Chapter 8.8
Ubiquitous Services and Business Processes

Alistair Barros
SAP Research, Australia

ABSTRACT

In the commercial world, the value of ubiquitous computing applications is proportional to the range of business services that can be accessed in device-consumptive ways. Services originate in legacy applications of organizations, and are developed and operated typically in heterogeneous environments. Service-oriented architecture (SOA), supported by a complex stack of Web services standards, addresses ways in which software components of diverse applications can be homogeneously interacted with and composed. Thus, SOA provides a crucial mechanism for making services accessible to ubiquitous computing applications. In this chapter, we shed light on what SOA entails, based on Web services interfaces and messaging, and service composition through single-party process orchestration and multiparty choreography languages. For the latter, concrete patterns are used to describe the capabilities of prospective standards. Ways in which SOA needs be extended to allow wider and more flexible service trading, typified in current developments through service marketplaces, are then discussed. Such extensions, we argue, converge with directions in ubiquitous computing through so-called ubiquitous service networks and service ecosystems.

INTRODUCTION

Over recent years, service-oriented architecture (SOA) has gained mainstream acceptance as a strategy for consolidating and repurposing legacy applications to dynamic market needs through self-contained, reusable and configurable services. As fostered through the *Web Services* standards, services, once in place, can interoperate with other services and be composed into long-running business processes spanning intra- and inter-organizational boundaries.

As Web services technologies mature, and commercial-scale SOAs shift from early adoption to mainstream development, a new revolution of service-orientation is emerging. Beyond the coordination of Web services in supply and value chains, a strategic trend for flexibly trading Web services into new and unforseen markets is

emerging. This development is shaping ubiquitous computing (UC) initiatives such as digital communities, service marketplaces and dynamic trading networks.

In this chapter, we provide insights into the ways in which Web services support the development of SOA, and how these need to evolve in the more ambitious UC setting.

In section "Service-Oriented Architecture Fundamentals", we present a background on SOA and details of *software interfacing* through Web services interface definition (WSDL) and *messaging* through simple object access protocol (SOAP). In section "Web Services Composition", we look at mechanisms for supporting the composition of Web services through business processes. Insights into intra-organizational, process *orchestration* are presented in terms of the widely referenced Web services business process execution language (WS-BPEL) and its support of workflow patterns. The inter-organizational process *choreography* layer is discussed through currently developing modelling concepts, service interactions and insights into a particular standards effort Web services choreography definition language (WS-CDL). In section "Scaling SOA for UC Applications", we chart the vision of SOA for UC applications, discussing how the recent developments of software-as-as-service applications are giving rise to wide-spanning UC service networks and service ecosystems. Accordingly, we discuss some open issues and future challenges which identify where present capabilities of SOA need scaling for UC. Finally, the chapter is concluded with a summary.

SERVICE-ORIENTED ARCHITECTURE FUNDAMENTALS

Fundamentals

Broadly speaking, there are two aspects of software architectures which impact the development and operation of applications. The first is software layering made possible by decoupling application functionality into separate and stand-alone parts known as software components (Szyperski, 2004). This fosters flexible reuse of otherwise unwieldy application monoliths, through independent deployment of individual components and their composition into more value-added service offerings. The second aspect is keeping applications focussed on business logic and relying on dedicated middleware to support platform functions such as naming and directory service, remote procedure calls, security, messaging, and persistence.

Distributed computing frameworks such as CORBA and, over recent years, J2EE and .NET based application servers have emerged in support of enterprise application integration (EAI) strategy. Through EAI, standard mechanisms are available for accessing and interacting with applications functions in different and heterogeneous environments in the enterprise's IT landscape. In addition, related functions across different applications, for example, for purchase order processing, are grouped to yield coherent and reusable business components.

With the proliferation of Web services, we nowadays speak increasingly of service-oriented architectures (SOA). Naturally the question arises as to whether Web services and SOA are substantially different from what has come before. As trite as it may sound, the answer is: yes and no. Understanding this lies at the heart of the whole posit of Web services. Consider collaborating partners (e.g., retailer, supplier, and shipper) in a supply chain with shared business processes spanning different applications operating in the separate IT environments. To coordinate the processes in the supply chain, different applications need to interoperate. However, the mechanism for interoperability is not obvious because of the variety of middleware products (for naming and directory service, remote procedure call and the like) operating at the different partners.