

# **JADE Tutorial for beginners**



**Part 2 - USING JADE**  
**Fabio Bellifemine, TILAB**

# Table of content

- What is JADE
- Main features of JADE
- How to install and use JADE
- Graphical tools to monitor and debug agent systems
- Configuring JADE



# JADE

- **JADE is an agent platform that implements the basic services and infrastructure of a distributed multi-agent application:**
  - agent life-cycle and agent mobility
  - white & yellow-page services
  - peer-to-peer message transport & parsing
  - agent security
  - scheduling of multiple agent tasks
  - set of graphical tools to support monitoring, logging, and debugging
- **JADE allows faster time-to-market for new services by making key functionality available across multiple applications**
  - terminal2terminal and multi-party communication (N:M)
  - where needed, communication based on MSISDN-identity & mobile terminals providing (as well as accessing) services
  - pro-active applications
- **Some relevant features:**
  - is extremely light-weight, ported to J2ME-CLDC-MIDP 1.0
  - enables interoperability through FIPA compliance
  - is an Open Source project originated by TILAB and currently governed by an International Board
  - is used by several R&D projects



## Faster time to market: example of source code JADE and JXTA.

```
public class AgentThatSearchesAndUseAService
    extends jade.core.Agent {
public void setup() {
    DFAgentDescription dfd = new DFAgentDescription();
    dfd.setType("SearchedService");
    DFAgentDescription[] agents =
        DFService.search(this,dfd);
    ACLMessage msg = new
        ACLMessage(ACLMessage.REQUEST);
    msg.addReceiver(agents[0].getAID());
    msg.setContent("execute service");
    send(msg);
    System.out.println(blockingReceive());
}
}
```

```
public class PeerThatSearchesAndUsesAService {
    private void startJxta() {
        netPeerGroup =
            PeerGroupFactory.newNetPeerGroup();
        discoSvc =
            netPeerGroup.getDiscoveryService();
        pipeSvc = netPeerGroup.getPipeService();
    }
    private void startClient() {
        Enumeration enum1 =
            discoSvc.getLocalAdvertisements(
                DiscoveryService.ADV, "SearchedService",
                SERVICE);
        Enumeration enum2 =
            discoSvc.getRemoteAdvertisements(
                null, DiscoveryService.ADV,
                "SearchedService", SERVICE, 1, null);
        Enumeration enum = <enum1 + enum2>;
        ModuleSpecAdvertisement mdsadv =
            (ModuleSpecAdvertisement)enum.nextElement();
        StructuredTextDocument doc =
            (StructuredTextDocument)
                mdsadv.getDocument(new
                    MimeMediaType("text/plain"));
        PipeAdvertisement pipeadv =
            mdsadv.getPipeAdvertisement();
        Pipe sendPipe = pipeSvc.createOutputPipe(
            pipeadv, 10000);
        msg = pipeSvc.createMessage();
        msg.setString(TAG, "Request Service");
        sendPipe.send(msg);
        Pipe myPipe =
            pipeSvc.createInputPipe(pipeadv);
        System.out.println(myPipe.waitForMessage());
    }
    public void main() {
        startJxta();
        startClient();
    }
}
```

