

# Ambient-Oriented Programming

Jessie Dedecker      Tom Van Cutsem      Stijn Mostinckx

Wolfgang De Meuter      Theo D'Hondt



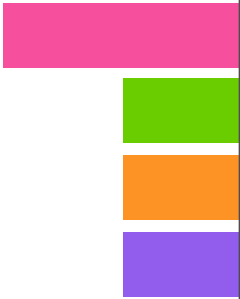
Programming Technology Laboratory  
Vrije Universiteit Brussel



Invited Talk SEPS at ICPS, Lyon, June 2006 (c) Programming Technology Lab, Vrije Universiteit Brussel, Belgium (2006)

# Overview

- Context
- Ambient-Oriented Programming
- AmbientTalk
- Wrap-up



# Context

# Mobile Networks



# Mobile Networks



# Mobile Networks



# Mobile Networks



# Issues

- Hardware Issues:
  - Miniaturisation
  - Device Autonomy
  - Interoperability
  - Processor Speed
  - Limited Memory
  - Integration
  - Cost
- Software Issues:
  - Context-awareness
  - Interaction with real world
  - Portability
  - New user interfaces
  - Standards
  - Distributed Applications



# Issues

- Hardware Issues:
  - Miniaturisation
  - Device Autonomy
  - Interoperability
  - Processor Speed
  - Limited Memory
  - Integration
  - Cost
- Software Issues:
  - Context-awareness
  - Interaction with real world
  - Portability
  - New user interfaces
  - Standards
  - Distributed Applications

# Issues

- Hardware Issues:
  - Miniaturisation
  - Device Autonomy
  - Interoperability
  - Processor Speed
  - Limited Memory
  - Integration
  - Cost
- Software Issues:
  - Context-awareness
  - Interaction with real world
  - Portability
  - New user interfaces
  - Standards
  - Distributed Applications

A Programming Language Approach



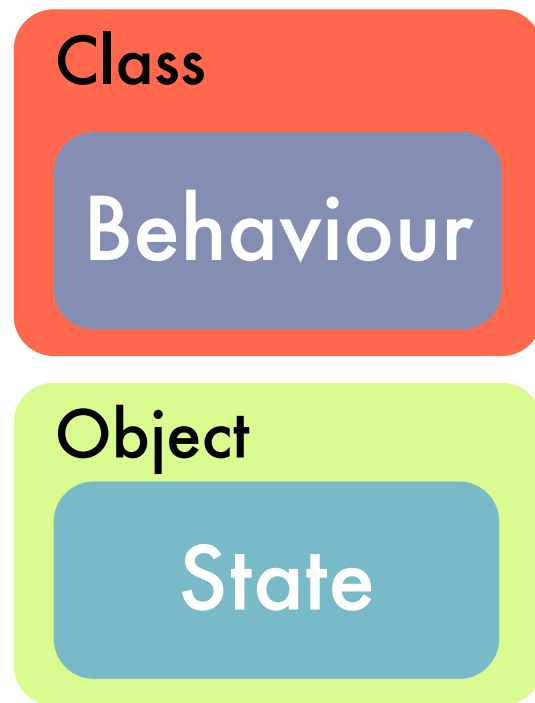
# Ambient-Oriented Programming

## Paradigm

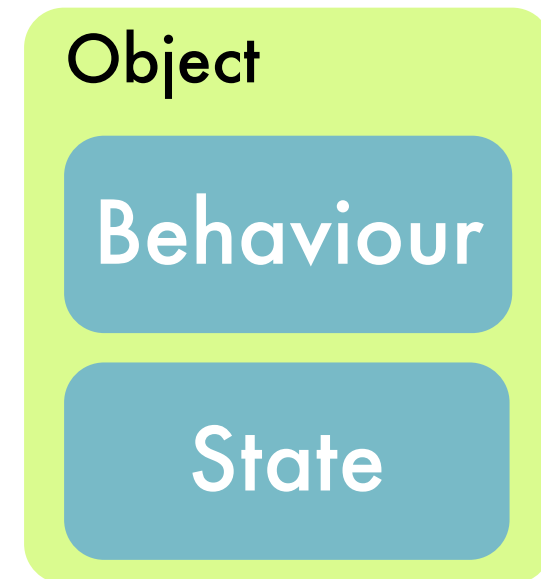
- Object-based Languages
- Non-Blocking Communication
- Reified Communication Traces
- Reified Environmental Context

# Object-based Model

## Class-based Models

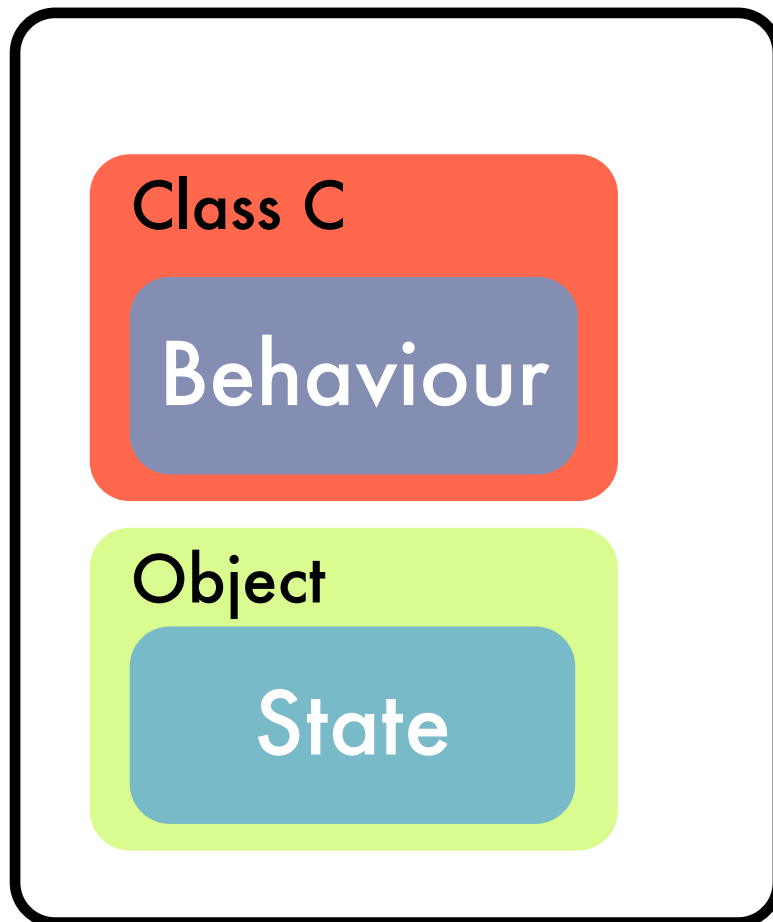


## Object-based Models

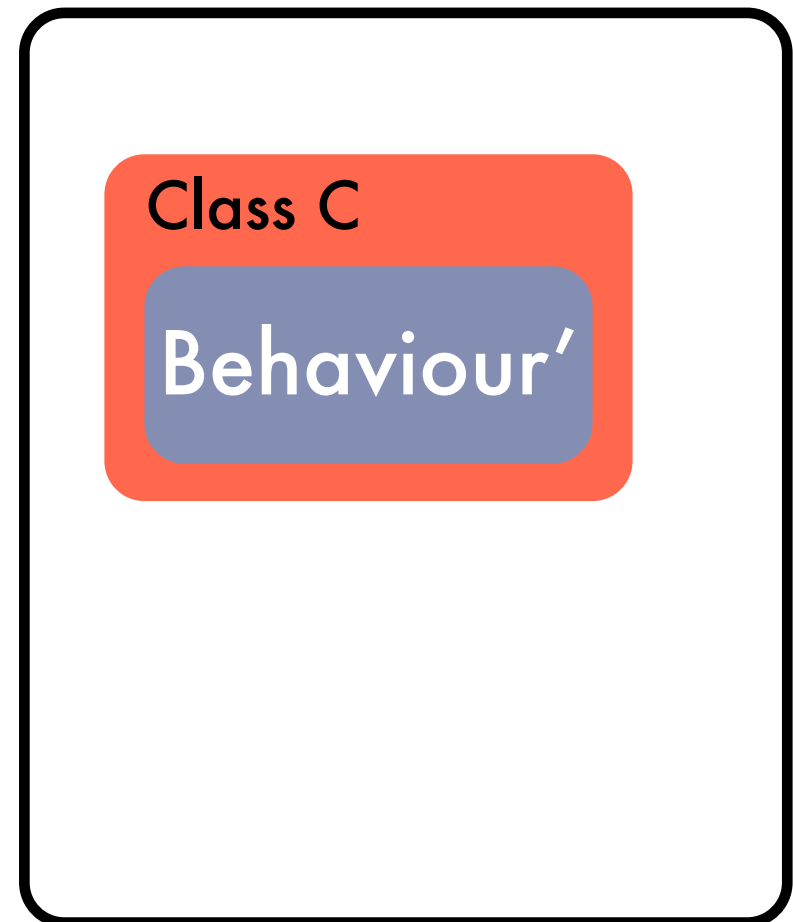


# Object-based Model

Virtual Machine A

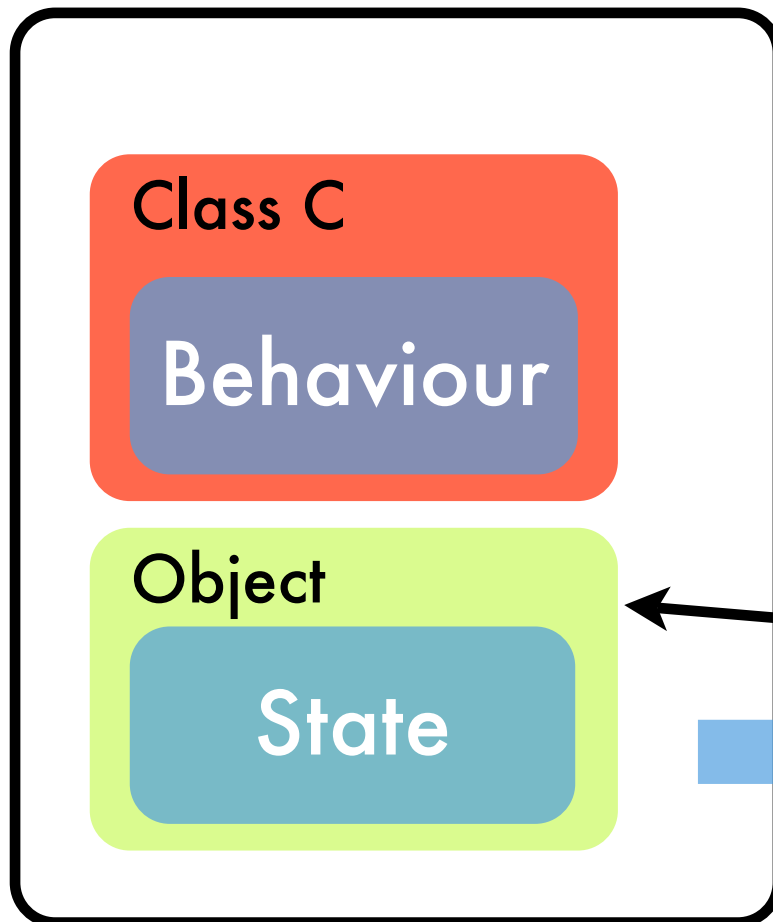


Virtual Machine B

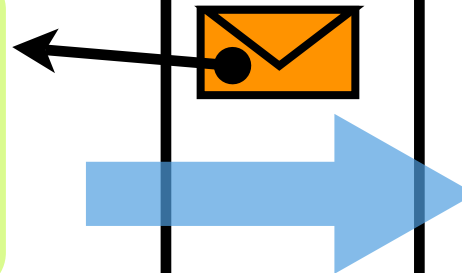
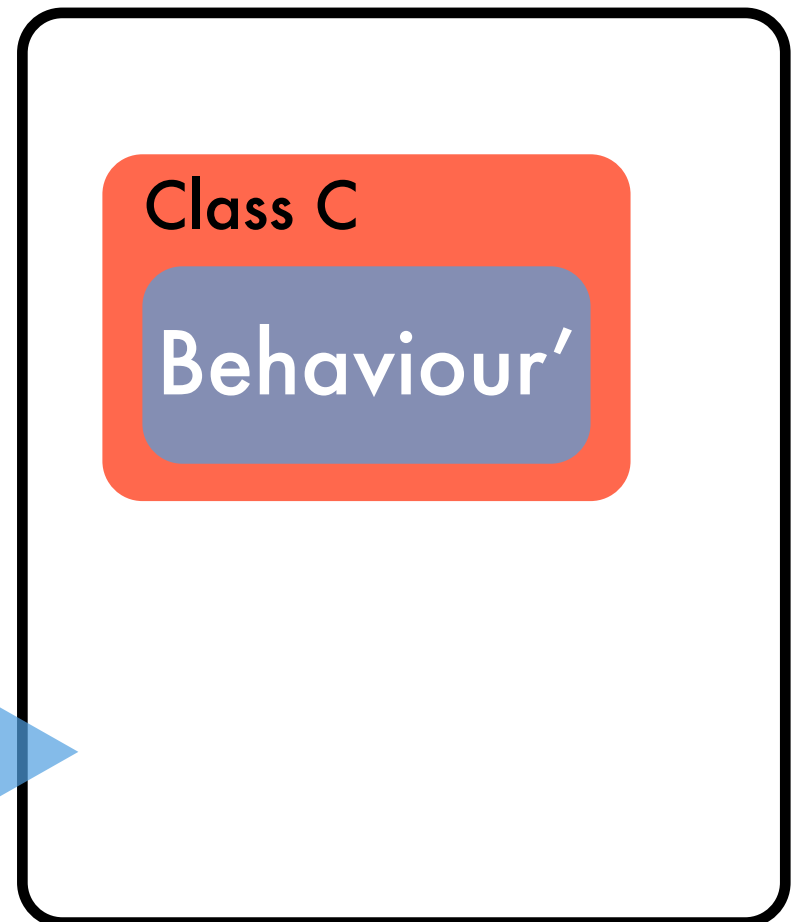


