Multi-agent Platforms

Cosmin Carabelea

Cosmin.Carabelea@emse.fr
SMA/SIMMO
ENS Mines Saint-Etienne
September 30th, 2003

Outline

- **Why multi-agent platforms?**
- Standards in MAS : FIPA
- Examples of multi-agent platforms
- Conclusions

Existing problems

- More and more applications are developed using MAS.
- **BUT :**
  - There are few multi-agent oriented implementation tools
  - Few agent programming languages
  - MAS Design relies on existing languages and programming techniques
  - It's often hard to develop MAS (implementation, distribution, communications, …)

➔ The trend of the work is towards Multi-Agent Oriented Programming, meaning programming MAS with MAS tools

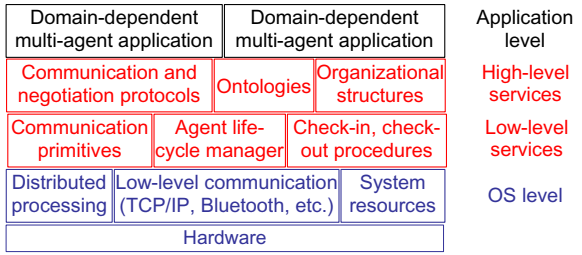
A new platform is created "everyday". Too many multi-agent development tools exist.
How to choose? How to compare?

Existing problems (2)

- Agent's point of view:
  - it needs support to perform the tasks it is required to do.
  - how to sense and act upon its environment.
  - how to communicate and cooperate (unique ids, service discovery, security)
- Developer's point of view:
  - the agents are software programs, their creation passes through the four phases:
    - analysis
    - design
    - development
    - deployment.
- Final user's point of view:
  - how to use the multi-agent application.
  - should he/she trust the agents to do the tasks they are required to do?

What is a multi-agent platform?

- A multi-agent platform is a software infrastructure used as an environment for agents' deployment and execution.



Outline

- Introduction
- **Standards in MAS : FIPA**
- Examples of multi-agent platforms
- Conclusions

Do we need MAS standards?

- Why ?
  - Standards favor development of the market.
  - Market development speed up development of technology.
- Why now ?
  - Technology is under development
    - Lack of large agreement on definitions, models, theories, …
    - Several research axes
  - Needs are recognized
  - Stress on the expected features : *openness, heterogeneous systems, emergent properties, …*

Standards in MAS

- **FIPA** Foundation for Intelligent Physical Agents
  - www.fipa.org
- **MASIF** - OMG (Object Management Group) : OMG effort to standardize mobile agents - middleware services and internal middleware interfaces
  - www.omg.org
- **Knowledge Sharing Effort** The DARPA Knowledge Sharing Effort
  - citeseer.nj.nec.com/pati192darpa.html

Standards in MAS (2)

- **Ontology : DAML, OIL, OWL, …**
  - http://www.daml.org
  - http://www.ontoknowledge.org/oil/
- **Other Standards (De Facto)**
  - dynamic discovery of services :
    - Jini (www.sun.com/jini),
    - UPnP (www.upnp.org),
    - UDDI (www.uddi.org),
    - Salutation (www.salutation.org)
  - mobility : Aglets (www.trl.ibm.com/aglets/)
  - coordination rules : JavaSpace (www.sun.com/jini)