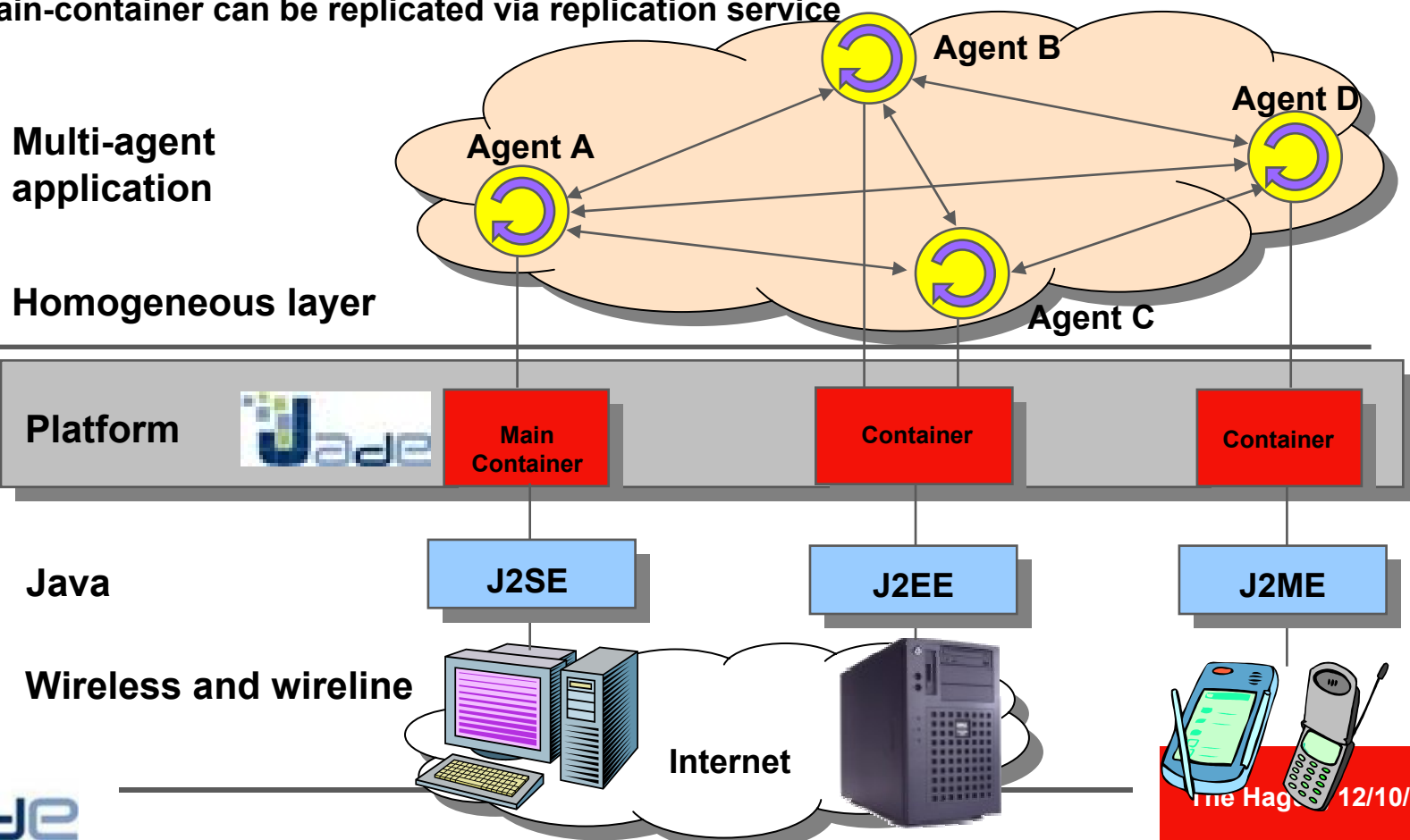


# JADE Hides FIPA From Programmers!

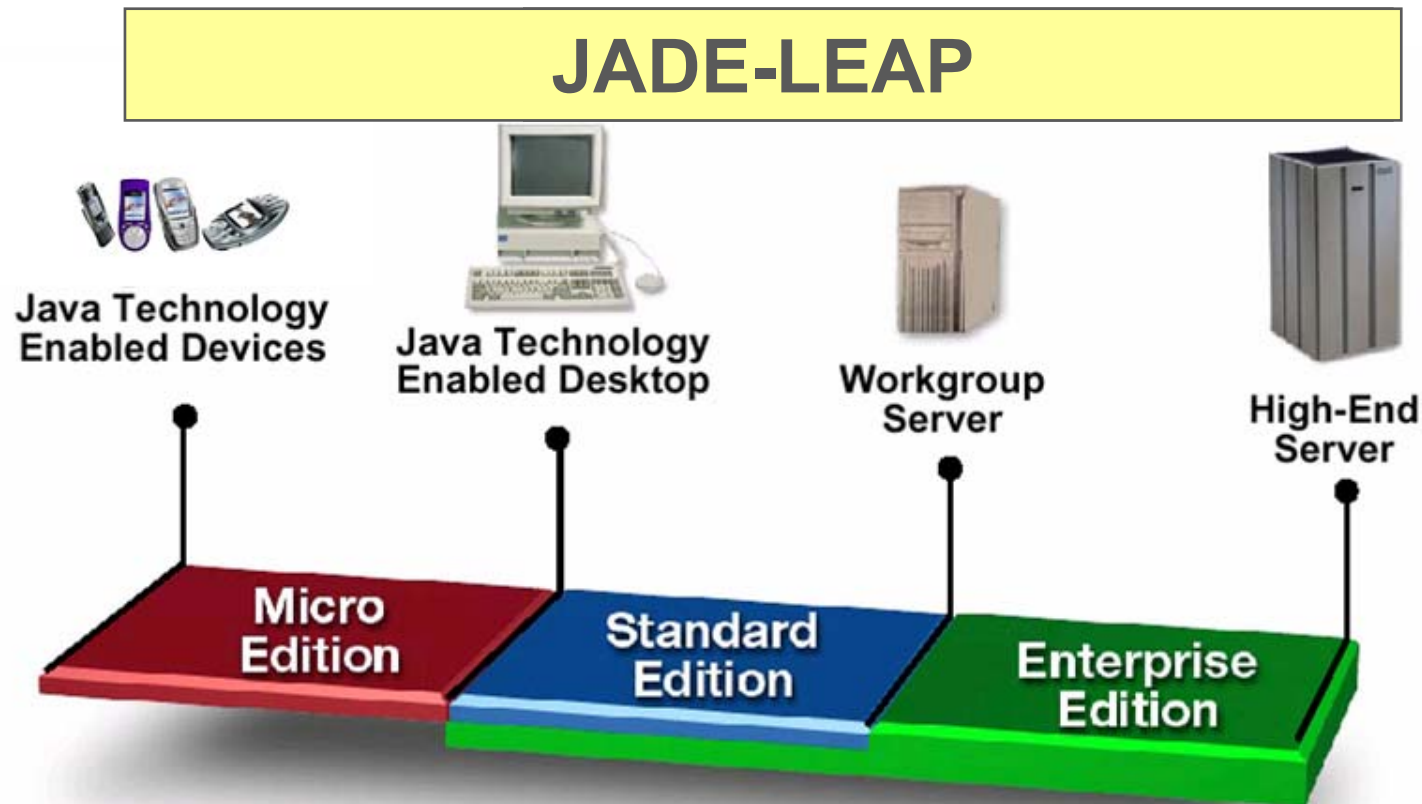
- **No need to implement the Agent Platform**
  - AMS, DF, and ACC executed at start-up
- **No need to implement agent-management ontology and functionalities**
  - An agent is registered with the AP within its constructor
    - It is given a name and an address
  - The DFService class provides a simplified interface to access the services of the DF (registration, searching, lease-renewal, ...)
- **No need to implement Message Transport and Parsing**
  - Automatically (and possibly efficiently) done by the framework when sending/receiving messages
- **Interaction Protocols must only be extended via handle methods**
- **AND it is standard FIPA**

# The architectural model

- ✓ A JADE-based application is composed of a collection of active components called Agents
- ✓ Each agent has a unique name
- ✓ Each agent is a peer since he can communicate in a bidirectional way with all other agents
- ✓ Each agent lives in a container (that provides its run time) and can migrate within the platform
- ✓ One container plays the role of main (where AMS, DF live)
- ✓ The main-container can be replicated via replication service



# Java 2 Platform and JADE



- footprint of the JADE-LEAP run-time on mobile phones:
  - 10-30 Kbyte if compiled with the JVM (ROMizing)
  - 40-100 Kbyte otherwise
- tested over almost all Java mobile phones
- integrated with Operator APN Radius Server to allow SIM-based addressing and authorization





