

An Introduction to Software agents

UiA 18.09.2007



Agenda

- Who's this guy? What's Integrasco?
- What is a software agent?
- Multi-agent systems (MAS)
- Areas of applications
- Why software agents?
- Software agent frameworks
- Java Agent Development Framework (JADE)
- Demonstration: Developing an agent with JADE
- Exercise



Jaran Nilsen

- Finished my Master degree in ICT June 2007
- Working part time for Integrasco A/S since 2004
- Now working full time for Integrasco A/S
- Intelligent agents and web mining

Contact: jaran.nilsen@integrasco.no



Integrasco A/S

- Working with analysis of online Word of Mouth
- Development areas:
 - Java Enterprise
 - Web mining
 - Machine learning and pattern classification
- All of our development is based on free and open source applications and frameworks:
 - Apache products
 - Spring Framework
 - Hibernate
 - Eclipse

What is a software agent?



What is a software agent?

- A little history

- Concept of agents has been around at least since the 1970's
 - Their physical instantiation – robots – has been around even longer
- Idea: *The future would bring personal assistant agents which would do anything!*
 - Arrange meetings for you
 - Book flights and hotel when you're going away
 - Today's agenda while you eat your breakfast
 - "That shirt doesn't really match your pants, sir."
 - Your fridge would shop for you
- Apple's Knowledge Navigator (1987)
 - [Knowledge Navigator \(~5 min\)](#)

What is a software agent?

- Latin: *Agere* – *to do, agreement to act on one's behalf* [1]
- Software entity capable of acting independently to a certain degree
- Designed to act without user intervention – decides for itself when to perform it's assigned tasks
- Definition of an agent[3]:
 - *Anything that can be viewed as perceiving its environment through sensors and acting upon that environment through effectors*
- Even humans can be seen as *agents* from this definition

What is a software agent?

- There exists no agreed-upon definition of agents
- Therefore it is often difficult to classify software as agents or not
- However, a few common concepts exist:
 - Persistent: Not executed on demand, but decides for itself
 - Autonomous: Perform tasks with little or no user intervention
 - Social: Able to communicate with other agents and cooperate
 - Reactive: Perceive the environment / context and act appropriately
- Grey zone example of an agent: Web Browser – often considered to be a *user agent*, but never acts without a user telling it to do something.

What is a software agent?

- Another familiar example: Anti-virus software
- All software agents are programs, but not all programs are software agents[2]
 - **I'm not an agent:** Regular spell checker in a text processor
 - **I'm an agent:** Spell checker which automatically detects mistakes *and* corrects them without the user telling it to do so

What is a software agent?

- Properties of software agents:
 - Reactiveness and adaptability
 - Communication and cooperation
 - Reasoning based on collected knowledge
 - Intelligence and learning abilities
 - Mobility
 - Autonomy



What is a software agent?

- A “Software agent” is *not* some highly advanced and complicated development method.
- A class of software usually performing more complex tasks than the average desktop application.



Multi-agent systems (MAS)

- Agents is an old concept – MAS is a newer area of research
- Idea: Systems designed to act on behalf of humans – represent our interests
- Software agents working together to achieve a more complex task
- Emerged from distributed systems
 - Distribution coupled with the need for systems to represent and act in our best interest requires them to cooperate and reach agreements
- Agentcities.org

Areas of application

- On your personal computer:
 - Web browsers / e-mail clients
 - Acts on behalf of the user
 - Less autonomous but still considered agents to a certain degree
 - Anti-Virus software, spam filter
 - Reacts to changes to files / incoming streams and decides on an appropriate action based on knowledge of viruses and spam messages
 - Able to “learn” from previous encounters with viruses and spam
 - Bayesian Spam filters
 - Windows Update
 - Acts on behalf of the user
 - Keeps your computer up to date with the latest patches so you won't have to do it yourself

Areas of application

- Data mining applications
 - Web crawlers and other data collecting agents
 - Supplied with a target host and will collect data based on sets of rules
 - Learning / Intelligence – The way a web crawler agent collects data from the target host may be based on learning and intelligent algorithms
 - Autonomous – often running constantly without any user intervention
 - Handle and recover from unexpected connection problems, re-routing
 - Data analysis agents
 - Classification of text based on machine learning and pattern recognition
 - Learning / Intelligence – Uses sophisticated learning algorithms
 - Autonomous – runs as long as there are data left to process
 - Common advantage of software agents for these two areas of application:
 - Distribution of the task
 - Cooperation between several agents to get the job done quicker

Areas of application

- Outer space
 - Spacecraft control – NASA DS1
 - Agents and their physical instantiation – robots - perform tasks in environments and settings which are unsuitable and dangerous for humans
 - DS1 – The agents were used to test equipment and analyze systems
- In your car
 - GPS module
 - Senses the environment through it's GPS receiver
 - “Intelligence” - calculate the most timesaving route, or the shortest route
 - Able to adapt and react to route changes

Areas of application

- Other areas of application
 - Crowd Simulations
 - A set of software agents are implemented with artificial intelligence which guides the agents based on what they sense
 - The agents can communicate and cooperate to handle situations
 - Used in “Lord of the Rings” to simulate the huge armies in the battles
 - Simulation of crowds in a city landscape
 - Computer games
 - Used to “control” Non-Player Characters
 - “Bots” - agents take on the role of another player

Why software agents?