# Object-based Model

Virtual Machine A

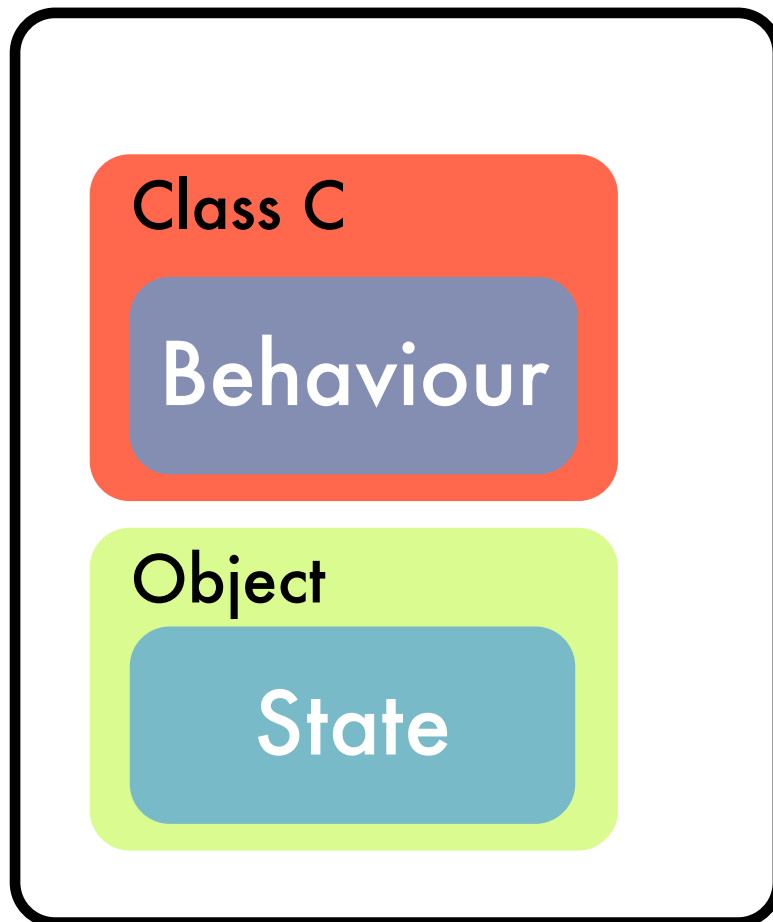


Virtual Machine B

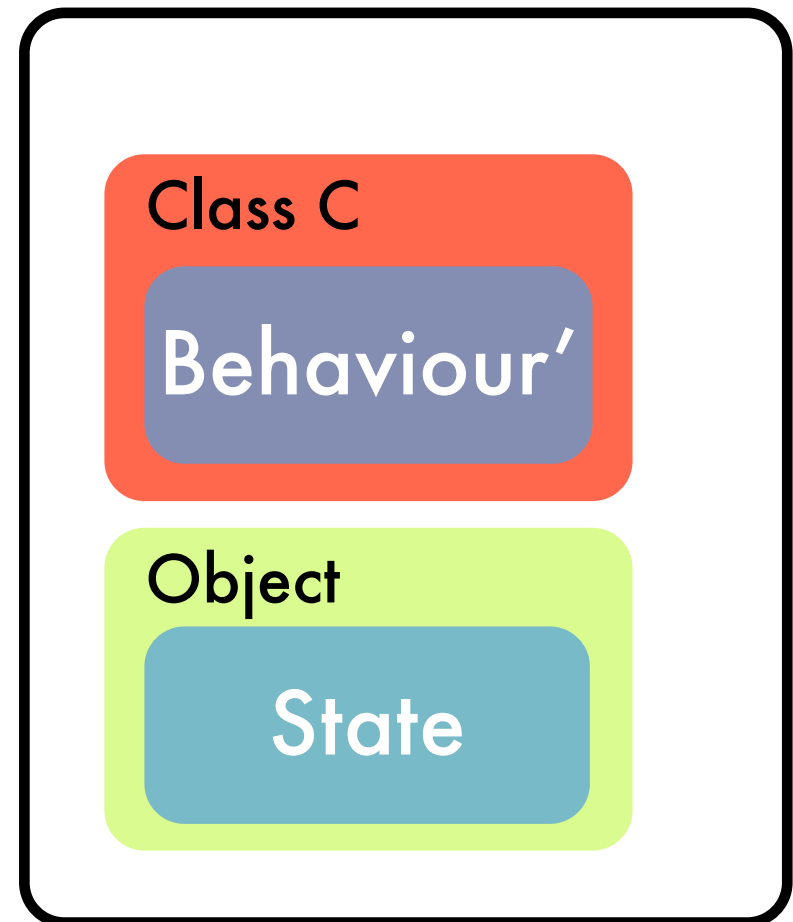


# Object-based Model

Virtual Machine A

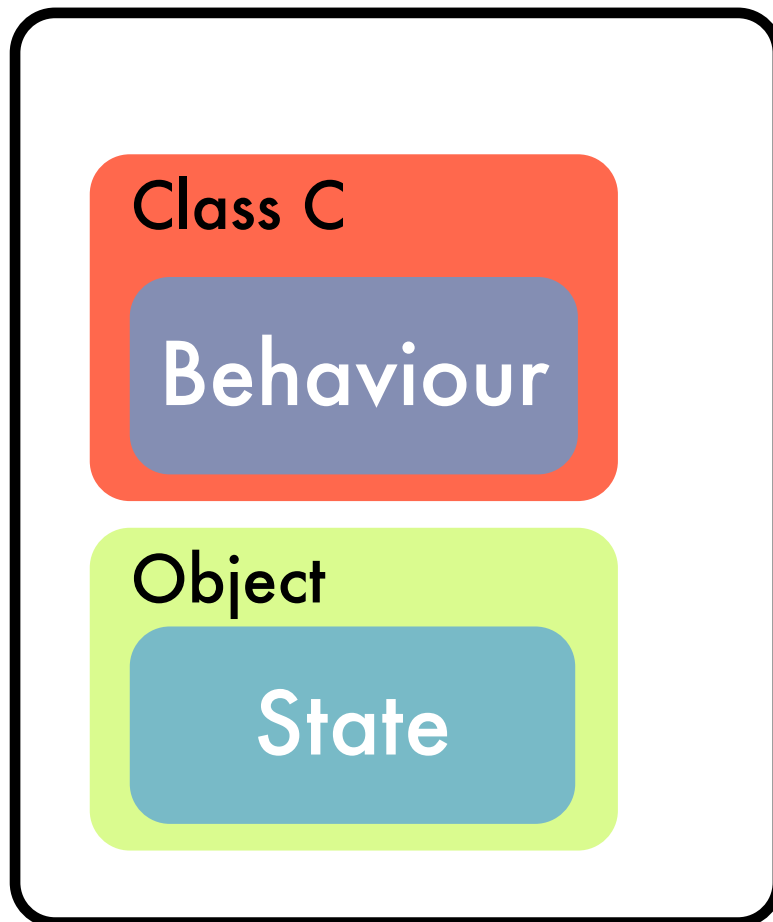


Virtual Machine B

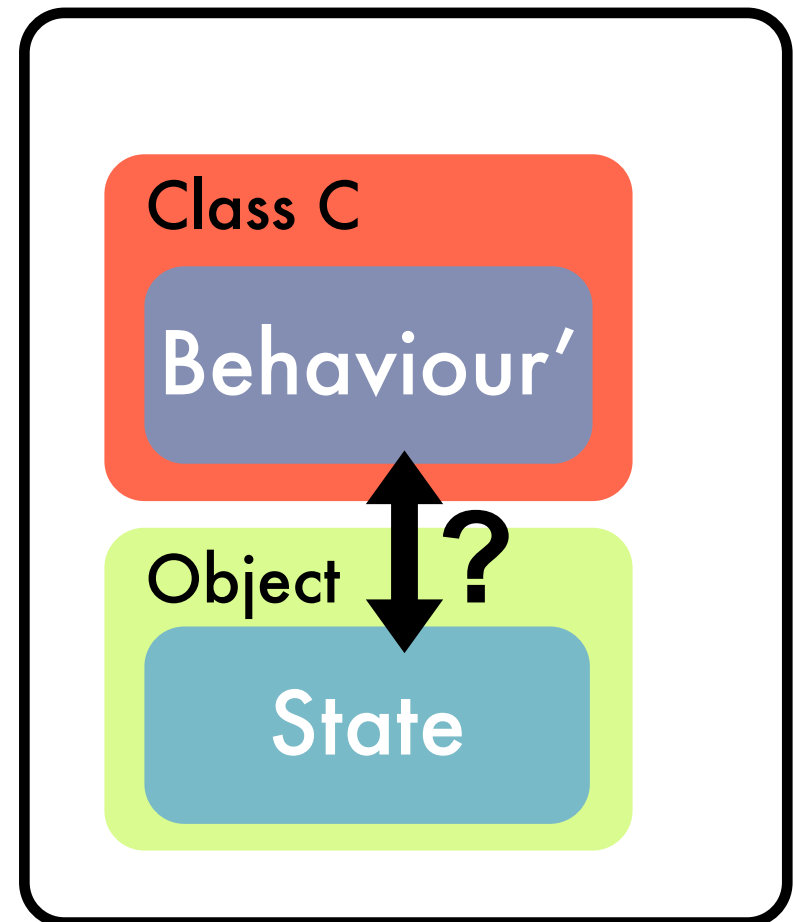


# Object-based Model

Virtual Machine A



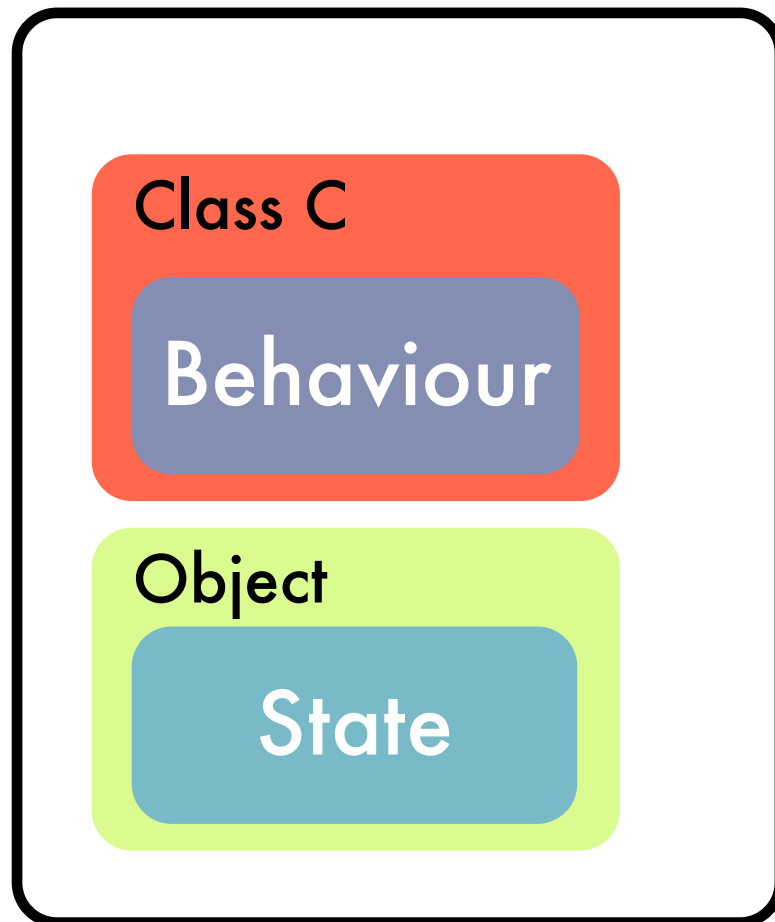
Virtual Machine B



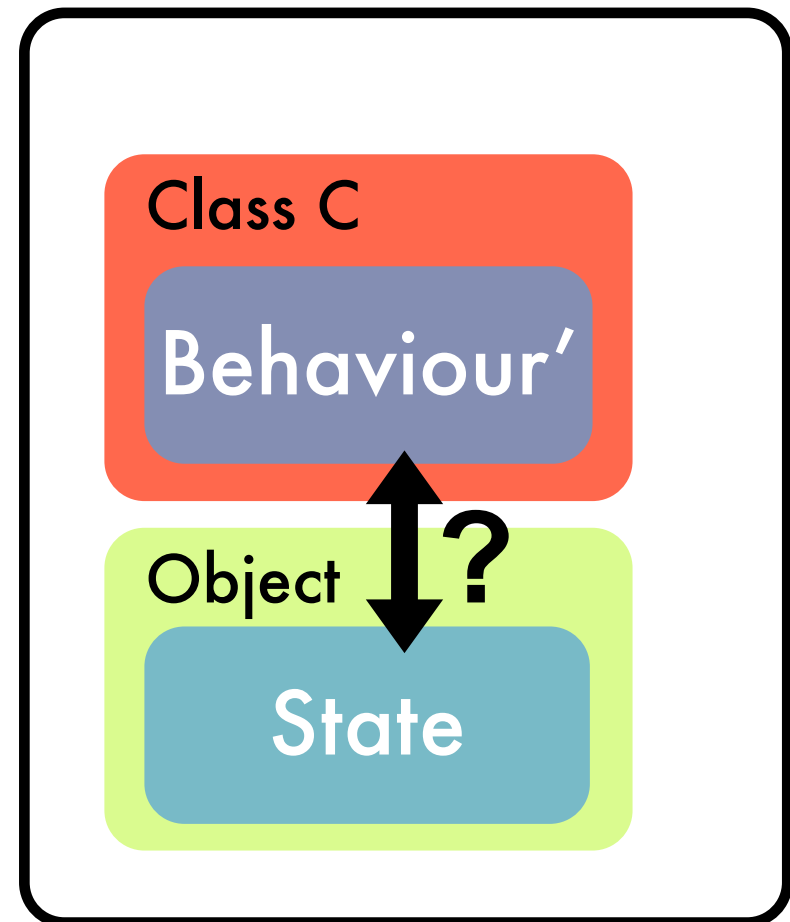


# Object-based Model

Virtual Machine A

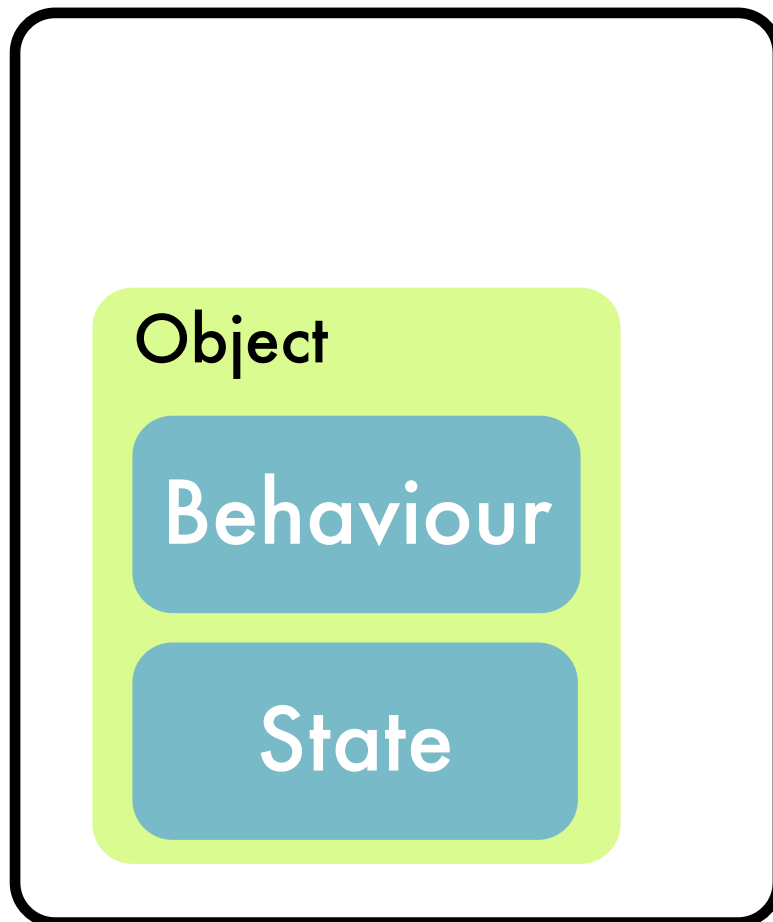


Virtual Machine B

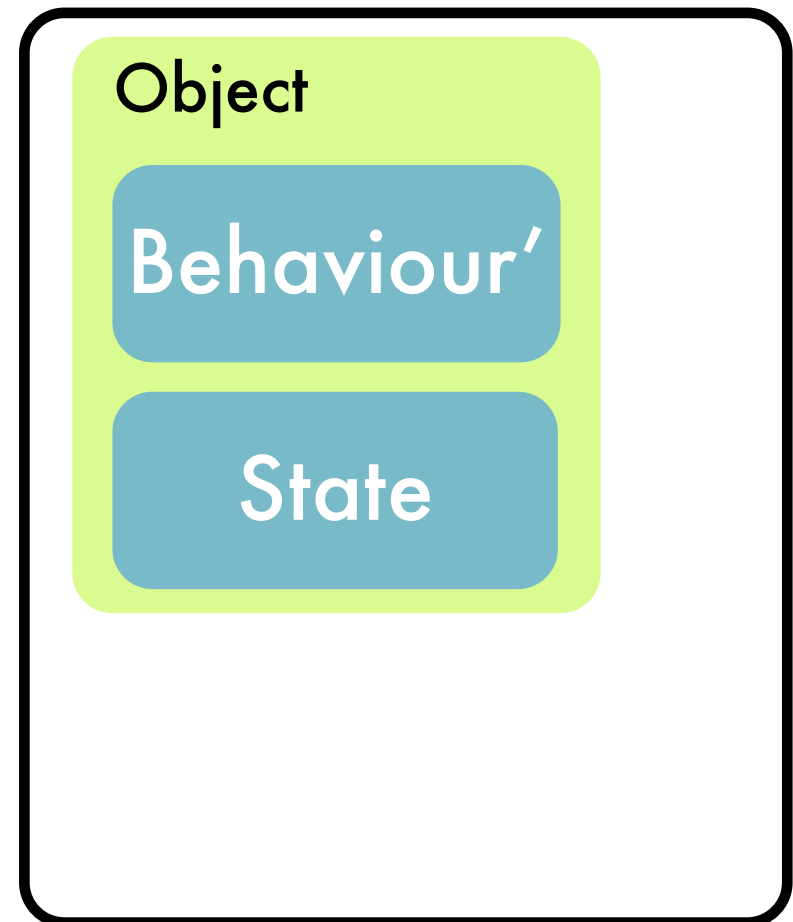


# Object-based Model

Virtual Machine A

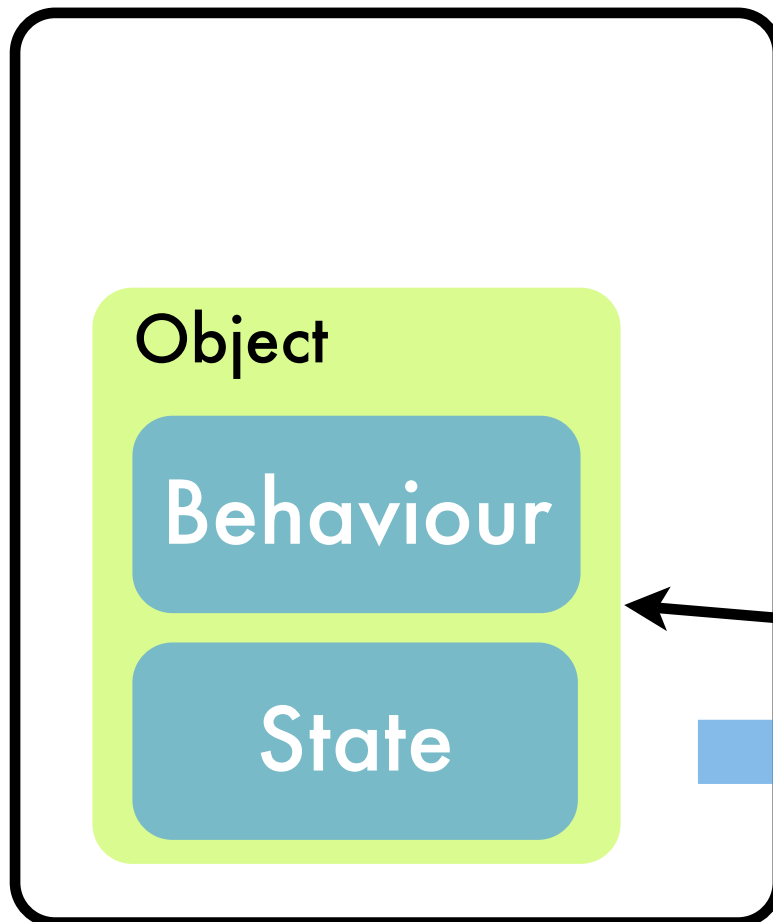


Virtual Machine B

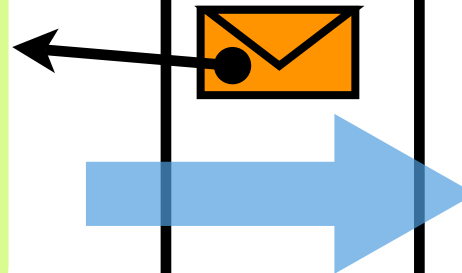
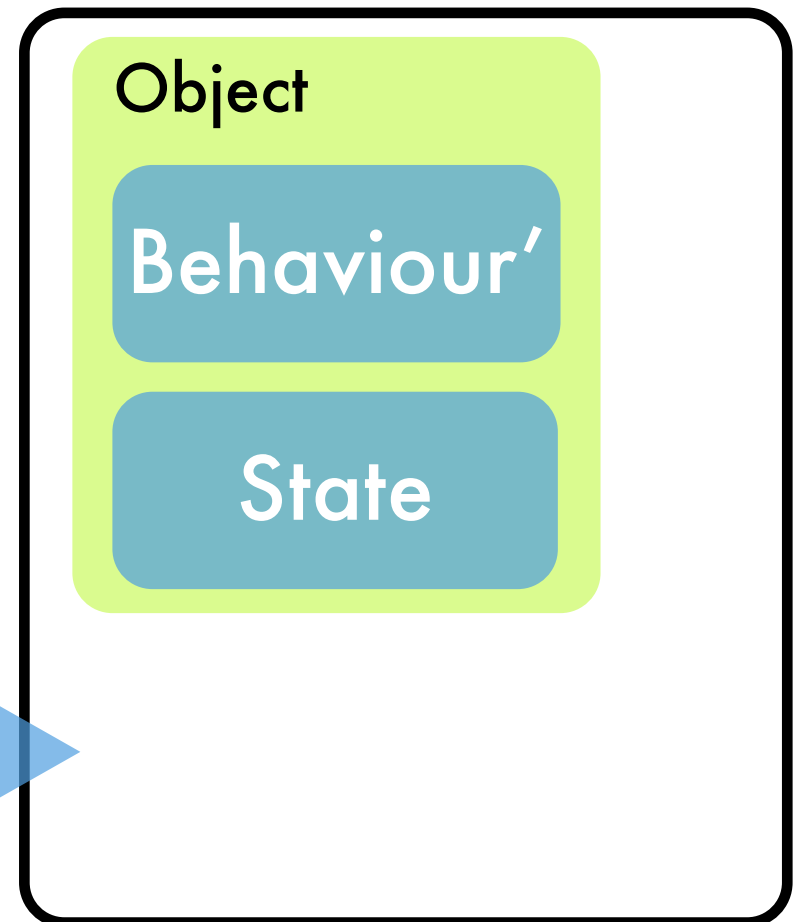


# Object-based Model

Virtual Machine A

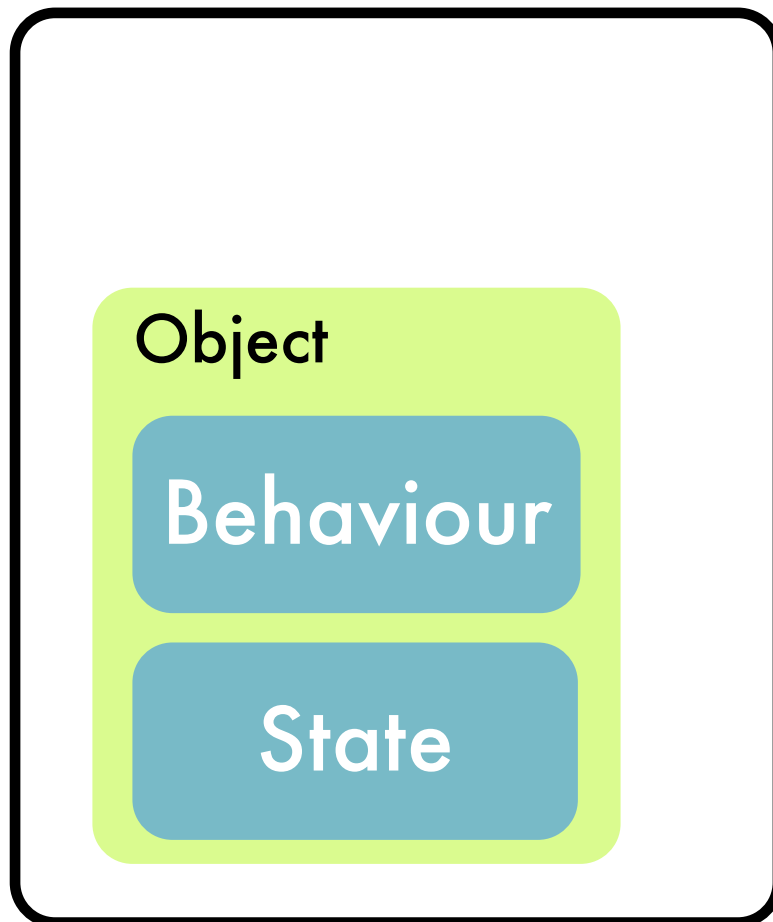


Virtual Machine B

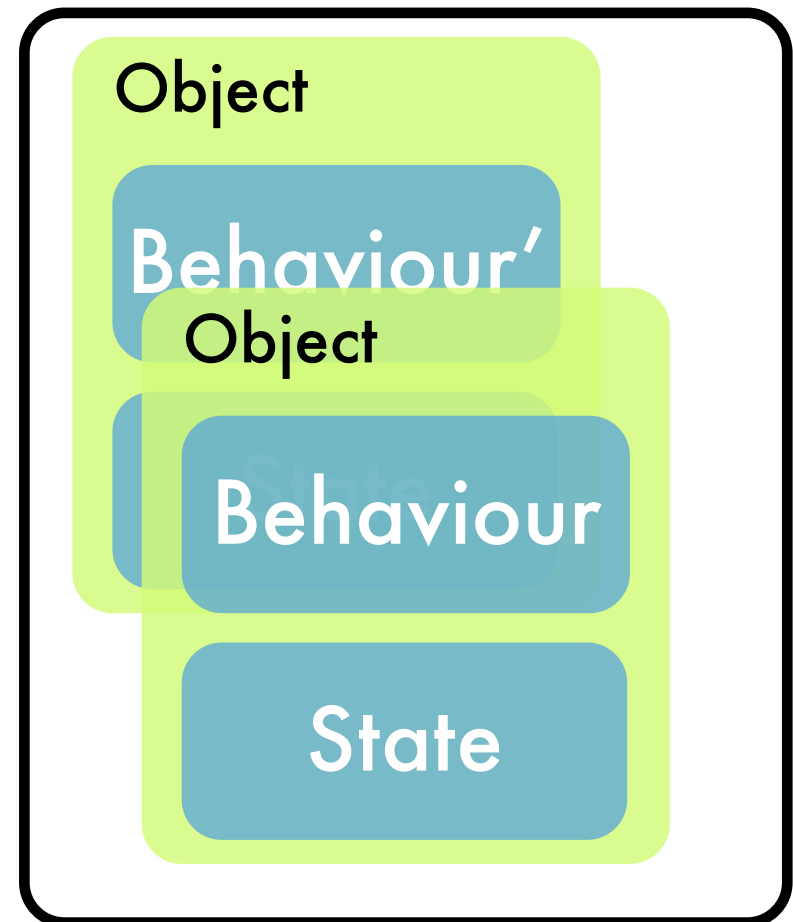


# Object-based Model

Virtual Machine A



Virtual Machine B



# Object-based Model

- Objects are self-sufficient
- No need to synchronise shared classes

Ambient-Oriented Programming deals with:



Autonomous Concurrent Devices



Volatile Connections

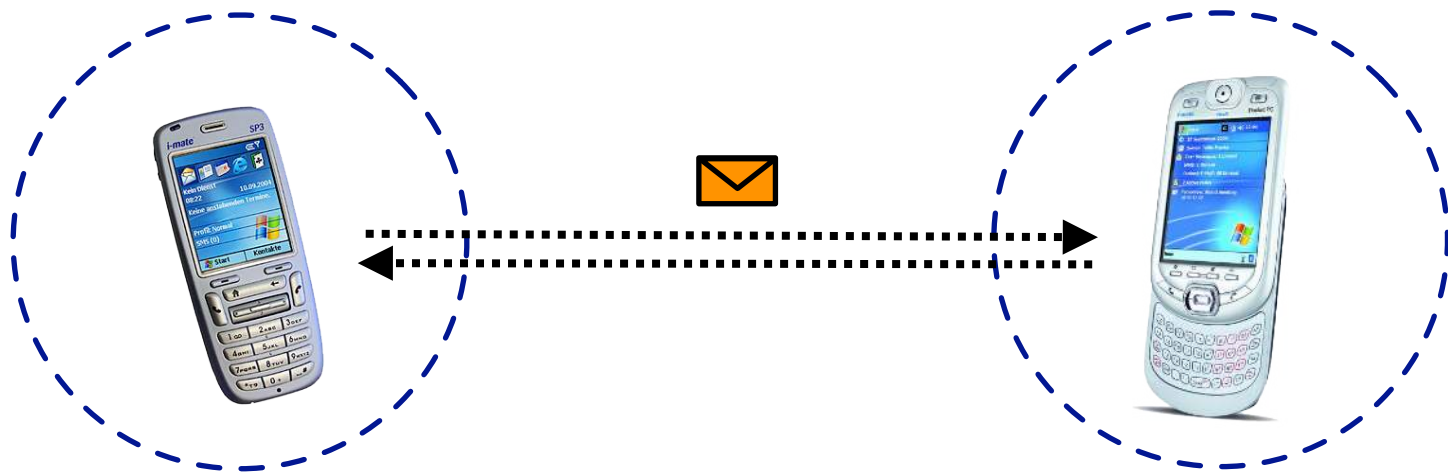
# Non-Blocking Communication

Observation: resumable communication



# Non-Blocking Communication

Observation: resumable communication



# Non-Blocking Communication

Observation: resumable communication





# Non-Blocking Communication

Observation: resumable communication



Consequence: tolerate disconnections

*Disconnections are no longer exceptional  
but become part of the paradigm!*

# Blocking Communication



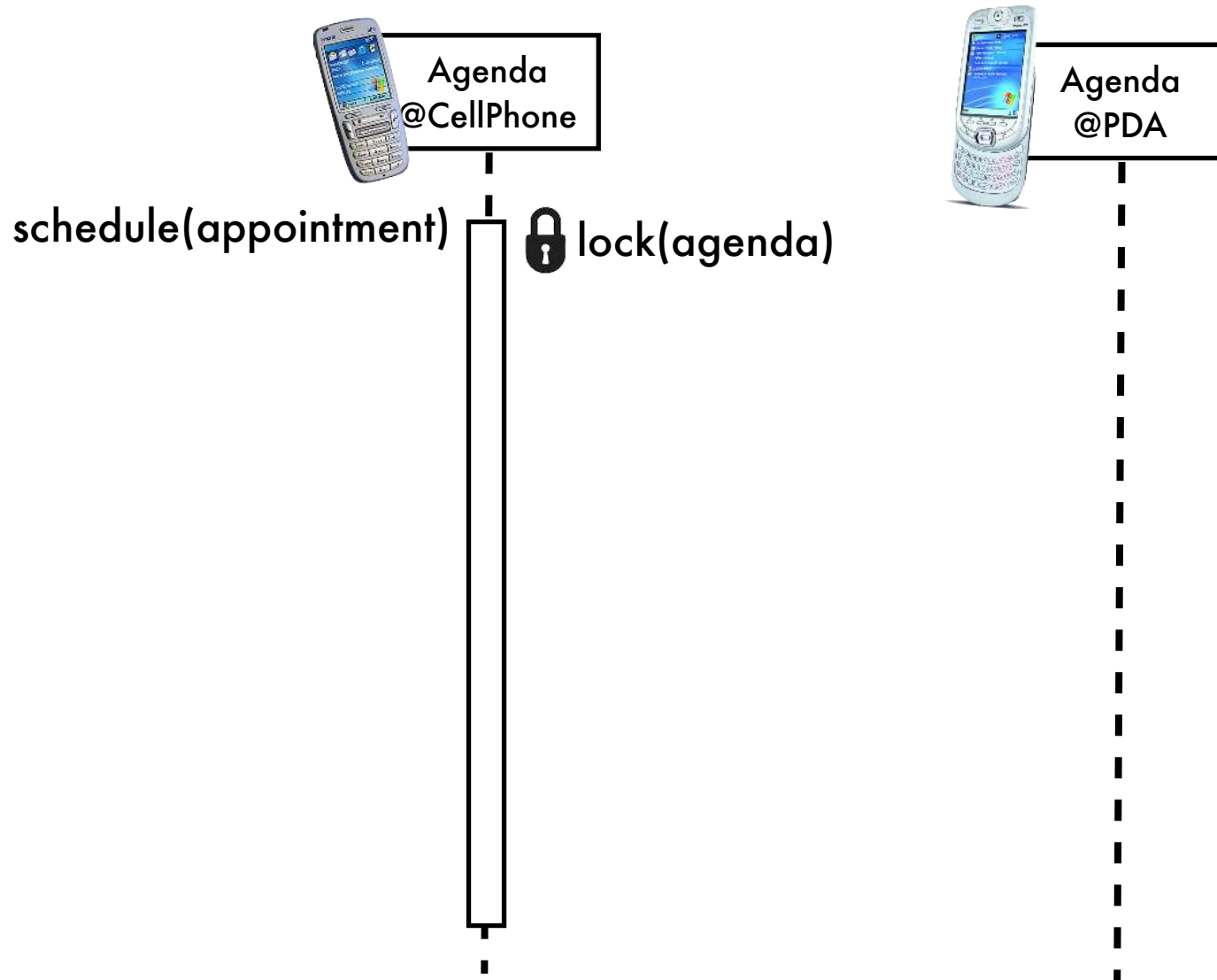
Agenda  
@CellPhone

schedule(appointment)