# Downloading JADE – content of the files



The screenshot shows the JADE website's download page. At the top, the JADE logo is on the left, and the text 'Java Agent Development Framework' and 'an Open Source platform for peer-to-peer agent based applications' is in the center. A green banner on the right says 'The latest official version is JADE 3.2'. Below this is a navigation menu with links like 'description & value', 'community & developers', 'JADE board', 'application & business', 'papers & resources', 'news', 'faq', 'mailing lists', 'add-ons & 3rd party sw', 'bugs & suggestions', 'to do list', and 'the running agent platform'. A 'sections' sidebar on the left lists 'faq', 'mailing lists', 'bugs & suggestions', 'to do list', 'the running agent platform', 'logout', and 'edit your profile'. The main content area is titled 'download' and contains the following text:

From this page you can now download JADE. We recommend to unzip the files by using the 'jar xvf' command rather than the winzip application because some incompatibilities have been reported in the past.

Current version of JADE is 3.2 (26th July 2004). No patches are available for this version.

*Note: All the binaries (lib/\* .jar) have been generated by using the JDK1.4 compiler.*

File	~ File size	Description of the content
<a href="#">jadeAll.zip</a>	7.9 MB	This file contains all JADE, i.e. it is just composed of the 4 files below. If it is too large for downloading, the 4 files below might be downloaded instead.
<a href="#">jadeBin.zip</a>	1.4 MB	This file contains JADE already compiled and ready to be used, i.e. a set of JAVA archive JAR files.
<a href="#">jadeDoc.zip</a>	4.9 MB	This file contains all the JADE documentation included the Administrator's Guide and and the Programmer's Guide. <b>NOTICE THAT</b> all the documentation is also available on-line.
<a href="#">jadeSrc.zip</a>	1.6 MB	This file contains all the JADE source code.
<a href="#">jadeExamples.zip</a>	237 KB	This file contains the source code of the examples and a simple demo. All the examples and demo must be compiled.

At the bottom of the page, there is a Google search bar with the text 'Google Search' and a 'site map' link.





# JADE command line arguments

Usage: `java jade.Boot [options] [agent specifiers]`

- **most used options:**
  - `-help`
  - `-container` creates a container and joins it to an existing platform
  - `-host <hostname>` specifies the host of the platform to be joined
  - `-port <port number>` specifies the port number “ ” “ ” “ ” “ ”
  - `-gui` launches the remote monitoring agent
  - `-nomtp / -mtp` lists of MTPs (by default HTTP is launched)
  - `-conf <file name>` creates/loads a configuration file
  - `-<key> <value>`
- **agent specifiers:**
  - list of agents to launch, separated by a space
  - `<agentName>:<agentClass>(<agentParams>)`

e.g. `java jade.Boot -gui -nomtp -port 1200 W1:x.y.W(20) W2:x.y.W(10)`

*Note: refers to the JADE Administrator's Guide for the full list of options*



# The main graphical tools of JADE

- **supports the management, control, monitoring, and debugging of a multi-agent platform**
  - RMA (Remote Monitoring Agent)
  - Dummy Agent
  - Sniffer Agent
  - Introspector Agent
  - Log Manager Agent
  - DF (Directory Facilitator) GUI

# Remote Management Agent (RMA)

## Provided functionalities:

- monitor and control the platform and all its remote containers
- remote management of the life-cycle of agents (creating, suspending, resuming, killing, migrating, cloning)
- compose and send a custom message to an agent
- launch the other graphical tools
- monitor (just read operations) other FIPA-compliant platforms

The screenshot shows the 'JADE Remote Agent Management GUI' window. The title bar reads 'RMA@hpi11170:1099/JADE - JADE Remote Agent Management GUI'. The menu bar includes 'File', 'Actions', 'Tools', 'Remote Platforms', and 'Help'. The toolbar contains several icons, with four arrows pointing to them from labels above: 'Start Sniffer Agent' (purple icon), 'Start Dummy Agent' (blue icon), 'Start Log Manager Agent' (red icon), and 'Start Introspector Agent' (green icon). The main area is split into a tree view on the left and a table on the right.

