

OpenFusion® CORBA Services

Version 4.1

Product Guide



OpenFusion®

CORBA Services

PRODUCT GUIDE



Part Number: OFCOR-PRDG-41

Doc Issue 18, 13 July 2004

Notices

Copyright Notice

© 2004 PrismTech Limited. All rights reserved.

This document may be reproduced in whole but not in part.

The information contained in this document is subject to change without notice and is made available in good faith without liability on the part of PrismTech Limited or PrismTech Corporation.

All trademarks acknowledged.

Preface

About the Product Guide

The *Product Guide* is included with OpenFusion CORBA Services' *Documentation Set*. This guide is the starting point for anyone using or developing applications with any of the OpenFusion services and interfaces.

The *Product Guide* contains:

- general information about the OpenFusion product suite and its range of services and interfaces
- the list of documents in the OpenFusion *Documentation Set* and how to use it
- the instructions for installing and running OpenFusion, as well as information on running the examples and application with the OpenFusion CORBA Services
- details of where additional information can be found, such as the OpenFusion FAQs, Knowledge Base, bug reports, etc.

Intended Audience

The *Product Guide* is intended to be used by anyone who wishes to use the *OpenFusion CORBA Services*.

Conventions

The conventions listed below are used to guide and assist the reader in understanding the *Product Guide*.



Item of special significance or where caution needs to be taken.



Item contains helpful hint or special information.



Information applies to Windows (e.g. NT, 2000, XP) only.



Information applies to Unix based systems (e.g. Solaris) only.

Hypertext links to WWW and other internet services are shown as *[blue italic underlined](#)*.

On-Line (PDF) versions of this document: Items shown as cross references to other parts of the document, e.g., *Contacts* on page vi, behave as hypertext links: readers can jump to that section of the document by clicking on the cross reference.

% *Commands or input which the user enters on the command line of their computer terminal*

Courier fonts indicate programming code and file names.

Extended code fragments are shown in shaded, full width boxes (to allow for standard 80 column wide text), as shown below:

```
NameComponent newName[] = new NameComponent[1];  
  
// set id field to "example" and kind field to an empty string  
newName[0] = new NameComponent ("example", "");  
  
rootContext.bind (newName, demoObject);
```

Italics and ***Italic Bold*** are used to indicate new terms, or emphasise an item.

Arial Bold is used to indicate user related actions, e.g. **File | Save** from a menu.

Step 1: Indicates that this item is a step or stage of completing a task by a user.

Contacts

PrismTech can be contacted at the following contact points for information and technical support. Users of the on-line version of this manual can *click* the e-mail addresses below to launch their e-mail client or Web browser to send e-mail direct to PrismTech. Letter and fax communications for technical support should be sent to ***Technical Support*** at the contact points listed below.

Corporate Headquarters

PrismTech Corporation
6 Lincoln Knoll Lane
Suite 100
Burlington, MA
01803
USA

Tel: +1 781 270 1177
Fax: +1 781 238 1700

Web: <http://www.prismtechnologies.com>
General Enquiries: info@prismtechnologies.com
Support Enquiries: <http://www.prismtechnologies.com/Contacts>

European Head Office

PrismTech Limited
PrismTech House
5th Avenue Business Park
Gateshead
NE11 0NG
UK

Tel: +44 (0)191 497 9900
Fax: +44 (0)191 497 9901



Contents

Table of Contents

	Notices	iii
	Preface	v
	About the Product Guide	v
	Contacts	vi
	About OpenFusion	1
1	Why OpenFusion?	3
1.1	Why Use It?	3
	Enterprise Application Integration	4
	OpenFusion Provides the Solution	4
1.2	What is OpenFusion?	5
	Special Features	5
	Available Services	5
	CORBA Services	5
2	Product Details	7
2.1	ORBS, Platforms, and Portability	7
2.2	Supporting Tools	7
2.3	Service Descriptions.	7
	CORBA Services	7
	COS Notification Service	7
	COS Naming Service	8
	COS Trading Service	8
	COS Time Service	8
	Telecom Event Domain Service	9
	OpenFusion CORBA Service Extensions	9
	Extended Notification Service	9
	Extended Naming Service	9
	OpenFusion Security	10

