
FIPA-OS V2.1.0 Distribution Notes

Reference FIPA-OS V2.1.0

Open Source Copyright Notice and License: FIPA-OS

1. The programs and other works made available to you in these files ("the Programs") are Copyright (c) 1999 - 2000 Nortel Networks Corporation, 8200 Dixie Road, Suite 100, Brampton, Ontario, Canada L6R 5P6. All rights reserved.
2. Your rights to copy, distribute and modify the Programs are as set out in the Nortel Networks FIPA-OS Public License, a copy of which can be found in file "Nortel_FIPA_OS_Public_Licence.txt" and the latest version can also be found at <http://fipa-os.sourceforge.net/>. By downloading the files containing the Programs you accept the terms and conditions of the Public License. You do not have to accept these terms and conditions, but unless you do so you have no rights to use the Programs.

The Original Code is Nortel Networks' FIPA-OS (Foundation for Intelligent Physical Agents - Open Source).

The Initial Developer of the Original Code is Nortel Networks Corporation. Portions created by Nortel Networks Corporation or its subsidiaries are Copyright (c) 1999 - 2000 Nortel Networks Corporation. All Rights Reserved.

Contributor(s):

Emorpha agree to provide Modifications to Nortel Networks' FIPA-OS Covered Code under Nortel Networks' FIPA-OS Public License. All Emorpha's Modifications remain Copyright (c) 2001 Emorpha Limited. All Rights Reserved.

Publication History

31 August 1999

First release. Forms part of the FIPA-OS v1.00 distribution.

13 September 1999

Second release. Forms part of the FIPA-OS v1.01 distribution which includes a complete set of the distribution notes.

24 September 1999

Minor updates to reflect the correct documentation. Forms part of the FIPA-OS v1.01a distribution which includes a complete set of the distribution notes.

29 October 1999

Updated to reflect bugs found in v1.01a and the new features and bug fixes included in v1.02.

19 January 2000

Updated to reflect bugs found in v1.02 and the new features and bug fixes included in v1.03.

20 March 2000

Updated to reflect bugs found in v1.03 and the new features and bug fixes included in v1.1.0. This release also includes guidelines on the version numbers for releases which is reflected for the first time in this release.

15 June 2000

Updated to reflect bugs found in v1.1.0 and the new features and bug fixes included in v1.2.0. Details of the package changes made in this release are also given.

18 August 2000

Updated to reflect bugs found in v1.2.0 and the new features and bug fixes included in v1.3.0. Many changes and fixes have been made. Most notably is FIPA-OS support of FIPA Experimental specifications approved in the Baltimore FIPA meeting.

6 September 2000

Updated to reflect new installation process and bug fixes to FIPA-OS v1.3.1

21 September 2000

Updated to reflect bug fixes to FIPA-OS v1.3.2

4 September 2000

Clarified some of the configuration details with regards to the Configurator. Removed details of how to manually configure profiles.

6 November 2000

Prepared for FIPA-OS v1.3.3 release

29 January 2001

Updated for FIPA-OS v1.4.0 release – updated file names etc...

16 March 2001

Updated for FIPA-OS v2.0.0 release

2 July 2001

Deprecated use of Configurator in preference to the Wizard

Table of Contents

About this document	v
Pre-installation	1
Requirements	1
Hardware Requirements	1
Software Requirements	1
Installation	3
Unpacking the FIPA-OS Installation	3
Self-extracting JAR	3
ZIP/GZ Archive	4
FIPA-OS Directory Structure	5
Using the FIPA-OS Configuration Tool (Configurator)	7
ACC Profile Configuration	8
Platform Profile Configuration	9
SetupFIPAOS Script Configuration	10
Default Profile Configuration	11
Agent Loader Profile	13
Configuration of Other FIPA-OS Components	14
Naming Service Batch/Script Files	14
Using the FIPA-OS Agent Platform	17
Starting the Platform Agents (AMS, DF)	17
FIPA Reference Model	17
Starting the Agent Platform using the Agent Loader	18
Starting the ACC	18
Testing the Platform Agents (AMS, DF)	19
Testing the AMS	19
Testing the DF	19
DF Cross Registration GUI	20
Swing DF GUI Interface Agent	20
Introduction	20
Using the Swing DFGUI	20
User Constructed Agents	22
Agent Shell	22
Using JESS Agent	22
Agent Development Tutorials	23
Updating FIPA-OS	24
Sharing Updates with FIPA-OS Users	24
FIPA-OS Packages	24
How do I know which package to use?	25
FIPA-OS Release Types	25
FIPA-OS Version Numbers	26
Compiling FIPA-OS Source Code	26
Obtaining the Latest FIPA-OS Source Code	26
Current Issues and Future Plans	27
Current Issues	27
Future Plans	27
FIPA2000	27
Profiles	27
Agent Naming convention	27

MTS.....	27
AgentLoader.....	28
ACC.....	28
DFGUI.....	28
FAQ.....	28

Bibliography	29
---------------------------	-----------

About this document

What is this document?

This document provides a brief explanation of:

1. The files included in the FIPA-OS download.
2. How the files should be installed.
3. How to start the FIPA-OS Agent Platform.
4. Testing for successful installation using a Test Agent.
5. How the platform can be built to incorporate changes to the code.
6. How to write further agents and how these agents can be executed on the platform.

Intended Audience

Developers installing, using and extending the FIPA-OS distribution.

Reading Guide

It is strongly recommended that the reader should look at the FIPA-OS web site at <http://fipa-os.sourceforge.net/> to understand the rationale behind this platform and for information on future updates. The installation, configuration, start-up and test instructions are written assuming that the developer will be using a Windows95/NT or Unix based system to run FIPA-OS.

Developers using FIPA-OS are encouraged to provide extensions, bug fixes and feedback to help improve the planned future releases. All such input should be contributed to the Open Source project via the SourceForge site at <http://sourceforge.net/projects/fipa-os/>. You are required to register as a developer to access some of the services at the SourceForge site. General issues and thoughts can be discussed via the FIPA-OS mailing list on fipa-os-developers@lists.sourceforge.net although you must register at <http://lists.sourceforge.net/mailman/listinfo/fipa-os-developers> on this list before you can send and receive messages. An archive of the messages sent to this list can also be viewed from <http://www.geocrawler.com/redirect.php3?list=fipa-os-developers>. Should you experience difficulties using this list, then please contact the FIPA-OS co-ordinators at fipaos@emorphia.com. Please consult the *FIPA_OS_Public_Licence.txt* file for further details on the requirements for using, extending and evolving FIPA-OS.

See the Change Log available from the FIPA-OS project web site at <http://fipa-os.sourceforge.net/docs/changes.html> for details of the technical changes made in this and past releases of FIPA-OS. Details of the current issues can be found in Bug Tracker, which is also available at the project web site at http://sourceforge.net/bugs/?group_id=3819&set=open&order=date.

Conventions used

Within the text filenames appear in *italics*. In examples where users should enter data, the suggested data appears in **bold**. For examples of entering data at the command prompt, variables are encapsulated in < and > and optional data is encapsulated in [and], e.g. [<comms-transport>] is an optional parameter which can be specified at the command prompt.

Terminology

AID	Agent Identifier [2]
ACC	Agent Communication Channel [1]
ACL	Agent Communication Language
AMS	Agent Management System [2]
AP	Agent Platform [2]
DF	Directory Facilitator [2]
DFGUI	DF Graphical User Interface
FIPA	Foundation for Intelligent Physical Agents
GUI	Graphical User Interface
HAP	Home Agent Platform [2]
HTTP	Hypertext Transmission Protocol
IDL	Interface Definition Language
IIOB	Internet Inter-ORB Protocol
IP	Internet Protocol
IOR	Interoperable Object Reference
IPMT	Internal Platform Message Transport
JESS	Java Expert System Shell
JRE	Java Runtime Environment
JVM	Java Virtual Machine
MTS	Message Transport Service [1]
MTP	Message Transport Protocol [1]
NS	Naming Service
ORB	Object Request Broker
Profile	Configuration details for the platform and agents
RDF	Resource Description Framework
RMI	Remote Method Invocation
URL	Universal Resource Locator
VM	Virtual Machine
XML	Extended Mark-up Language

Chapter 1

Pre-installation

Requirements

Hardware Requirements

The distribution has been tested primarily in Windows 95 and NT environments, although FIPA-OS has also been used successfully in Linux and Solaris environments. The recommended minimum hardware specification for PC's is:

- Pentium 166Mhz processor
- 64 MB Memory
- 4 MB disk space

Software Requirements

All of the 3rd-party software used by FIPA-OS is available for free download. Instructions for installing each of the 3rd party products are included in each of the specific downloads. Some third-party products may have to be licensed for commercial applications.