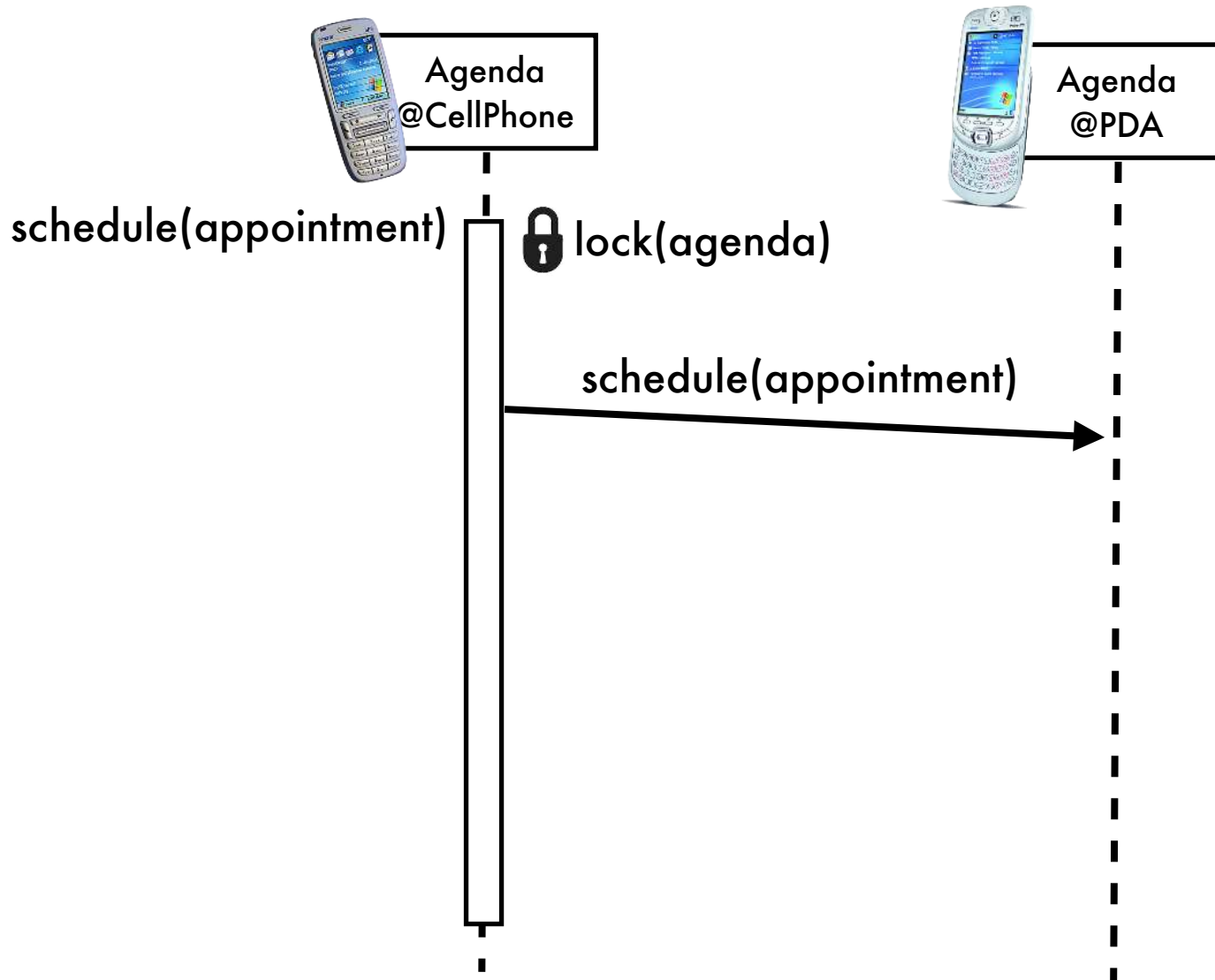


Agenda  
@PDA

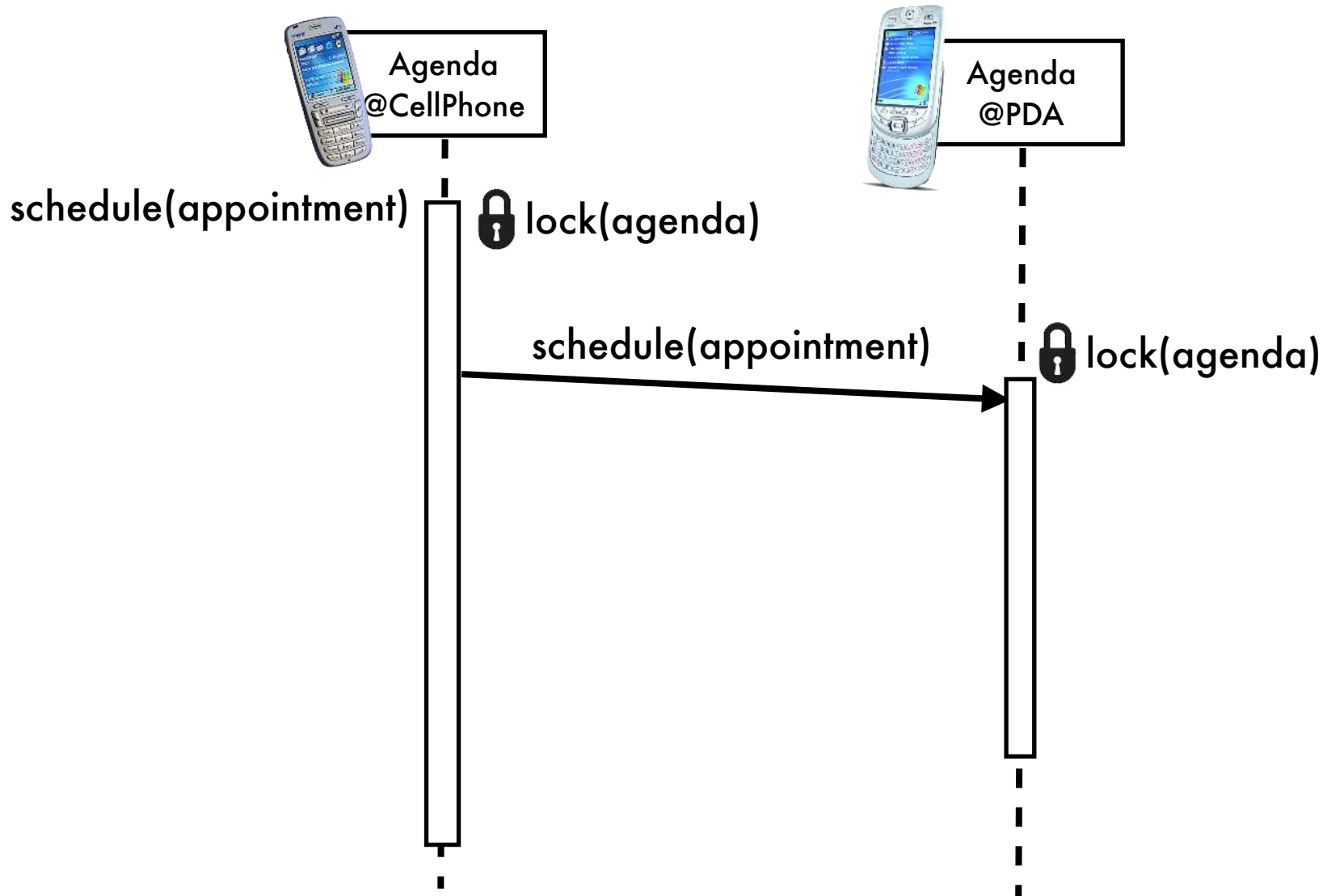
# Blocking Communication



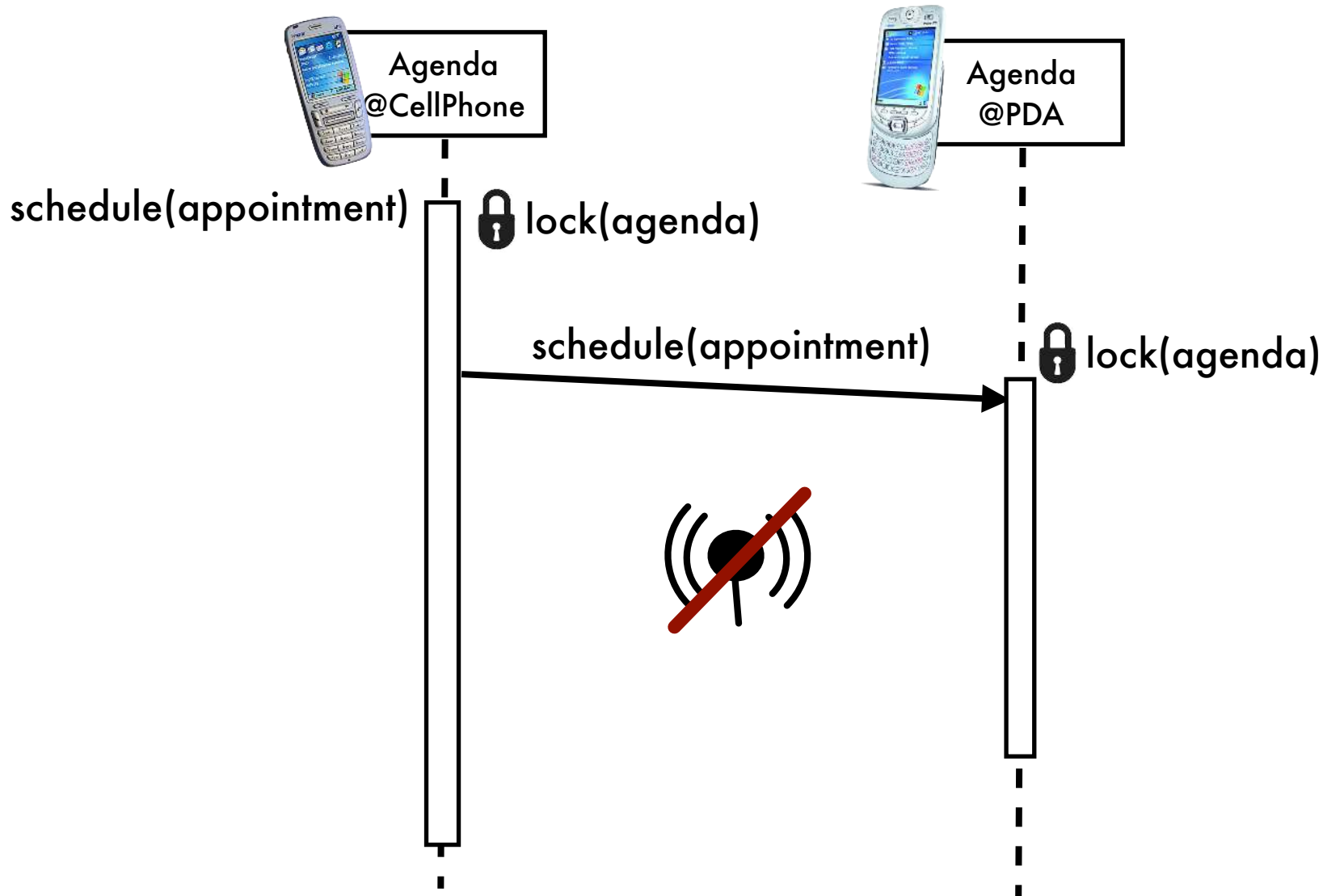
# Blocking Communication



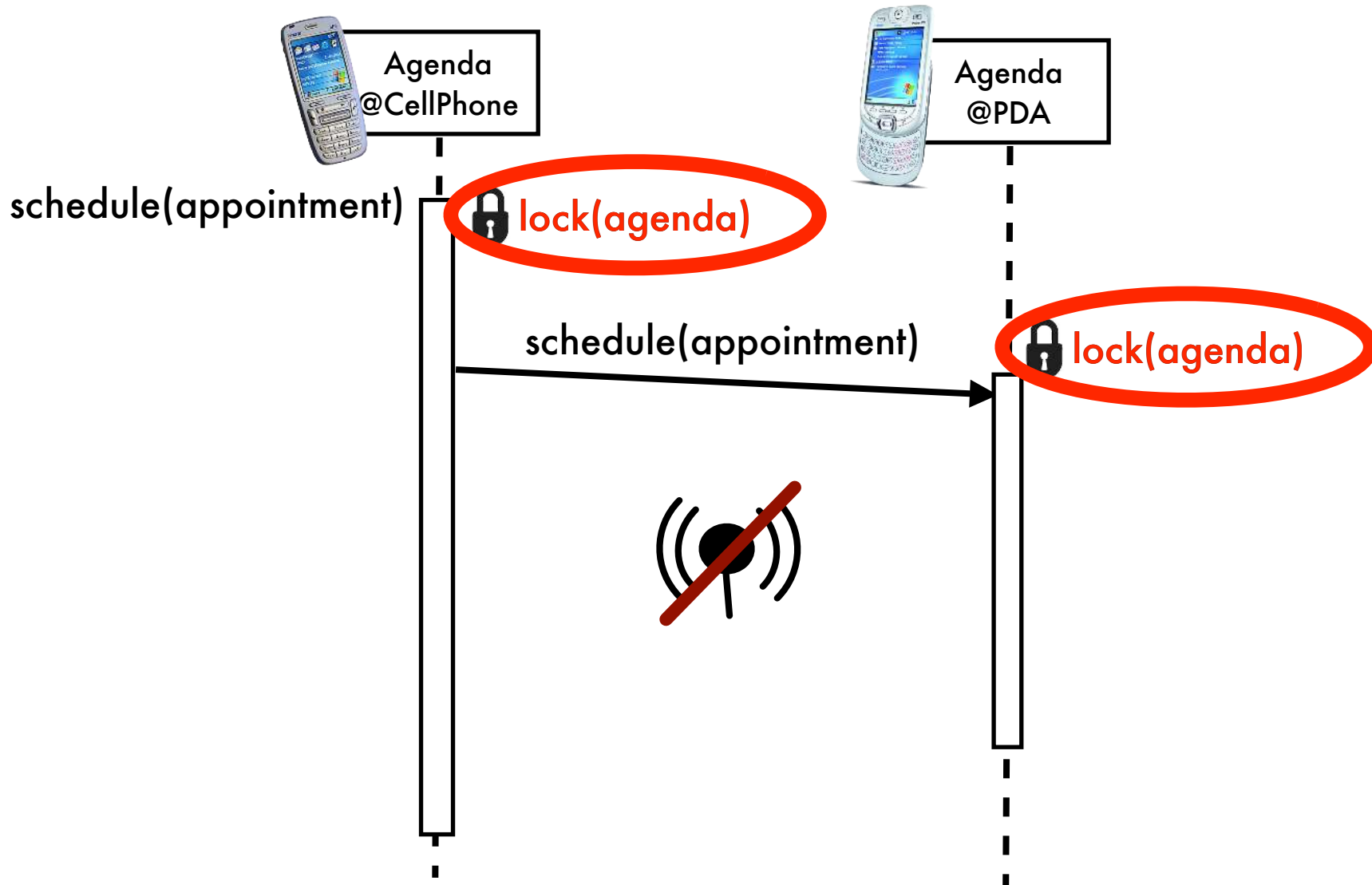
# Blocking Communication



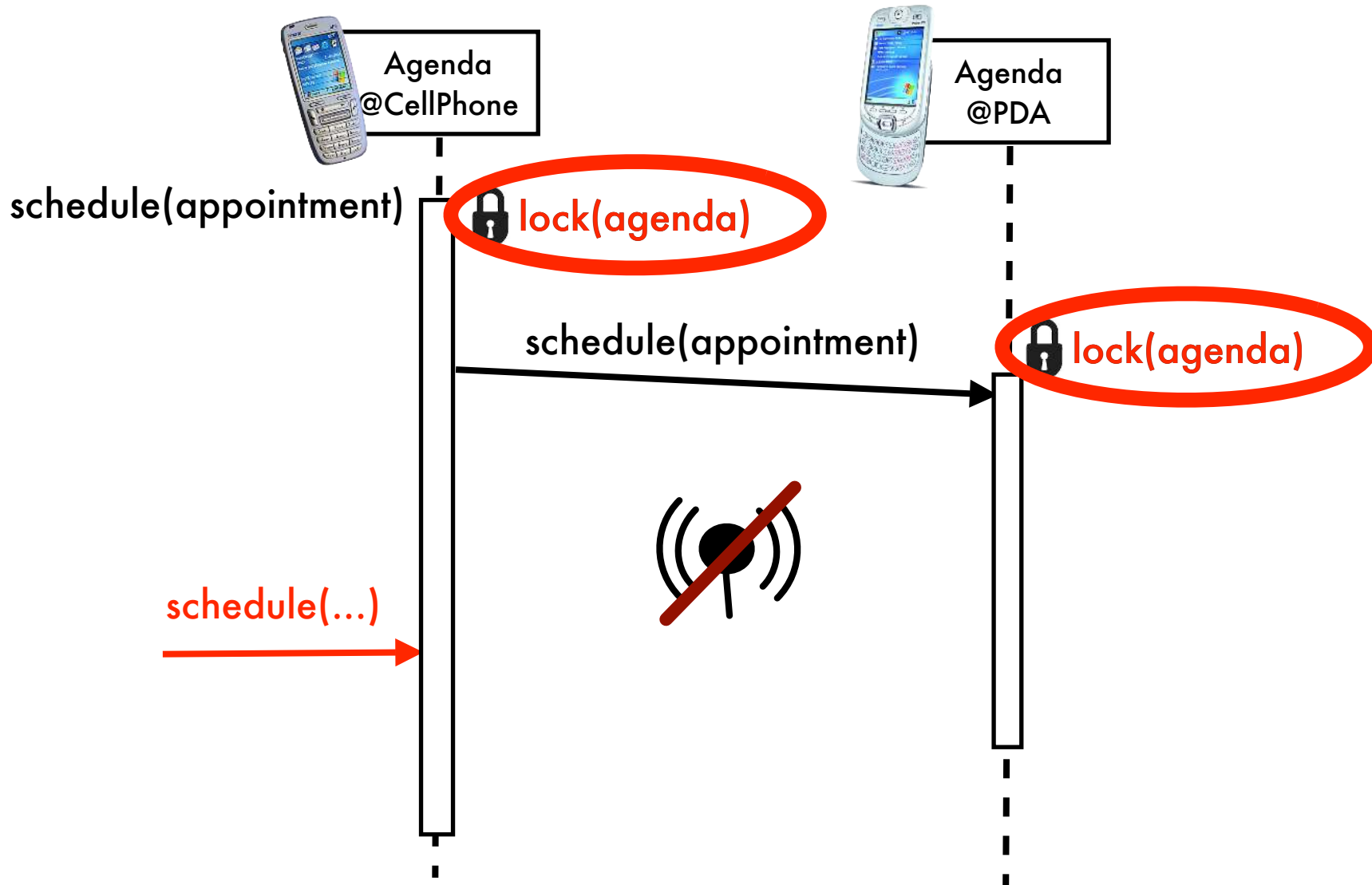
# Blocking Communication



# Blocking Communication

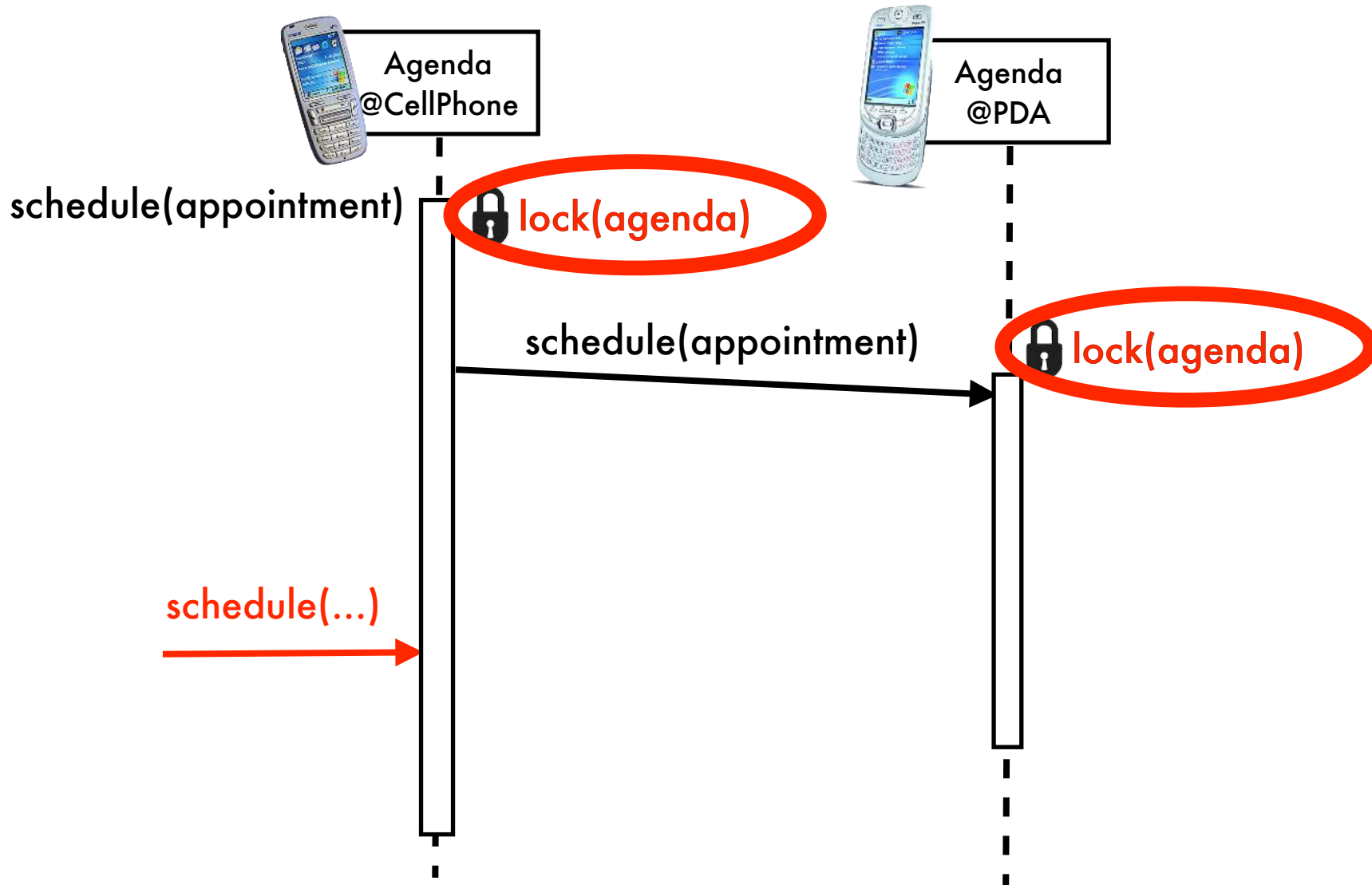


# Blocking Communication

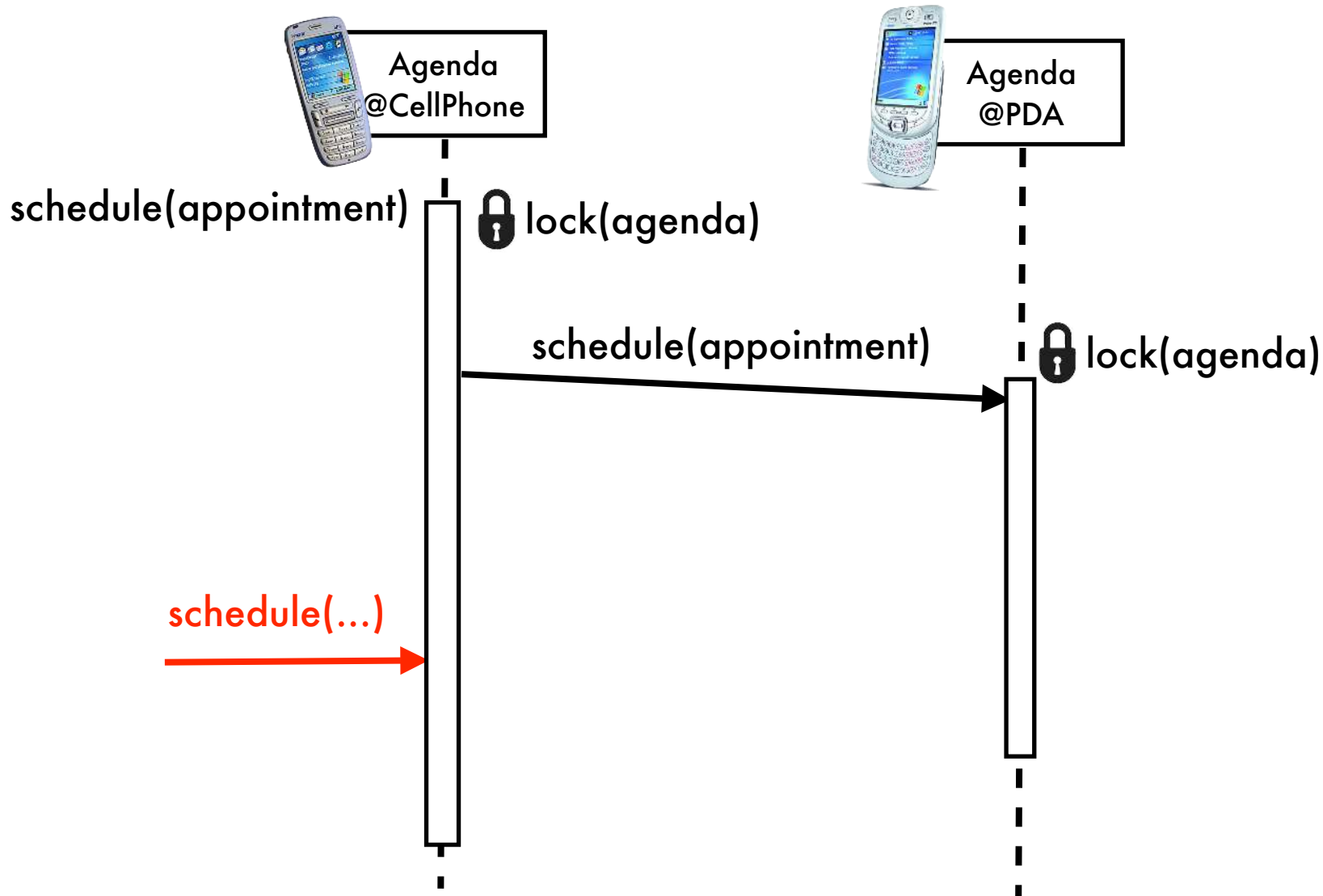




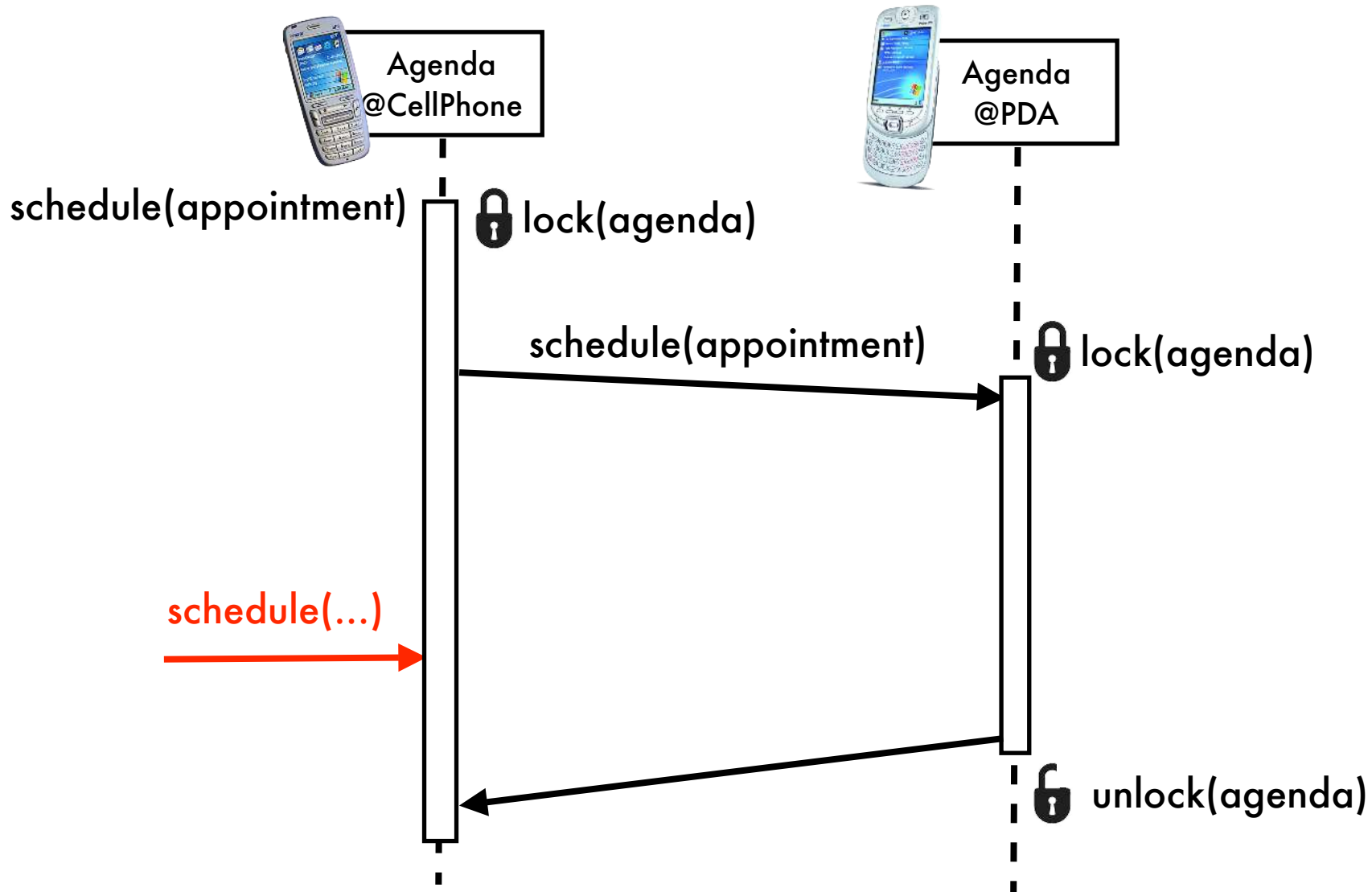
# Blocking Communication



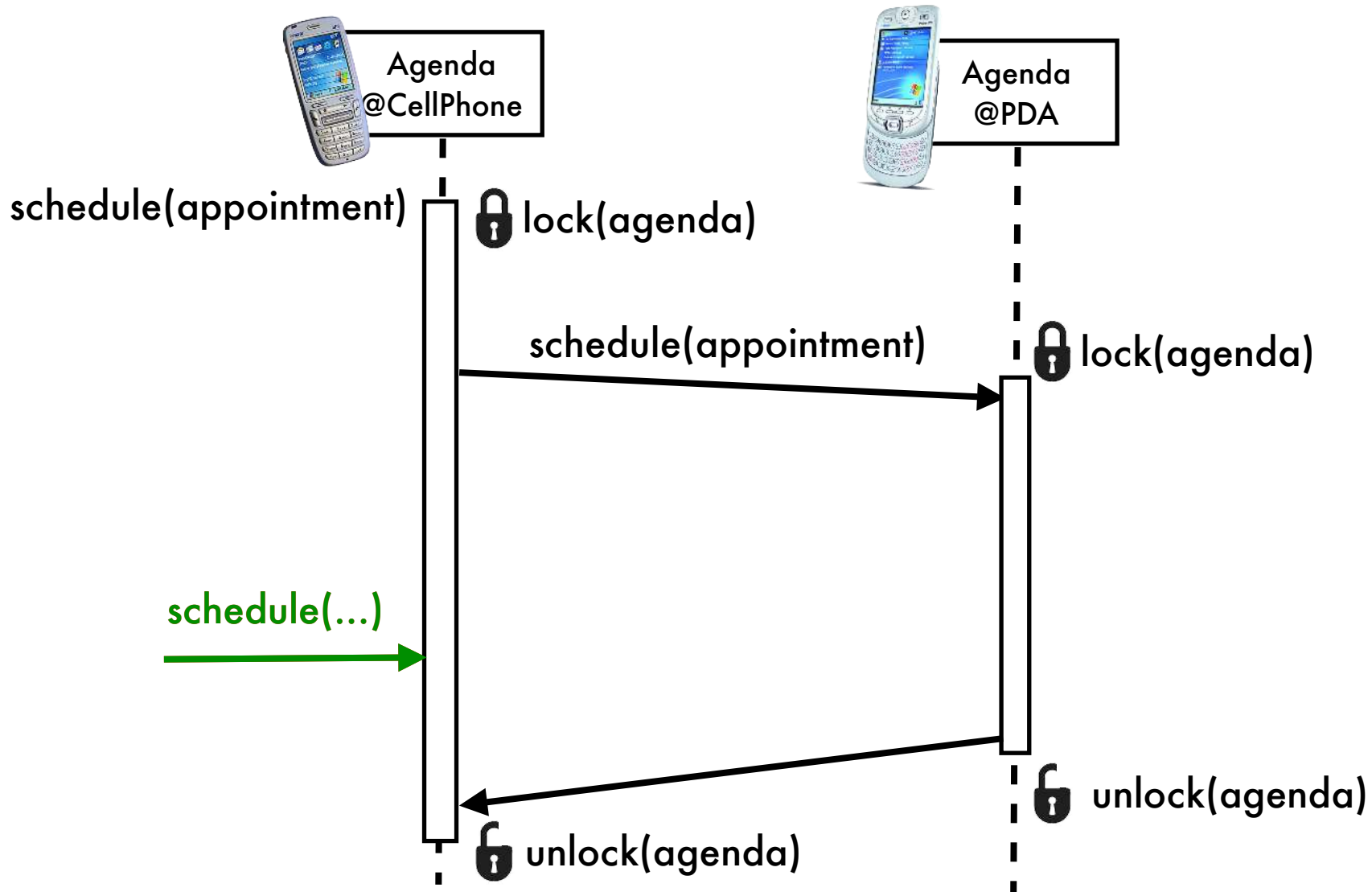
# Blocking Communication



# Blocking Communication



# Blocking Communication



# Non-blocking communication

Decouples message sending from transmission

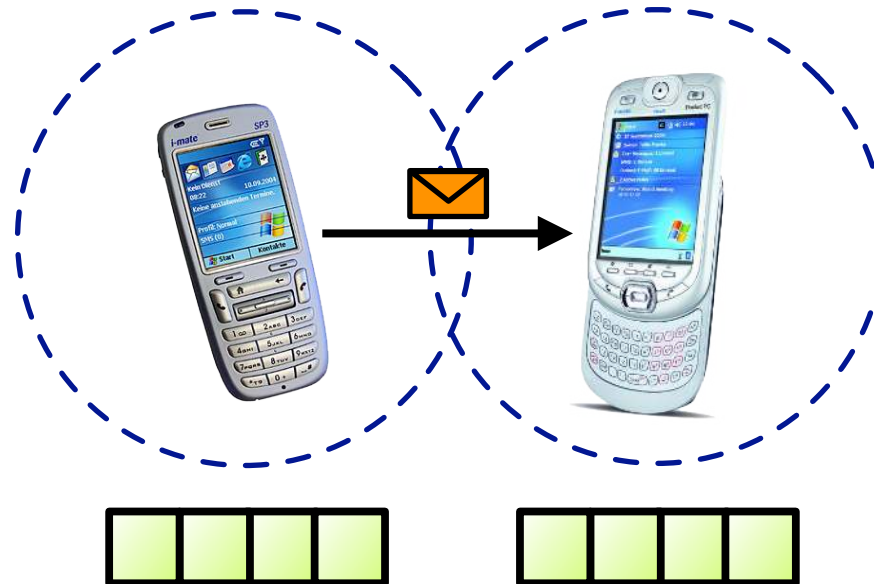
Resilient to disconnections