FIPA Overview

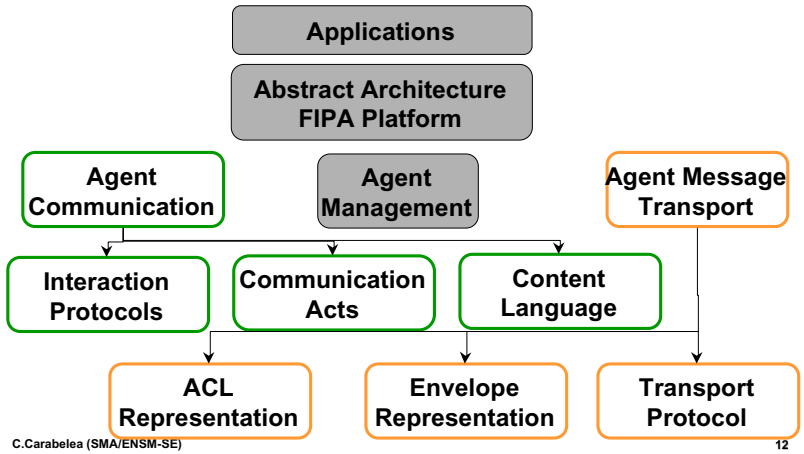
www.fipa.org

- Aim :
  - to create International Standards body in order to promote the development of agents applications
- Structured in :
  - FIPA Architecture Board, Technical Committees, Working Groups
- Funded in 1996
  - 62 member companies with heavy involvement from telecommunications companies in particular.
    BT, EPFL, France Télécom, Fujitsu, HP, Hitachi, IBM, Imperial College, Intel, Motorola, NASA, Nec, NHK, Nortel Networks, NTT, Philips, Siemens, SNCF, SUN Microsystems, Telecom Italia, Toshiba …
- First standards in 1997 - FIPA97, since then - FIPA98 and FIPA2000

FIPA Overview (2)

- FIPA specifies the interfaces of the different components in the environment with which an agent can interact:
  - humans
  - other agents
  - non-agent software
  - the physical world.
- FIPA produces two kinds of specifications:
  - **normative** specifications mandating the external behaviour of an agent and ensuring interoperability with other FIPA-specified subsystems;
  - **informative** specifications of applications providing guidance to industry on the use of FIPA technologies.

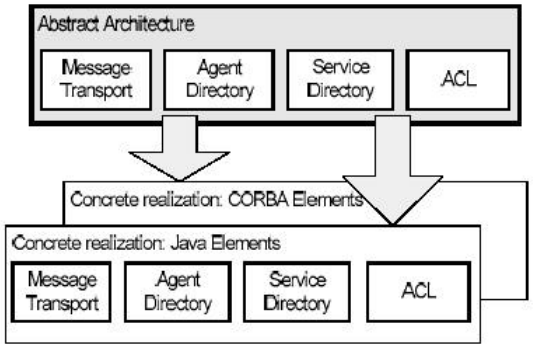
FIPA Specifications (#80)



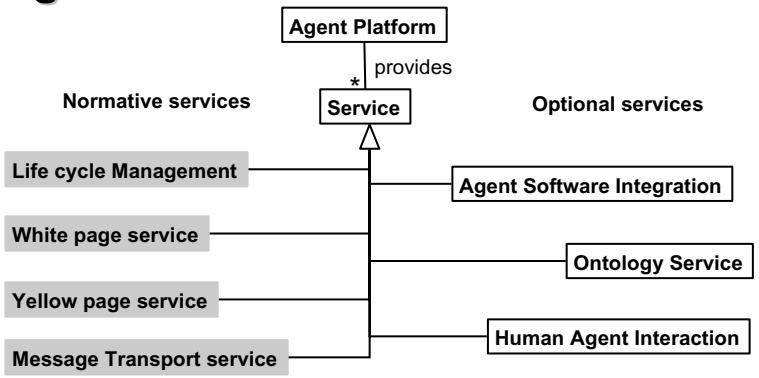
Abstract Architecture specifications

- Definition of an abstract architecture
  - Message transport interoperability.
  - Supporting various forms of ACL representations.
  - Supporting various forms of content language.
  - Supporting multiple directory services representations.
- Modelling of the abstract elements and their relationships.