NOTE: The Java 1 and Java 2 implementations of FIPA-OS (provided as of FIPA-OS v1.3.3) have differing requirements, and are separate downloads.

Java 2 Runtime Environment (Mandatory)¹

This FIPA-OS build has been created using Java 1.2.2. The installation instructions assume that a JRE is installed on your computer, and your PATH environment variable is set-up to include the JRE bin directory.

JDK 1.1 Compatible Runtime Environment (Mandatory)²

Along side the standard FIPA-OS build is the JDK 1.1 FIPA-OS implementation, which requires the use of a JDK1.1 compatible JVM.

SiRPAC & SAX 1.0 XML Parser (Mandatory - included within FIPA-OS distribution)

The content language and agent profile parsers currently support XML\RFC encoding. In the current build of FIPA-OS the RDF parser (SiRPAC 1.14 from W3C at <http://www.w3c.org/RDF/Implementations/SiRPAC/>) and the XML parser (Xerces from Apache at <http://xml.apache.org>) are included in the distribution and will be installed automatically with FIPA-OS.

Enhydra XML Databinding Sub-system (Mandatory – included within FIPA-OS distribution)

This is a sub-set of the Enhydra project codebase, which provides XML-specific databinding. Data binding is the process of converting back and forth between a runtime object and a representation of that object that can be stored persistently or sent as part of a message. It is not tied to any programming language or to any constraint and instance document formats. See <http://www.enhydra.org/> or <http://zeus.enhydra.org/> for more details.

¹ For Java 2 FIPA-OS implementation only

² For JDK 1.1 FIPA-OS implementation only

JDOM Parser (Mandatory – included within FIPA-OS distribution)

JDOM is a simplified XML API used by the Enhydra databinding code. For more information, visit <http://www.jdom.org>.

JDK 1.1 Collection Classes (Mandatory)²

In order to enable the same functionality across both Java 2 and JDK1.1 versions of FIPA-OS, the JDK1.1 Collection classes are required. These are available from the JavaSoft InfoBus web-page @ <http://java.sun.com/products/javabeans/infobus/>

JDK 1.1 Swing Classes (Mandatory)²

In order to enable the same functionality across both Java 2 and JDK1.1 versions of FIPA-OS, the Swing 1.1.x classes are required. These are available from the JavaSoft Java Foundation Classes web-page @ <http://java.sun.com/products/jfc/>

Web-server (Optional)

A web server is one mechanism used to support the initial distribution of the transport addresses for remote agent platforms - this enables the required bootstrapping process for platform interoperability. A web server must be installed before the ACC can be started. There are no specific requirements for which web server to employ, nor is it necessary that a dedicated web server be installed in order to support FIPA-OS. It is envisaged that an existing web server could, quite satisfactorily, be employed in this role. However, if an existing web server is not available, the following open source web servers are suitable for all platforms:

- Apache HTTP Server Project (<http://www.apache.org/httpd.html>)

JESS (Optional)

JessAgent is a special Agent Shell that incorporates the reasoning abilities provided by JESS (Java Expert System Shell). To utilise this agent you must first download and install JESS from <http://herzberg.ca.sandia.gov/jess/>. The required distribution file (version 5.0) can be found by selecting the "downloads" option. Installation instructions can be found in Chapter 2 of these notes, as it is not mandatory to install this software before installing and using FIPA-OS.

Java Secure Sockets Extension (Optional)¹

This is an extension to the core Java API which adds SSL support to any Java 2 platform. It can be located at <http://java.sun.com/products/jsse/>

Chapter 2 Installation

FIPA-OS distributions starting with version 1.3.0 uses an improved installation system. The installation is divided into 2 stages:

- Unpacking the FIPA-OS installation into a user-selected directory.
- Using the FIPA-OS Configuration Wizard to customise the installation for the local environment.

Unpacking the FIPA-OS Installation

There are two types of download available from the FIPA-OS website:

- Self-extracting JAR file containing of FIPA-OS
- ZIP or TAR/GZ archive of FIPA-OS

It is recommended that the self-extracting JAR file be used if you are unfamiliar with WinZIP, tar, or FIPA-OS.

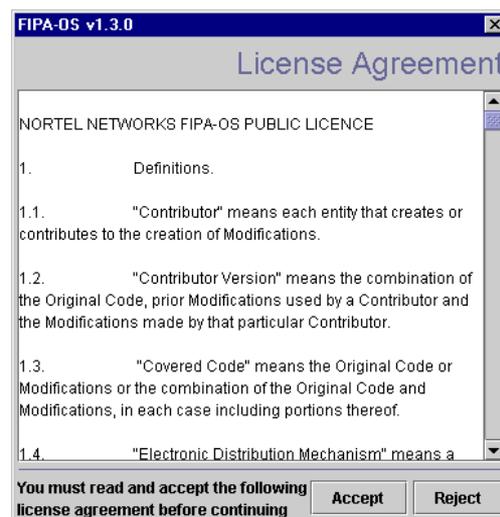
Self-extracting JAR

The self-extracting JAR file contains Java classes that take care of the process of installing FIPA-OS on your system, and starting the FIPA-OS Wizard. (NOTE: This option is currently only available for the Java 2 FIPA-OS implementation distribution).

There are two mechanisms for starting the installation process:

- Double-click the JAR file (Windows only) – this will only work if the default file-type created by the JRE installer is unchanged
- Type `java -jar FIPA_OSv2_1_0_Installer.jar` at a command prompt (All operating systems)

After a short while you will be presented with a license agreement screen. You must agree with the license agreement to continue:



The installer will then prompt you for an installation location:



You may either select an existing directory, or create a new directory. In the later case, you should click the “Create New Folder” button and then rename the new folder as you wish. In order to select a directory it must be highlighted within the dialog box when you click “Open”.

If the selected directory already existed, you will be prompted if you wish to backup its contents, and whether you want to delete its contents (this is recommended, since installing new versions of FIPA-OS over an older version may result in unpredictable behaviour).

If you select “Yes”, the contents of the directory will be placed into a JAR file in its parent directory.

If there are any problems removing the directory or files it contains, the install process will abort. In this case simply remove the directory manually, and restart the installer.

Once FIPA-OS has completed installation, the Wizard will be automatically started. In the event that it fails to start after successful installation, please follow the instructions in the *ZIP/GZ Archive* section of this chapter for starting the Wizard.

On Unix/Linux systems, the following step may need to be carried out in order to ensure that the SH scripts provide are executable. The permissions of the files in the *bat* directory should be set so they are treated as executable files. This can be achieved by typing `chmod -R 755 bat/*` at a command prompt in the FIPA-OS installation directory.

ZIP/GZ Archive

It is recommended that FIPA-OS is NOT installed on top of an older version, since this may lead to unforeseen compatibility issues between the two versions (applies only to the ZIP/GZ archive versions of FIPA-OS only)

These archives should simply be extracted into the desired installation directory. The Wizard should then be used to correctly configure FIPA-OS. The following are step-by-step guides for various operating systems:

Windows (e.g Windows 95, 98, NT and 2000)

1. Download the FIPA-OS distribution (http://download.sourceforge.net/FIPA-OS/FIPA_OSv2_1_0.zip)
2. After downloading, the distribution file (FIPA_OSv2_1_0.zip), it should be extracted using an archive tool such as WinZip into the desired installation directory.
3. Once the contents of the distribution file has been extracted invoke the *StartWizard.bat* file which is located in the 'bat' sub-folder of the installation directory to start the Wizard.

Alternatively start the *StartConfig.bat* file and follow the instructions in the *Using the FIPA-OS Configuration Tool (Configurator)* section.

UNIX (e.g. Linux, Solaris, etc.) Instructions:

1. Download the FIPA-OS distribution (http://download.sourceforge.net/FIPA-OS/FIPA_OSv2_1_0.tar.gz). You will need to right-click this link and select "Save Link As..." or "Save Target As...", depending on your browser, and then select a destination in which to save the file.
2. Open a shell and 'cd' to the directory where you downloaded the distribution file. To extract the contents of the FIPA-OS distribution file type `tar -xvz -fFIPA_OSv2_1_0.tar.gz`. NOTE: the contents of the distribution file will be expanded in your current directory.
3. The permissions of the files in the 'bat' directory should be set so they are treated as executable files. This can be achieved by typing: `chmod -R 755 bat/*` at a command prompt in the FIPA-OS installation directory.
4. The Java 1.2 runtime must be in your path in order to complete the installation. You can add this information to your system PATH variable (if it is not already present) using the 'export' command. For example, if the JDK was installed in '/opt/software/jdk1.2.2' then typing the following command at the prompt will add this information to your PATH: `export PATH=/opt/software/jdk1.2.2/bin:$PATH`
4. At the prompt, navigate to the 'bat' sub-folder of the installation directory and type: `sh ./StartWizard` to start the Configuration Wizard.

