

JADE: Java Agent DEvelopment Framework Overview

Stefano Mariani
`s.mariani@unibo.it`

Dipartimento di Informatica – Scienza e Ingegneria (DISI)
ALMA MATER STUDIORUM – Università di Bologna a Cesena

Academic Year 2015/2016



- 1 What is JADE?
- 2 JADE Architecture
- 3 JADE Tools



Disclaimer

All the material presented in these slides is rearranged by the author from a collection of documents kindly made available by the JADE team.

Credits for all the stuff (text & images) go to the JADE team, in particular to Giovanni Caire.

Credits for all the mistakes go to the author.



Outline

1 What is JADE?

2 JADE Architecture

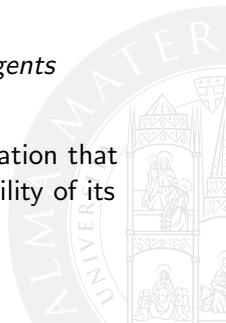
- JADE & FIPA
- JADE Agents
- JADE ACC

3 JADE Tools



JADE in Short I

- JADE stands for *Java Agent DEvelopment Framework*
<http://jade.tilab.com/>
- JADE is a Java-based framework to develop agent-based applications in compliance with the **FIPA specifications** for interoperable, intelligent, multi-agent systems
- FIPA stands for *Foundation for Intelligent Physical Agents*
<http://www.fipa.org/>
- FIPA is the IEEE Computer Society standards organisation that promotes agent-based technology and the interoperability of its standards with other technologies



JADE in Short II

JADE Goals

As an **agent-oriented middleware**, JADE pursues the twofold goal of being

- a full-fledged FIPA-compliant *agent platform*. Hence, it takes charge of all those application-independent aspects – such as agent lifecycle management, communication, distribution transparency, etc. – necessary to develop a MAS
- a simple yet comprehensive *agent development framework*. Therefore, it provides Java developers a set of APIs to build their own customisations



JADE Main Ingredients

Java

Being fully implemented in Java, JADE is a notable example of a distributed, **object-based**, **agent-oriented** infrastructure, hence an interesting example about how to face a design/programming *paradigm shift*

FIPA

Being compliant to FIPA standards, JADE is a *complete* and *coherent* agent platform providing all the necessary facilities to deploy MAS



JADE Main Features

JADE offers (among many)

- a distributed agent platform, where “distributed” means that a single (logical) JADE system can be split among different networked hosts
- transparent, distributed **message passing** service
- transparent, distributed **naming** service
- white pages & yellow pages **discovering** facilities
- intra-platform agent **mobility** (code & context, to some extent)
- debugging & **monitoring** graphical tools
- ... and much more



Outline

1 What is JADE?

2 **JADE Architecture**

- JADE & FIPA
- JADE Agents
- JADE ACC

3 JADE Tools



JADE Architecture Overview

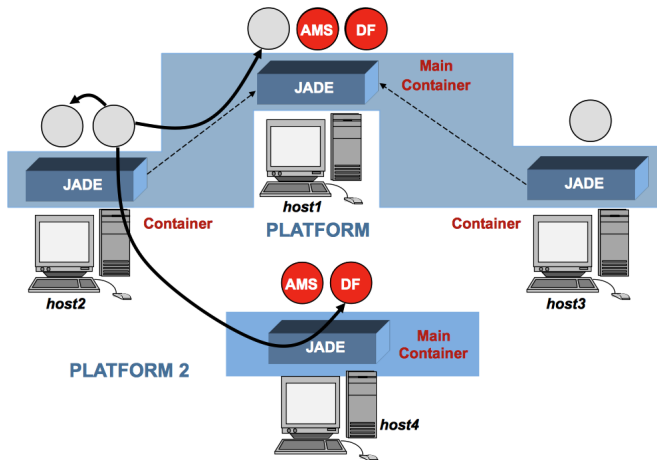


Figure: JADE system overview

Outline

- 1 What is JADE?
- 2 **JADE Architecture**
 - JADE & FIPA
 - JADE Agents
 - JADE ACC
- 3 JADE Tools



FIPA Architecture I

According to FIPA, the agent platform can be split on several hosts given that:

Containers

- each host acts as a **container** of agents, that is, provides a complete *runtime environment* for JADE agents execution—lifecycle management, message passing facilities, etc.
- (at least) one of these containers is the **main container** (actually, the first started), responsible to maintain a *registry* of all other containers in the same JADE platform—through which agents can discover each other

Hence, JADE promotes a *P2P* interpretation of a MAS.



FIPA Architecture II

Agent Management System

For a given JADE platform, a single **Agent Management System** (AMS) exists, which:

- keeps track of all other agents in the same JADE platform—even those living in remote containers
- should be contacted by JADE agents prior to any other action (they do not even exist until registered by the AMS)

Hence, the AMS provides the **white pages** service—that is, a *location-transparent* naming service.