# Non-blocking communication

Decouples message sending from transmission

Resilient to disconnections



# Non-blocking communication

Decouples message sending from transmission

Resilient to disconnections



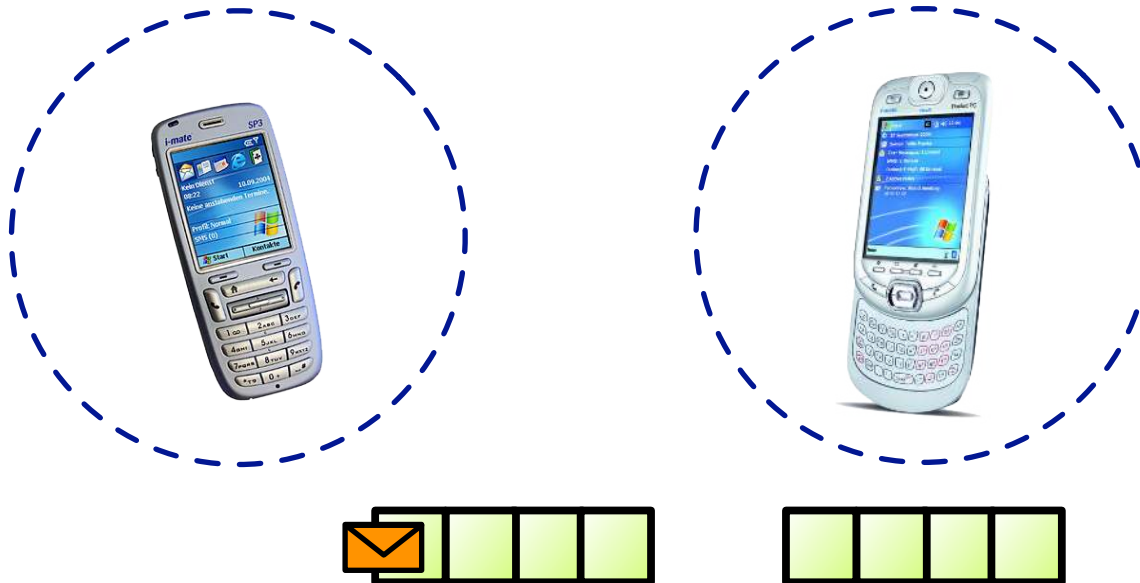
asynchronous  
send



# Non-blocking communication

Decouples message sending from transmission

Resilient to disconnections





# Non-blocking communication

Decouples message sending from transmission

Resilient to disconnections



# Non-blocking communication

Decouples message sending from transmission

Resilient to disconnections



# Non-blocking communication

Decouples message sending from transmission

Resilient to disconnections



asynchronous  
receive

# Non-Blocking Communication



Agenda  
@CellPhone

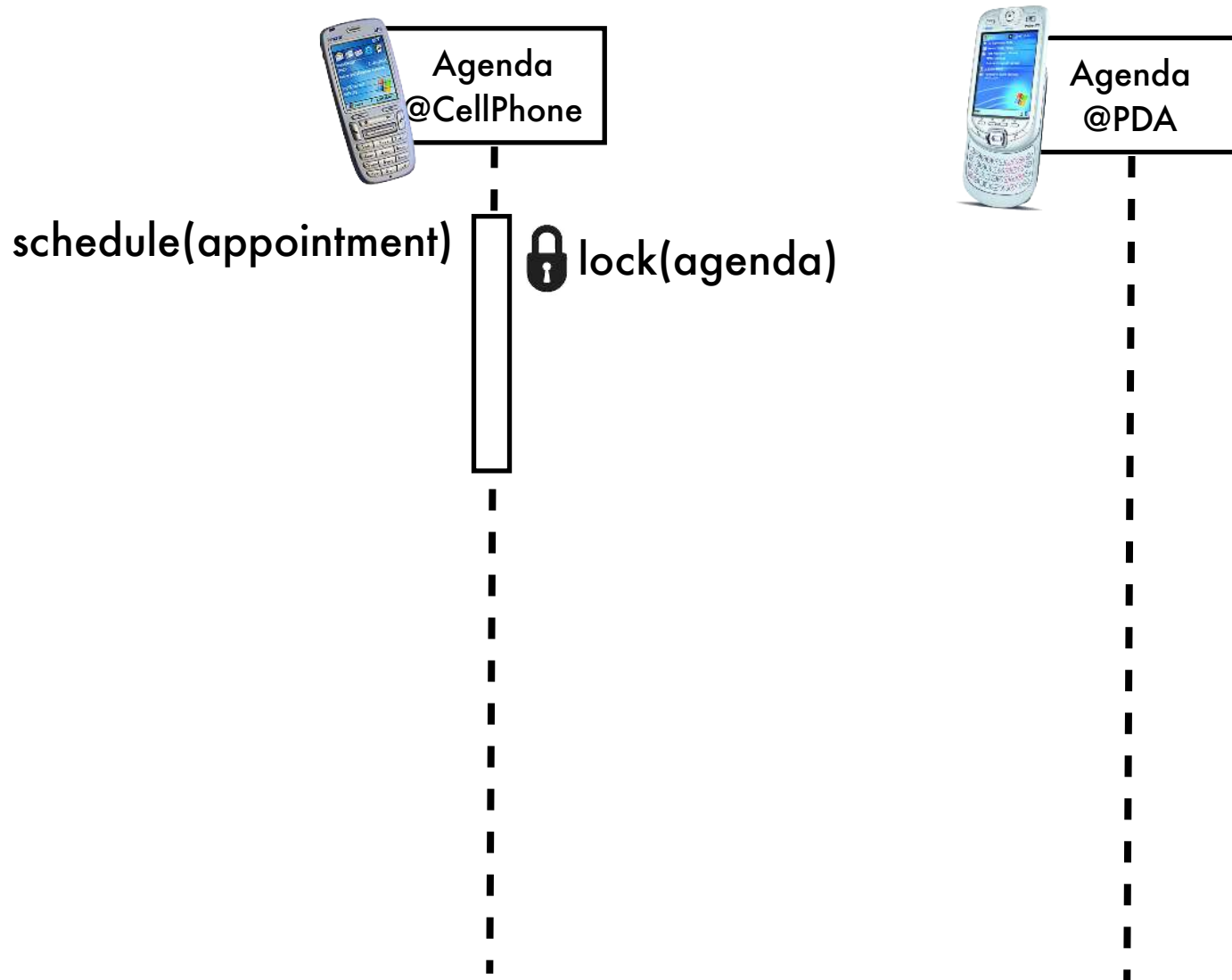
schedule(appointment)