Using OpenFusion	11
3 Documentation	13
3.1 The Documentation Set	13
4 Installing OpenFusion	17
4.1 Prerequisites	17
Operating Systems	17
ORB and Java Runtime Environment	18
System Variables	18
CLASSPATH and ORBs	19
C++ Client Development	19
4.2 Installation Procedure	19
General	19
Preparation	20
Installation	20
Installing Using GUI Mode	20
Installing Using Command Line Mode	21
Install the Licence File	21
4.3 Running and Configuration	22
Running the ORB Daemons	22
JacORB 1.4	22
JacORB 2.1	23
VisiBroker	23
Starting the Administration Manager	23
Running Servers	23
Using the Administration Manager	24
Using the Server Command	24
4.4 Uninstalling	25
4.5 Running as a Windows Service	26
OpenFusion	26
Installing	26
Stopping	27
Troubleshooting	27
JacORB	27

5	Running Applications	29
5.1	General	29
	The run Script and PATH	29
	CLASSPATH	30
5.2	Configuration and Running	30
	Basic Steps	30
	OpenFusion Examples Specifics	32
	CLASSPATH and Application Run Directory	32
	PATH	32
	User-compiled Application Specifics	32
	CLASSPATH and Application Run Directory	33
5.3	Examples of Running Applications	33
6	Information Sources	35
6.1	PrismTech Product Information	35
	FAQs	35
	Knowledge Base	35
	Additional Technical Information	35
6.2	PrismTech Product Support	36

The background of the slide is a close-up, low-angle photograph of a computer keyboard. The keys are white and slightly raised. A white grid pattern is overlaid on the entire image, creating a sense of depth and perspective. The lighting is soft, and the overall color palette is muted, with a slight purple tint on the right side.

About OpenFusion

1 Why OpenFusion?

The OpenFusion CORBA Services are PrismTech's implementation of the main Common Object Services (COS) defined by the Object Management Group. The OpenFusion CORBA Services product also includes a number of additional components which enable application developers to easily integrate systems developed on both the CORBA (Common Object Request Broker Architecture) and J2EE (Java 2 Enterprise Edition™) platforms. These additional components are based on standards defined by Sun Microsystems™.

1.1 Why Use It?

Organisations face a host of challenges today to compete successfully in both established and new markets. Frequent changes to business models take place and are caused by, for example:

- the rapid evolution of internet-based commerce, telecommunications, and information technologies
- the demand for integrated supply-chains and customer relationship management
- changes to organizational and systems structure resulting from mergers, acquisitions, and divestments
- the requirements for improved product time-to-market and return on investment

End-to-end integration and continual evolution of your business and the systems that support it are critical for competing today. A range of different requirements from customers, partners, and employees need to be flexibly accommodated and efficiently processed using both front- and back-office functions.

In these fast moving environments, a competitive strategy is built on quick deployment, evolution and productive use of reliable, flexible, integrated business applications, and information. How do you build and maintain an integrated system infrastructure to support your business? You have to deal with existing applications, data, different legacy technologies, and the need to quickly integrate new functions to stay competitive. Loosely-coupled, integrated applications and data, and the use of standards-based components, are the keys.

The OpenFusion range of CORBA products are accepted as the best fully compliant implementation of the CORBA standards available. OpenFusion CORBA products are used and deployed across technology domains and business sectors by some of the largest companies in the world. No matter what scenario or problem you're trying to address, the comprehensive OpenFusion CORBA product range will meet

your requirements. OpenFusion CORBA products support both the C++ and Java development communities and offer unrivalled interoperability with other leading distributed architectures and technologies such as J2EE and XML.

Enterprise Application Integration

Virtually all organisations face the problem of integrating new and existing systems at some point in time. For cost reasons, it may not be acceptable to simply replace an existing application or business process. Many companies would prefer the option of retaining their existing systems while integrating them with new applications in order to enhance and extend their functionality.

CORBA is the dominant technology for developing high performance, secure, distributed applications and also for integrating applications written in different programming languages. Equally, J2EE has emerged as the leading technology for organisations with rapidly changing business requirements who need to make their applications, products and services available over the Internet.

Many organisations faced with implementing Enterprise Application Integration (EAI) solutions will, inevitably, need to resolve the issue of integrating systems and business processes built on both the CORBA and J2EE architectures.

An open, standards-based infrastructure is a key success enabler. Open standards bring vendor independence and a commonly-agreed baseline.

