Service-Oriented Architectures for Pervasive Computing

Elias S. Manolakos, University of Athens, Greece
Demetris G. Galatopoullos, Northeastern University, USA

ABSTRACT

The vision of pervasive computing is to create and manage computational spaces where large numbers of heterogeneous devices collaborate transparently to serve the user tasks all the time, anywhere. The original utility of a computer is now changing from a stand-alone tool that runs software applications to an environment-aware, context-aware tool that can enhance the user experience by executing services and carrying out his/her tasks in an efficient manner. However, the heterogeneity of devices and the user’s mobility are among the many issues that make developing pervasive computing applications a very challenging task. A solution to the programmability of pervasive spaces is adopting the service-oriented architecture (SOA) paradigm. In the SOA model, device capabilities are exposed as software services thus providing the programmer with a convenient abstraction level that can help to deal with the dynamicity of pervasive spaces. In this chapter the authors review the state of the art in SOA-based pervasive computing, identify existing open problems, and contribute ideas for future research.

INTRODUCTION

Although at times we may not realize it, networked devices of varying computational power are everywhere around us and heavily involved in our daily lives. From embedded sensors in the foundations of a bridge that communicate data about its structural integrity, to set-top boxes and game consoles providing entertainment at home, to large-scale servers offering access to specialized services over the Internet, the vision of pervasive computing as narrated by Mark Weiser (1991) is in fact becoming a reality. Distributed computing can now reach a wide-range of heterogeneous devices that may disappear in the background while still offering their capabilities as services through the network “all the time, everywhere” (Saha & Mukherjee, 2003; Satyanarayanan, 2001).

Pervasive computing is a multidisciplinary field that evolved from distributed and mobile computing (da Costa, Yamin, & Geyer, 2008; Saha et al., 2003). It differs from traditional computing in terms of how the computational devices are perceived and utilized in applications. We live in an era where computing gradually moves from stand-alone software programs running on a single machine to orchestrated composite services running across distributed interconnected cyberspaces. A pervasive environment evolves from a virtual environment of interconnected computers into a collaborative information-rich physical space. The diverse and distributed services that devices immersed in a pervasive environment offer to the developer of pervasive applications or to the

DOI: 10.4018/978-1-61520-686-5.ch007
end-user are called pervasive services. Pervasive devices become an integral part of a pervasive environment and unlike traditional computers they collectively offer location and context awareness as core services to user applications in a seamless and continuous manner (Gupta, Lee, Purakayastha, & Srimani, 2001; Saha et al., 2003). Pervasive computing principles have already been utilized in several application domains such as smart homes, where intelligent appliances may expose their capabilities as services and can be combined with sensors to provide a comfortable, safe or rehabilitative living environment. Another example is the smart cars that promise an enhanced driving experience for a safer and more enjoyable ride. Health care, manufacturing, and social networks are also areas where pervasive computing is currently exploited.

Since pervasive computing evolved from distributed and subsequently mobile computing, it is only natural that it inherited a set of their problems (da Costa et al., 2008). Developing software applications that engage pervasive devices is not easy and requires new application development paradigms. The high degrees of device heterogeneity and dynamicity call for device interoperability models and protocols. Moreover, current standard Integrated Development Environments (IDEs) are not suitable for developing pervasive applications (Helal, 2005). This is largely due to the fact that using pervasive devices introduces a set of unique issues not present in traditional software development, such as the integration of large numbers of heterogeneous entities, ubiquitous connectivity etc., that so far are only performed at small scale and in an ad-hoc manner. These difficulties generate the need for new middleware technologies and software engineering methodologies that will allow developers to integrate available devices from a pervasive-enabled space into their applications easily and at a large scale without the need of using a specialized API for each new device type. Middleware is generally defined as a set of services that lie between the application layer and the connectivity layer (Vinoski, 2002). The middleware layer hides the complexity details of the lower layers so that developers can implement portable and interoperable applications. Middleware does not deal with the underlying hardware of the device it is executing on, something that the Operating Systems (OS) does, but it rather mediates between software layers. In Pervasive computing, middleware architectures provide a set of core services that hide the heterogeneity of devices in ambient environments and enable their services to be exposed at the application layer (Saha et al., 2003). Examples of these services include service discovery and execution, context-awareness, trust and security and semantic matching. Currently middleware technologies such as the Universal Plug and Play (UPnP) (Presser et al., 2008) and the Open Services Gateway initiative (OSGi) (OSGi Alliance, 2008; Marples & Kriens, 2001) contribute towards this objective but only at the local area network level.

Extending pervasive computing to the enterprise level requires exploiting the Service-Oriented Architecture (SOA) paradigm that offers the potential to handle the heterogeneity and scalability challenges in a device, platform and language agnostic manner. The World Wide Web (W3C) consortium (W3C, 2008) defines SOA as a service-centered platform-neutral distributed systems architecture. Similarly, the Organization for the Advancement of Structured Information Standards (OASIS) (OASIS, 2008) defines SOA as a set of practices, principles and patterns related to service-aware, enterprise-level, distributed computing. In their seminal paper, Papazoglou, Traverso, Dustdar, and Leymann (2007) define SOA as a methodology for building software that can deliver services to other services or end-users. All these definitions agree in that an SOA is not a protocol but rather a software architectural style. Software services in an SOA are advertised as public interfaces that can be discovered and composed to yield new more complex services. This model fits well in the pervasive computing vision where a multitude of heterogeneous devices should be weaved together to naturally and transparently serve the end-user’s application (task).
Related Content

[www.igi-global.com/chapter/level-crossing-sampling-energy-conservation/53790?camid=4v1a](www.igi-global.com/chapter/level-crossing-sampling-energy-conservation/53790?camid=4v1a)
Social Media and Autonomous Learning at a Latin America University: The Social Neuroscience Role of the Developing Metacognitive Capacities
www.igi-global.com/article/social-media-and-autonomous-learning-at-a-latin-america-university/209371?camid=4v1a

Adaptive Content Delivery in E-Learning Systems using Mobile Agents
www.igi-global.com/chapter/adaptive-content-delivery-learning-systems/56424?camid=4v1a

Modern Educational Technique Center Educational Media Management Based on Design and Practice of Questionnaire
www.igi-global.com/article/modern-educational-technique-center-educational/62294?camid=4v1a