- Why software agents are good:
 - Frameworks based on standards
 - Already implemented functionality for
 - Messaging
 - Execution
 - Lifecycle management
 - Easy access to monitor and manage a complete platform
- Why software agents are bad (common criticism):
 - Testing – testing of agents can be difficult, but far from impossible!
 - No standard for communication with non-agent systems
 - No standard for inter-MAS communication

Software Agent frameworks

- Voyager
- Concordia (Mitsubishi)
- Grasshopper
- Able (IBM)
- IMPACT
- JADE
- +++



JADE

- Java Agent Development Framework
- Middleware for developing distributed multi-agent systems[4]
- Developed and maintained by Telecom Italia Labs
- Open source – robust active community around the framework
- Implementation of the FIPA specifications

FIPA

- Foundation of **I**ntelligent and **P**hysical **A**gents
- Specifications for several aspects of agents
 - Agent Lifecycle management
 - Message transport
 - Message structure
 - Interaction protocols
 - Ontologies
 - Security



FIPA

- Inside the scope of FIPA:
 - Platform services provided to the agents
 - Message Transport System
- Outside the scope of FIPA:
 - The agent itself

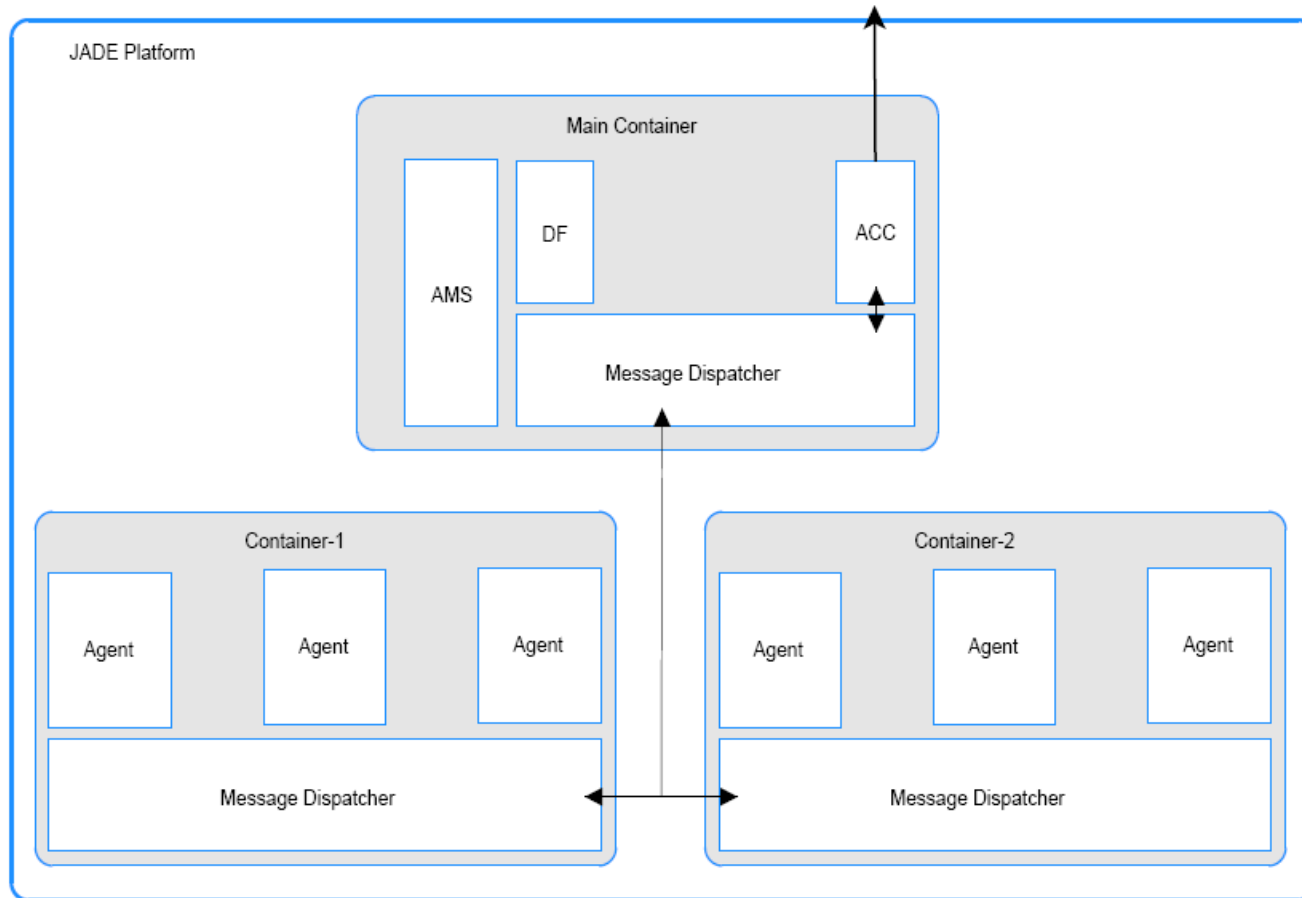


JADE – Architecture

- Agent platform – consists of containers
- Agent container – consists of agents
- Boundaries of the platform is decided by the containers connected to it.
- If a new container is connected – boundary is extended.
- Core elements of a JADE agent platform:
 - The containers
 - The agents
 - The Agent Management System (AMS)
 - The Directory Facilitator
 - The Message Transport System



JADE – Architecture



JADE – Services

- The JADE Framework provides the following services:
 - Yellow / White pages service - Directory Facilitator (DF)
 - Efficient communication via Agent Communication Language (ACL)
 - Ontologies – For *advanced* agent communication
 - Intra-platform *mobility* – this is cool! :)
 - Useful tools managing, monitoring and debugging agents

A JADE Agent

- Each agent executed as a thread
- Implemented through the `jade.core.Agent` class

```
public class MyAgent extends Agent {
    protected void setup ( ) {

        // TODO Initialize the agent and
        // add behaviours
    }

    protected void takeDown() {

        // TODO Perform any clean up before the
        // agent is destroyed
    }
}
```



A JADE Agent

- The agents actions are implemented as *behaviours*
 - `jade.core.behaviour` package
 - Behaviour – base class of all other behaviours
 - OneShotBehaviour – executes once
 - TickerBehaviour – executes periodically
 - WakerBehaviour – executes once, at a given time
 - ParallelBehaviour – concurrent children scheduling