Tree View:

- AgentPlatforms
  - "hpi11170:1099/JADE"
    - Main-Container
      - RMA@hpi11170:1099/JADE
      - df@hpi11170:1099/JADE
      - ams@hpi11170:1099/JADE

*java jade.Boot -gui*

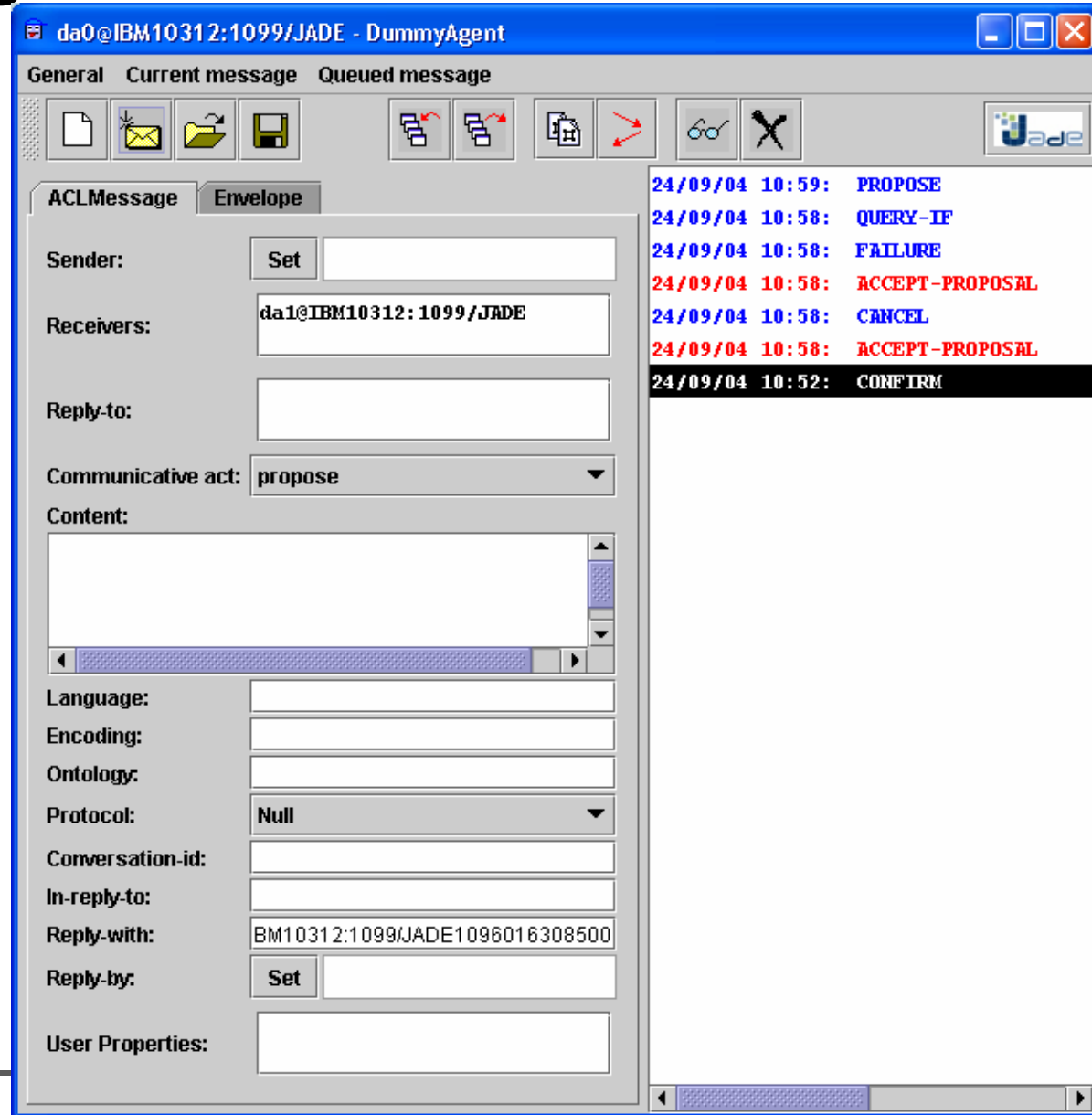
The Hague, 12/10/04



# Dummy Agent

## Provided functionalities:

- compose and send a custom messages
- load/save the queue of messages from/to a file