FIPA Architecture III

Directory Facilitator

A singleton **Directory Facilitator** (DF) exists for each JADE platform, that:

- keeps track of all advertised services provided by all the agents in the same JADE platform
- should be contacted by JADE agents who wish to publish their capabilities

Hence, provides the default **yellow pages** service—*publish/subscribe* paradigm.



FIPA Architecture IV

Agent Communication Channel

For a given JADE platform, a distributed message passing system exists—which is called **Agent Communication Channel**:

- it controls exchange of messages within the JADE platform, be them local or remote
- it implements all the needed facilities to provide *asynchronicity* of communications
- it manages all aspects regarding **FIPA ACL** (Agent Communication Language) message format, such as serialization and deserialization



FIPA Architecture V

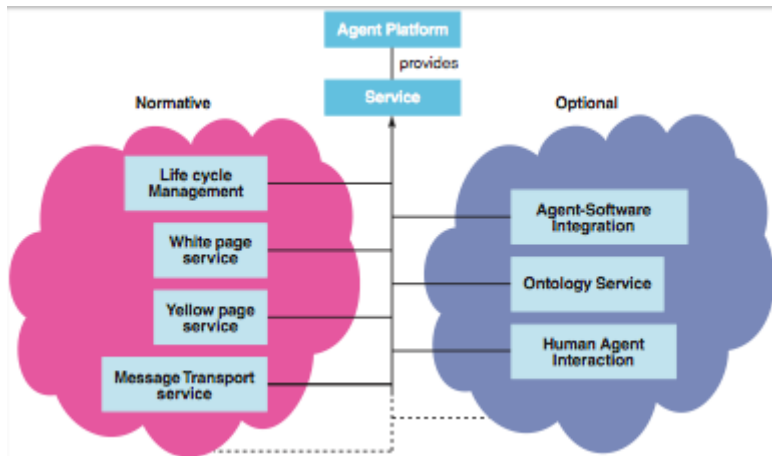


Figure: FIPA required services

Outline

- 1 What is JADE?
- 2 **JADE Architecture**
 - JADE & FIPA
 - **JADE Agents**
 - JADE ACC
- 3 JADE Tools



What is an Agent in JADE I

An agent is a Java object executed by a Java thread

Being an *object-based middleware*, JADE agents are first of all Java objects:

- user-defined agents must extend `jade.core.Agent` class, inheriting some ready-to-use methods
- a JADE agent is executed by a single Java *thread* (there is an exception, though)



What is an Agent in JADE II

An agent is more than a Java object

JADE agents have a wide range of features enabling their **autonomy**—despite being still Java objects

- all JADE agents must have a *globally unique name* (aid), which is (by default) the concatenation – by symbol '@' – of their *local name* and of the JADE platform name
- agents *business logic* must be expressed in terms of *behaviours*
- JADE agents can communicate by exchanging FIPA *ACL* messages



FIPA Agents Lifecycle I

Agent Life Cycle I

According to FIPA, a JADE agent can be in one of several states during its lifetime:

- Initiated** the agent object has been built, but cannot do anything since it is not registered to the AMS yet—it has no *aid* even
- Active** the agent is registered to the AMS and can access all JADE features—in particular, it is executing its behaviour(s)
- Waiting** the agent is blocked, waiting for something to happen (and to react to)—typically, an *ACL* message

⋮ ⋮

FIPA Agents Lifecycle II

Agent Life Cycle II

⋮ ⋮

Suspended the agent is stopped, therefore none of its behaviours are being executed

Transit the agent has started a *migration* process—it will stay in this state until migration ends

Unknown the agent is dead—it has been deregistered to the AMS



FIPA Agents Lifecycle III

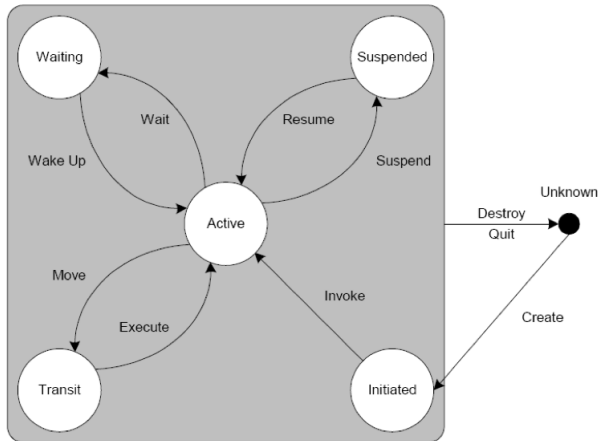


Figure: FIPA Agent Life Cycle

Agent Behaviours I

Why behaviours?