Abstract Architecture specifications (2)

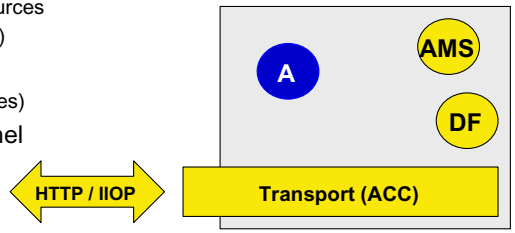


FIPA : conceptual model of an Agent Platform



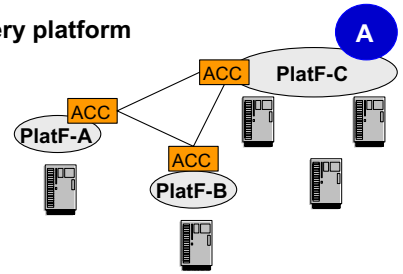
FIPA Agent Platform

- Software that implements the set of FIPA specifications.
- **FIPA-compliant** ⇔ implements at least the **Agent Management** and **Agent Communication Language** specifications.
- **A**gent **M**anagement **S**yst.
  - Authentication, Resources
  - White pages (naming)
- **D**irectory **F**acilitator
  - Directory (yellow pages)
- **A**gent **C**omm. **C**hannel
  - Message transport



FIPA Platforms Inter-operability

- **Communication between agents can be**
  - Platform Internal – non-standard technologies
  - Platform-Platforms – uses the ACC and standard FIPA Message transport protocols.
- **Agent Environment on every platform**
  - Different languages
  - Different APIs
  - Different support features
  - Different agent architectures
- **Same**
  - Base services
  - Same transports
  - Same languages



Outline

- Introduction
- Standards in MAS : FIPA
- **Examples of multi-agent platforms**
  - **JADE**
  - Madkit
  - AgentTool
  - Zeus
  - Multi-agent platforms for small devices
- Conclusions

JADE: Introduction

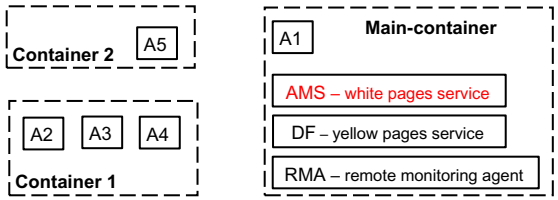
- **J**ava **A**gent **D**evelopment **F**ramework http://jade.cselt.it
- Open Source, written in Java
- Goal: developing multi-agent systems and applications conforming to FIPA standards for intelligent agents.
- Includes two main products:
  - a FIPA-compliant agent platform
  - a package to develop Java agents.
- Utility agents:
  - *DummyAgent tool* allows users to interact with JADE agents in a custom way.
  - *Sniffer Agent* is basically a FIPA-compliant agent used to sniff messages.

JADE: Agent development

- A JADE agent is simply an instance of a user defined Java class that extends the base *Agent* class.
- Each agent can have several behaviours, obtained by inheriting the *Behaviour* class (or one of its subclasses).
- Each agent and each agent's behaviour is a thread.
- An agent can send/receive Java objects, that represent FIPA-ACL messages within the scope of interaction protocols.
- JADE hides all message coding (encoding/parsing).
- The Message Transport Protocol (MTP) module automatically selects the best way to send a message: method invocation, RMI, TCP/IP, etc.
- Developers can create new MTPs.

JADE: containers

- Each agent lives inside a container.
- A container is a JVM and it:
  - provides a complete runtime environment for agent execution
  - allows several agents to run concurrently
  - controls the life-cycle of agents
  - deals with communication.

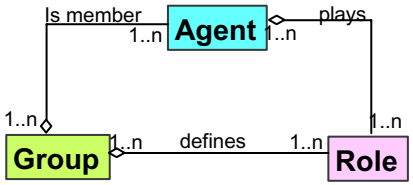


Outline

- Introduction
- Standards in MAS : FIPA
- **Examples of multi-agent platforms**
  - JADE
  - **Madkit**
  - AgentTool
  - Zeus
  - Multi-agent platforms for small devices
- Conclusions

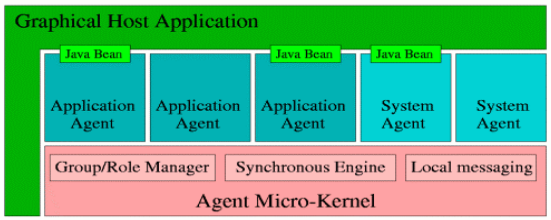
MADKIT: Introduction

- Multi-Agent Development Kit www.madkit.org
- Developed by LIRMM lab, free for educational use.
- Java multi-agent platform based on the Aalaadin organizational model:



