Ambient-Oriented Programming in AmbientTalk: Combining mobile hardware with simplicity and expressiveness

Jessie Dedecker* Tom Van Cutsem*
Stijn Mostinckx† Wolfgang De Meuter Theo D'Hondt
Programming Technology Laboratory
Department of Computer Science
Vrije Universiteit Brussel, Belgium
jededeck | tvcutsem | smostinc | wdmeuter | tjdhondt@vub.ac.be

ABSTRACT

A new field in distributed computing, called Ambient Intelligence, has emerged as a consequence of the increasing availability of wireless devices and the mobile networks they induce. Developing software for such mobile networks is extremely hard in conventional programming languages because the network is dynamically defined. We demonstrate a new distributed programming language named AmbientTalk, a research vehicle to experiment with expressive high-level language constructs for *Ambient-Oriented Programming*.

Categories and Subject Descriptors

D.3.2 [**Programming Languages**]: Language Classifications—*distributed languages*; D.3.3 [**Programming Languages**]: Language Constructs and Features—*Frameworks*

General Terms

Design, Languages

Keywords

ambient intelligence, mobile networks, actors, language kernel

1. INTRODUCTION

Software development for mobile devices (such as smart phones and PDAs) is given a new impetus with the advent of mobile networks. Mobile networks surround a mobile device equipped with wireless technology and are demarcated dynamically as users move about. Mobile networks turn the applications running on mobile devices from mere isolated programs into smart applications that can cooperate with their environment. As such, mobile networks take us one step closer to the world of ubiquitous computing envisioned by Weiser[5]; a world where (wireless) technology is gracefully integrated into the everyday lives of its users. Recently, this vision has been termed Ambient Intelligence (AmI for short) by the European Council's IST Advisory Group.

Mobile networks that surround a device have several properties that distinguish them from other types of networks. The most important ones are that connections are volatile (because the communication range of the wireless technology is limited) and that the network is open (because devices can appear and disappear unheraldedly). Dealing with these low-level network concerns at the application-level clutters the code significantly and puts an additional burden on software developers.

Although low-level system software and networking libraries (such as JXTA [3] and M2MI [4]) tend to provide uniform interfaces to wireless network technologies, they do not alleviate the task of developing application software for mobile networks. One of the main reasons for this is that current-day programming languages lack abstractions that avoid the cluttering of application and network-related code. The need to deal with the network characteristics in a way that *can be* made oblivious to the programmer justifies the need for a new Ambient-Oriented Programming paradigm [2] (AmOP for short) that consists of programming languages that explicitly incorporate potential network failures in the very heart of their computational model.

AmbientTalk [1] is a first scion of this AmOP programming language family. The power of AmbientTalk lies in its simplicity and expressiveness. It offers the programmer features to deal with the conceptual properties of wireless networks without having to deal with their technological characteristics. The basis of AmbientTalk consist of:

- Concurrent Object Model combining the power of
  - prototype-based programming
  - active objects based on both actors and ordinary objects

*Research Assistant of the Fund for Scientific Research Flanders, Belgium (F.W.O.)

†Author funded by a doctoral scholarship of the "Institute for the Promotion of Innovation through Science and Technology in Flanders (IWT Vlaanderen"

OOPSLA'05, October 16–20, 2005, San Diego, California, USA.
ACM 1-59593-193-7/05/0010.

- A system of first-class mailboxes that allow both the reification of an objects computational history and future, and the reification of the hardware surrounding an object.
- Reflective Properties such as a MOP and a language extension facility which allow us to experiment with new, reflectively implemented language features.

The demo showcases a simple instant messenger that is designed to operate in wireless ad hoc networks. Without requiring additional infrastructure, different instant messenger clients can discover each other and engage in spontaneous interactions. These interactions gracefully handle users coming online or going offline at any instant. The demo setup consists of an alternation between

- showing code snippets that acquaint the audience with AmbientTalk, as well as providing a short overview of some of its features. Furthermore these code snippets illustrate how the use of AmbientTalk language constructs allows a software developer to abstract away from the AmI hardware characteristics.
- demonstrating how the instant messenger actually behaves in a real-life context. This will be done by showing the execution of the programs on several portable devices and subsequently introducing some typical network failures.

2. REFERENCES

[1] Dedecker, J., Van Cutsem, T., Mostinckx, S., De Meuter, W., and D'Hondt, T. Ambienttalk: A small reflective kernel for programming mobile network applications. Tech. Rep. VUB-PROG-TR-05-06, Programming Technology Laboratory, Department of Informatica, Vrije Universiteit Brussel, 2005.

[2] Dedecker, J., Van Cutsem, T., Mostinckx, S., D'Hondt, T., and De Meuter, W. Ambient-oriented programming. In *OOPSLA '05: Companion of the 20th annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages and Applications. San Diego, U.S.A. ACM Press.* (2005), ACM Press.

[3] Gong, L. JXTA for J2ME extending the reach of wireless with JXTA technology. Tech. rep., SUN Microsystems, http://www.jxta.org/project/www/docs/JXTA4J2ME.pdf, 2002.

[4] Kaminsky, A., and Bischof, H.-P. Many-to-many invocation: A new object oriented paradigm for ad hoc collaborative systems. *17th Annual ACM Conference on Object Oriented Programming Systems, Languages, and Applications (OOPSLA 2002)* (2002).

[5] Weiser, M. The computer for the 21st century. *Scientific American 265*, 3 (1991), 66–75.