- By definition, agents are **autonomous** entities, therefore they should act independently and in parallel with each other
- The need for *efficiency* drives toward the execution of JADE agents as a single Java thread each
- ! However, agents need to perform complex activities, possibly composed by multiple tasks—even concurrently

How to conciliate this contrasting needs?



Agent Behaviours II

With behaviours!

What are behaviours?

- A behaviour can be seen as *an activity to perform with the goal of completing a task*
- A behavior can represent both a *proactive* activity – started by the agent on its own – as well as a *reactive* activity—performed in response to some events (timeouts, messages, etc.)
- ! JADE implements behaviours as Java objects, which are executed concurrently (still by a single Java thread) using a **non-preemptive, round-robin scheduler** (internal to the agent class but hidden to the programmer)

Agent Behaviours III

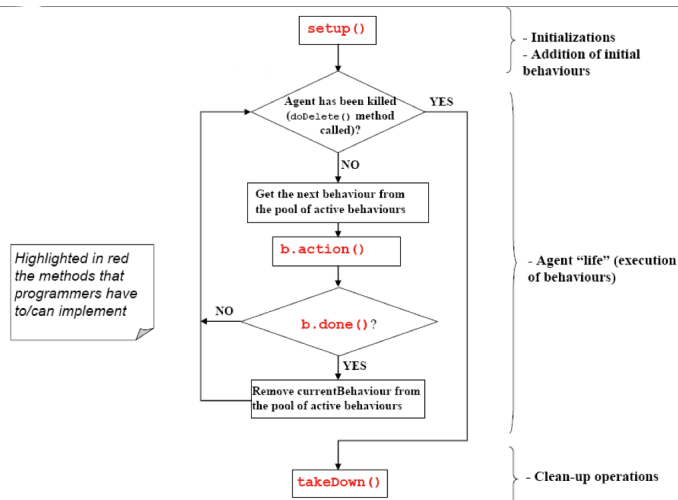


Figure: JADE non-preemptive scheduling policy

Outline

1 What is JADE?

2 JADE Architecture

- JADE & FIPA
- JADE Agents
- JADE ACC

3 JADE Tools



The Agent Communication Channel I

JADE messaging runtime

According to the FIPA specification, JADE agents communicate via **asynchronous** message passing:

- each agent has a *message queue* (a sort of mailbox) where the JADE ACC delivers $\mathcal{ACL}$ messages sent by other agents
- whenever a new entry is added to the mailbox, the receiving agent is *notified*—it does not need to block nor to continuously ask either
- ! *if and when* the agent actually processes a message is up to the agent itself (or the programmer)—for the sake of agents **autonomy**



The Agent Communication Channel II

ACL-compliant messages

- To *understand* each other, it is crucial that agents agree on the **format** and **semantics** of the messages they exchange
- Hence, an *ACL* message contains:
 - :sender** who sends the message—automatically set
 - :receiver** who the message targets—may be many
 - :performative** the name of the **communication act** the agents want to carry out—constrained by a FIPA ontology
 - :content** the actual information conveyed by the message
 - :language** the syntax used to encode the **:content**
 - :ontology** the semantics upon which the **:content** relies
 - :** **:**
 - others** fields...

The Agent Communication Channel III

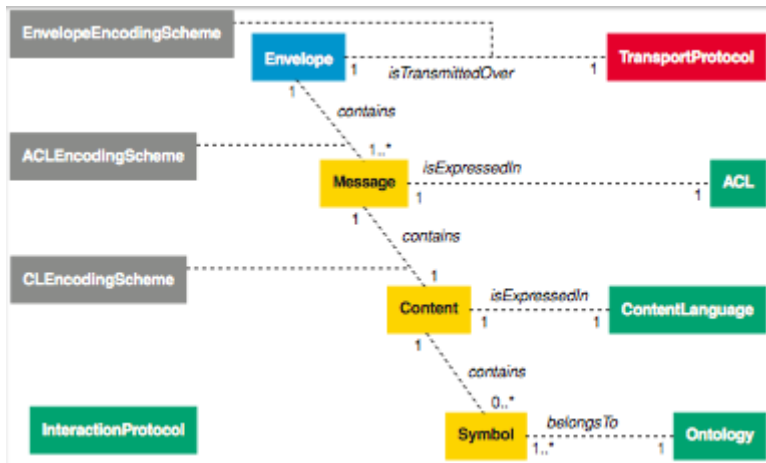


Figure: FIPA communication model abstractions

The Agent Communication Channel IV

JADE communication primitives

To interact, JADE agents have a number of ready-to-use methods:

send to send a message to a recipient agent

receive to **asynchronously** retrieve the first message in the mailbox (if any)

timed receive to perform a *timed*, **synchronous** receive on the mailbox—timeout causes agent to resume execution

selective receive to retrieve a message from the mailbox which *matches* a given *message template*—message queue order is bypassed

All these methods are **distribution-transparent**, that is they choose the proper address and transport mechanism based upon sender and receiver locations.