- A tutorial can be found at http://www.emse.fr/~carabele/master/tp_sma.html

MADKIT: micro kernel



MADKIT (continued)

- **Analysis**:
  - no specific analysis method
- **Design**:
  - organizational model (groups, roles)
  - interaction model (protocols, messages)
  - tasks, goals, etc.
  - no software tools.
- **Development**:
  - no agent model (to be implemented in Java from scratch).
- **Deployment**:
  - use of the G-box (eq. Sandbox)

Outline

- Introduction
- Standards in MAS : FIPA
- **Examples of multi-agent platforms**
  - JADE
  - Madkit
  - **AgentTool**
  - Zeus
  - Multi-agent platforms for small devices
- Conclusions

AgentTool: Introduction

- http://www.cis.ksu.edu/~sdeloach/ai/download-agentool.htm
- Developed at Kansas State University, free for research and academic use.
- Written in Java 1.2.
- Conversation verification (requires some additional tools to be downloaded).
- Automatic code generation (up to 90% of the agent).
- Methodology used: MASE – Multi-agent Systems Engineering.
- The methodology covers the phases of analysis and design.

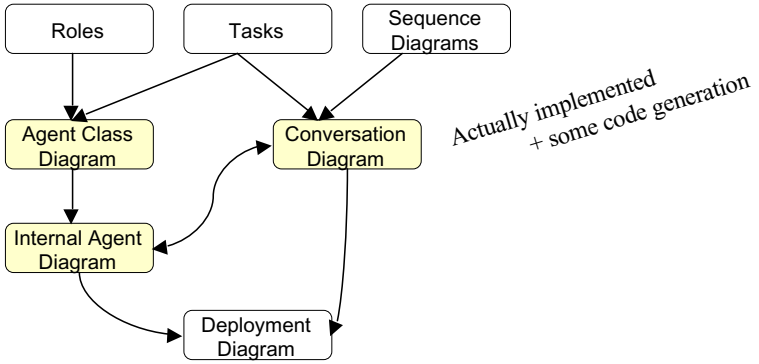
AgentTool: MaSE process

1. Capturing Goals
   - initial system specification ⇒ struct. hierarchy of goals
   - i.e. similar to requirement specification
2. Applying Use Cases (i.e. UML)
   - Use cases and sequence diagrams based on spec.
   - Use cases – represent logical interaction path
   - Sequence diagrams – number of messages needed
3. Refining Roles
   - Creates roles corresponding to the goals (or a set of goals)
   - Creates tasks – how to solve goals related to the role

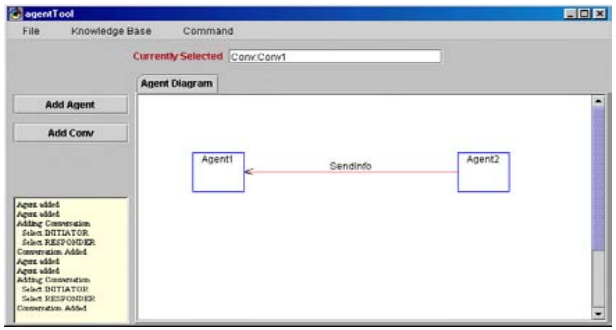
AgentTool: MaSE process (2)

4. Creating Agent Classes
   - Maps roles to agent classes in an agent class diagram
   - Resemble object class diagrams, but semantics is high-level conversation versus inheritance (and encapsulation)
5. Constructing Conversations
   - Defines coordination protocols for interaction with state diagrams
6. Assembling Agent Classes
   - Internal functionalities of classes created
   - Based on either BDI, reactive, planning, knowledge-based and user-defined architecture.
7. System Design
   - Create instances of the agent classes presented in a deployment diagram

