

# A Simple Infrastructure for Ubiquitous Computing

G. Navarro<sup>1</sup>, J. Peñalver<sup>2</sup>, J.A. Ortega-Ruiz<sup>2</sup>, J. Ametller<sup>1</sup>, J. Garcia<sup>1</sup> and J. Borrell<sup>1</sup>

<sup>1</sup>Dept. of Information and Communications Engineering  
Universitat Autònoma de Barcelona, Spain.

<sup>2</sup>Institute for Space Studies of Catalonia,  
80034 Barcelona, Spain.

INTELLCOMM 2005, Montreal.

# Outline

- 1 Outline
- 2 Introduction
- 3 Identities and Groups
- 4 Possession paradigm
- 5 Conclusions

# Introduction

## AMAPOLA

Simple **A**gent-based **MA**nagement for **P**ervasive **cO**mputing.

- Provides simple infrastructure to implement applications in pervasive environment.
- Naming schema based on local names.
- Management protocols.

# Identities

- Naming schema influenced by *trust management*.
- Each entity is denoted as **poppy** and identified by the *pID*.
  - strong poppy** the *pID* is a **public key** or its hash.
    - has a pair of asymmetric cryptographic keys.
  - weak poppy** the *pID* is the **hash** of an object.
    - unable to perform asymmetric cryptography operations.

# Local names

- Each entity as a **name container** for defining local names.

```
(<pID>, <local-name>)
```

# Local names

- Each entity as a **name container** for defining local names.

( <pID> , <local-name> )

- Example: poppy  $PK_0$  defines a local name *partner* for the poppy  $PK_1$ :

$PK_0$ : ( $PK_1$ , partner)

# Local names

- Each entity as a **name container** for defining local names.

( <pID> , <local-name> )

- Example: poppy  $PK_0$  defines a local name *partner* for the poppy  $PK_1$ :

$PK_0$ : ( $PK_1$ , partner)

- Third parties can use a *fully qualified name* to refer to  $PK_0$ 's *partner*.

$PK_0$  partner

# Local names

- Each entity as a **name container** for defining local names.

$$(\langle \text{pID} \rangle, \langle \text{local-name} \rangle)$$

- Example: poppy  $PK_0$  defines a local name *partner* for the poppy  $PK_1$ :

$$PK_0: (PK_1, \text{partner})$$

- Third parties can use a *fully qualified name* to refer to  $PK_0$ 's *partner*.

$$PK_0 \text{ partner}$$

- Name assertion: local names can be made public through a signed SAML assertion:

$$\{(PK_1, \text{partner})\}_{PK_0^{-1}}$$

# Group Management

- Groups/roles created as local names:

$$\{(PK_1, \text{friends})\}_{PK_{adm}^{-1}}$$

$$\{(PK_2, \text{friends})\}_{PK_{adm}^{-1}}$$

$$\{(PK_0 \text{ partner}, \text{friends})\}_{PK_{adm}^{-1}}$$

# Group Management

- Groups/roles created as local names:

$$\{(PK_1, \text{friends})\}_{PK_{adm}^{-1}}$$

$$\{(PK_2, \text{friends})\}_{PK_{adm}^{-1}}$$

$$\{(PK_0 \text{ partner}, \text{friends})\}_{PK_{adm}^{-1}}$$

- Support for group/role **hierarchies** by means of inclusion.  
→ support for RBAC-like systems.

$$\{(PK_{adm} \text{ family}, \text{friends})\}_{PK_{adm}^{-1}}$$

# Group Management (II)

- Three supported types of group/role management:

**single-leader** group: a single poppy manages a group.

**set-leader** group: as set of leader poppies can manage the group. New leaders can be added to the set.

**threshold-leader** group:  $(k, n)$  – *threshold*-like schema, new group members need the approval from  $k$  of  $n$  leader ( $k \leq n$ ).

# Possession paradigm

- Two types of poppies (management role):
  - Simple poppy: any kind of poppy.
  - Control Station (CS): *strong poppy* capable coordinate several poppies to perform a concrete and complete task.
- Each poppy may have:
  - One **owner**: static and immutable property.
  - Several **holders**: a holder uses the poppy to perform (or coordinate) an specific task.

# Possession protocols

- Possession protocols: allows a CS to take possession of other poppies to perform an specific task.

