IDEA GROUP PUBLISHING

701 E. Chocolate Avenue, Suite 200, Hershey PA 17033-1240, USA
Tel: 717/533-8845; Fax 717/533-8661; URL-http://www.idea-group.com

ITJ2731

RAWS Architecture:
Reflective and Adaptable Web Service Model

Javier Parra-Fuente, Pontifical University of Salamanca, Madrid, Spain
Salvador Sánchez-Alonso, Pontifical University of Salamanca, Madrid, Spain
Oscar Sanjuán-Martínez, Pontifical University of Salamanca, Madrid, Spain
Luis Joyanes-Aguilar, Pontifical University of Salamanca, Madrid, Spain

ABSTRACT

Web services are static components, which implies that before a change in their structure or behavior can be made, the source code — or a decoder of compiled code — is needed. The full process consists of three steps: editing and modifying the source code, compiling it again, and redeploying it in the server. Reflection, a powerful tool for the adaptation of applications at runtime, may help in creating more flexible and dynamic Web services. In this paper, we introduce RAWS (Reflective and Adaptable Web Service) Architecture, a Web service design model based on a reflective architecture multilevel. RAWS allows both the dynamic modification of the definition and implementation structure of the Web service, and the dynamic modification of the Web service behavior in order to change the existing code or to add new functionalities. All these dynamic modifications are performed directly on the code during execution, with no need to have the Web service source code. We also introduce an automatic generator of the reflective infrastructure that is needed for the implementation of the RAWS architecture. This infrastructure will make possible that any Web service can automatically behave like a Reflective and Adaptable Web Service.

Keywords: *adaptability; architecture; behavioral reflection; customization; introspection; maintainability; structural reflection; Web service*

INTRODUCTION

Web services are programmable components of applications that use SOAP (Gudgin et al., 2003) as an access protocol, regardless their client and component technology (a drawback in DCOM) and regardless the language in which both communication ends are written (a drawback in RMI). SOAP generally uses the HTTP transport protocol, over the port 80 for request/response, thus crossing corporate firewalls (a drawback in CORBA or DCOM) and facilitating the interoperability of applications that work with different technologies.

Currently, the modification of a Web service implies the availability, edition, recompilation and redeployment of the source code. Depending on the application server, the deployment task can be a simple or a complicated one. If the application

server supports the dynamic load of applications, then the deployment will be simple task; but if that is not the case, it will imply to stop the execution of the Web service, replacing the old version with the new one, and deploying the new version.

Reflection is a property of computational systems that allows them to reason and act by themselves and to modify their behavior (Maes, 1987). Although this concept has been successfully applied to other fields such as distributed systems (Ledoux, 1999; McAffer, 1995), concurrent programming (Masuhara, Matsouka, & Yonezawa, 1993), aspect-oriented programming (Pawlak, Duchien, & Florin, 1999; Tanter et al., 2003), and etcetera, its application to Web service design has not been addressed yet.

Reflection can be applied to Web services in order to enhance their adaptability and flexibility. We propose in this paper a Web service reflective architecture, RAWS, which allows one to dynamically modify a Web service during its execution.

In this paper, we introduce the basic concepts of reflection that will be applied to Web services (the introspective characteristics and the analysis of the structural and behavioral reflection of the Web service), the architecture model of a reflective and adaptable Web service, and the automatic generation mechanism to obtain the reflective infrastructure needed for a Web service to be dynamically adaptable.

REFLECTIVE WEB SERVICES

In general terms, reflection in a system can be classified into three groups, depending on the information that the system can reflect:

- *Introspection* (Foote, 1992): the system is able to observe and reason on the system elements, but it is unable to modify them.
- *Structural Reflection* (Foote, 1989): the system can enquire and modify its structure at runtime.
- *Behavioral Reflection* (Ferber, 1989): the system can manipulate and modify its behavior.

The RAWS architecture will prove how all the aforementioned kinds of reflection can be successfully applied to Web service design. Reflection is usually represented, as in *Figure 1*, by a two-level architecture: a base level that contains the modules that solve the problem, and a meta-level containing the representation of the base level. An application is represented in the base level and can be manipulated by the meta-level. Both levels are joined together by a causal connection (Smith, 1984), so that the changes brought about in the base level are reflected in the meta-level.

On the other hand, in order to manipulate the information of the meta-level, the computational behavior of the base level object is transformed into data. This process is called reification (Smith, 1982). The reificated information makes up the meta-information, thus allowing the reflective behavior.

The association of both the base and meta-levels and the implementation of reflection can take place in two ways: (a) *explicitly*, if the base level object activates the reflection, or (b) *implicitly*, if the system activates the meta-object.

As already stated, the abovementioned architecture can be applied to Web service design in order to enhance flexibility and adaptability. In the case