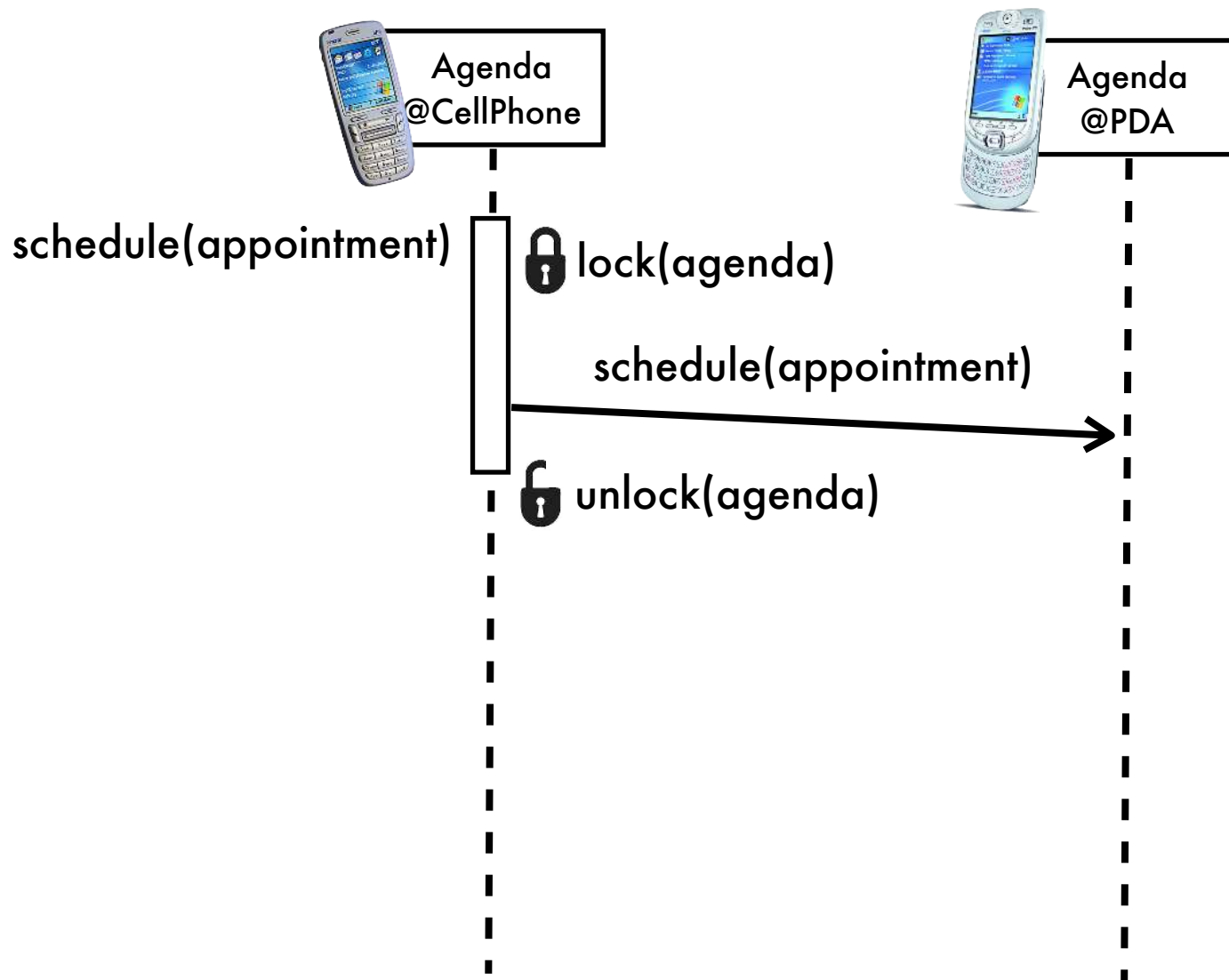
Agenda  
@PDA



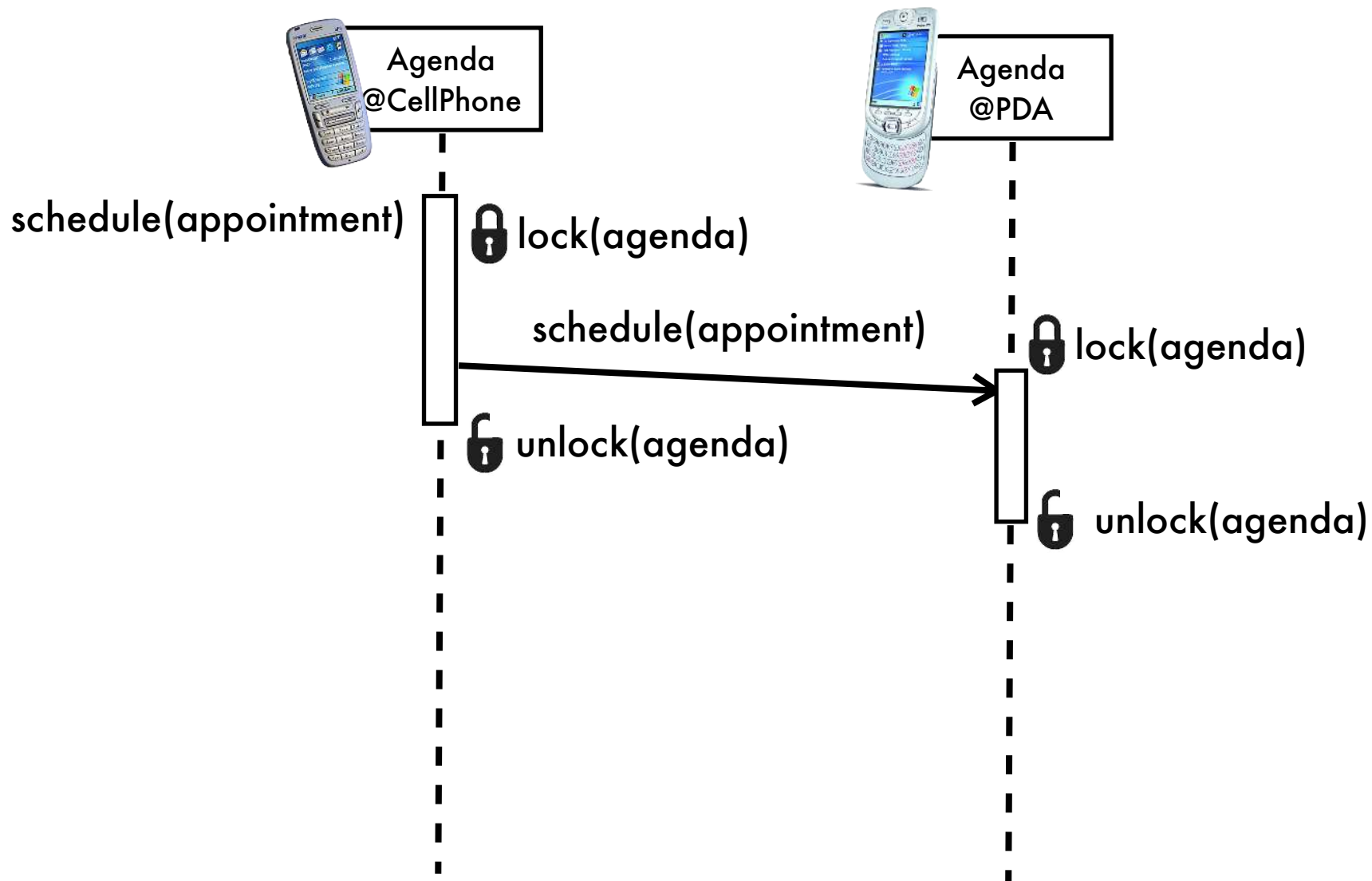
# Non-Blocking Communication



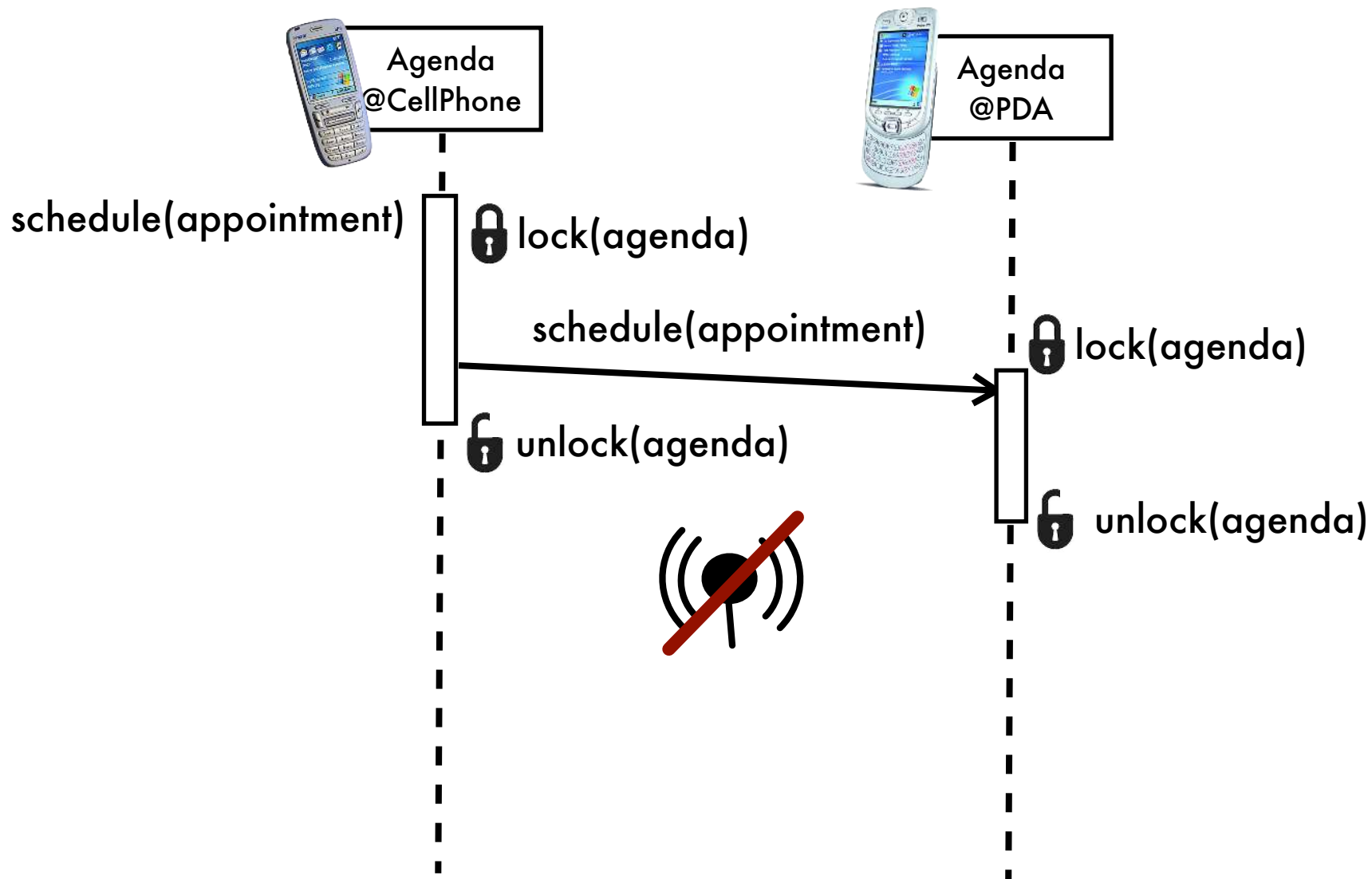
# Non-Blocking Communication



# Non-Blocking Communication

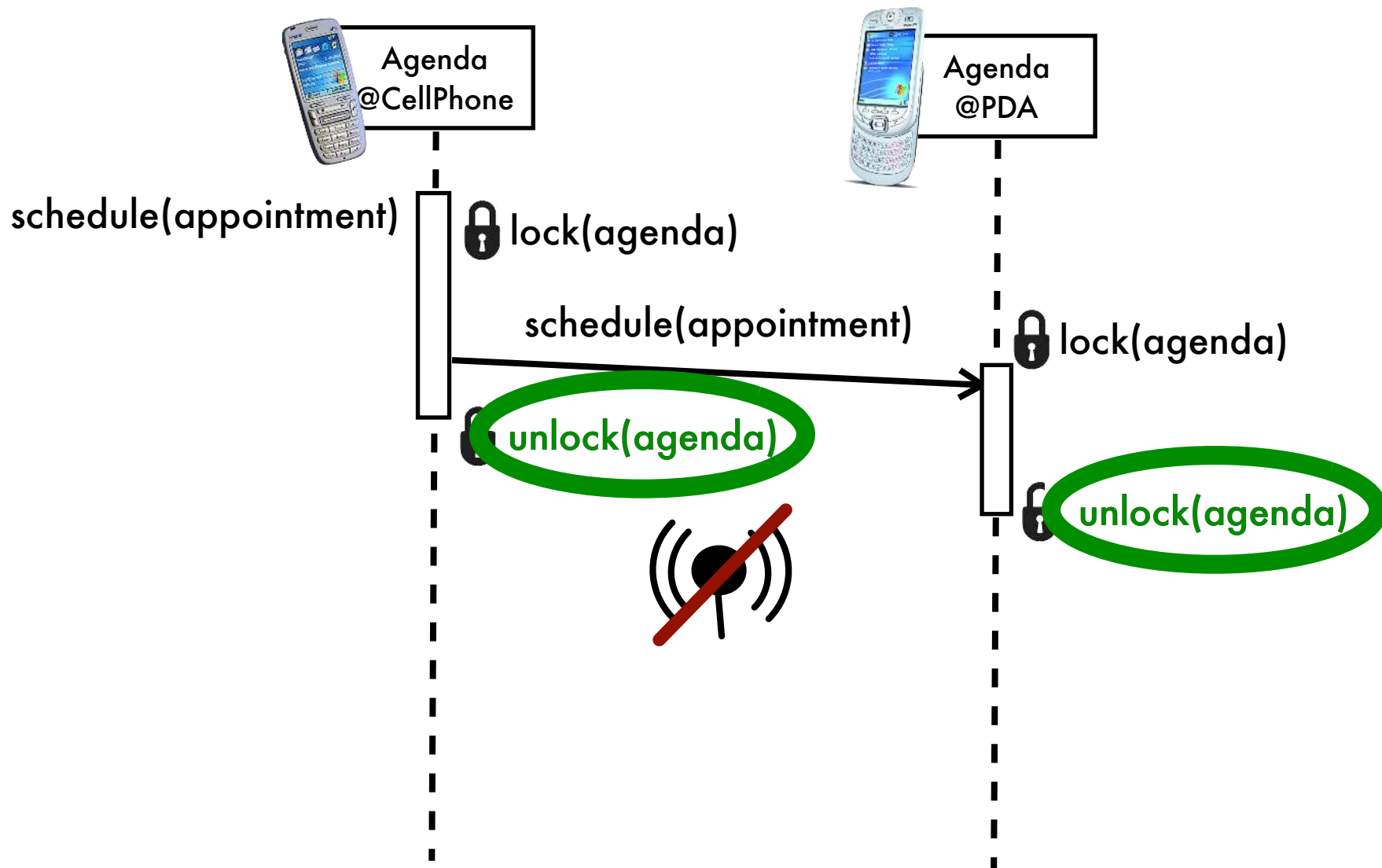


# Non-Blocking Communication

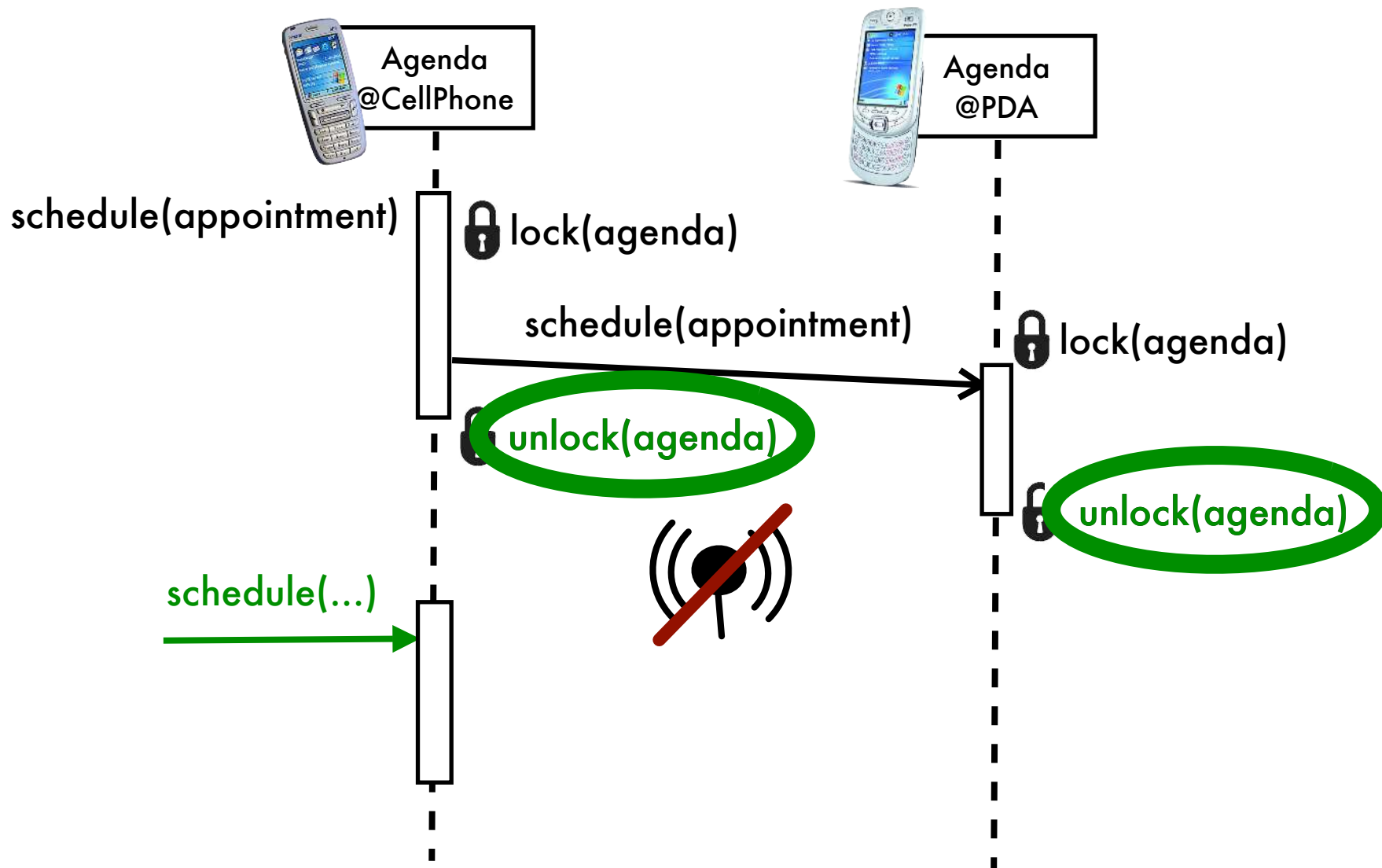




# Non-Blocking Communication



# Non-Blocking Communication



# Non-Blocking Communication

- Non-blocking **send**: asynchronous
- Non-blocking **receive**: event-driven
- Communication **!=** synchronisation

Ambient-Oriented Programming deals with:



Autonomous Concurrent Devices



Volatile Connections

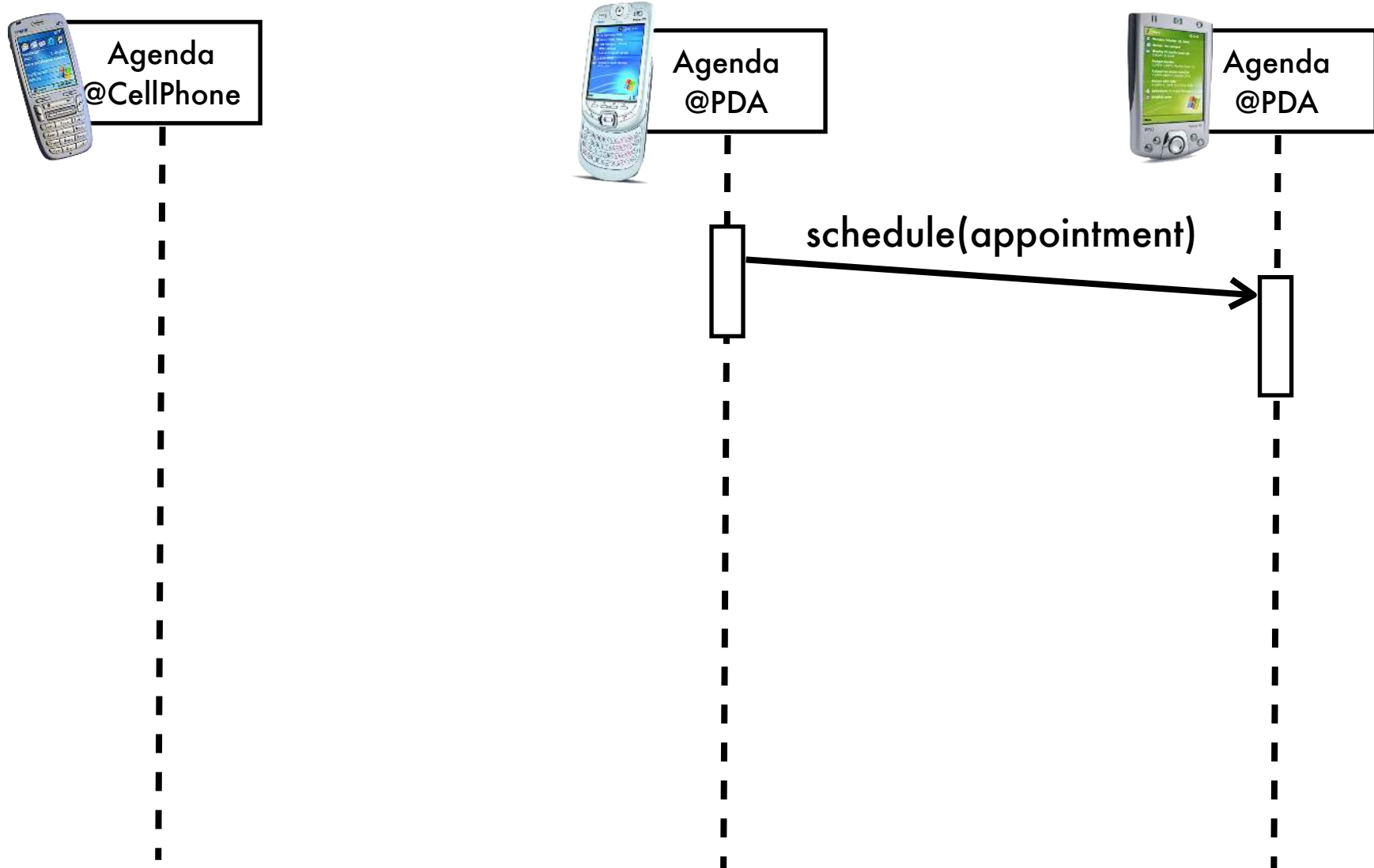
# Reified Communication Traces

- A representation of **past** and **future** process communication
- Why? **Synchronization**: rollback, retry, cancel, postpone, replicate communication