OpenFusion Provides the Solution

The OpenFusion CORBA Services product is the most comprehensive suite of enterprise strength implementations of the main OMG CORBA Services available in the market today. OpenFusion CORBA Services provide the premier implementations of the CORBA Notification, Naming, Trading, and Time Services available anywhere.

Additionally, the OpenFusion CORBA Services product provides CORBA users with unrivalled connectivity to J2EE, using solutions built on standards. Seamless CORBA/J2EE integration is also provided by the OpenFusion Naming Service. The Naming Service is a unified COS Naming/JNDI implementation and can be used to store and resolve both CORBA and JAVA object references held in the same name space.

OpenFusion CORBA Services is fully standards compliant and written entirely in Java. It is both platform and vendor independent; it runs on a comprehensive range of Unix and Windows platforms, and supports all leading ORB platforms.

1.2 What is OpenFusion?

OpenFusion is a set of integrated CORBA and J2EE services. It also has additional features which make development with, and management of, these services easier and more efficient.

Special Features

- **CORBA/J2EE Integration** - OpenFusion provides seamless integration between CORBA and J2EE through object location with the OpenFusion Naming Service.
- **Security** - OpenFusion provides its own security service that can be used to provide secure access to both the CORBA and J2EE Services.
- **Flexibility through plug-ins** - All key resources required by OpenFusion services are accessed through generic *plug-ins*. These enable OpenFusion to flexibly accommodate, databases (such as Oracle, Sybase, Informix, and SQL Server) and other products. Custom plug-ins can be created with OpenFusion's open APIs to suit clients' particular needs.
- **Graphical user interfaces** - The OpenFusion services can be easily configured, tested, and managed using the graphical user interfaces; at run-time the GUIs enable dynamic observation of the services' behaviour.
- **Instrumentation** - A set of controls enabling remote, dynamic, and efficient monitoring, to suit client needs, with no performance overhead.

Available Services

CORBA Services

The Object Management Group's (OMG) CORBA based Common Object Services (COS) are system level services in the OMG model for distributed objects, otherwise known as the Object Management Architecture (OMA).¹

OpenFusion currently supports two categories of the CORBA services:

- Common Object Services which are services common to the Object Management Architecture
- services from the Telecom OMA vertical domain

1. For more information about the OMG architecture and the services, please consult the relevant OMG material.

The services provided by OpenFusion are strictly CORBA-compliant at both the interface and implementation level. This means that the services are portable across a number of different ORB implementations and operating systems.

All of these services conform to the relevant OMG CORBA standards, but some have been extended to provide additional functionality. These extensions are proprietary to PrismTech and are implemented in separate modules. The extensions are thus purely optional and do not affect the functionality of the standard OMG-defined services or interfaces.

Where the OpenFusion product contains an implementation of both COS and extended Services (e.g. the OpenFusion Notification Service), a developer may use whichever service meets their requirements. When a developer wishes to be strictly OMG compliant, only the COS service implementation should be used. Nonetheless, a developer is free to use the extended implementation when additional functionality is required.

OpenFusion includes the following range of CORBA services:

OpenFusion CORBA Services

Service	Standards Supported
Notification Service*	COS Notification Service COS Event Type Repository
Event Domain Service	Telecom Event Domain Service
Naming Service*	COS Naming Service COS Interoperable Naming Service
Trading Service	COS Trading Service
Time Service	COS Time Service COS Timer Event Service
* Including PrismTech extensions	

2 Product Details

2.1 ORBS, Platforms, and Portability

OpenFusion has been designed and engineered to be a portable, ORB- and platform-independent product providing a unique OMG compliant Naming Service which can be used as a location service by both CORBA applications and J2EE/Java based applications

OpenFusion's strategy for ORB vendor independence includes an *ORB Abstraction Layer*. The OpenFusion ORB Abstraction Layer isolates, or hides, the detailed differences of the various ORB vendors' particular implementations. This valuable feature removes the need for application developers to alter their code to suit different ORBs - or different versions of the same ORB - when using the OpenFusion Services.

2.2 Supporting Tools

OpenFusion includes features to help developers and administrators develop, use, and manage applications in conjunction with OpenFusion. These features include:

- **OpenFusion Graphical Tools** - A set of graphical user interfaces (GUI) which are used to configure, test, and manage the OpenFusion CORBA Services.
- **Instrumentation** - Features which allow remote, dynamic, and efficient management using JMX™ and other methods.