The screenshot shows the 'da0@IBM10312:1099/JADE - DummyAgent' window. It has tabs for 'General', 'Current message', and 'Queued message'. The 'Current message' tab is active, showing fields for Sender (Set), Receivers (da1@IBM10312:1099/JADE), Reply-to, Communicative act (propose), Content, Language, Encoding, Ontology, Protocol (Null), Conversation-id, In-reply-to, Reply-with (BM10312:1099/JADE1096016308500), Reply-by (Set), and User Properties. A toolbar with icons for file operations and message actions is visible. On the right, a log displays the following messages:

24/09/04	10:59:	PROPOSE
24/09/04	10:58:	QUERY-IF
24/09/04	10:58:	FAILURE
24/09/04	10:58:	ACCEPT-PROPOSAL
24/09/04	10:58:	CANCEL
24/09/04	10:58:	ACCEPT-PROPOSAL
24/09/04	10:52:	CONFIRM



# Sniffer Agent

## Functionalities:

- display the flow of interactions between selected agents
- display the content of each exchanged message
- save/load the flow on/from a file

The screenshot displays the Sniffer Agent interface. On the left, a tree view shows the hierarchy: AgentPlatforms > "hpi11170:1099/JADE" > Main-Container > RMA@hpi1170:1099/JADE. The main area shows a message flow diagram with three agents: 'Other', 'df', and 'a'. The flow consists of three messages: 0: INFORM:0 (591 075) from 'Other' to 'df'; 1: REQUEST:0 (591) from 'df' to 'Other'; 2: INFORM:0 (591 813) from 'df' to 'Other'. A dialog box titled 'ACL Message' is open, showing details for a message from 'da0@IBM10312:1099/JADE' to 'df@IBM10312:1099/JADE'. The message content is 'alive', the language is 'PlainText', and the protocol is 'ping-protocol'. The dialog also shows fields for 'Reply-to', 'Communicative act' (confirm), 'Conversation-id' (conversation1), 'In-reply-to', 'Reply-with' (reply1), and 'Reply-by' (20041024T085219000Z).



# Introspector Agent

## Functionalities:

monitoring  
agent internal  
state

- received/sent/  
pending msg
- scheduled  
behaviours  
(active,  
blocked) and  
sub-  
behaviours
- agent state

debugging  
execution

- step-by-step
- slowly
- break points

The screenshot displays the Introspector Agent GUI for the agent 'df@IBM10312:1099/JADE'. The interface is divided into several sections:

- Agent Platforms:** A tree view on the left showing the hierarchy of agent platforms, including 'Main-Container' and various agent instances like 'Introspector0@IBM10312:1099/JADE', 'b@IBM10312:1099/JADE', 'a@IBM10312:1099/JADE', 'da0@IBM10312:1099/JADE', 'da1@IBM10312:1099/JADE', 'sniffer0-on-Main', 'sniffer0@IBM10312:1099/JADE', 'RMA@IBM10312:1099/JADE', and 'df@IBM10312:1099/JADE'.
- Current State:** A vertical list of state options: Active (selected), Suspended, Idle, Waiting, Moving, and Dead.
- Change State:** A set of buttons for 'Suspend', 'Wait', 'Wake Up', and 'Kill'.
- Incoming Messages:** A panel with 'Pending' and 'Received' tabs. The 'Pending' tab shows a message 'CONFIRM'.
- Outgoing Messages:** A panel with 'Pending' and 'Sent' tabs. The 'Sent' tab shows two 'INFORM' messages.
- Behaviours:** A tree view showing the agent's behaviours, including 'GuiHandlerBehaviour', 'DFFipaAgentManagementBehaviour', 'DFJadeAgentManagementBehaviour' (selected), 'DFAppletManagementBehaviour', and a list of actions: 'Send-notifications', 'Receive-subscription', 'Send-response', 'Prepare-response', and 'Handle-cancel'.
- Details:** A text area on the right showing details for the selected behaviour: 'name: DFJadeAgentManagementBehaviour', 'class: jade.domain.DFJadeAgentManagementBehaviour', and 'id: DFJadeAgentManagementBehaviour'.



