Chapter 7.9
Towards a Programming Model for Ubiquitous Computing

Jorge Barbosa
Universidade do Vale do Rio dos Sinos (Unisinos), Brazil

Fabiane Dillenburg
Universidade Federal do Rio Grande do Sul, Brazil

Alex Garzão
Universidade do Vale do Rio dos Sinos (Unisinos), Brazil

Gustavo Lermen
Universidade do Vale do Rio dos Sinos (Unisinos), Brazil

Cristiano Costa
Universidade do Vale do Rio dos Sinos (Unisinos), Brazil

ABSTRACT

Mobile computing is been driven by the proliferation of portable devices and wireless communication. Potentially, in the mobile computing scenario, the users can move in different environments and the applications can automatically explore their surroundings. This kind of context-aware application is emerging, but is not yet widely disseminated. Based on perceived context, the application can modify its behavior. This process, in which software modifies itself according to sensed data, is named Adaptation. This constitutes the core of Ubiquitous Computing. The ubiquitous computing scenario brings many new problems such as coping with the limited processing power of mobile devices, frequent disconnections, the migration of code and tasks between heterogeneous devices, and others. Current practical approaches to the ubiquitous computing problem usually rely upon traditional computing paradigms conceived back when distributed applications where not a concern. Holoparadigm (in short Holo) was proposed as a model to support the development of distributed systems. Based on Holo concepts, a new programming language called HoloLanguage (in short, HoloL) was created. In this chapter, we propose the use of Holo for developing and executing ubiquitous applications. We explore the HoloL for ubiquitous programming and propose a full platform to develop and execute Holo programs. The language supports mobility, adaptation, and context awareness. The execution environment is based on a virtual machine that implements the concepts proposed by Holo. The environment supports distribution and strong code mobility.

DOI: 10.4018/978-1-61350-456-7.ch7.9

INTRODUCTION

Nowadays, studies focusing mobility in distributed systems are being stimulated by the proliferation of portable electronic devices (for example, smart phones, handheld computers, tablet PCs, and notebooks) and the use of interconnection technologies based on wireless communication (such as WiFi, WiMAX, and Bluetooth). This new mobile and distributed paradigm is called *Mobile Computing* (Satyanarayanan, 1996). Moreover, mobility together with the widespread use of wireless communication enabled the availability of computational services in specific contexts – *Context-aware Computing* (Dey et al., 1999). Furthermore, researches related to adaptation brought the possibility of continuous computational support, anytime and anywhere. This characteristic is sometimes referred as *Ubiquitous Computing* (Weiser, 1991; Grimm et al., 2004; Saha & Mukherjee, 2003; Satyanarayanan, 2001). Despite the relevance of context awareness and ubiquitous computing, there is no support of the existing programming models for the developer to think and specify his/her application using a more general abstraction that integrates both application and contexts into an integrated logic, including also the mobility and adaptation aspects.

Holoparadigm (in short, *Holo*) was proposed as a development model for traditional distributed systems (Barbosa, Yamin, Augustin, Vargas & Geyer, 2002). A blackboard (called *history*) implements the coordination mechanism and a new programming entity (called *being*) organizes encapsulated levels of *beings* and histories. Based on Holo concepts, a new programming language called HoloLanguage (in short, HoloL) was proposed. The original execution platform was oriented to grid systems and used Java as the intermediate language (Barbosa, Costa, Yamin & Geyer, 2005). HoloL supports mobility, adaptation, and context awareness. These characteristics were not fully used in the original platform, because they were not critical to grid computing systems. However, they are highly relevant in the development of ubiquitous software.

In this chapter we propose the use of Holo for developing and executing ubiquitous applications. HoloL is used as a language for ubiquitous programming, and a new execution environment supports the distributed execution of programs exploring the potential of Holo to ubiquitous computing. The environment is based on a virtual machine and supports distributed *beings*, native strong code mobility, and dynamic behavior of *beings*.

This chapter is organized as follows. Section two summarizes the Holoparadigm concepts. The third section uses sample codes to present the HoloLanguage, and also discusses its characteristics associated with ubiquitous programming. The execution environment is presented in the fourth section. The fifth section presents experimental results and performance analysis. The sixth discusses related works. Finally, last two sections draw future research directions and some conclusions.

BACKGROUND

Holoparadigm is based on an abstraction called *being*, which is used to support mobility. There are two kinds of *beings*: *elementary being*, which is an atomic *being*, without composition levels, and *composed being*, which is a *being* composed by other *beings*. An elementary *being* is organized in three parts: *interface*, *behavior,* and *history*. The *interface* describes the possible interactions between beings. The *behavior* contains actions, which implement the *being's* functionality. The *history* is a shared tuple space in a *being*. A composed *being* (Figure 1a) has the same organization of an elementary one, but may also contain other *beings* (which are named *component beings*). Each *being* has its own encapsulated history.

A composed *being's* history is also shared with its component *beings*. In this way, several levels