2.3 Service Descriptions

CORBA Services

COS Notification Service

The COS Notification Service is a powerful extension to the standard Event Service. The Notification Service adds these features to the Event Service:

- *Event filtering* - The Notification Service includes a structured event type and a powerful constraint language. The constraint language can be used to filter both structured events and events of type any.
- *Wide selection of Quality of Service properties* - This enables the Notification Service to be configured and fine-tuned to support many different application requirements. The QoS properties include coverage for reliability, queue management and event management.

- *Sharing of subscription and offered event types* - Suppliers and consumers can obtain run-time information about events that are currently required or offered by an event channel. This can be used to reduce both the number of events and the overhead of event filtering.
- *Event Type Repository* - The Notification Service provides an Event Type Repository that contains information about the event types used in the distributed system.

COS Naming Service

The COS Naming Service provides the ability to bind a name to an object relative to a *naming context*. (A naming context is an object that contains a set of name bindings in which each name is unique. The COS Naming Service is the easiest and most straight-forward method of finding and using named objects.

Graphs of naming contexts can be supported in a distributed, federated fashion. The scalable design allows the distributed, heterogeneous implementation of names and naming contexts.

The COS Naming Service also supports the Interoperable Naming Service (INS), enhancing compatibility between individual ORB vendors' products and applications. Load Balancing is also available as an optional feature.

COS Trading Service

The COS Trading Service provides access to objects based on their capabilities rather than name or interface type. The service makes it easier for client objects to offer and discover instances of services of particular types.

A *trader* is an object which supports the trading object service within a distributed environment. Objects can advertise their capabilities to other objects and find objects which can satisfy their needs by using the trader.

The COS Trading Service is scalable, as well as able to accommodate a diverse range of needs. Traders can also be linked to each other, enabling the trading object service to be distributed.

COS Time Service

The COS Time Service enables the current time to be obtained, with an associated degree of accuracy, and provides facilities for manipulating representations of time and time intervals. The service can use either the system clock of the machine it is running on or the Network Time Protocol (NTP) as a time source. The service can be configured to return either secure or insecure time according to the security of the host machine's system clock.

The COS Timer Event Service provides operations for managing time-triggered event handlers and the events they handle. The COS Timer Event Service sends events to event channels of the *push* type as defined by the Event Service or the Notification Service.

Telecom Event Domain Service

The Management of Event Domains specification was developed by the Telecom task force within the OMG. It describes standard interfaces for managing and federating a set of Notification Service event channel objects, or a set of Log Service log objects.

OpenFusion CORBA Service Extensions

The OpenFusion Extended Object Services provide additional functionality and features to some of the standard OMG services. These services have been specified where:

- additional Quality of Service properties have been provided
- there is no existing equivalent OMG service
- the existing OMG service specification is incomplete
- a higher level of abstraction is required

A brief description of each of the OpenFusion extended services is given below. The OpenFusion extended services are layered upon the relevant OMG- defined services and do not affect the use of the standard services.

Extended Notification Service

The CORBA Notification Service has the concept of extensible qualities of service. The additional qualities of service supported by the PrismTech implementation are defined in this extension.

Extended Naming Service

The Extended Naming Service is an extension of the COS Naming Service. It supports a Load Balancing interface that allows many objects to be bound to the same name. The service has been layered on top of the Java Naming and Directory Interface (JNDI) thereby allowing it to store its persistent data in a database, file, LDAP or (temporarily) in memory. The Naming Service may optionally browse a JNDI hierarchy, and standard JNDI clients can access the persisted Naming Service hierarchy.

OpenFusion Security

OpenFusion provides a security service which enables users to apply access control to both the OpenFusion CORBA and J2EE Services. Access control is based upon a client's identify being verified by plug-in authentication modules.

The security service can be used to control access to individual object types, object instances or even specific methods on an object type or instance. A suite of GUI based tools is also provided that can be used to manage and administer the security profiles for each service.

Using OpenFusion

3 Documentation

3.1 The Documentation Set

OpenFusion CORBA Services includes a number of documents which provide detailed information about each service, the OpenFusion Graphical Tools, configuration and administration.

The following table, *The Documentation Set*, describes all of the documentation and manuals included with the OpenFusion CORBA Services product.