FIPA-OS Directory Structure

Path and filename	Description
index.html	HTML index of the distribution documentation. Start here.
bat\	Directory containing distribution batch/script files (for launching Agents and configuration)
bat\StartFIPAOS	Batch/script file to start FIPA-OS
certificates	Security certificate key-stores for FIPA-OS agents are located here. Currently these are only used by the RMI over SSL implementation
classes\FIPA_OSv2_1_0.jar	Compiled FIPA-OS core without diagnostic support.
classes\FIPA_OSv2_1_0_debug.jar	Compiled FIPA-OS core with diagnostic support.
databases	This is the default location where persistent databases will be stored by Agents
imports\	Directory containing the third-party components SiRPAC and Xerces. All JAR or ZIP files in this directory are added to the classpath by SetupFIPAOS – simply add third party libraries to this directory to make use of them.
src\	Directory containing all source code for the FIPA-OS Platform.
javadocs\	Directory containing the JavaDoc's describing the FIPA-OS classes
docs\	Directory containing documents including tool instructions and this document.

Path and filename	Description
docs\licenses	Contains licenses referenced by the FIPA-OS code-base and associated documentation
tools\ profiles\ examples\ FIPA_OS_Copyright.txt	Directory containing associated tools. Directory containing Agent profiles for each of the Agents included in this distribution. Example ACL messages for various situations. Copyright Notice.
FIPA_OS_Public_License.txt	Public Licence.
FIPA_OS_Installation_Marker.txt	Text file used by the installation procedure – denotes build number of FIPA-OS. Do NOT delete this file.

Using the FIPA-OS Configuration Tool (Configurator)

Please note that the FIPA-OS Wizard is now the preferred mechanism for configuring the most common options of FIPA-OS. The Configurator should not be needed in order to correctly configure a normal installation.

If you wish to run Configurator after installation of FIPA-OS, the following information may be useful:

- The classes for Configurator are in the package `fipaos.tool.configurator` and will be included in the file `FIPA_OSv2_1_0.jar` of the distribution.
- The script that starts Configurator is called *StartConfig.bat* (windows) or *StartConfig* (UNIX or Solaris)
- Configurator modifies the following core FIPA-OS configuration files:
 - *acc.profile* – Provides configuration information for the ACC of a platform, including which MTP's the platform is using, additional (external) MTP's it should use for inter-platform communication, details of other platforms that should be contacted at start-up and details of which database type to use to store this information.
 - *platform.profile* – Describes information about the FIPA-OS platform, including its name, the "location" of the AMS and the location of other profiles used by entities within the platform (i.e. Agents & ACC).
 - *default.profile* – Provides default Agent configuration information for Agents without a profile. It defines the MTP's used by the platform which an Agent on that platform should bind to, and the database type that should be used by default when constructing databases.
 - *loader.profile* – Describes the types of agent that the AgentLoader can start.
 - *SetupFIPAOS.bat* (windows) or *SetupFIPAOS* (UNIX or Solaris) – Sets up the necessary environment variables to enable FIPA-OS to execute on the host OS it is contained within (e.g. sets up classpath, path and arguments for the Java/JRE executable).
- Configurator requires that the SiRPAC and Xerces classes, or jars, be in the system classpath before it is run.
- Configurator can only modify existing versions of the configuration files; it cannot create new versions.

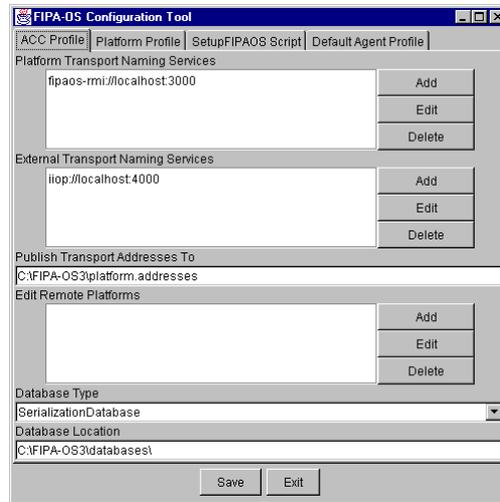
Configurator assumes it is being run from the *bat* directory within the FIPA-OS installation. If it cannot find the FIPA-OS installation directory (indicated by the presence of the file *FIPA_OS_Installation_Marker.txt*) then it will prompt the user to enter the correct path to the FIPA-OS installation directory.

Configurator loads the *SetupFIPAOS* script or batch file, the ACC profile, default profile, and the platform profile. If any of these files cannot be found then a prompt window will appear asking for the directory containing the missing file.

The Configurator GUI will appear once the configuration files have all been loaded. The GUI consists of four panes that enable the core aspects of the agent platform to be configured. The first time the Configurator is executed, the "Save" button MUST be pressed, otherwise the initial configuration information provided will NOT be saved.

ACC Profile Configuration

This panel allows the user to edit the information used to configure the Agent Communication Channel for the platform.



Platform Transport Naming Services

This field contains details of the platform MTP's that the ACC should bind into upon start up. Each MTP requires a URL describing the physical location of a NS for that transport of the form:

```
<protocol>://<ns-hostname>:<ns-port-number>/
```

where the available protocols are:

- **fipaos-rmi**

External Transport Naming Services

This field contains details of the external MTP's that the ACC should bind into upon start up. Each MTP requires a URL describing the physical location of a NS for that transport of the form:

```
<protocol>://<ns-hostname>:<ns-port-number>/
```

where the available protocols are:

- **iiop.**

Publish Transport Addresses To

This field specifies the **filename** to which the ACC publishes its MTP's addresses, which provides the boot-strapping mechanism for other platforms – please see the FIPAOS Inter-platform Communication Configuration Guide for more details of this process.

Remote Platforms



Each remote platform description contains details about a remote agent platform. Each remote platform description object specifies the two values:

- Platform Name - the name of the remote platform (as used as the HAP name within AID's of Agents on that platform [2])
- Location of published addresses - the location where the addresses of the remote platform are published – this could be a URL from where the remote platforms' addresses file can be accessed, or a path to file. For the remote platform this will be its "Publish Transport Addresses To" file, as defined within its configuration (assuming the platform is a FIPA-OS platform).

Database Type

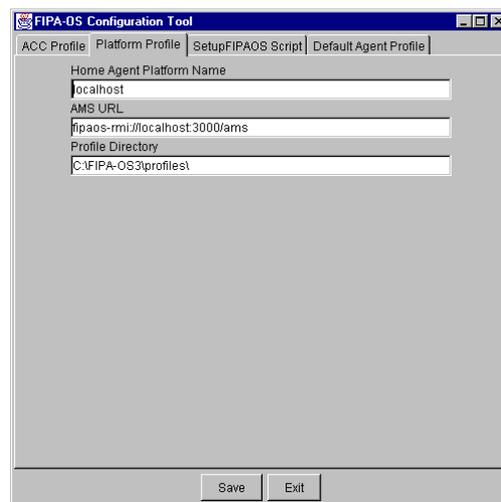
This field selects the type of database used by the ACC. If MemoryDatabase is chosen then remote platform registration information exists only in the runtime memory and will be lost when the system shuts down. If SerializationDatabase is chosen, remote platform registration information will be preserved in a file.

Database Location

This field gives the location in which to persist the internal data of the ACC. This field is only used if the database type specified is persistent.

Platform Profile Configuration

The Platform Profile is used to store the configuration details for the FIPA Agent Platform that the installation of FIPA-OS will belong to.



Home Agent Platform Name

The HAP name [2] used by Agents on the platform this installation will be part of - this should be globally unique (a sensible value is the fully qualified hostname of the host on which the AMS/ACC will be running). The HAP is simply a unique identifier and will not be used to resolve the platform location

AMS URL

This is the internal MTP address via which the AMS can be contacted. It should be of the form:

```
<protocol>://<ns-hostname>:<ns-port-number>/<agent-name>
```

where the available protocols are:

- **fipaos-rmi**
- **fipaos-ssl-rmi**
- **iiop**

and the agent-name is **ams**.

