tATAmI-2 – a Flexible Framework for Modular Agents

Andrei Olaru – cs@andreioraru.ro

University Politehnica of Bucharest

28.05.2015
tATAmI-2 – a Flexible Framework for Modular Agents

overview
Problem Context

- Context: building a MAS framework for MAS-based AmI applications

We require a framework with a lot of flexibility:

- agents must be able to run on various devices (PC, Android, iOS, Arduino)
- agents must be able to use various communication methods
  - TCP/IP · web services · web sockets · queues · other?
- agents structure must be able to be very light or more complex
  - behaviors · S-CLAIM · other AOP languages
Problem Context

- How to model agents regardless of their internal structure?
- How to model communication and mobility services?
- How to control agents and platforms?
- How to correctly load platforms and agents?
What are we doing?

- **2010**
  - tATAmI-1
  - S-CLAIM
  - Jade
  - A. Olaru
  - M.T. Benea
  - T.T.N. Nguyen

- **2013**
  - tATAmI-2
  - S-CLAIM
  - ContextKB
  - Jade
  - A. Olaru
  - M.T. Benea
  - E. Sevastian

- **2015**
  - tATAmI-2.5 platforms
  - A. Olaru
  - C. Mihai
What are we doing?

S-CLAIM
Knowledge
Visualization
Base
Jade agent

tATAmI-1
class inheritance layers

tATAmI-2
flexible modules/components

knowledge

visualization

control

communication
tATAmI-2 Architecture

The tATAmI-2 framework connects all platforms and agents, across multiple machines.

A machine that is part of the framework; it hosts one or more containers, which host agents.

A platform spans multiple machines and offers communication, discovery and mobility services to agents.

An agent runs inside a container, being loaded on a platform.

A component runs inside an Composite agent an implements functionality.
tATAmI-2 Architecture

The platform is an entity that offers various types of services to agents.

- tATAmI-2 sees it as:
  
  **PlatformLoader**
  
  - start()
  - stop()
  - loadAgent()
  - recommendComponent()

- an agent sees it as *platform link*
  - can only be used by specialized components

Loading a platform:
create instance → start → create containers
  → create *link* agents → load agents
tATAmI-2 – a Flexible Framework for Modular Agents

The agent is a persistent, autonomous entity that perceives, acts and communicates

- tATAmI-2 sees it as:
  
<table>
<thead>
<tr>
<th>AgentManager</th>
</tr>
</thead>
<tbody>
<tr>
<td>.start()</td>
</tr>
<tr>
<td>.stop()</td>
</tr>
<tr>
<td>.setPlatformLink()</td>
</tr>
</tbody>
</table>

- the platform is contacted by the agent’s specialized components

An agent is loaded by an AgentLoader:
create the agent loader → pre-load the agent → load the agent
→ load the agent on the platform → start → enrol → start simulation
tATAml-2 – a Flexible Framework for Modular Agents

---

**tATAml-2 Architecture**

---

**Boot**
- platform(s) boot

**Create Agents**
- start Sim Link Agents
- create containers

**Start Simulation**
- load agents
- load platforms
- pre-load agents
- start platforms

---

**Clear Agents**
- start Sim Link Agents
- load agents onto platforms
- start agents
- enrol agents with Sim Link Agents
- signal simulation to all agents
- starts event timeline
- signal simulationstop to all agents
- stop Sim Link Agents
- stop platforms
- close GUI
- system exit

---

Andrei Olaru – cs@andreioralu.ro
CSCS’20
Bucharest, Romania 28.05.2015
Use an XML scenario file to **completely** specify the initial configuration

```xml
<scene:platform>
  <scene:parameter name="name" value="local" />
</scene:platform>

<scene:initial>
  <scene:container name="Container"

  <scene:agent>
    <scene:component name="parametric" />
    <scene:component name="visualizable" />
    <scene:component name="messaging" />
    <scene:component name="testing" classpath="...PingBackTestComponent">
      <scene:parameter name="other_agent" value="AgentB" />
      <scene:parameter name="initiator" value="true" />
    </scene:component>
  </scene:agent>

  <scene:agent>
    <scene:component name="parametric" />
    <scene:component name="visualizable" />
    <scene:component name="messaging" />
    <scene:component name="testing" classpath="...PingBackTestComponent">
      <scene:parameter name="loader" value="composite" />
    </scene:component>
  </scene:agent>

  <scene:agent>
    <scene:component name="parametric" />
    <scene:component name="visualizable" />
    <scene:component name="messaging" />
    <scene:component name="testing" classpath="...PingBackTestComponent">
      <scene:parameter name="loader" value="composite" />
    </scene:component>
  </scene:agent>

</scene:initial>
```
Composite agents are formed of components which communicate by means of an event queue.

- and agent sees it as:
  \[\text{AgentComponent} \quad .\text{parentChangeNotifier()} \quad .\text{signalEvent()}\]

- tATAmI-2 sees it as:
  \[\text{.initialize()} \quad \text{.preload()}\]

Loading a component: initialize → preload → add to agent → agent start → simulation start
Example: the **Messaging Component** – abstracts messaging services

- A message is abstracted as a content sent between two endpoints
- An endpoint has an *external path* and an *internal path*

```
jade:platform-1/AgentA /VISUALIZATION/CONTROL
```

- can be extended by any component offering messaging services
- provides methods such as `send()`, `registerMessageHandler()`, `getAgentAddress()`
- is able to access the platform by using the *platform link*
- each platform is able to recommend a corresponding messaging component
Implementation

- tATAmI-2 core
- local messaging platform + corresponding component
- Jade messaging platform + corresponding component
- WebSocket messaging platform + corresponding component
- visualization, control, S-CLAIM interpreter
- various test components
tATAmI-2 – a Flexible Framework for Modular Agents

Implementation

Andrei Olaru  
- architecture
- main development

Marius-Tudor Benea  
- S-CLAIM development
- Android development

Amal El-Fallah Seghrouchni  
- tATAmI-1 coordination

Emma Sevastian  
- scenario implementation

Cosmin Mihai  
- WebSocket messaging

Thi Thuy Nga Nguyen  
- tATAmI-1 development

Adina Magda Florea  
- coordination

Andrei Olaru – cs@andreiolaru.ro
CSCS’20
Bucharest, Romania 28.05.2015
Implementation

- multiple platforms running at the same time; same agent communicating through different means
- web service messaging
- conversation support
- Android deployment (supported in tATAmI-1)
Thank You!

Any Questions?