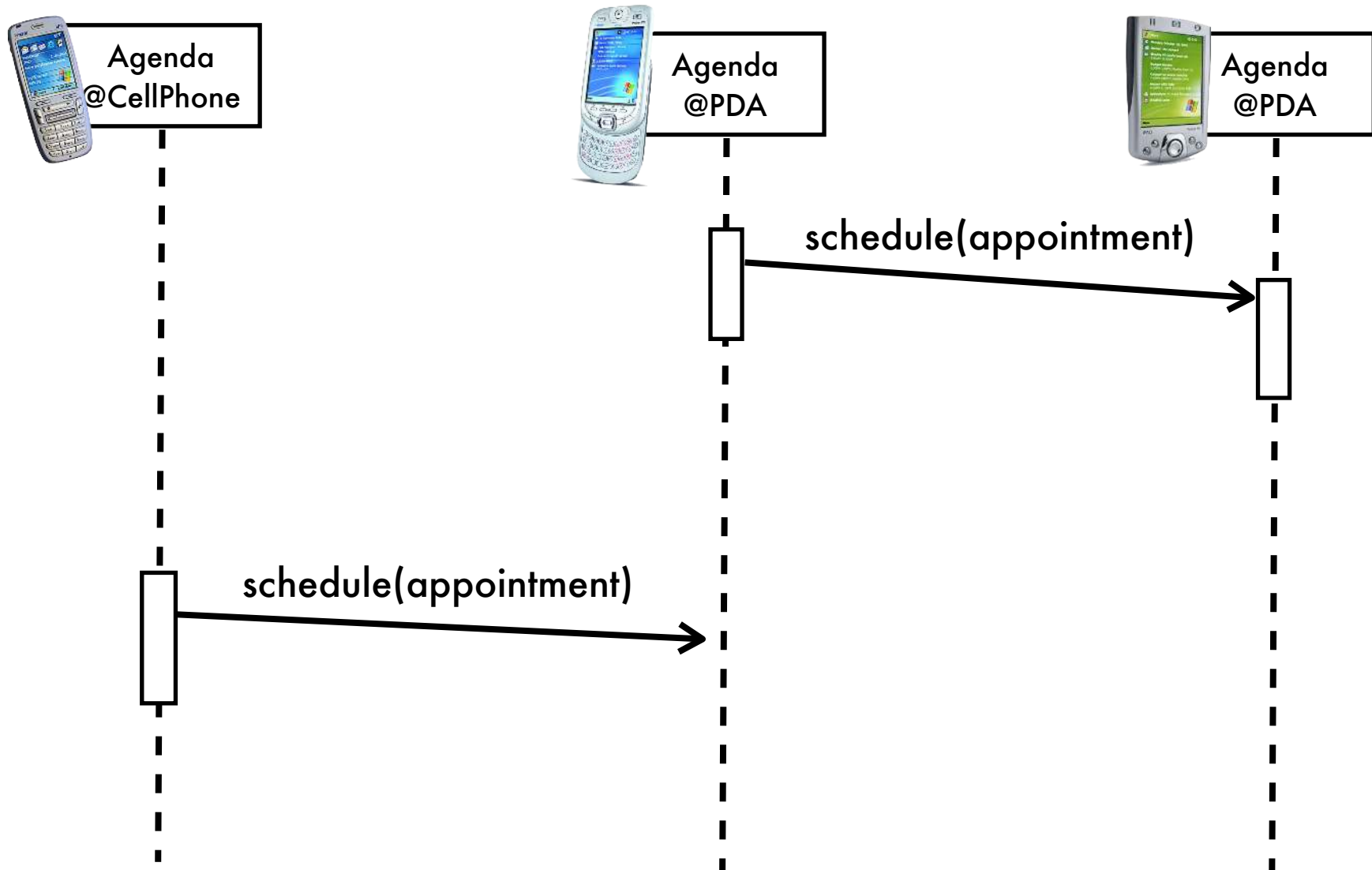
# Example: rollback



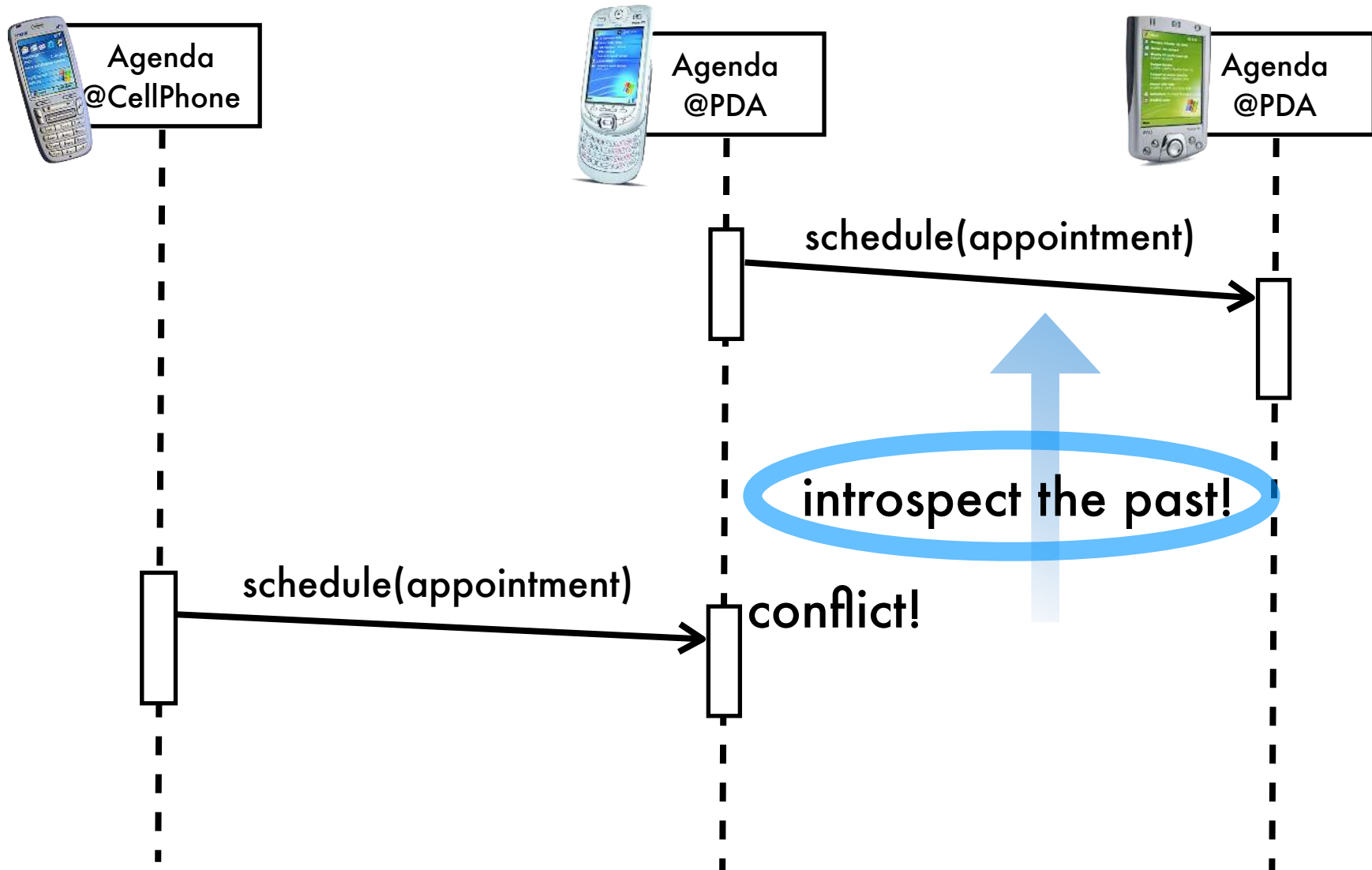
# Example: rollback



# Example: rollback

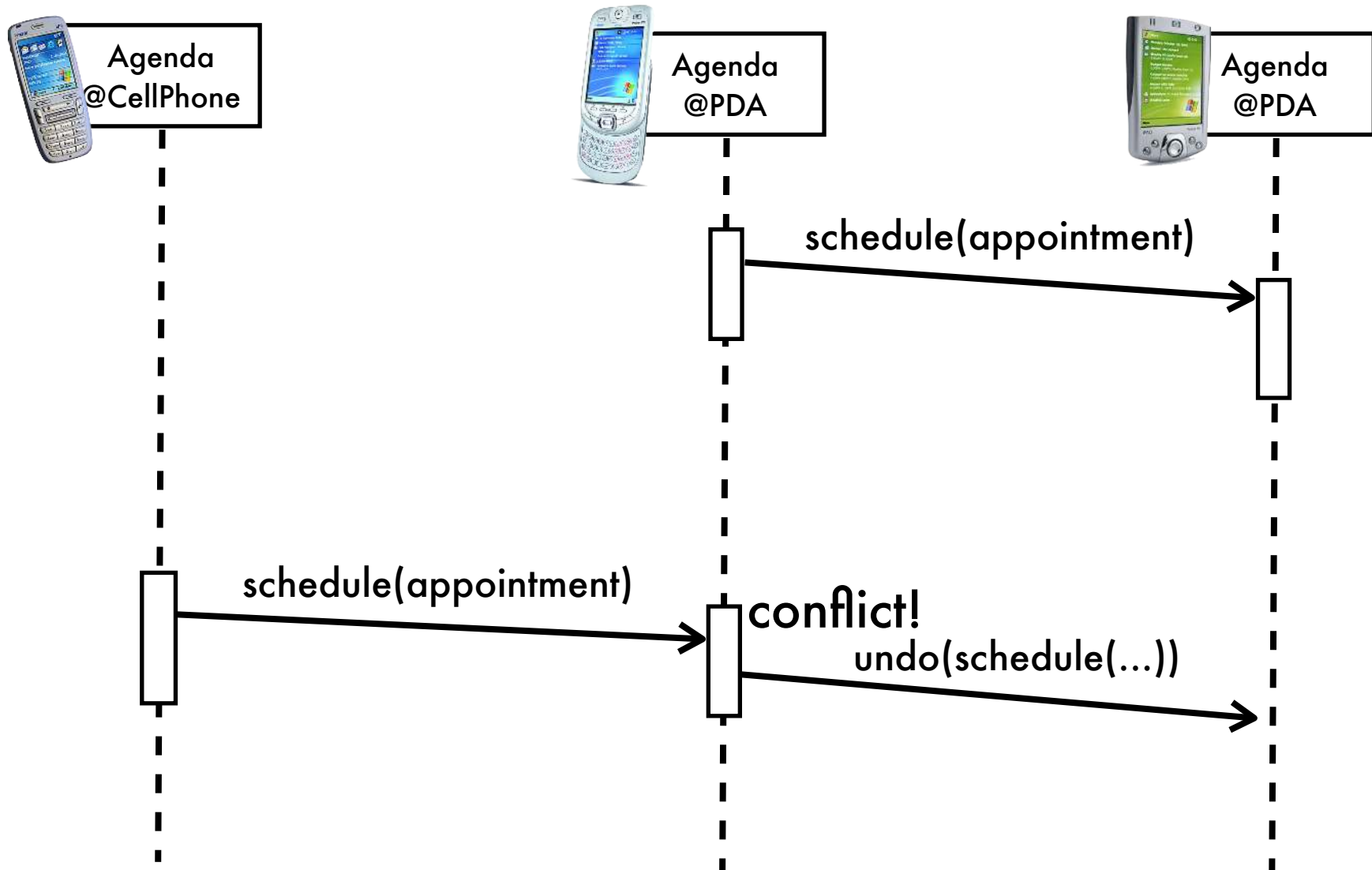


# Example: rollback

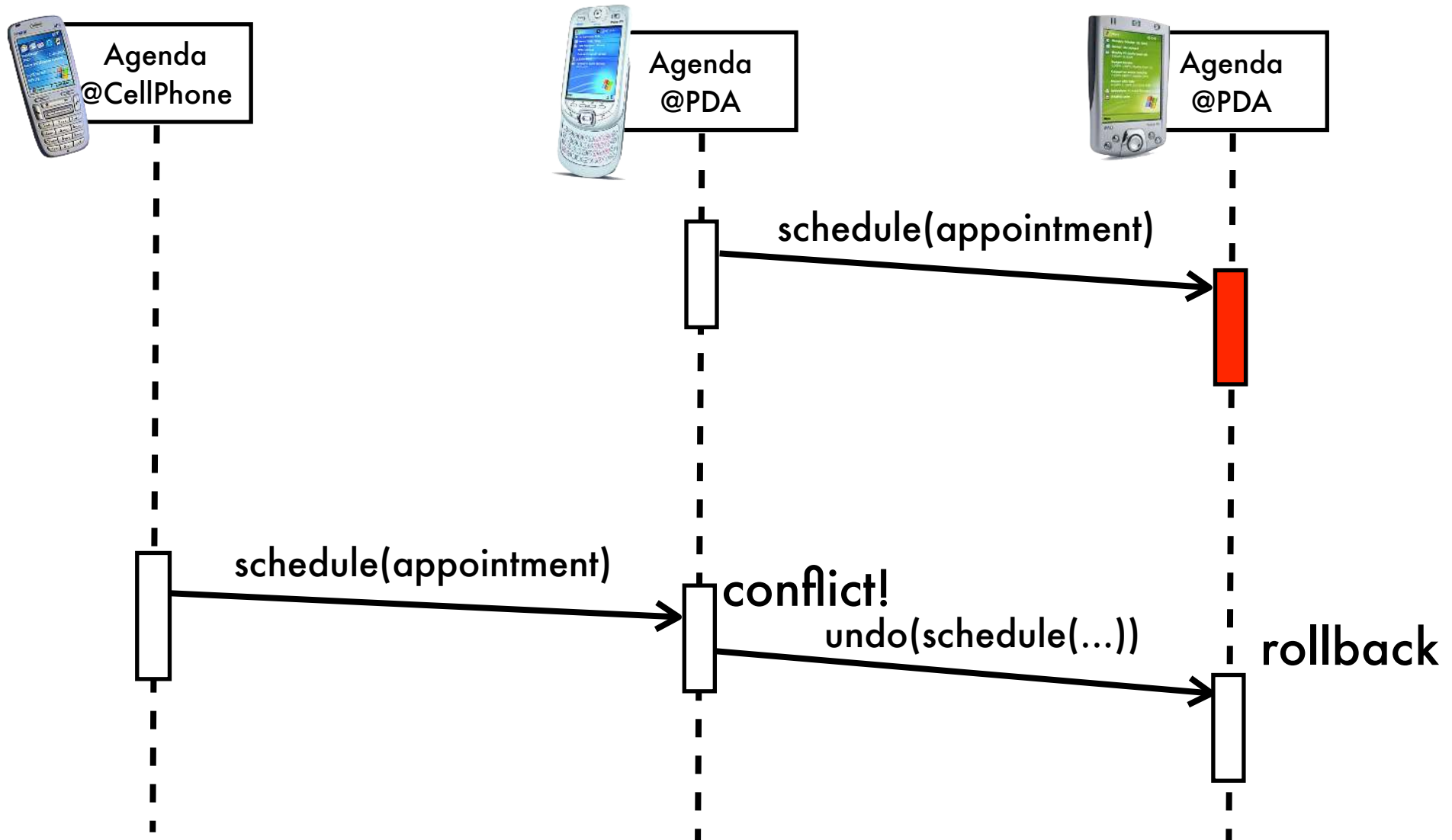




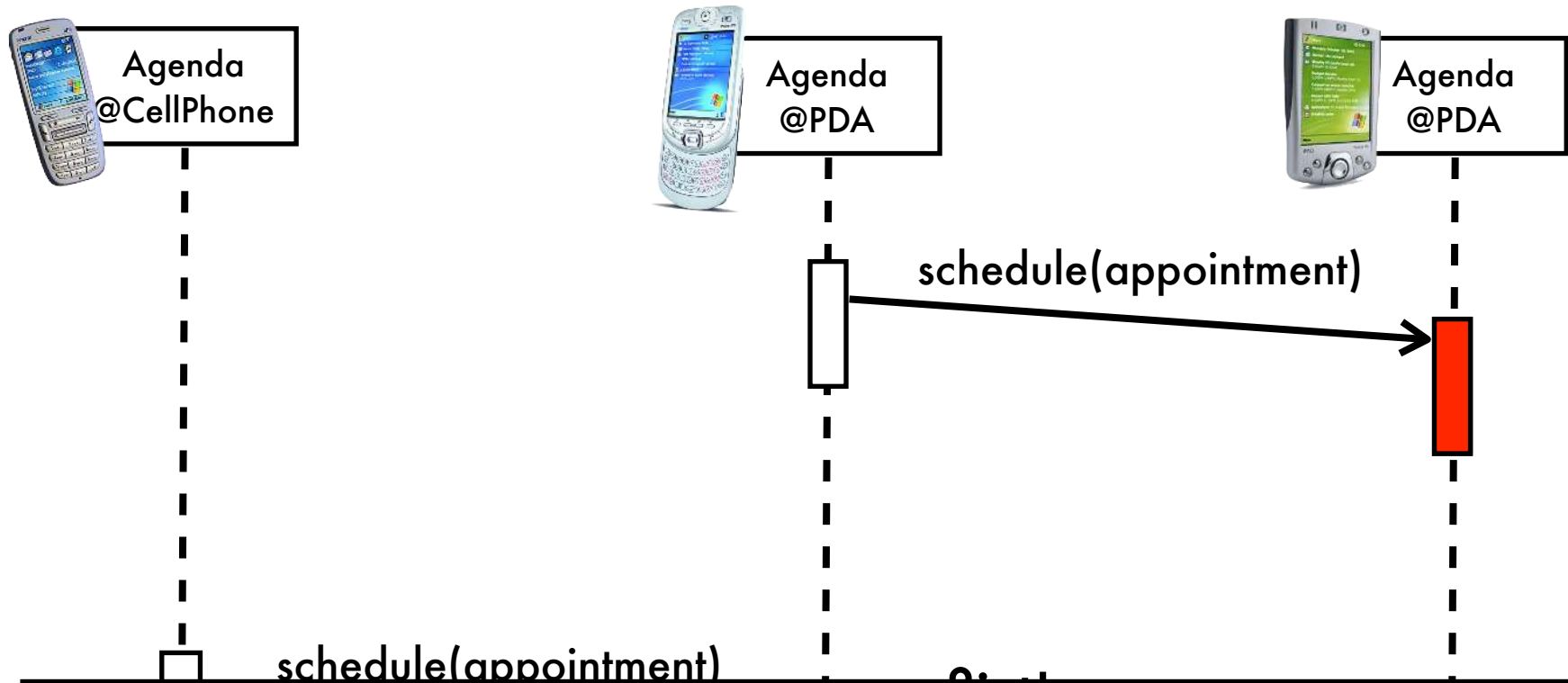
# Example: rollback



# Example: rollback



# Example: rollback



Ambient-Oriented Programming deals with:



Autonomous Concurrent Devices

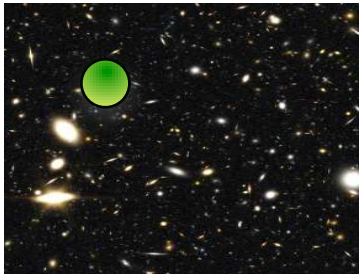


Volatile Connections

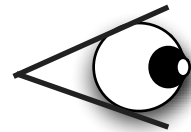
back

# Reified Environmental Context

Software



Ambient

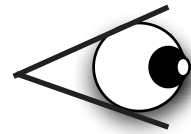


# Reified Environmental Context

Software

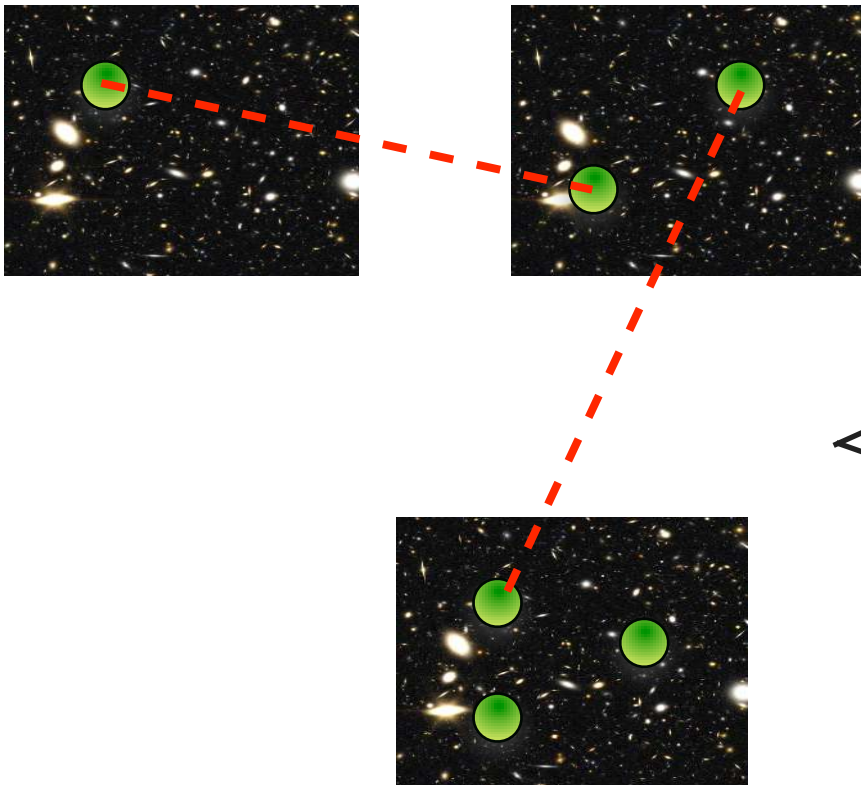


Ambient



# Reified Environmental Context

Software



Ambient

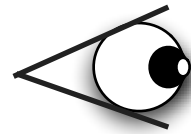


# Reified Environmental Context

Software



Ambient



# Reified Environmental Context

Software



Ambient



Ambient-Oriented Programming deals with:



Ambient Resources



# Ambient-Oriented Programming

- Object-based Languages
- Non-Blocking Communication
- Reified Communication Traces
- Reified Environmental Context

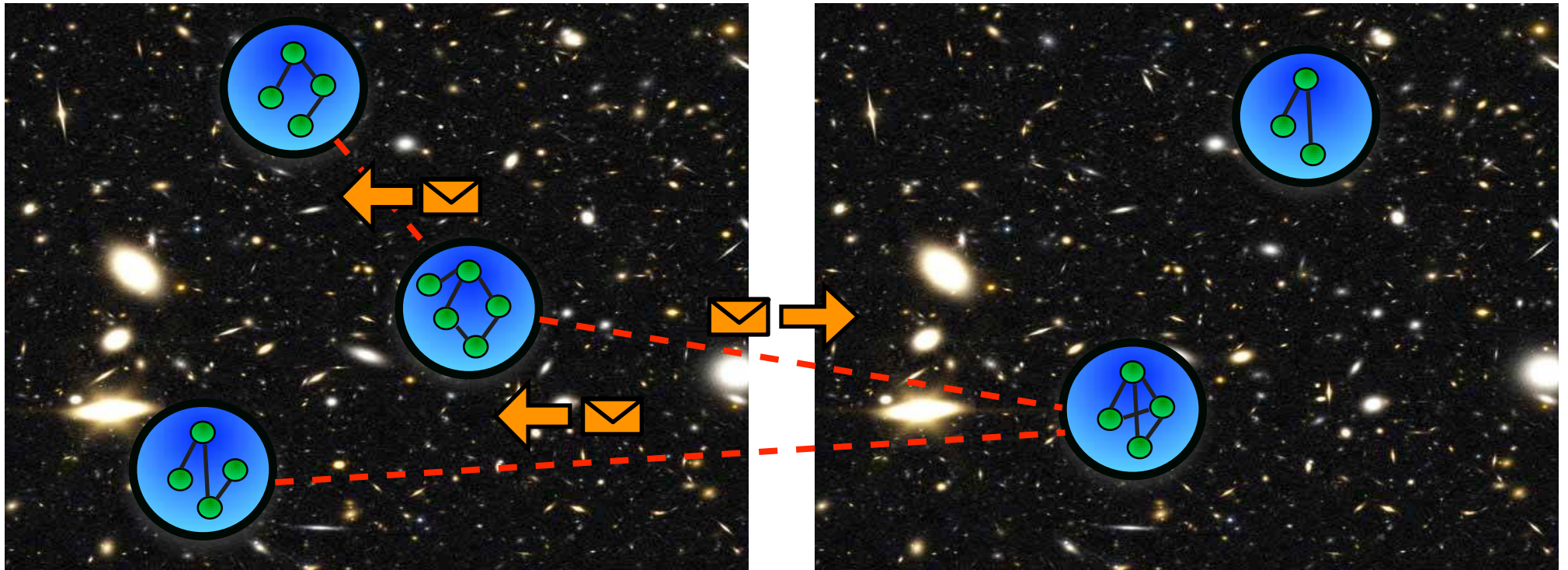


# AmbientTalk

# AmbientTalk in a Nutshell

- Object-based language
- Based on Agha & Hewitt's **actor** model
- Asynchronous message passing
- First-class message queues (**mailboxes**)

# Objects and Actors



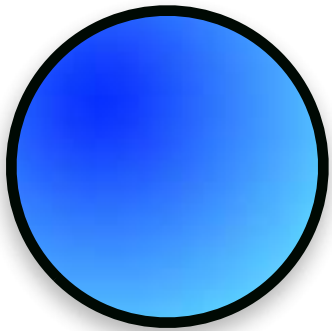
actors encapsulate **objects**

**asynchronous** message passing

Invited Talk SEPS at ICPS, Lyon, June 2006 (c) Programming Technology Lab, Vrije Universiteit Brussel, Belgium (2006)