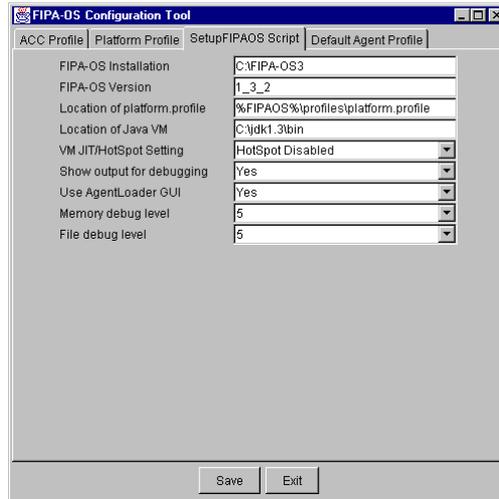
The NS hostname and port are determined by the NS used by the platform.

Profile Directory

This specifies where the profiles for entities belonging to this platform can be located.

SetupFIPAOS Script Configuration

This GUI panel configures the parameters in the FIPA-OS platform start-up script. These parameters are used to set the environment variables for the platform.



FIPA-OS Installation

This sets the environment variable FIPAOS, which contains the directory where FIPA-OS is installed.

FIPA-OS Version

This sets the environment variable FIPAOS_VER, which indicates the version of FIPA-OS JARs being used. A value of “1_00” will select the FIPA_OSv1_00.jar. A value of “1.3.0” would select FIPA_OSv1.3.0.jar and so on. This allows multiple versions of the FIPA-OS JARs to be installed in the FIPA-OS classes directory, enabling a user to switch versions by simply changing this variable.

Location of platform.profile

This sets the environment variable PLATFORM, which specifies where the “platform.profile” file can be located for use by the Agent Platform. This can take the form of either a local filename or a URL. The default value is “%FIPAOS%\profiles\platform.profile” or “\$FIPAOS\profiles\platform.profile” depending on the operating system in use..

Location of VM

This sets the environment variable JDK, which selects the location of the Java virtual machine you wish to use. This variable is put at the start of the system path variable and can be used to override any Java virtual machine already in the system path. If a Java virtual machine is already in the system path then this variable can be left blank.

VM JIT/HotSpot Setting

This sets the environment variable NOJIT, which is used as a switch to the Java virtual machine. Its purpose is to allow the disabling of any JIT or HotSpot performance compiler used by the VM. By

disabling JIT compilation you will be able to trace exceptions to the exact line of source code that they occurred on, but this will cause a reduction in the performance of the virtual machine.

Show output for debugging

This sets the environment variable DEBUGVER. If this variable is set to “On”, then the diagnostic versions of the FIPA-OS JARs will be used. If it is set to “Off” then the non-debug versions of the jars will be used. (NOTE: if some errors occur, it will be much easier to track down the problem using the debug JAR’s).

Use AgentLoader GUI

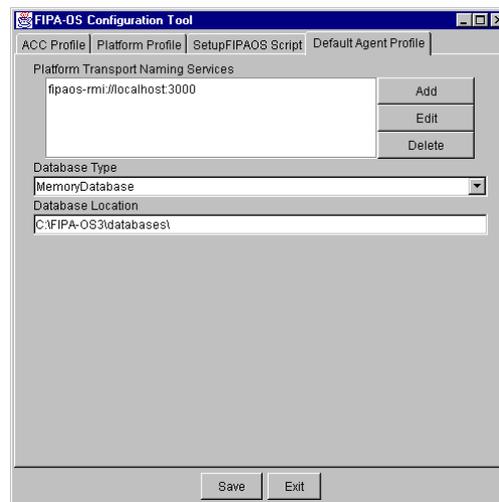
This selection allows selection of whether the AgentLoader should attempt to create a GUI when it starts. If this is set to “No”, the AgentLoader will simply load the Agents it would normally load at startup.

Memory debug level & File debug level

These settings allow the level of DIAGNOSTICS output sent to the screen or the DIAGNOSTICS.txt file to be configured. Level 1 provides the most detailed results, whilst Level 5 displays only the most important information (i.e. such as exceptions). These levels will only have effect on FIPA-OS classes if the debug JAR’s are in use.

Default Profile Configuration

This panel allows the user to edit the information used to configure the profile for agents that do not have their own profiles.



Platform Transport Naming Services

This field contains details of the platform MTP’s that Agents should bind into upon start up. Each MTP requires a URL describing the physical location of a NS for that transport of the form:

```
<protocol>://<ns-hostname>:<ns-port-number>/
```

where the available protocols are:

- **fipaos-rmi**
- **fipaos-ssl-rmi**

Database Type

This field selects the type of database used by agents using the default profile. If MemoryDatabase is chosen then remote platform registration information exists only in the runtime memory and will be

lost when the system shuts down. If `SerializationDatabase` is chosen, remote platform registration information will be preserved in a file.

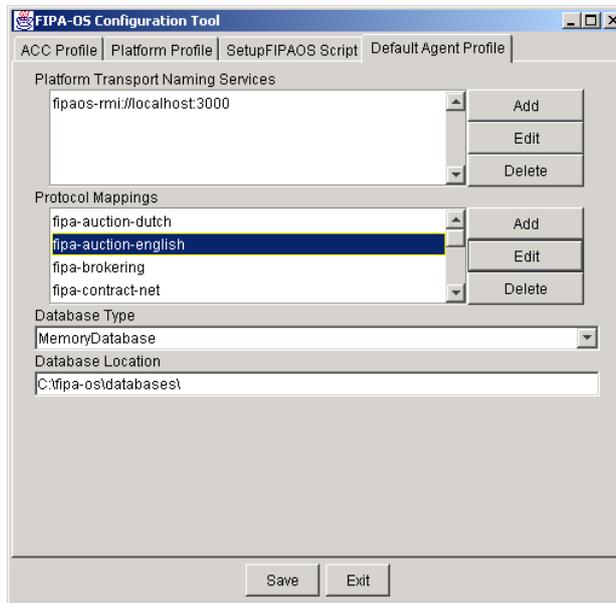
Database Location

This field gives the location in which to persist the internal data of agents using the default agent profile. This field is only used if the database type is `SerializationDatabase`.

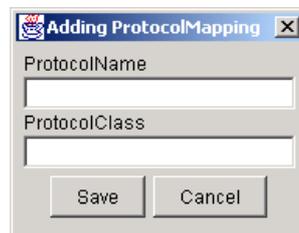
Dynamic Protocol Mappings

With release 2.0.0 of the platform, FIPA-OS now supports dynamic mappings between conversation protocol identifiers and the classes that implement those protocols. This adds support for dynamic agent protocol learning behaviour where an agent can learn to understand new conversation protocols during its lifetime.

The FIPA-OS configuration tool has been updated to allow editing of the `ProtocolProfile` member of the default agent profile.



The default agent profile editor now has a new section for editing an agent's `ProtocolProfile`. The Add button allows you to add new protocol mappings to the `ProtocolProfile`:

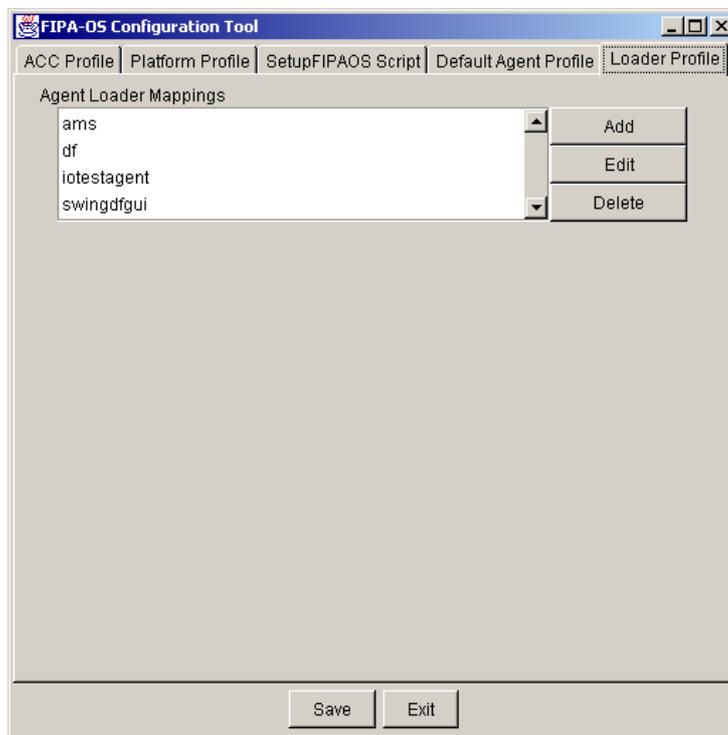


The Edit button allows you to modify an existing mapping in the `ProtocolProfile`:

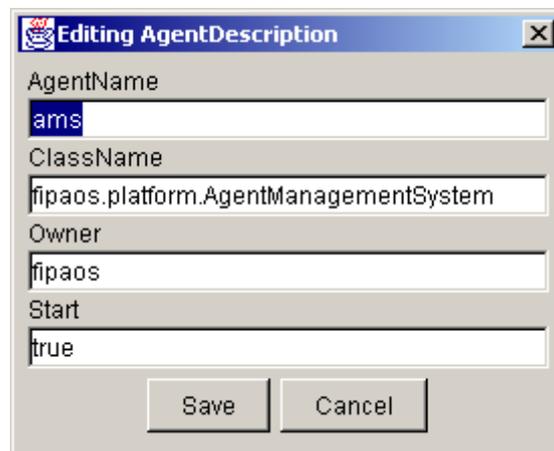


Agent Loader Profile

New in FIPA-OS v2.0.0, this panel allows you to modify the Agents available within the AgentLoader. Agents can be setup to start automatically with the AgentLoader, or manually from the AgentLoader GUI. A list of currently known agents is displayed. Select Add, Edit or Delete to modify the list appropriately.