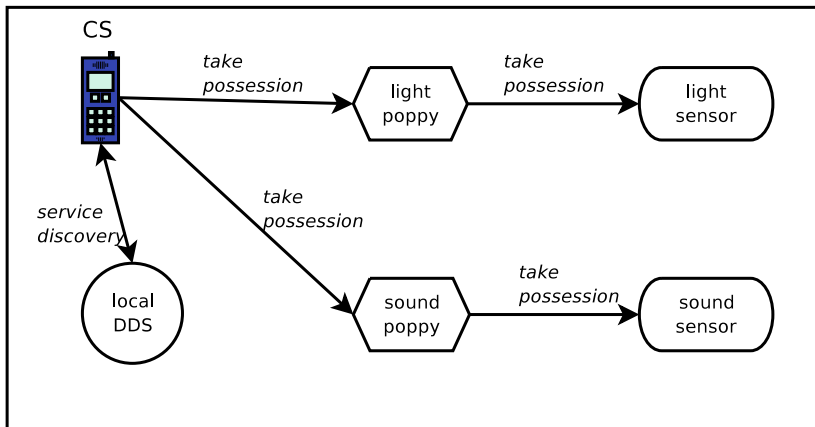
**take-possession** a CS takes possession (becomes the holder) of another poppy.

**terminate-possession** the CS terminates the possession.

**revoke-possession** the possessed poppy terminates the possession.

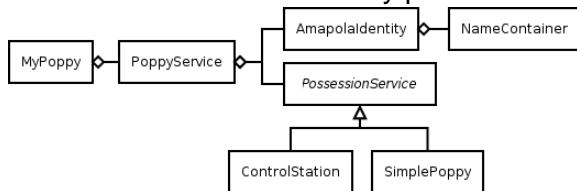
**delegate-possession** the CS delegates the possession of a given poppy to another CS.

# Possession protocols sample



# Implementation details

- Current prototype implemented on top of JADE/JADE-LEAP.
- API with the main functionality presented as services.



- Makes use of SAML for expressing assertions and the possession protocols.
- It also provide tools for debugging and testing.

## CS-console

ControlStation@think:1099/JADE - Control Station Entity GUI  
 General Message Queue Results Queue Help

SAMAP Management SAMAP Protocols DDS Query & Information

**Take-possession**

Of: SE2@think:1099/JADE

**Selected Entity Info**

Service: Sound Player  
 Samap Identity: 7bybsr2zaASIDlhmffR91Q==

**Terminate-possession**

My possessions: SE4@think:1099/JADE

**Selected Entity Info**

Service: Sound Player  
 Samap Identity: 9yXoHjWrtjujt7QStUPKNA==

**Delegate-possession**

My possessions: SE4@think:1099/JADE  
 To: SE3@think:1099/JADE

**Message Queue**

INCOMING FIPA-QUERY INFORM 5/12/05 8:25 PM ((action (agen  
 OUTGOING FIPA-QUERY for Identity QUERY-REF 5/12/05 8:25 PM  
 INCOMING FIPA-QUERY INFORM 5/12/05 8:25 PM ((action (agen  
 OUTGOING FIPA-QUERY for Identity QUERY-REF 5/12/05 8:25 PM

**Possession Entities**

Samap Identity	Service	AID
9yXoHjWrtjujt7QStUPK...	Sound Player	SE4@think:1099/JADE
7bybsr2zaASIDlhmffR...	Sound Player	SE2@think:1099/JADE

**DDS: Registered Samap Entities**

Samap Identity	Service	AID
45KcbVGl6OzatPDF2...	Sound Player	SE3@think:1099/JADE
5rXGgCIR5Mw5q6jJK...	ControlStation	ControlStation@think:...
nEgDnhMpuCPMDsmj...	Sound Player	SimpleEntity@think:1...
zi9DlKEN0YgVTettxu...	ControlStation	CS2@think:1099/JADE

**Results Queue**

Identity query response: 5/12/05 8:25 PM Identitat Samap de (agent-identifrier :name SE2@think:1099/JADE :addresses (sequence http://think  
 Identity query response: 5/12/05 8:25 PM Identitat Samap de (agent-identifrier :name SE3@think:1099/JADE :addresses (sequence http://think  
 Take possession response = TAKE\_POSSESSION\_GRANT 5/12/05 8:25 PM Nova possessio sobre l'entitat 7bybsr2zaASIDlhmffR91Q==  
 Termination possession response: 5/12/05 8:25 PM Eliminada possessio sobre 7bybsr2zaASIDlhmffR91Q==  
 Termination possession response: 5/12/05 8:25 PM Eliminada possessio sobre 45KcbVGl6OzatPDF2f/8KA==  
 Delegation possession response AGREE 5/12/05 8:25 PM L'entitat desti esta processant la Request!  
 Take possession response = TAKE\_POSSESSION\_GRANT 5/12/05 8:25 PM Nova possessio sobre l'entitat 9yXoHjWrtjujt7QStUPKNA==

# Conclusions

- AMAPOLA provides a **simple** infrastructure to implement applications in pervasive environments.
- Intended to provide a simple API for the programmer.
- Provides group/role management and possession protocols.
- Influenced by trust management (SPKI/SDSI).

# Thanks

Thanks. Questions?