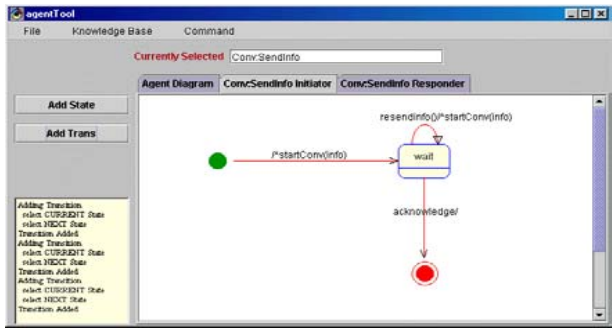
MaSE in agentTool



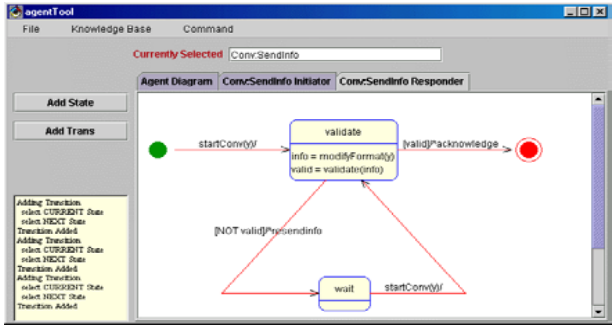
AgentTool : Agent Diagram



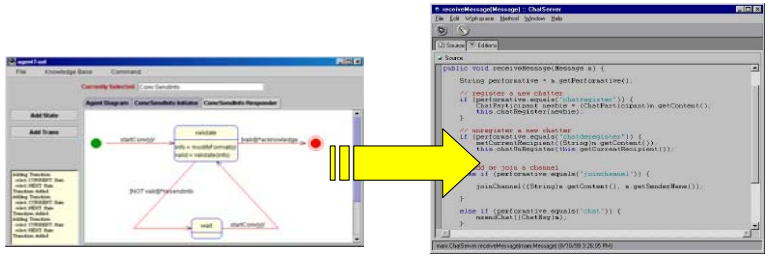
AgentTool : Conversation Diagram
(half a conversation)



AgentTool : Conversation Diagram
(the other half)



AgentTool : Code Generation



- Automatic from Agent and Conversation Diagrams
- Select platform-dependent components such as a messaging system

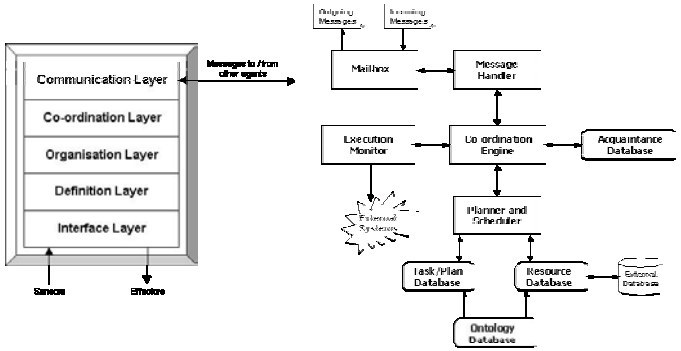
Outline

- Introduction
- Standards in MAS : FIPA
- **Examples of multi-agent platforms**
  - JADE
  - Madkit
  - AgentTool
  - **Zeus**
  - Multi-agent platforms for small devices
- Conclusions

ZEUS: Introduction

- Agent Building Toolkit (www.labs.bt.com/projects/agents/zeus)
- Open Source Licence (Mozilla)
- Developed by Agent Research Programme of BT Intelligent Research Lab.
- Integrated environment.
- Strong emphasis on the importance of methodology.
- Three libraries:
  - Utility Agents
  - Agent Building Tool
  - Agent Component Library

ZEUS : Agent architecture



ZEUS : Predefined agents

- Utility agents:
  - Nameserver agent – white-pages service
  - Facilitator agent – yellow-pages service
- Visualiser agents:
  - Society Viewer : all agents, organisational relationships, messages.
  - Reports Tool : decomposition/distribution of active tasks and the execution states of the various tasks.
  - Agent Viewer : observes and monitors the internal states of agents.
  - Control Tool : to remotely review and/or modify the internal states of individual agents.
  - Statistics Tool : displays individual agent and society-wide statistics in a variety of formats.