When adding or editing the entries, the following dialog is displayed:



The parameters required are:

- AgentName: Name that the agent will use when started
- ClassName: Name of the class which contains the agent implementation
- Owner: The owner of this agent.
- Start: If set to “true”, will start the agent when the AgentLoader starts

Configuration of Other FIPA-OS Components

Naming Service Batch/Script Files

It may be necessary to modify the StartSUNCORBANamingService and StartRMINamingService or StartFIPAOS script or batch files to start the NS's on the correct ports if you are not using the Wizard to configure your FIPA-OS installation. On the hosts which will run the NS's, open these batch/script files, and modify the port number used by the NS's. For the StartRMINamingService bat/script, change:

```
java fipaos.mts.rmi.internal.ns.RMIAgentNamingServiceImpl 3000
```

so that the value given (in this case 3000) matches the port to be used by the. For the StartSUNCORBANamingService bat/script change:

```
tnameserv -ORBInitialPort 4000
```

so that the value given (in this case 4000) matches the port to be used by the NS.

The Configuration Wizard will automatically configure StartFIPAOS based upon your selections – in this event you don't need to modify these files. See [4] for more details.

Set-up of RMI over SSL

The RMI over SSL transport of FIPA-OS utilizes the Java Secure Sockets Extension and Sun's reference implementation of SSL.

Quick Start

FIPA-OS come pre-configured to use SSL – all that you have to do is:

- Select the “fipaos-ssl-rmi” protocol for use by your Agents
- Install the JAR's provided as part of JSSE into the FIPA-OS imports directory – they will then be automatically added to the classpath.

FIPA-OS will take care of the rest. **NOTE: In this scenario, you'll be using the certificates that ship as part of FIPA-OS to authenticate SSL connections. Since this certificate is available publicly and the private key is easily accessible, you should use these keys for testing purposes only.**

How the keystores are used by FIPA-OS

Keystores used by FIPA-OS include:

- *certificates/agent_keys*
Certificates used by an Agent to authenticate itself when sending/receiving a message should be added to this keystore.
- *certificates/trusted_keys*
Certificates that are used by trusted agents should be placed here. An SSL exception is generated if an SSL connection is made to an Agent that doesn't have any trusted certificates (defined in their agent_keys file).

By default, the password for these keystores is set to “changeme” – it is recommended that these are changed if the keystores are on a non-secure medium. If the passwords are changed from the default value, the new passwords should be passed into VM’s using the RMI over SSL using the following Java properties:

- `fipaos.agent_keys.pass`
Password for the *agent_keys* keystore
- `fipaos.trusted_keys.pass`
Password for the *trusted_keys* keystore

E.g. if the password for the *agent_keys* file were changed to “newpass”, “-Dfipaos.agent_keys.pass=newpass” should be added as an option to the Java command

Advanced Configuration

In order to improve the security of the system, you should add your own certificate to the *agent_keys* keystore, and delete the existing key. You should also add your certificate to the *trusted_keys* keystore and delete the existing key.

The keystores are in standard Java KeyStore format, and can be manipulated using the *keytool* program that comes with the JDK/JRE.

To delete the default certificates from the keystore:

```
C:\FIPAOSv200\certificates>keytool -delete -alias fipaos-agent -keystore agent_keys
Enter keystore password: changeme
```

```
C:\FIPAOSv200\certificates>keytool -delete -alias fipaos-agent -keystore trusted_keys
Enter keystore password: changeme
```

Then generate your own certificate, type the following and response appropriately:

```
C:\FIPAOSv200\certificates>keytool -genkey -alias fipaos-agent -keystore agent_keys
Enter keystore password: changeme
What is your first and last name?
[Unknown]: FIPA-OS Agent Key
What is the name of your organizational unit?
[Unknown]: Software Development
What is the name of your organization?
[Unknown]: BigOrg
What is the name of your City or Locality?
[Unknown]: Harlow
What is the name of your State or Province?
[Unknown]: Essex
What is the two-letter country code for this unit?
[Unknown]: UK
Is <CN=FIPA-OS Agent Key, OU=Software Development, O=BigOrg, L=Harlow, ST=Essex,
C=UK> correct?
[no]: yes

Enter key password for <fipaos-agent>
(RETURN if same as keystore password):
```

The next step self-certifies your certificate, although you may wish to be certified by a certification authority such as VeriSign – please see the Java KeyTool documentation for more information:

```
C:\FIPAOSv200\certificates>keytool -selfcert -alias fipaos-agent -keystore agent_keys
Enter keystore password: changeme
```

The newly created certificate needs to be added to the *trusted_keys* keystore – it must be exported from *agent_keys*, and imported into *trusted_keys*:

```
C:\FIPAOSv200\certificates>keytool -export -alias fipaos-agent -keystore agent_keys -file key
```

```
Enter keystore password: changeme
```

```
Certificate stored in file <key>
```

```
C:\FIPAOSv200\certificates>keytool -import -alias fipaos-agent -keystore trusted_keys -file key
```

```
Enter keystore password: changeme
```

```
Owner: CN=FIPA-OS Agent Key, OU=Software Development, O=BigOrg, L=Harlow, ST=Essex, C=UK
```

```
Issuer: CN=FIPA-OS Agent Key, OU=Software Development, O=BigOrg, L=Harlow, ST=Essex, C=UK
```

```
Serial number: 3aa50a0f
```

```
Valid from: Tue Mar 06 16:02:23 GMT 2001 until: Mon Jun 04 16:02:23 GMT 2001
```

```
Certificate fingerprints:
```

```
MD5: 5C:15:B8:2C:D4:0E:CA:76:D5:A5:4E:54:D2:34:CF:72
```

```
SHA1: C3:AE:58:D4:92:E8:B7:60:00:EE:EB:F2:78:9E:35:EF:3D:2A:0A:4E
```

```
Trust this certificate? [no]: yes
```

```
Certificate was added to keystore
```

Your new certificate has now been added, and can be used by the JSSE SSL API to authenticate connections.

Chapter 3

Using the FIPA-OS Agent Platform

To start the FIPA Agent Platform, FIPA-OS must first be installed and configured as described in Chapter 1. In addition to the platform configuration it is suggested that the `autoexec.bat` (or equivalent Unix script such as `.cshrc`) 'path' environment variable is set to include the path of the `\FIPA-OS\Bat\` directory.