# Log Manager Agent

Is the GUI to modify at run-time the logging of the platform.

It is based upon `java.util.logging` and it allows to:

- browse all Logger objects on its container (both JADE-specific and application-specific)
- modify the logging level
- add new logging handlers (e.g. files)

da0@hpi11170:1099/JADE - LogManagerAgent

Logger Name	Set Level	Handlers	Set log file
jade.content.lang.sl.SL0Ont...	INFO	java.util.logging.ConsoleHandler	
jade.content.lang.sl.SL1Ont...	INFO	java.util.logging.ConsoleHandler	
jade.content.lang.sl.SL2Ont...	INFO	java.util.logging.ConsoleHandler	
jade.content.lang.sl.SLOntol...	INFO	java.util.logging.ConsoleHandler	
jade.content.onto.BasicOntol...	SEVERE	java.util.logging.ConsoleHandler, java.util.logging.FileHandler	myLog.txt
jade.content.onto.Ontology	INFO	java.util.logging.ConsoleHandler	
jade.content.onto.Serializabl...	INFO	java.util.logging.ConsoleHandler	
jade.content.schema.AgentA...	INFO	java.util.logging.ConsoleHandler	
jade.content.schema.Aggreg...	INFO	java.util.logging.ConsoleHandler	
jade.content.schema.Conce...	INFO	java.util.logging.ConsoleHandler	
jade.content.schema.Conte...	INFO	java.util.logging.ConsoleHandler	
jade.content.schema.IRESc...	FINER	java.util.logging.ConsoleHandler	
jade.content.schema.IRESc...	INFO	java.util.logging.ConsoleHandler	
jade.content.schema.Object...	INFO	java.util.logging.ConsoleHandler	
jade.content.schema.Predic...	SEVERE	java.util.logging.ConsoleHandler	
jade.content.schema.Primiti...	WARNING	java.util.logging.ConsoleHandler, java.util.logging.FileHandler	log.txt
jade.content.schema.TermS...	INFO	java.util.logging.ConsoleHandler	
jade.content.schema.Variabl...	CONFIG	java.util.logging.ConsoleHandler	
jade.core.AgentContainerImpl	FINE	java.util.logging.ConsoleHandler	
jade.domain.DFGUIManage...	FINER	java.util.logging.ConsoleHandler	
jade.domain.DFMemKB	FINER	java.util.logging.ConsoleHandler	
jade.domain.FIPAAgentMan...	FINEST	java.util.logging.ConsoleHandler	
jade.domain.FIPAAgentMan...	ALL	java.util.logging.ConsoleHandler	
jade.domain.JADEAgentMan...	INFO	java.util.logging.ConsoleHandler	
jade.domain.ams	INFO	java.util.logging.ConsoleHandler	
jade.domain.df	INFO	java.util.logging.ConsoleHandler	