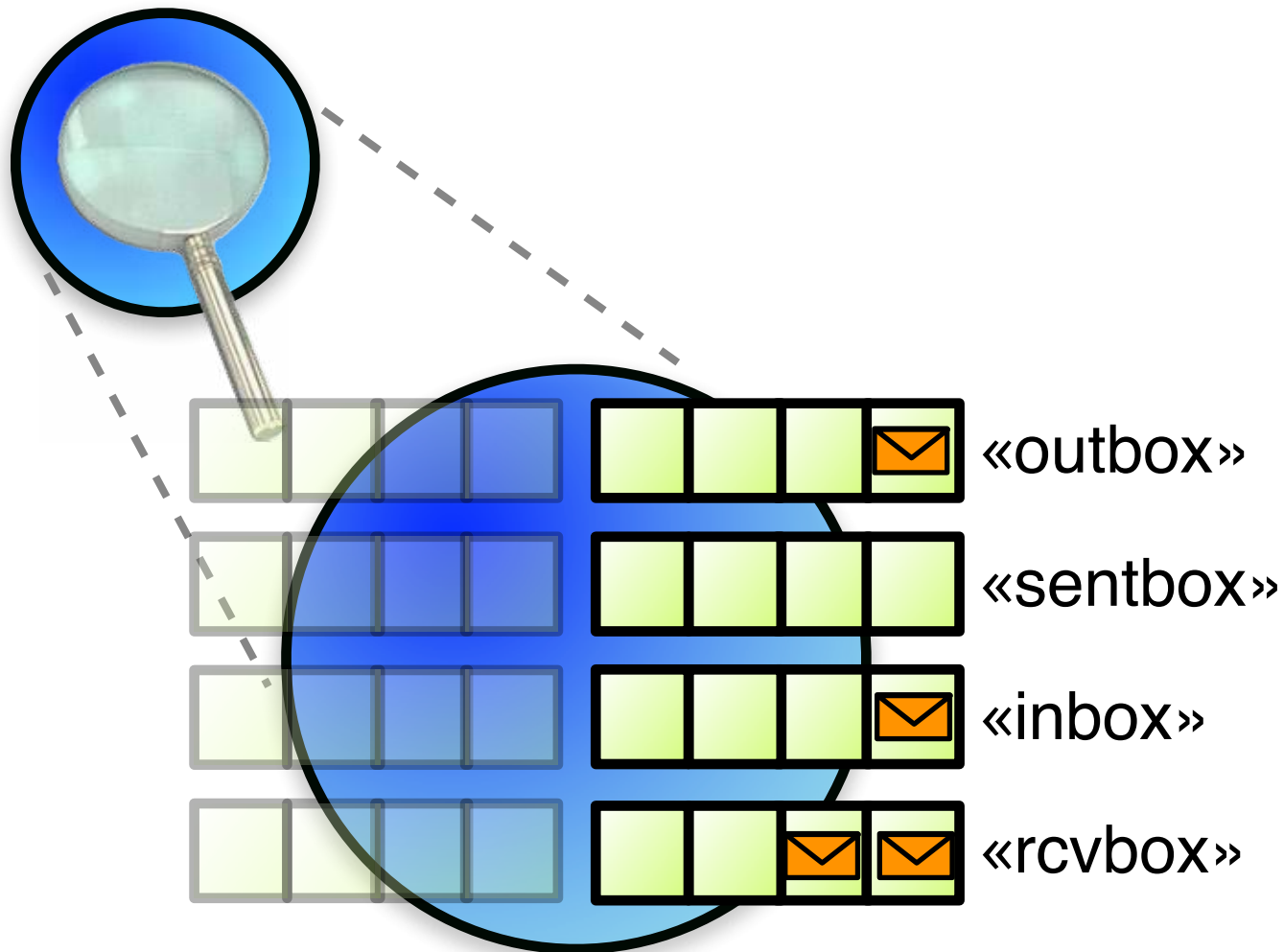
# First-class Mailboxes

## 1) Reified Communication Traces



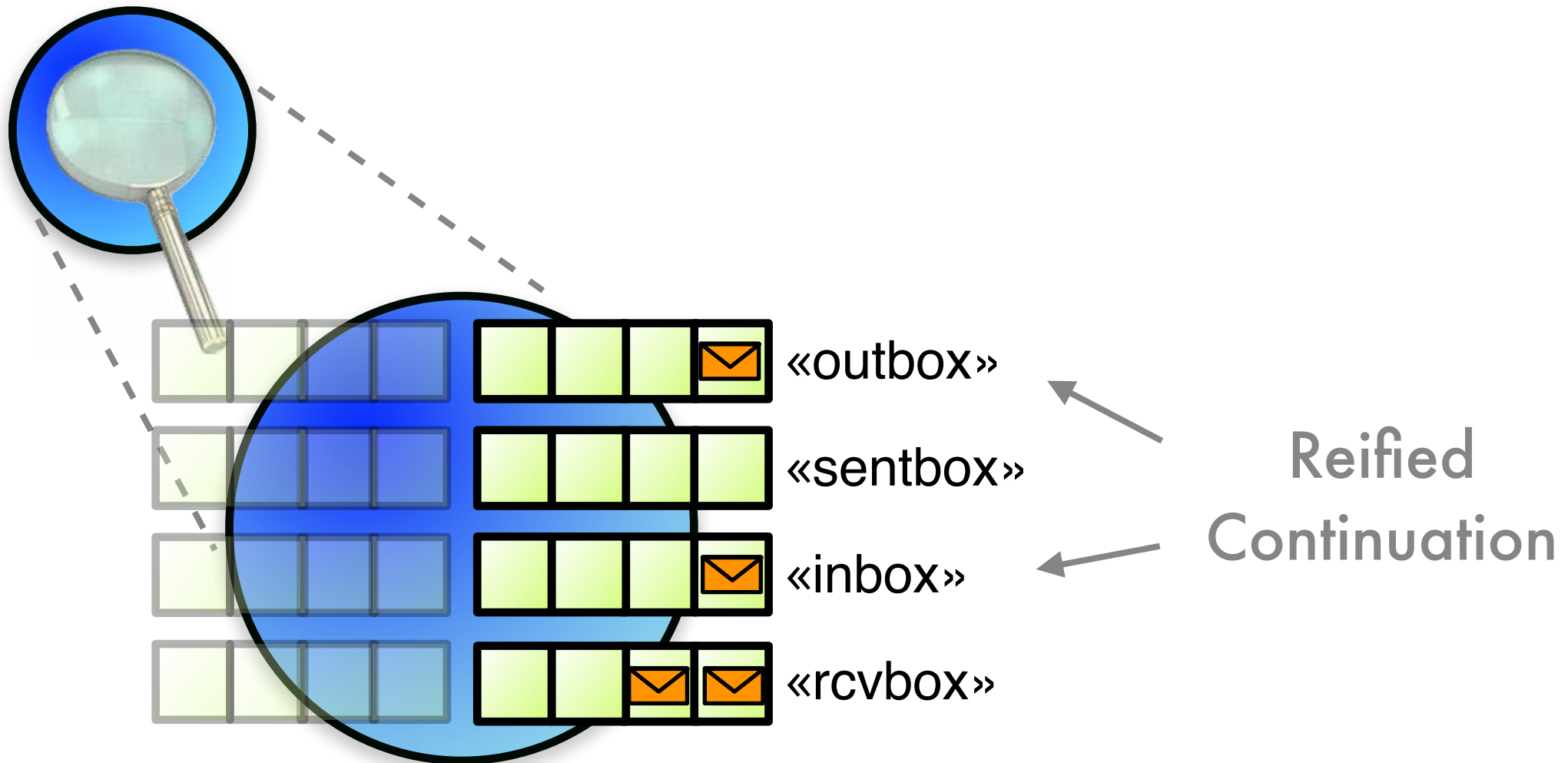
# First-class Mailboxes

## 1) Reified Communication Traces



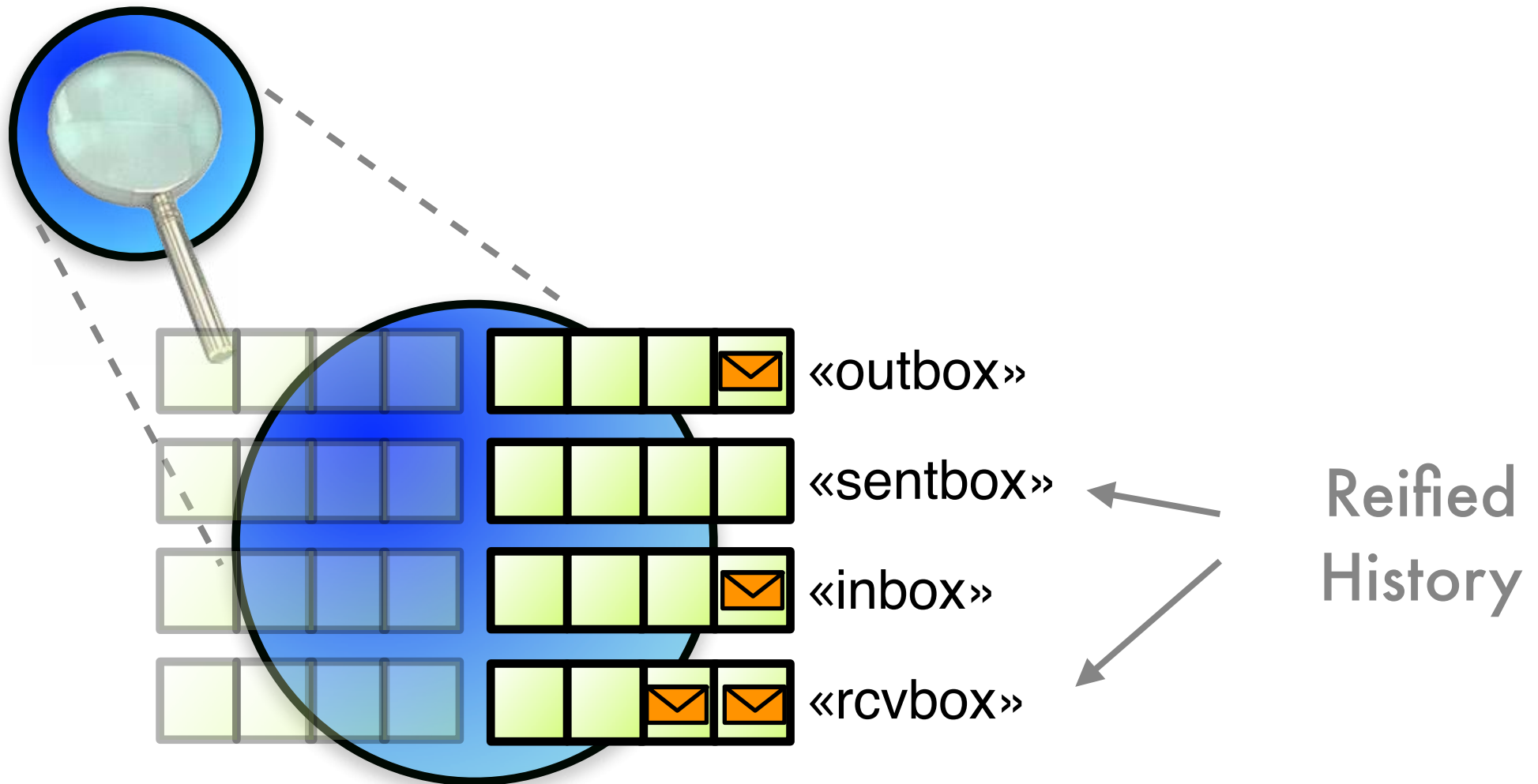
# First-class Mailboxes

## 1) Reified Communication Traces



# First-class Mailboxes

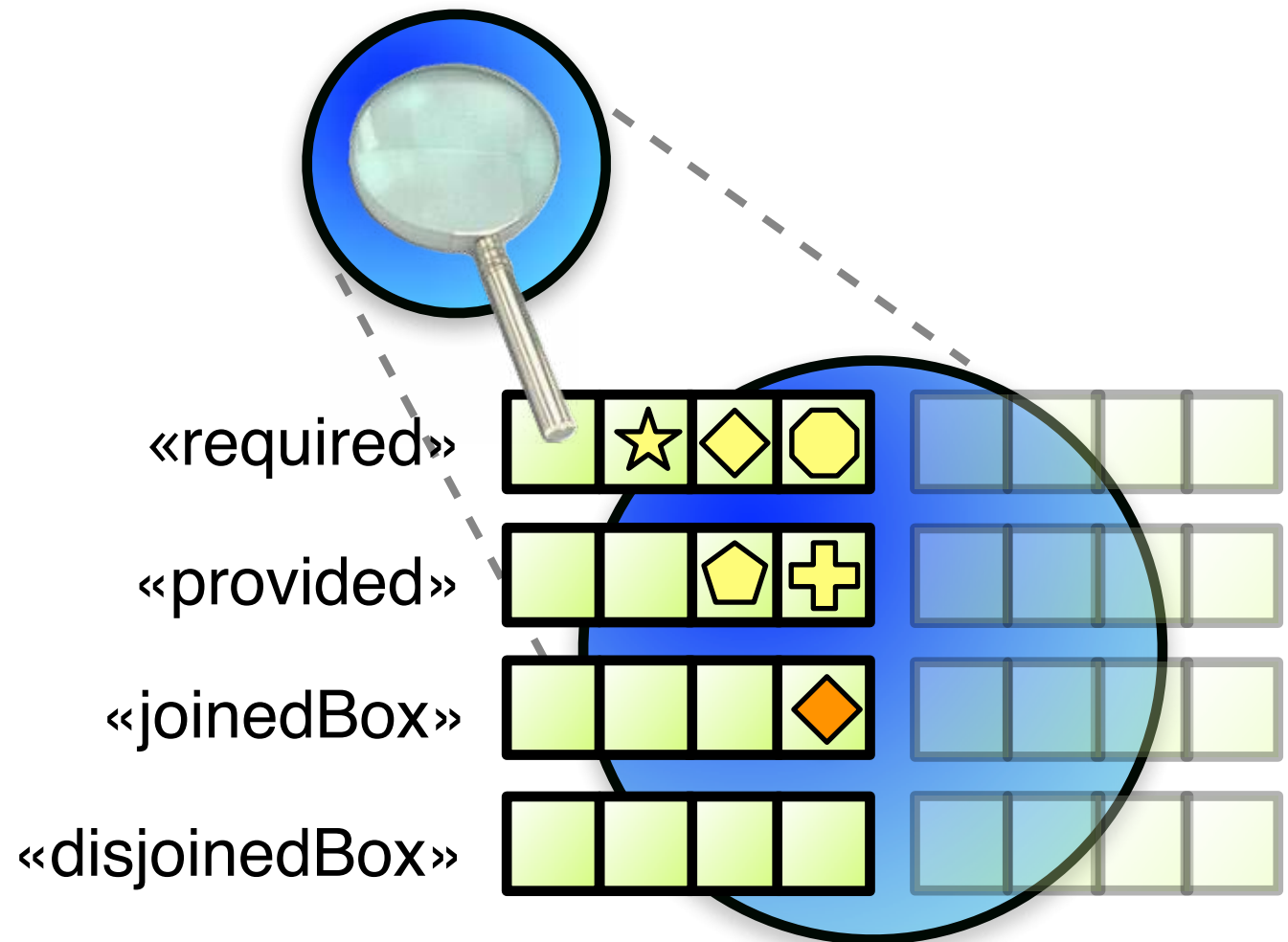
## 1) Reified Communication Traces





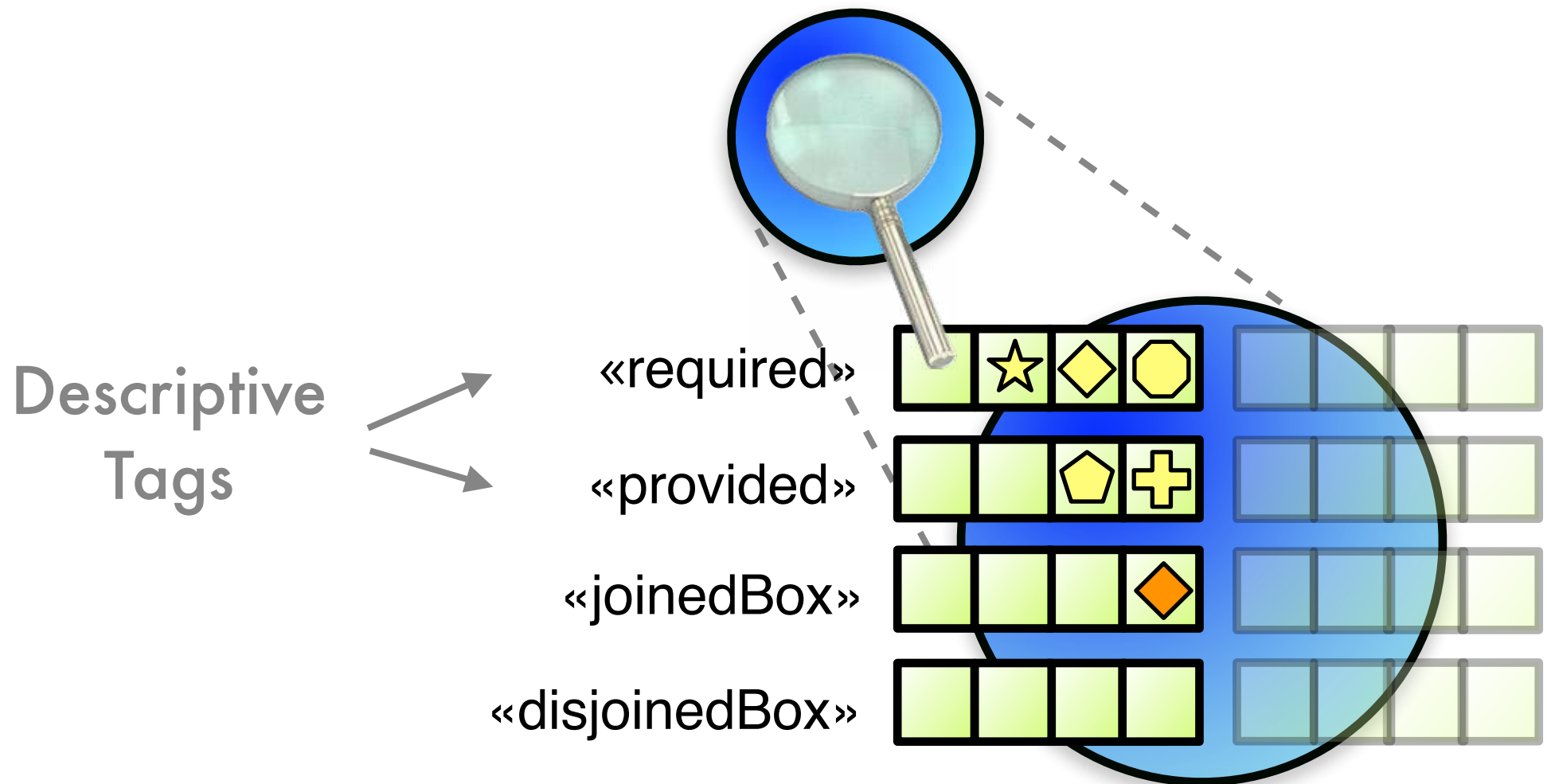
# First-class Mailboxes

## 2) Reified Environmental Context



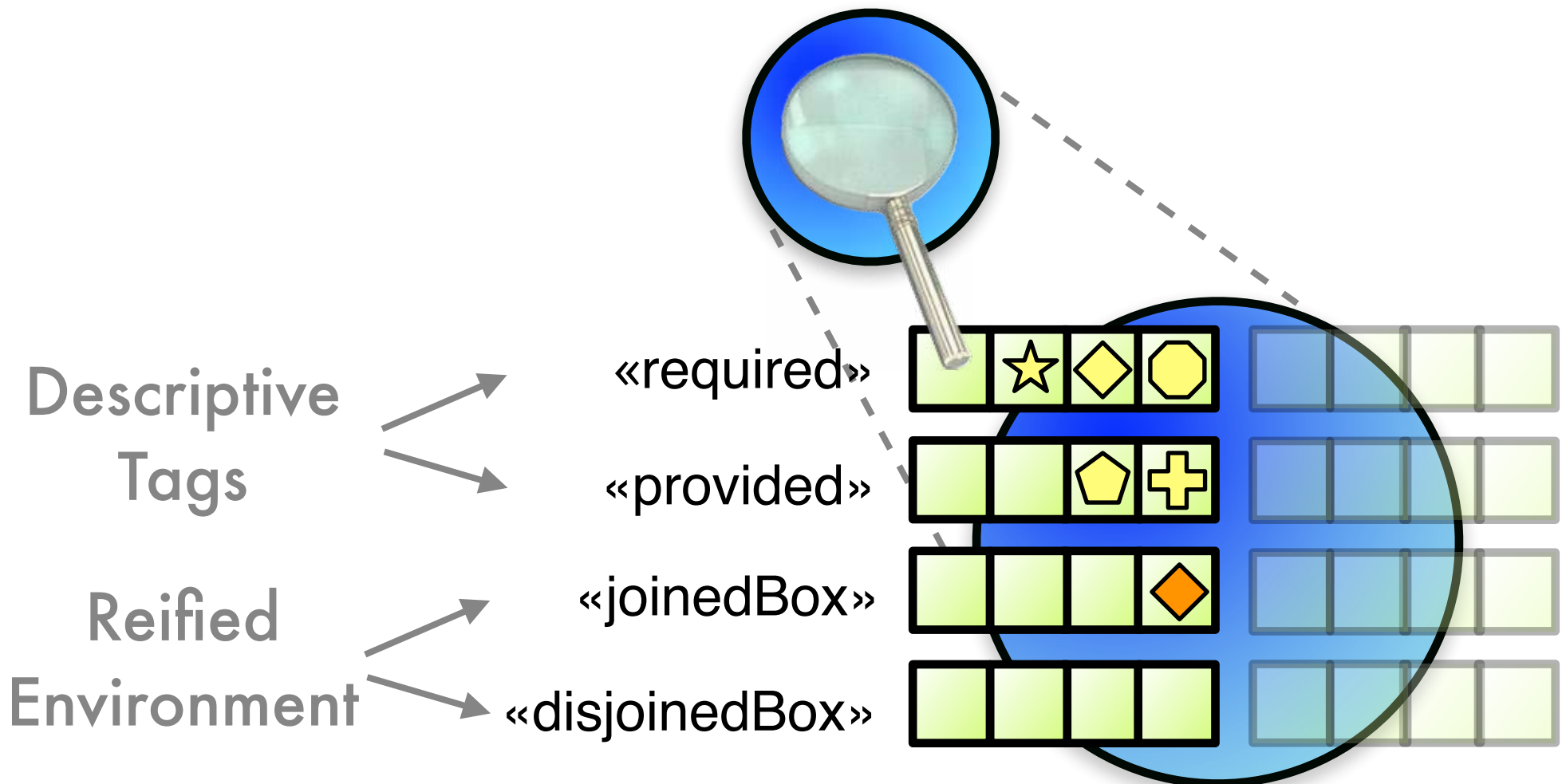
# First-class Mailboxes

## 2) Reified Environmental Context



# First-class Mailboxes

## 2) Reified Environmental Context



# Language Constructs

AmbientTalk programs

Language  
Constructs

AmbientTalk  
Kernel Language

Metaobject  
Protocol

# Language Constructs

AmbientTalk programs

Language  
Constructs

AmbientTalk  
Kernel Language



Metaobject  
Protocol

- Objects and Actors
- Non-blocking communication
- P2P Discovery

# Language Constructs

AmbientTalk programs

Language  
Constructs

AmbientTalk  
Kernel Language

Metaobject  
Protocol



- Access to mailboxes
- Intercept messages
- Syntax extensions

# Language Constructs

AmbientTalk programs

Language  
Constructs

AmbientTalk  
Kernel Language

Metobject  
Protocol

- Futures as return values
- Discovery abstractions
- Broadcast/multicast
- Weak Object Replication
- Failure Handling constructs
- ...

# Implementation

- Interactive Interpreter
- Written in Java
- Runs on top of J2ME









# Wrap-Up

# Conclusion

- Distributed Application Support for Pervasive Computing / AmI
- Hardware: Mobile Networks
  -  (Mobile) Autonomous Devices
  -  Volatile Connections
  -  Ambient Resources

# Conclusion

- Software: AmOP Paradigm
  - Object-based languages
  - Non-blocking communication
  - Reified communication traces
  - Reified environmental context
- AmbientTalk: experimental AmOP language



<http://prog.vub.ac.be/amop>