Bridging Together Mobile and Service-Oriented Computing

Loreno Oliveira
Federal University of Campina Grande, Brazil

Emerson Loureiro
Federal University of Campina Grande, Brazil

Hyggo Almeida
Federal University of Campina Grande, Brazil

Angelo Perkusich
Federal University of Campina Grande, Brazil

INTRODUCTION

The growing popularity of powerful *mobile devices,* such as modern cellular phones, smart phones, and PDAs, is enabling *pervasive computing* (Weiser, 1991) as the new paradigm for creating and interacting with computational systems. Pervasive computing is characterized by the interaction of mobile devices with embedded devices dispersed across *smart spaces,* and with other mobile devices on behalf of users. The interaction between user devices and smart spaces occurs primarily through services advertised on those environments. For instance, airports may offer a notification service, where the system registers the user flight at the check-in and keeps the user informed, for example, by means of messages, about flight schedule or any other relevant information.

In the context of smart spaces, *service-oriented computing* (Papazoglou & Georgakopoulos, 2003), in short SOC, stands out as the effective choice for advertising services to mobile devices (Zhu, Mutka, & Ni, 2005; Bellur & Narendra, 2005). SOC is a computing paradigm that has in services the essential elements for building applications. SOC is designed and deployed through *service-oriented architectures* (SOAs) and their applications. SOAs address the flexibility for dynamic binding of services, which applications need to locate and execute a given operation in a pervasive computing environment. This feature is especially important due to the dynamics of smart spaces, where resources may exist anywhere and applications running on mobile clients must be able to find out and use them at runtime.

In this article, we discuss several issues on bridging mobile devices and service-oriented computing in the context of smart spaces. Since smart spaces make extensive use of services for interacting with personal mobile devices, they become the ideal scenario for discussing the issues for this integration. A brief introduction on SOC and SOA is also presented, as well as the main architectural approaches for creating SOC environments aimed at the use of resource-constrained mobile devices.

BACKGROUND

SOC is a distributed computing paradigm whose building blocks are distributed services. Services are self-contained software modules performing only pre-defined sets of tasks. SOC is implemented through the deployment of any software infrastructure that obeys its key features. Such features include loose coupling, implementation neutrality, and granularity, among others (Huhns & Singh, 2005). In this context, SOAs are software architectures complying with SOC features.

According to the basic model of SOA, service providers advertise service interfaces. Through such interfaces, providers hide from service clients the complexity behind using different and complex kinds of resources, such as databanks, specialized hardware (e.g., sensor networks), or even combinations of other services. Service providers announce their services in service registries. Clients can then query these registries about needed services. If the registry knows some provider of the required service, a reference for that provider is returned to the client, which uses this reference for contacting the service provider. Therefore, services must be described and published using some machine-understandable notation.

Different technologies may be used for conceiving SOAs such as grid services, Web services, and Jini, which follow the SOC concepts. Each SOA technology defines its own standard machineries for (1) service description, (2) message format, (3) message exchange protocol, and (4) service location.

In the context of pervasive computing, services are the essential elements of smart spaces. Services are used for interacting with mobile devices and therefore delivering personalized services for people. Owning to the great benefits that arise with the SOC paradigm, such as interoperability, dynamic service discovery, and reusability, there is a strong and increasing interest in making mobile devices capable of providing and consuming services over wireless networks (Chen, Zhang, & Zhou, 2005; Kalasapur, Kumar, & Shirazi, 2006; Kilanioti, Sotiropoulou, & Hadjiefthymiades, 2005). The dynamic discovery and invocation of services are essential to mobile applications, where the user context may change dynamically, making different kinds of services, or service implementations, adequate at different moments and places.

However, bridging mobile devices and SOAs requires analysis of some design issues, along with the fixing of diverse problems related to using resources and protocols primarily aimed at wired use, as discussed in the next sections.

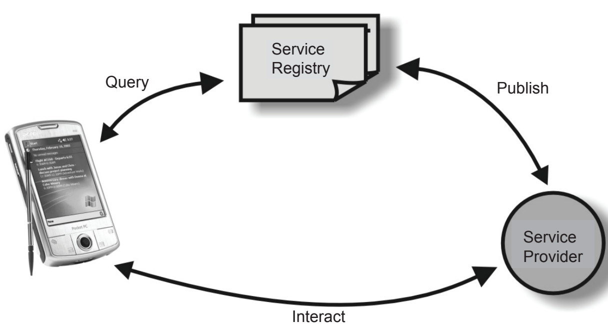
INTEGRATING MOBILE DEVICES AND SOAS

Devices may assume three different roles in a SOA: service provider, service consumer, or service registry. In what follows, we examine the most representative high-level scenarios of how mobile devices work in each situation.

Consuming Services

The idea is to make available, in a wired infrastructure, a set of services that can be discovered and used by mobile devices. In this context, different designs can be adopted for bridging mobile devices and service providers. Two major architectural configurations can be derived and adapted to different contexts (Duda, Aleksy, & Butter, 2005): direct communication and proxy aided communication. In Figure 1 we illustrate the use of direct communication.

Figure 1. Direct communication between mobile client and SOA infrastructure



In this approach, applications running at the devices directly contact service registries and service providers. This approach assumes the usage of fat clients with considerable processing, storage, and networking capabilities. This is necessary because mobile clients need to run applications coupled with SOA-defined protocols, which may not be suited for usage by resource-constrained devices.

However, most portable devices are rather resource-constrained devices. Thus, considering running on mobile devices applications with significant requirements of processing and memory footprint reduces the range of possible client devices. This issue leads us to the next approach, proxy-aided communication, illustrated in Figure 2.

In this architectural variation, a proxy is introduced between the mobile device and the SOA infrastructure, playing the role of mobile device proxy in the wired network. This proxy interacts via SOA-defined protocols with registries and service providers, and may perform a series of content adaptations, returning to mobile devices results using lightweight protocols and data formats.

This approach has several advantages over the previous one. The proxy may act as a cache, storing data of previous service invocations as well as any client relevant information, such as bookmarks and profiles. Proxies may also help client devices by transforming complex data into lightweight formats that could be rapidly delivered through wireless channels and processed by resource-constrained devices.

Advertising Services

In a general way, mobile devices have two choices for advertising services (Loureiro et al., 2006): the push-based approach and the pull-based approach. In the first one, illustrated in Figure 3, service providers periodically send the descriptions of the services to be advertised directly to potential clients, even if they are not interested in such services (1). Clients update local registries with information about available services (2), and if some service is needed, clients query their own registries about available providers (3).

In the pull-based approach, clients only receive the description of services when they require a service discovery. This process can be performed in two ways, either through centralized or distributed registries. In the former, illustrated in Figure 4, service descriptions are published in central servers (1), which maintain entries about available services (2). Clients then query this centralized registry in order to discover the services they need (3).

In the distributed registry approach, illustrated in Figure 5, the advertisement is performed in a registry contained in each provider (1). Therefore, once a client needs to discover a service, it will have to query all the available hosts in the environment (2) until discovering some service provider for the needed service (3).