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

Tom Van Cutsem, Stijn Mostinckx, Jessie Dedecker, Wolfgang De Meuter, Theo D'Hondt
Programming Technology Laboratory
Vrije Universiteit Brussel, Belgium

PROG — Programming Technology Lab

AMBIENTTALK

Teaser

Properties of Pervasive, Mobile Computing Hardware

The hardware environment of the near future ...

Engenders

Communication — Autonomous Concurrent Devices

Interconnected via ad hoc wireless networks

Failure — Volatile Wireless Connections

Devices frequently disconnect

Discovery — Ambient Resources

I provide a printing service

I require a printing service

Devices (dis)appear in the 'Ambient' Spontaneous Service Discovery

Ambient-Oriented Programming Paradigm

... requires programming languages that provide adequate language constructs.

Concretised by

Communication — Non-blocking Asynchronous Communication Primitives

send
transmit
receive

Based on Actors (Agha, 1986)

Failure — Reified Communication Traces

agenda slot available?
yes
Agenda Actor
reserve slot
Agenda Actor
reservation confirmed?
Agenda Actor

E.g. rollback strategies for failure recovery

Discovery — Reified Environmental Context

Provided service patterns

Required service patterns

description of service interface

Actors specify services required from and provided to the ambient

AmbientTalk

A first AmOP programming language supporting such language constructs.

Communication — Communication Mailboxes reify actor-actor interaction

«outbox»
«sentbox»
«inbox»
«rcvbox»

Metaobjects — First-class Mailboxes

interaction between mailboxes governed by a metaobject protocol

onSend
onTransmit
onJoin
onReceipt
onDisjoin
onProcess

Meta-level events trapped using mailbox observers

Discovery — Discovery Mailboxes reify actor-ambient interaction

«required»
«provided»
«joinedBox»
«disjoinedBox»

```
PrintFileBehavior :: object({
  printer : void;
  file    : void;

  new(aFile,dpi) :: copy({
    printer := AmbientRef("Printer");
    file := aFile
  });

  printIt() :: {
    when(printer#print(file), {
      display("Doc Printed!")
    })
  }
})

PrintFileActor(file,dpi) :: {
  actor(PrintFileBehavior.new(file,dpi))
}

{ p: PrintFileActor(aFile,300);
  p#printIt() }
```

▶ Dynamically typed language
▶ Prototype-based object model
▶ Actor-based concurrency model
▶ VM runs on Java J2SE and J2ME platforms
▶ VMs communicate via sockets over WLAN
▶ Plug-and-play: no additional setup
▶ Tested and deployed on QTek9090 PocketPC smartphones

Actors
AmbientTalk VM
Java

Further Reading
[1] Dedecker J., Van Cutsem T., Mostinckx S., De Meuter W., D'Hondt T. *Ambient-Oriented Programming in AmbientTalk*. In proceedings of the 20th European Conference on Object-oriented Programming (ECOOP), LNCS pp. 230-254. Nantes, France, 2006.

[2] Dedecker J., Van Cutsem T., Mostinckx S., D'Hondt T., De Meuter W. 2005. *Ambient-Oriented Programming*. In Companion of the 20th ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages and Applications, pp 31-40, San Diego, USA. Onward! Presentation.

Download & Play!
http://prog.vub.ac.be/amop