Outline

- 1 What is JADE?
- 2 JADE Architecture
 - JADE & FIPA
 - JADE Agents
 - JADE ACC
- 3 JADE Tools



JADE Management Tools I

RMA

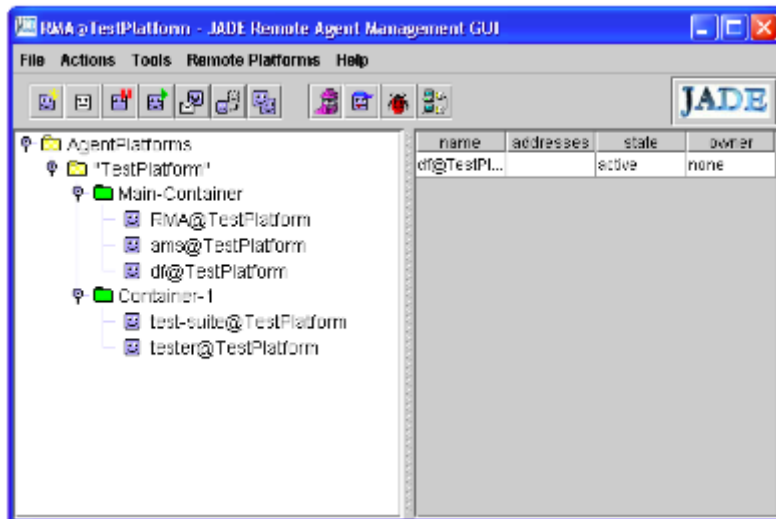
The **Remote Monitoring Agent** (RMA) allows to control the life cycle of the agent platform and of all the registered (possibly, remote) agents

RMA features

RMA allows to:

- start, stop, kill agents
- send them messages
- clone and/or migrate agents
- add, remove, shutdown (remote) platforms
- ... and much more

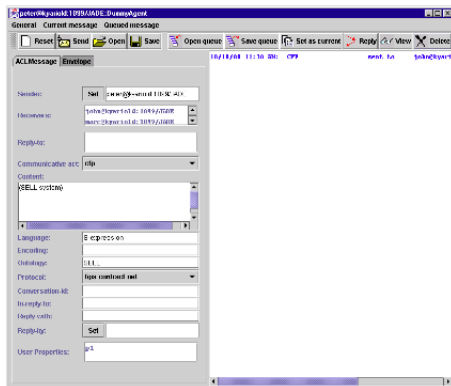
JADE Management Tools II



JADE Management Tools III

Dummy Agent

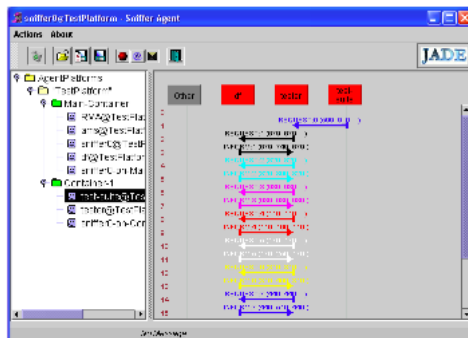
The **Dummy Agent** allows a human user to interact with JADE agents by sending, inspecting, recording custom *ACL* messages



JADE Management Tools IV

Sniffer Agent

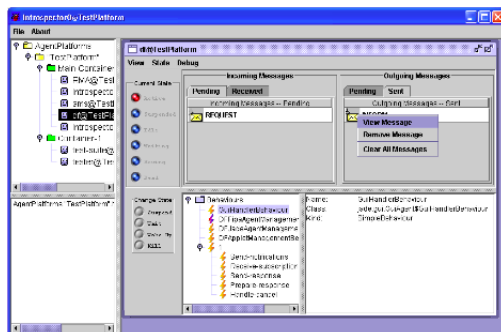
The **Sniffer Agent** allows a user to *sniff* an agent or a group of agents, which means that every message directed to/from that agent / agent group is tracked and displayed



JADE Management Tools V

Introspector Agent

The **Introspector Agent** allows to monitor and control both the queue of sent and received messages as well as the queue of behaviours—including executing them step-by-step



JADE: Java Agent DEvelopment Framework Overview

Stefano Mariani
`s.mariani@unibo.it`

Dipartimento di Informatica – Scienza e Ingegneria (DISI)
ALMA MATER STUDIORUM – Università di Bologna a Cesena

Academic Year 2015/2016