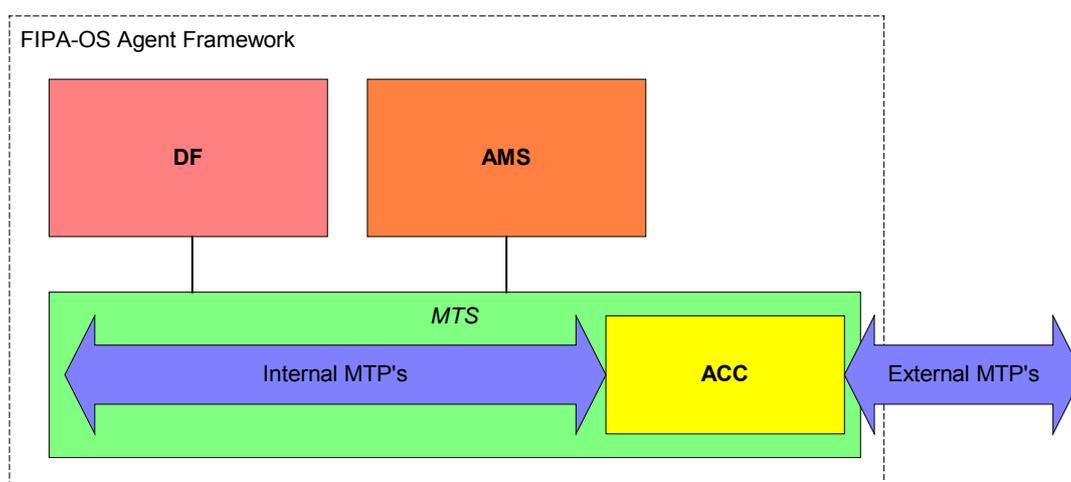
Starting the Platform Agents (AMS, DF)

An Agent Loader is included in the distribution, which allows for arbitrary agents to be loaded into the same VM. One VM is used by each instance of the Agent Loader. The Agent Loader enables the Platform Agents to be started and any other arbitrary agent as required. The Agent Loader supports additional functionality, which enables agents to be shut down via a GUI. Agents requiring Agent Loader start-up support have to be written using a constructor with three parameters, e.g:

```
public MyAgent(String platform_profile_location, String name, String owner)
```

FIPA Reference Model

The FIPA reference model shown below illustrates the core components of the FIPA-OS distribution. The agent reference model provides the normative framework within which FIPA Agents exist and operate. Combined with the Agent Life cycle, it establishes the logical and temporal contexts for the creation, operation and retirement of Agents.



The Directory Facilitator (DF) and Agent Management System (AMS) are specific types of agents, which support agent management, and the Agent Communication Channel (ACC) is a lower-level entity that is part of the MTS (Message Transport Service). The DF provides "yellow pages" services to other agents. The AMS provides platform management functionality, such as monitoring Agent lifecycles and ensuring correct behaviour of entities within, and upon, the platform. The ACC supports interoperability both within and across different platforms; therefore, it is viewed as a component of the MTS. The MTS provides a message routing service for agents on a particular platform. Such Agents must be reliable, orderly and adhere to the requirements specified in the FIPA MTS Specification [1].

The AMS, MTS and DF form what will be termed the Agent Platform (AP). These are mandatory, normative components of the model. For further information on the FIPA Agent Platform see the FIPA Agent Management Specification [2].

FIPA-OS handles the initial bootstrapping required to enable agents on multiple, potentially heterogeneous, FIPA Agent Platforms to interoperate via the use of a web server. The ACC included in

the current FIPA-OS distribution uses the web servers that it knows about (as configured in its agent profile) to obtain the required MTP (Message Transport Protocol) addresses for the initial platforms with which it requires interoperability. The interaction with the web servers is performed at start-up only and not when each message is routed to a remote Agent Platform. In the situation where the ACC has to be restarted it uses an *inform* message to notify all known ACCs of its new MTP addresses. Likewise, the ACC assumes that remote ACCs will notify it of their new MTP addresses should they be changed due to them being re-initialised. See [4] for more detail/examples.

Starting the Agent Platform using the Agent Loader

It is recommended that the Agent Loader be used to start all agents. This enables agents to be managed by a human user as required. Use of the Agent Loader is consistent with starting agents using the batch files or Unix scripts, but has the added advantage of a visual tool to control the lifecycle of the agents at run-time.

To start the Agent Loader (plus naming services and ACC, dependant upon your configuration) use:

```
C:\>StartFIPAOS
```

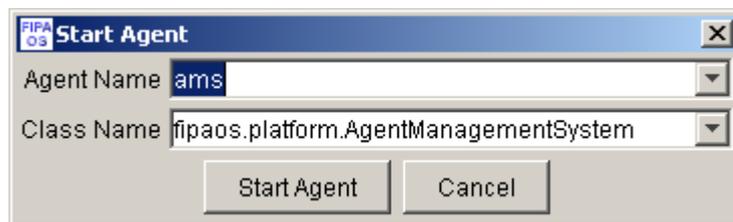
This will result in the Agents that have been specified in the loader.profile (see profile section) being started and a loader GUI being displayed. This will show the Agents that are running as a list on the left-hand side of the window, and a list of ready-to-start Agents on the right:



To start an Agent (or multiple Agents), select the Agent(s) from the list on the right, and press the Start button.

To shutdown an agent (or multiple Agents), select the Agent(s) from the list on the left and click the shutdown button.

To start a previously undefined Agent, select Start other, a dialog window will be displayed:



Enter or select the name of the Agent to start and the class to use for the Agent. Then select Start Agent to attempt to start the Agent.

Starting the ACC

If you've used the Wizard to configure FIPA-OS, the ACC will automatically start when you execute StartFIPAOS.

In the event that you are not using the Wizard or StartFIPAOS script, you should use either the StartACC.bat batch file (for Windows installations) or StartACC shell script (for UNIX installations) in order to run the ACC for the platform. The ACC requires the naming services for the MTPs it is using to be started (where appropriate) before it starts.

The ACC is only required when interoperating with other FIPA compliant Agent platforms.

Testing the Platform Agents (AMS, DF)

The FIPA-OS build includes an IOTestAgent, which enables ACL messages to be manually composed and sent, such that the functionality of the FIPA Platform Agents (described in the FIPA Reference Model) can be demonstrated. These tests are intended to illustrate the correct functionality of the FIPA Platform Agents, but assume that the user has a working knowledge of how the Platform Agents should be used and the associated FIPA communication protocols.

Testing the AMS

To test the functionality of the AMS, the AMS must first be configured and initiated as per the instructions in chapters 1 and 2. The IOTestAgent can be launched from the AgentLoader, and the default loader.profile file provided is configured to enable this.

When the IOTestAgent starts up a GUI will be created. Choose File\Load from the IOTestAgent GUI to load one of the selection of example ACL test scripts, which have been included in this distribution.

The table below lists the scripts, which can be used by the IOTestAgent for communication with the AMS.

File Name	Use of messages between IOTestAgent and AMS
amsregister.txt	Register the IOTestAgent with the AMS
amsmodify.txt	Modify the state of the IOTestAgent from active to suspended
amsderegister.txt	Deregister the IOTestAgent

Once a script has been loaded into the GUI, the Agent IDs should be modified to reflect the actual names of the agents for the specific configuration of the Agent Platform being used. Alternatively, the scripts can be edited using a text editor. The test scripts in this FIPA-OS distribution contain agent IDs with names of the form *'iotestagent@localhst'*. To send a test message to the AMS, the **send** on the IOTestAgent GUI should be selected. Upon receipt of the test message the AMS will act upon it and respond to the IOTestAgent, which will display the message received in the IOTestAgent GUI. Correct operation will result in a message of the form *inform(done)* being received by the IOTestAgent from the AMS. For a further explanation of the operation of the AMS see [2].

Testing the DF

To test the functionality of the DF, the DF must first be configured and initiated as per the instructions in chapters 1 and 2. The IOTestAgent must be configured and used as described in the Testing the AMS section. The table below lists the scripts, which can be used by the IOTestAgent for communication with the DF.

File Name	Use of messages between IOTestAgent and DF
dfregister.txt	Register the IOTestAgent with the default DF
dfmodify.txt	Modify state of the IOTestAgent with the default DF
dfsearch.txt	Search the default DF for the IOTestAgent
dfderegister.txt	De-register the IOTestAgent from the default DF

Once a script has been loaded into the GUI, the Agent IDs should be modified to reflect the actual names of the agents for the specific configuration of the Agent Platform being used. Alternatively, the scripts can be edited using a text editor. To send a test message to the DF, the **send** on the

IOTestAgent GUI should be selected. Upon receipt of the test message the DF will act upon it and respond to the IOTestAgent, which will display the message received in the IOTestAgent GUI.

DF Cross Registration GUI

When the DF starts, it creates a GUI to enable registration of the DF with other DF's and vice-versa. This enables federated searches to be propagated by and to the DF. The GUI is only accessible by "activating" the DF (this is achieved by double-clicking on the name of a DF agent in the Started Agents list of the AgentLoader).



Simply enter the name (NOT the entire AID) of the remote DF into the text-field and either:

- Click "Register this DF with Remote DF" to send a DF registration to the DF specified containing details of the DF the GUI is attached to (see GUI title-bar). If this is successful, the name of the remote DF will appear in the list area above the text-field.
- Click "Add Remote DF to directory" to add an appropriate DF description for the specified DF to the directory maintained by the DF the GUI is attached to (see GUI title-bar). This step is immediate – the result will be shown in the status area only (bottom of the window)

It is envisaged that the DF GUI will eventually acquire this functionality, however some trust-model needs to be developed so that not just any arbitrary Agent can request that the DF registers with another DF (given the constraint that only the DF has the ability to register itself with another DF).