 - SerialBehaviour – serially executes children
 - ++



A JADE Agent

- WakerBehaviour example:

```
public class MyWakeupBehaviour extends WakerBehaviour {  
    public MyWakeupBehaviour(Agent agent, long wakeup) {  
        super(agent, wakeup);  
    }  
    protected void onWake () {  
        System.out.println("Hello, " + myAgent.getName ()  
            + " is awake!");  
    }  
}
```



JADE Agent communication

- `jade.lang.ac1.ACLMessage` class
- Ontologies – describes the format and meaning of your message
- Message types:
 - REQUEST
 - INFORM
 - CONFIRM
 - REJECT
 - +++
- Topic based communication also available



JADE Agent communication

- Sending a message:

```
ACLMessage msg = new ACLMessage(ACLMessage.REQUEST);
msg.addReceiver(new AID("SomeAgent", false));
msg.setSender(myAgent.getAID());
msg.setContent("Hello");
...
myAgent.send(msg);
```

- Receiving a message:

```
ACLMessage msg = receive();
if(msg == null) {
    block();
    return;
}
```

```
ACLMessage msg = blockingReceive();
...
```

JADE Agent Mobility

- Allows agents to move between containers on the same platform
- JADE supports weak mobility – agents move with their data state, not the complete execution stack
 - Strong mobility includes complete execution stack – the agent continues executing from the code line where it was before it moved.
- Moving an agent to another container is done with the **Agent.doMove(Location)** method
- **Agent.beforeMove()** - called by the framework right before the agent is moved.
- **Agent.afterMove()** - called by the framework on the agent in it's new location

To sum up...

- All software agents are programs, but not all programs are software agents
- Software agents are programs which are:
 - Persistent, autonomous, intelligent, social, reactive
- Using agent frameworks when developing applications with these properties eases the job because of services provided by the framework
- Standardization enables communication and cooperation with other agents, even when their operation is unknown.
 - You don't have to know how a person thinks to communicate with him / her

Curriculum

- Managing Software Agents in J2EE Application Servers
 - Chapter 2 and 3, except sections 3.2.5 and 3.5
- Is it an Agent, or just a Program?

- JADE Programmers Guide
 - *Not* part of the exam, but useful for exercises



References

- [1] Wikipedia.org
http://en.wikipedia.org/wiki/Software_agents
- [2] Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents, Franklin and Graesser, 1996
<http://www.msci.memphis.edu/~franklin/AgentProg.html>
- [3] Artificial Intelligence: A Modern Approach.
Stuart J. Russel and Peter Norvig, 1995
- [4] JADE – A White paper
F. Bellifemine, G. Caire, A. Poggi, G. Rimassa

