Chapter 1

Service-Oriented Computing: From Web Services to Service-Oriented Components

Stéphanie Chollet
Laboratoire d’Informatique de Grenoble, France

Philippe Lalanda
Laboratoire d’Informatique de Grenoble, France

Jonathan Bardin
Laboratoire d’Informatique de Grenoble, France

ABSTRACT

The visionary promise of Service-Oriented Computing (SOC) is a world-scale network of loosely coupled services that can be assembled with little effort in agile applications that may span organizations and computing platforms. In practice, services are assembled in a Service-Oriented Architecture (SOA) that provides mechanisms and rules to specify, publish, discover and compose available services. The aim of this chapter is to present the different technologies implementing the new paradigm of SOA: Web Services, UPnP, DPWS, and service-oriented component OSGi and iPOJO. These technologies have been developed and adapted to multiple domains: application integration, pervasive computing and dynamic application integration.

INTRODUCTION

The Service-Oriented Computing (SOC) represents today a solution of choice to deal with integration issues. SOC promotes the use of well-defined composition units – services – to support the rapid development of applications. The central objective of this approach is to reduce dependencies among composition units, where a unit is typically some remote functionality accessed by clients. By reducing such dependencies, each element can evolve separately, so the application is more flexible than monolithic applications (Papazoglou & Georgakopoulos, 2003), (Escoffier, 2007). Services are assembled in a Service-Oriented Architecture (SOA) that provides mechanisms and rules to
Service-Oriented Computing

specify, publish, discover and compose available services. Depending on the target application domain, different protocols and mechanisms are used to implement SOA. For instance, Web Services (www.w3c.org) are dominant to integrate IT applications. UPnP (UPnP Forum, 2008) or DPWS (Microsoft, 2006) are preferred in small area networks for devices. OSGi (OSGi Alliance, 2007) and iPOJO (Escoffier & Hall, 2007) are often used in centralized, embedded equipments.

The service-oriented approach has been implemented in many domains. Figure 1 illustrates the use of service technologies in different application domains.

This chapter is organized as follows. First, we present the Web Services technology and the requirements of application integration which are solved by Web Services. The second section deals with the pervasive computing and the technologies adapted to this domain: UPnP and DPWS. The third section introduces the Service-Oriented Computing in dynamic environment and the technologies implementing this new paradigm: OSGi and iPOJO. Before the conclusion, the fourth section proposes a comparison of the service technologies presented in the previous sections.

APPLICATION INTEGRATION

Web Services are the most popular and well-known technology for implementing Service-Oriented Architecture, both in the industry and the academia. In this section, we will present the main Web Services principles. Then, we will describe the three standards this technology uses: WSDL, UDDI and SOAP.

Web Services

The main purpose of Web Services was to render applications available via the Internet or from within an Intranet. Web Services comply with the service-oriented approach, meaning they can be described, published and discovered. A service provider can describe their service’s functional and non-functional characteristics in a WSDL (W3C, 2001) file and then registers the service description in an UDDI (OASIS, 2004) service registry. A client, or consumer, can search the UDDI registry for services that meet their requirements.

From the consumer perspective, a Web Service is a black box that provides no technical details on its implementation. The only available information includes the Web Service functionalities, certain properties, location and invocation instructions.

Consumers use the SOAP protocol (W3C, 2000) to communicate with Web Services. The Web Services architecture is illustrated by Figure 2.

This architecture hides the Web Services implementation complexity from the user. Hence, a Web Service may use other services (i.e. Web Services, or of other types) in order to provide its functionalities. The Web Service’s users remain completely unaware of such aspects. Certain coordination is required among the different service calls.

Web Services are a successful technology because of the loose-coupling it ensures between service consumers and service providers. As service providers describe offered service functionalities in a standard language, service consumers

Figure 1. Service technologies and application domains