Outline

- Introduction
- Standards in MAS : FIPA
- **Examples of multi-agent platforms**
  - JADE
  - Madkit
  - AgentTool
  - Zeus
  - **Multi-agent platforms for small devices**
- Conclusions

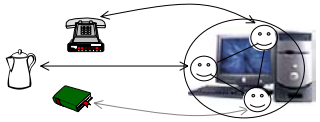
In the near future…

- Taxi-sharing scenario:
  - many users, equipped with mobile phones and PDAs, are waiting for taxis
  - their devices interact (without the user's intervention) to find persons with similar destinations
  - there are small processors embedded in cars → the taxi can inform the user via their devices if it is stuck in a traffic jam
- Intelligent house scenario:
  - there are small processors embedded in all objects in the house: cd-player, tv, refrigerator, etc.
  - the refrigerator can inform the user the milk is about to expire, etc.
  - when the user comes home, the tv and the cd-player can negotiate which one will turn on to entertain the user
  - the objects can act differently if the user is not alone…

Multi-agent platforms for small devices

Three types of multi-agent platforms for small devices:

- portal platforms:
  MobiAgent
- surrogate platforms:
  kSACI
  JADE-LEAP
- embedded platforms:
  JADE-LEAP
  Micro FIPA-OS
  MAE
  AgentLight



Multi-agent platforms for small devices (2)

Platform	MobiAgent	kSACI	LEAP	MAE	AgentLight	micro FIPA-OS
Connection to SD	portal	surrogate	surrogate/ embedded	embedded	embedded	embedded
Smallest targeted device	mobile phone	mobile phone	mobile phone	PDA	mobile phone	PocketPC
FIPA-compliant	it may be	no	yes	no	yes (?)	yes
No. of agents on device	0	1	several (pref.1)	several	several	several (pref.1)
Available for download?	no	yes	yes	no	yes	yes
JavaVM used	kVM	kVM	various	various	kVM	PersonalJava

Jade-Leap Platform

An add-on of the JADE platform since version 3.0.

- Works over fixed and wireless networks (WLAN or GPRS).
- Can be configured for different devices, OS and JavaVM.
- For the small devices there are two execution modes: stand-alone vs. split.
- Limitations:
  - the main container must be on a PC
  - no support for mobility.
- Keeps the FIPA-compliancy.
- Smallest device targeted: mobile phone

Outline

- Introduction
- Standards in MAS : FIPA
- Examples of multi-agent platforms
- **Conclusions**

Conclusions

- There is a need for tools that will ease the design, development, deployment, execution and utilization of multi-agent systems.
- Too many such tools exist, and they are not complete.
- We presented some of the most representative multi-agent platforms, but there are many more.
- What multi-agent platform to use?
- There is also a need for standards in multi-agent systems.
- Work is still in progress, but FIPA standards might ensure the interoperability needed by multi-agent systems.
- For example…

AgentCities

www.agentcities.org

- International Deployement of MAS Platforms (> 100)
  - Permanently accessible via Internet
  - Openness
  - FIPA Compliant
- Hosting multiple agent “services”
  - Interoperability between agent services
  - Experiment of composition of services / with added value
  - Experience on complex models and semantic descriptions

Bibliography

- *AgentCities*: http://www.agentcities.org
- *AgentLink’s review of agent software*: www.agentlink.org
- C.Carabelea, O.Boissier: “*Multi-agent platforms for small devices: Dream or reality?*”. In Proc. of the Smart Objects Conference (SOC’03), Grenoble, 2003.
- *Foundation for Intelligent Physical Agents* (FIPA): http://www.fipa.org
- The platforms presented:
  - *AgentTool*: http://www.cis.ksu.edu/~sdeloach/ai/download-agentool.htm
  - *JADE*: http://jade.cselt.it
  - *Madkit*: http://www.madkit.org
  - *Zeus*: http://www.labs.bt.com/projects/agents/zeus