Swing DF GUI Interface Agent

Introduction

Swing DF GUI interface agent³ is an agent that will give a user a way to interact with DFs. Currently, Swing DF GUI supports following:

- Subscribing to a *home* platform FIPA-OS DFs, so that the GUI will always show all the agents registered on those DFs.
- Querying remote FIPA-OS DFs – the information will be updated when Refresh is invoked.
- Viewing agents' DF agent descriptions.

Using the Swing DFGUI

Swing DF GUI can be started from the agent loader, and will start by subscribing to the platform's default DF, all other DFs in the platform will have to be subscribed to manually. Figure 1 shows the GUI that has subscribed to platform's default DF that has two agents registered with it.

³ Interface agent in this case means that despite agent properties like ACL communication, the agent doesn't offer services to any agents – only the user.

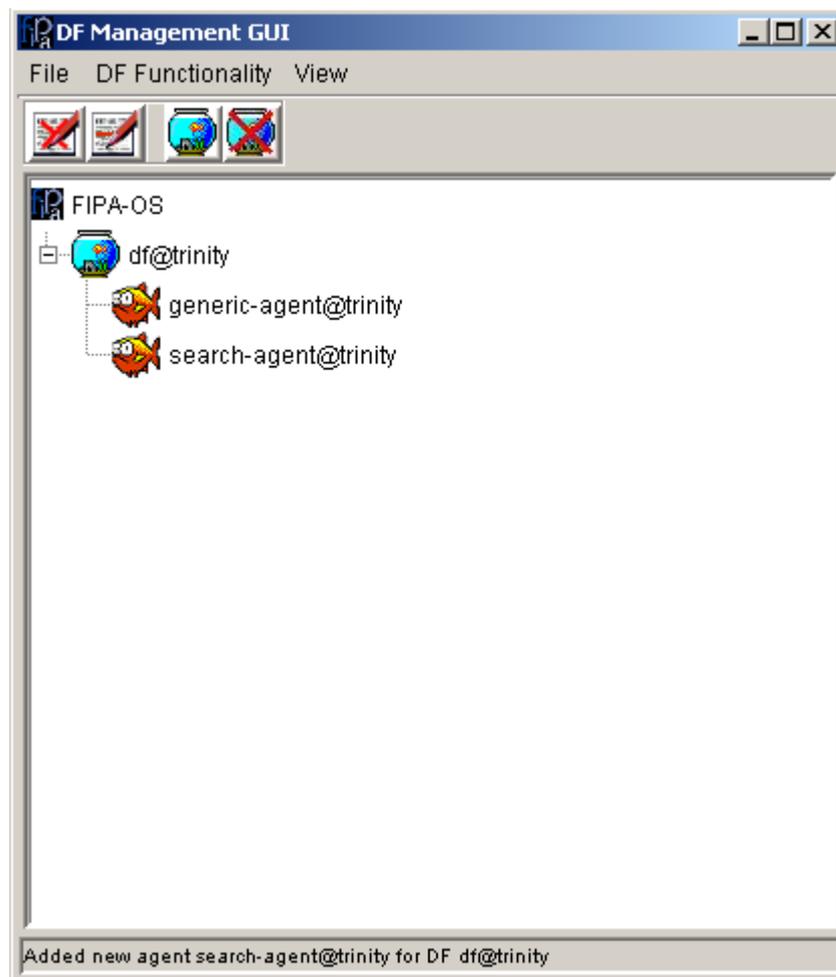


Figure 1: Swing DF GUI

When monitoring only home platform (the platform the GUI has been started from), there is no need to refresh the GUI, the subscribed DFs will send updates when agents register and deregister. To add other DFs to the list (it's possible to monitor remote platform DFs as well), select FIPA-OS from the list and either use context sensitive popup menu or menu File -> Add DF. Figure 2 shows the dialog where the user can specify the DF to be added, either by typing the Agent ID or load it from a file. Usually it's enough to specify just the name for the DF, but if dealing with inter-platform communication refer to the Inter-platform Communication Guide [4] – it may be necessary to specify a transport address to reach the DF.

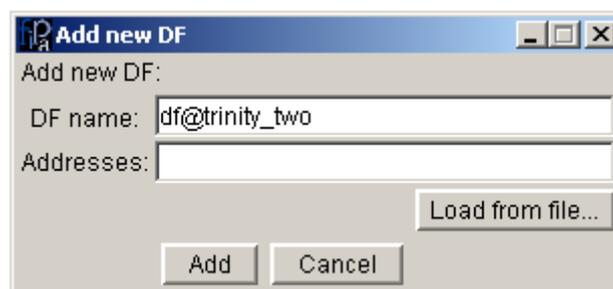


Figure 2: Adding a new DF

It's also possible to view agents' DF agent description, by selecting an agent from the list and then either using context sensitive popup menu or menu DF Functionality -> View. Figure 3 shows the dialog.



Figure 3: View DF agent description

User Constructed Agents

In addition to the mandatory components of the FIPA Reference Model, the FIPA-OS distribution includes an Agent Shell. This is an empty template for an agent. Multiple agents can be implemented, using the Agent Shell as a template; these agents can then communicate with each other using the FIPA-OS support facilities.

Agent Shell

All agents in the FIPA-OS distribution extend the *FIPAOSAgent* class (part of the *fipaos.agent* package). For an example of an agent, see *fipaos.tools.IOTestAgent*, which is a very basic agent that can send and receive ACL messages. The internal structure of an agent is not specified and is left to the individual agent designer. For further examples of how the agent shell, *FIPAOSAgent* can be extended to build agents the following object model is included to illustrate how the platform agents (AMS and DF) have been constructed by inheriting and extending the agent shell, *FIPAOSAgent*.

Using JESS Agent

JessAgent is a special Agent Shell that incorporates the reasoning abilities provided by JESS (Java Expert System Shell). To utilise this agent you must first download and install JESS from <http://herzberg.ca.sandia.gov/jess/>. The required distribution file (version 5.0) can be found by selecting the "downloads" option.

The best way to use the FIPA-OS JessAgent is to compile the JESS distribution and create a jar file as described here:

-
1. Download and extract JESS, then cd into the JESS root directory and at a command prompt type:

```
C:\ > javac -d <yourdirectory> jess/*.java jess/awt/*.java  
jess/factory/*.java jess/examples/**/*.java
```

Here *yourdirectory* is the directory into which you want to compile.

2. Then cd into the *yourdirectory* and type:

```
C:\ > jar -cvf jess.jar *.class
```

This should give you a jar file of all the classes.

3. Copy this jar file into the *imports* directory of FIPA-OS. Also, copy the file “*scriptlib.clp*” from the JESS distribution into your */imports/kb* directory. This is important because this file is a JESS knowledge base that the *jess.jar* will use when you instantiate the engine using the jar.

Once these steps have been completed it is possible to use the JessAgent within FIPA-OS. For information writing an agent with JESS capabilities, look at FIPA-OS Tutorial step five – JESS Agent (available separately). The JESS Agent needs to be extended and this tutorial shows you how.

Agent Development Tutorials

A growing number of tutorials to aid developers in constructing custom agents using the FIPA-OS toolkit can be found in a separate download, available via the FIPA-OS website (<http://fipa-os.sourceforge.net/>). Documentation describing these tutorials can be found within the **docs** directory of the tutorial archive.

Chapter 4

Updating FIPA-OS

Sharing Updates with FIPA-OS Users

Developers using FIPA-OS are encouraged to provide extensions, bug fixes and feedback to help improve the planned future releases. All such input should be contributed to the Open Source project via the SourceForge site at <http://sourceforge.net/projects/fipa-os/>. You are required to register as a developer to access some of the services at the SourceForge site. General issues and thoughts can be discussed via the FIPA-OS mailing list on fipa-os-developers@lists.sourceforge.net although you must register at <http://sourceforge.net/projects/fipa-os/> before you can send and receive messages. An archive of the messages sent to this list can be viewed from this site. Should you experience difficulties using this list, then please contact the FIPA-OS co-ordinators at fipaos@emorphia.com. Please consult the *FIPA_OS_Public_Licence.txt* file for further details on the requirements for using, extending and evolving FIPA-OS.