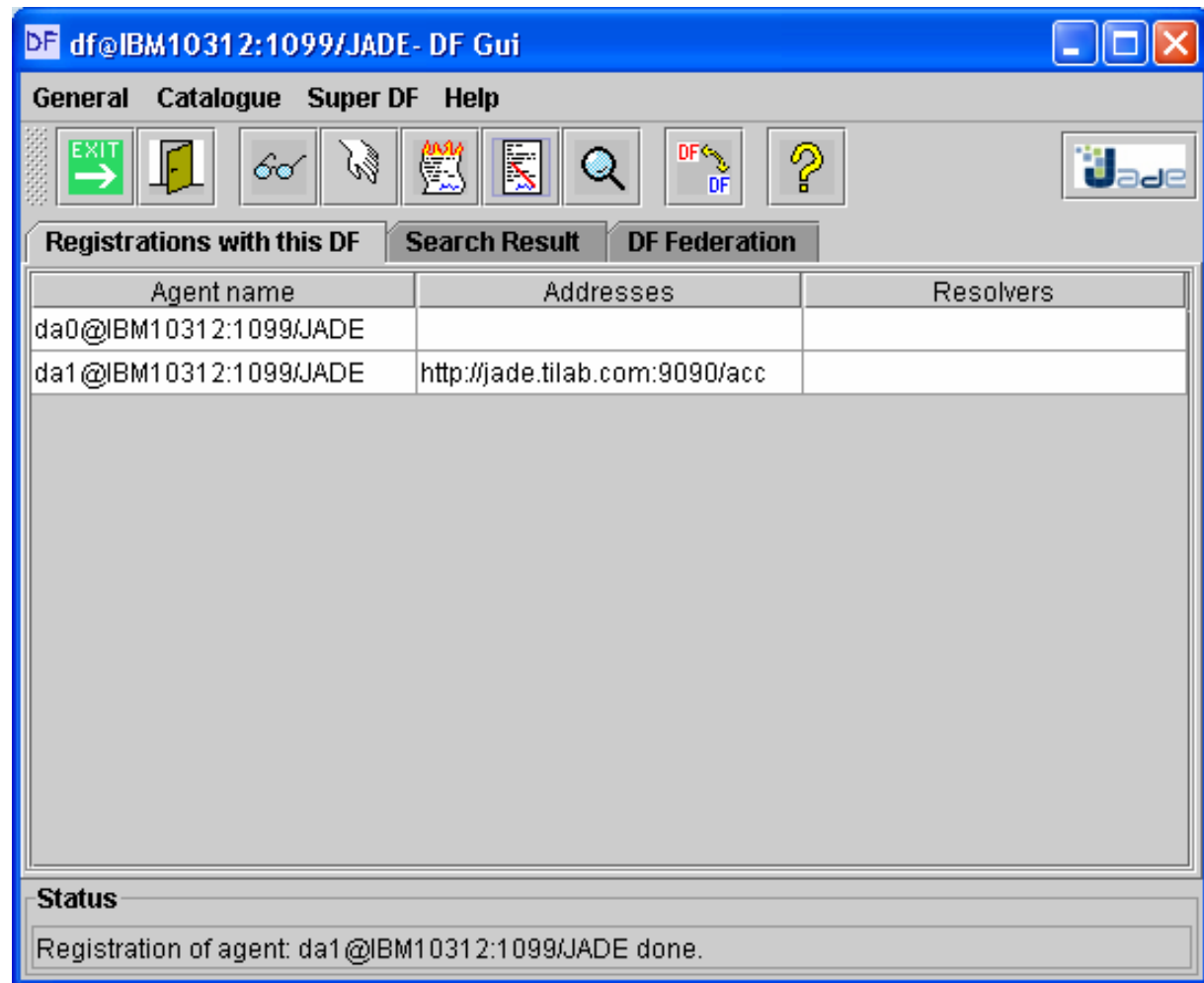




## DFGUI

GUI of the yellow-page service,  
it allows to:

- browse, register, deregister, modify, search agent descriptions
- federate with other DFs
- execute federated searches



# FYI – Some topics not fully covered by this tutorial

- **Integration with JESS (Java Expert System Shell)**
  - It allows reasoning about messages in JESS
  - It allows a JESS program to control sending/receiving messages and/or creating/destroying JADE behaviours
- **JADE and some Internet tools**
  - integration with servlets, applets, JSP
- **Advanced features**
  - distributed security, fault tolerance, support for replicated agents and services, persistence
  - application-specific persistent delivery filters & JADE kernel-level services
  - JADE and .NET
  - JADE, Protégé, XML, RDF and OWL

**Note: the documentation includes a tutorial for almost each of these aspects**

