web-services. This feature is essential for huge business corpora-tions, which use great number of different web-services in their industry as system integrators normally have to search or insert amendments in them manually for further work.

2. Middleware provider. This requirement is important for further functionality of web-services and communication between them.

3. Secured encryption of transmitted data. The feature is of critical importance for companies, whose data and information are confidential, so using unsecured software may deface the activi-ty of organization or its customers.

4. Composition of web-services. As was already mentioned, huge companies have huge num-ber of services, some of which should perform together. The automated search and composition of web-services may help to reduce the complexity of this task.

As it can be noticed, the systems listed in the paragraph 2 only partially satisfy these require-ments. In this connection, we have started the development of the automated integration system for web-services.

Pattern of work of our system is described below.

At first step user through web-interface specifies search and integration parameters, such as range of IP-addresses, specific ports, supported functions and operations, and so on. After that search subsystem attempts to find all web-services with defined parameters. Also, this subsystem tries to connect with external Internet cloud data providers and services, such as Cartographic and geographic services.

At second step, with use of integration algorithm, integration subsystem tries to compose founded web-services with each other and external services. Integration algorithm extracts de-scription of service if this one is provided or, in case of RESTful web-services, it may try to formu-late by itself based on resource identity.

At final step, our system provides recommendation for using of data and message transition middleware, in order to make connections between web-services more reliable and confidence. To ensure that, system encrypts data with AEAD block cipher mode of operation. This middleware can also work as secured cache manager and monitoring tool.

Overall, usage of our system will consent to line up secured environment of web-service data streams.

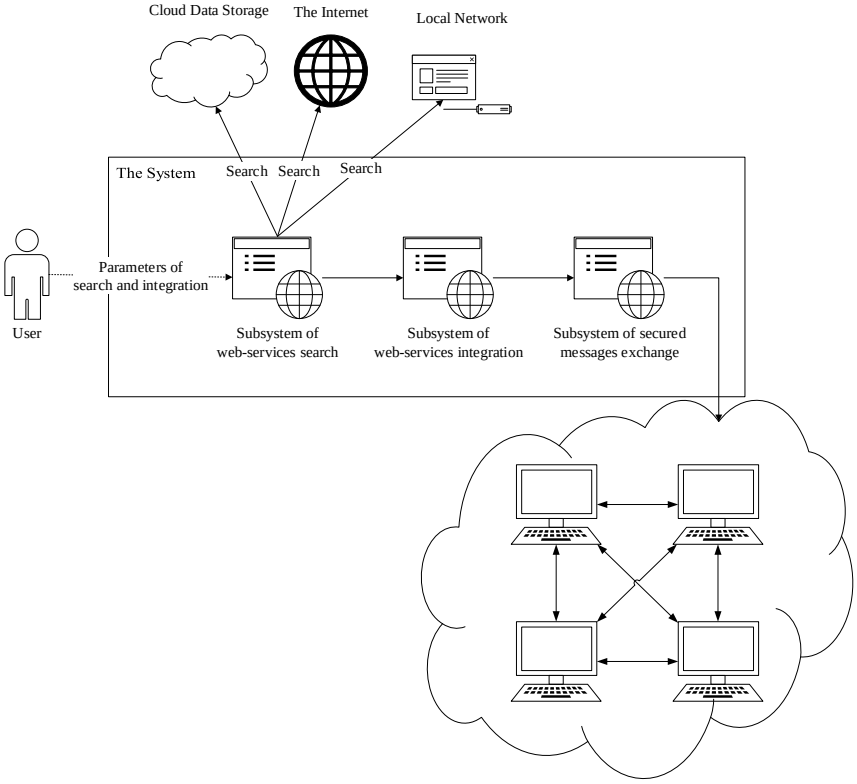

Figure 1: Architecture of the System.

Working scheme of the system is presented at the figure 1.

Acknowledgements

This work was supported by Competitiveness Growth Program of the Federal Autonomous Educational Institution of Higher Professional Education National Research Nuclear University MEPhI

(Moscow Engineering Physics Institute). The funding for this research was provided by the Russian Science Foundation, Grant RSF 15-11-30014.

References

[1] Web Services Glossary. (Retrieved 2011). *W3C.* Available at: https://www.w3.org/TR/2004/NOTE-ws-gloss-20040211/

[2] D. Romana,, J. Kopeckýb, T. Vitvarc, J. Dominguee, D. Fensel. WSMO-Lite and hRESTS: Lightweight semantic annotations for Web services and RESTful APIs. (2015). *J. Web Semant*, 31, 39–58.

[3] Q. Zhang, L. Cheng, and R. Boutaba, Cloud computing: state-of-the-art and research challenges. (2010). *Journal of Internet Services and Applications*, 1, 1, 7–18.

[4] L. Leite, C.E. Moreira, D. Cordeiro, M.A. Gerosa, F. Kon. Deploying large-scale service compositions on the cloud with the CHOReOS Enactment Engine. (2014). *IEEE 13th International Symposium on Network Computing and Applications (NCA),* 121 – 128.

[5] M. Caporuscio, M. Funaro, C. Ghezzi, and V. Issarny. UbiREST: A RESTful Service-Oriented Middleware for Ubiquitous Networking. (2014). *Springer New York*, 475-500.

[6] Pedrinaci, C., Liu, D., Maleshkova, M., Lambert, D., Kopecky, J., and Domingue, J. iServe: a Linked Services Publishing Platform. (2010). *In Proceedings of Ontology Repositories and Editors for the Semantic Web at 7th ESWC*, 71-82.