FIPA-OS Packages

FIPA-OS has been packaged to ensure that the structure is understandable, logical, and effective. Each package has a specific purpose. Correct use of the packages will result in more understandable code, and a more logical distribution of classes and source code. The intended use of each package is as follows:

fipaos.agent

The agent package is used exclusively for any mandatory components of an agent. For example, the Conversation Factory, Task Manager etc. Classes that form a large component should be grouped into a sub-package as seems most logical and appropriate (e.g. The Task Manager is placed in `fipaos.agent.task`). Classes in this package are required to run the platform.

fipaos.mts

This package is for the communications and transport classes of the platform. Classes exclusively of this type should be put into this package as appropriate (see existing package structure for guidance). Developers not working on the MTS/MTP classes should not need to use this package. Classes in this package are required to run the platform.

fipaos.ont

The ont package is for ontology specific classes. If a class belongs to an ontology component then it should be placed in a sub-package named after the ontology that it applies to. For example, the `fipa-agent-management` ontology classes are in `fipaos.ont.fipaman`. Classes in these sub-packages are required to run any components of the platform that use the specific ontology in the sub-package. Classes in the ont package (but not in a sub-package) and the `ont.fipaman` package are required to run the platform.

fipaos.ont.fipa

This package is for FIPA specific classes. Any classes that define or handle FIPA specific data or objects should be placed in this package. Classes in this package are required to run the platform.

fipaos.parser

The parser package is for classes that form language specific parsers. Parser components should be in sub-packages named by the language that they apply to. Classes in this package are required to run the platform.

fipaos.platform

The platform package is for platform specific classes that are not agent components. This includes platform agent classes. Classes in this package are required to run the platform.

fipaos.skill

The skill package is used for optional agent components that can be ‘plugged-in’ to the agent to provide it with a specific skill. For example, database access is a sub-package of the skill package. Classes in this package are not required to run the platform in general, although any platform agents that use specific skills will introduce a requirement for those specific skill classes.

fipaos.tool

The tools package is for platform and agent level tools that are not required for the use of the platform. This includes such tools as testing harnesses, information tools etc.

fipaos.util

The util package is used for agent classes that are not mandatory for use in an agent, but are provided as helper classes that are not application specific, and are not part of a skill component. Classes in the util package are required for correct operation of the platform. Examples of util classes are UTC time handlers and diagnostics output handlers.

How do I know which package to use?

If you have a class that does not appear to belong to any of these packages then follow this checklist:

1. Is the class specific to one of the package types (e.g. is it a mandatory agent component etc)? If yes, then package the class in the appropriate specific package.
2. Is the class related to any existing classes? If yes, then package it with those classes.
3. Is the class a non-mandatory helper class that does not form part of a larger package? If yes, then package it in `fipaos.util`.
4. Is the class part of a new optional component? If yes, then package it with the other classes that make up the new component in `fipaos.skill.component-name`.
5. Is the class *really* necessary? If yes, then make a judgement as to its best package location.

FIPA-OS Release Types

FIPA-OS version numbers are intended to reflect development of the platform. There are four types of release that can occur with FIPA-OS based on development. These are:

- Bug-fix release
- New features release
- Major release

The bug fix release is a properly built and packaged **public** release that only includes fixes and workarounds for known bugs. This will use the installer.

The new features release is a properly built and packaged **public** release that includes new features that have previously not been included in an *external* release of the platform. This type of release may or may not include bug fixes. This will use the installer.

The major release is a properly built and packaged **public** release that requires changes to the FIPA-OS API in a way that affects *external* agent developers. This may be a combination of new features and bug fixes, but these in themselves are not sufficient to warrant major release status. Examples of major releases would be a change in the core agent development API that requires large changes to existing agent code, or a complete redesign of the whole platform.

FIPA-OS Version Numbers

FIPA-OS has four components to its version number. These are:

- Major version (integer)
- Minor version (integer)
- Bug-fix version (integer)

These are represented in the following format: **major.minor.bug-fix**

For example, 1.0.3 denoted version 1 of the platform with no minor functionality updates and three bug fix releases.

Along with the version number of the platform, every release or build also has a build number. This denotes which physical build of the platform is being used. This number is updated every time the build process is run, even if that build is never used. The build number is only reset to zero when the major version number of the platform changes. This build number is not used as part of the FIPA-OS version number, but it should be included in the platform documentation. This build number can then be used to easily distinguish different versions of the platform in use by third parties.

Compiling FIPA-OS Source Code

We generally recommend compiling FIPA-OS code and Agents within an IDE, which will recompile parts of FIPA-OS when modified automatically. However, an ANT script [5] is included (**build.xml**) which will attempt to compile all of the FIPA-OS source code using the JAR's/ZIP's in the **imports** directory into the **classes** directory.

Configuring ANT to enable this is beyond the scope of this document.

Obtaining the Latest FIPA-OS Source Code

The FIPA-OS source code is stored using a publicly visible CVS server located at the SourceForge site (as of January 2001), enabling FIPA-OS developers to monitor and obtain the latest updates to FIPA-OS as soon as they have been made. Details of how to access our CVS repository can be found at <http://fipa-os.sourceforge.net/cvs.htm>.

Due to the nature of the code checked into the repository, there is absolutely no guarantee that any particular source file will work as expected, or even compile (work-in-progress code will most likely be checked into CVS as well as fully-tested code), or that the most up to date versions of any two source files will work together as expected. However, using the tagging capability of CVS, particular builds of FIPA-OS can be checked out that match the content of a particular distribution (see <http://fipa-os.sourceforge.net/cvs.htm> for details of the tags used), or particular groups of updated files that work together and form a patch for a particular build will be available (as announced on the FIPA-OS mailing list).

Chapter 5

Current Issues and Future Plans

This section describes the current issues and bugs associated with the build of FIPA-OS. Further, the section includes plans for addressing these known issues and plans for additional features expected to be included in future releases. All issues are numbered uniquely relating the issue to a build version. Each number relates to the Bug ID that was allocated to it on the sourceforge site - http://sourceforge.net/bugs/?group_id=3819.

Current Issues

Below is a list of known issues with FIPA-OS. The most up to date version of the list can be found at http://sourceforge.net/bugs/?group_id=3819&set=open&order=date. A brief description is provided, for more details follow the link provided above.

Bug ID	Summary
112101	ACC not immediately operational after start-up
112099	The ACC does not authenticate the sending agent
108809	Theres a limit of 20k(ish) when sending a message
107167	ACL Messages received out of order / protocol exception
130141	ConversationManager and performative not set
130480	CM doesn't deal with "reply-to" field at all!
130584	CM expects a single performative as start of Conversation
130149	DF Register message requires double brackets
130060	DFAgentDescription inconsistancies
111274	Not enough garbage collection in Agents
113224	Modify profiles to reference FIPA-NET at IC
114587	Agents don't get garbage collected

Future Plans

The following improvements and bug fixes are planned for the future.

FIPA2000

- Future releases of FIPA-OS will aim to incorporate an increasing number of FIPA 2000 specifications.

Profiles

- Improve the API provided to access profile information and allow Agents to re-write their profiles.

Agent Naming convention

- The ACL parser will be updated to allow agent names containing the underscore character to pass through the ACL parsing process.

MTS

- The parser factory will be updated so that alternative ACL encoding means will be supported, such as XML.
- Provide an optional MTP utilising the Java Messaging Service.

AgentLoader

- Future versions of FIPA-OS will hope to include an improved AgentLoader GUI, which is more user friendly and provide control over more than just the Agents in one JVM.
- The FIPAOSAgent constructor will be standardised so that the Agent Loader can be used to start all agents consistently.
- Add support to enable the list of agents loaded using the GUI to be reflected in the Agent Loader profile.

ACC

- The ACC will be modified so that it authenticates the sending agent with the AMS of its home platform before forwarding a message.

DFGUI

- Update the DFGUI & DF to enable modification of the contents of the DF
- Allow the DFGUI to instruct the DF of a remote DF to register with

FAQ

- It is intended that an FAQ will be made available from the FIPA-OS SorceForge covering installation/runtime problems associated with FIPA-OS in the near future

Bibliography

- [1] FIPA *Agent Message Transport Service Specification* (XC00067C). August 2000
<http://www.fipa.org/specs/fipa00067/XC00067C.doc>
- [2] FIPA *Agent Management Specification* (XC00023F). July 2000
<http://www.fipa.org/specs/fipa00023/XC00023F.doc>
- [3] S. Willmott, B. Faltings, M. Calisti, S. Macho-Gonzalez, O. Belahdar and M. Torrens *Constraint Choice Language (CCL), Language Specification v2.01*.
<http://liawww.epfl.ch/~willmott/CCL/>
- [4] *FIPA-OS Inter-platform Communication Configuration Guide* October 2000
http://fipa-os.sourceforge.net/docs/Interplatform_Configuration_Guide.pdf
- [5] ANT Website
<http://jakarta.apache.org/ant/>