The Documentation Set

Document	Description and Use
<i>Release Notes</i>	<p>Describes the latest OpenFusion updates, bug fixes, and any last-minute information.</p> <p>Should be read by anyone using or installing OpenFusion.</p>
<i>Product Guide</i>	<p>The starting point for understanding, installing and using the OpenFusion CORBA Services. The <i>Product Guide</i> contains:</p> <ul style="list-style-type: none">• general information about the product and its components• installation instructions• basic information for running applications and the OpenFusion examples• links to information sources, including OpenFusion FAQs, PrismTech Knowledge Base and related information. <p>For managers, administrators, and developers who need to understand, install, or use the product.</p>

The Documentation Set (Continued)

Document	Description and Use
<i>System Guide</i>	<p>Includes information about:</p> <ul style="list-style-type: none"> • Graphical Tools for configuring, testing, and managing the OpenFusion Services • details of common service properties • instrumentation features <p>For administrators, developers, and testers who need to configure, develop, test, and manage the OpenFusion Services.</p> <p>The <i>System Guide</i> is intended to be used together with the individual Service Guides.</p>
<i>Service Guides</i> <ul style="list-style-type: none"> • <i>Naming Service Guide</i> • <i>Notification Service Guide</i> • <i>Time Service Guide</i> • <i>Trading Service Guide</i> 	<p>Contains detailed conceptual and technical information about each service or interface.</p> <p>Contains details about service configuration properties and graphical tools specific to the service.</p> <p>For system and application designers and developers to aid understanding, using, and implementing the OpenFusion services</p>
<i>OpenFusion IDL API</i>	<p>The complete IDL API for the OpenFusion Services.</p> <p>This is a detailed reference for use by developers to help them to understand the details of the OpenFusion Service interfaces.</p>

The Documentation Set (Continued)

Document	Description and Use
<i>OpenFusion Usage Examples</i>	<p>Working examples, complete with source code, demonstrating how applications incorporating the OpenFusion Services can be written and used.</p> <p>The examples and related instructions are accessed through HTML pages included with the OpenFusion distribution.</p>
<i>White Papers and Data Sheets</i>	<p>Technical papers providing information about OpenFusion and selected services.</p> <p>These technical papers are in Adobe Acrobat PDF™ format and can be obtained from the PrismTech web site at:</p> <p>http://www.prismtechnologies.com</p>

4 Installing OpenFusion

The procedures for installing the OpenFusion CORBA Services for all platforms are described below. Please follow these procedures carefully.

The installation files are provided on the OpenFusion CORBA Services CD or can be downloaded from the PrismTech Web site (www.prismtechnologies.com).

Note

- The directory paths shown in the installation instructions use the UNIX forward-slash (/) separator convention; Windows™ users should substitute these separators with the standard DOS back-slash (\) separator.
- `<install_dir>` is used as a convention in this section to refer to the directory where OpenFusion is (or will be) installed
- Items which are unique to UNIX or Windows are shown using the *UNIX Only* or *Windows Only* icons, respectively. For example:

WIN

```
> SET CLASSPATH=.;%CLASSPATH%;
```

UNIX

```
% CLASSPATH=.:$CLASSPATH; export CLASSPATH
```

4.1 Prerequisites

OpenFusion depends on underlying services and technologies. If these services and technologies are not properly installed and configured, then the OpenFusion CORBA Services cannot perform as intended. Accordingly, please check that your system meets each of the prerequisites described below before installing OpenFusion CORBA Services.

i

The latest version details for the supported operating systems, platforms, ORBs, etc. are listed on the *Supported Platforms* HTML page. This page is in the OpenFusion product CD and installation.¹ Please refer to *Supported Platforms* and other *Release Notes* pages for the latest information affecting this distribution.

Operating Systems

The following operating systems are supported by OpenFusion CORBA Services.

- **UNIX** - including HP-UX, Redhat Linux, and Solaris

1. The *Supported Platforms* HTML page is accessible through the *index.html* page found in the root directory of the OpenFusion product CD and (after installation) the root directory where OpenFusion is installed.

- **Windows** - including NT4 (with Service Pack 3 or later), 2000, and XP

ORB and Java Runtime Environment

The OpenFusion CORBA Services requires the following correctly installed software combination prior to installing OpenFusion:

- JacORB v1.4 or v2.1 and the latest release of the Java Development Kit (JDK) 1.4
- i*
- If OpenFusion JacORB is included as part of your OpenFusion CORBA Services distribution, then the OpenFusion installation program can install OpenFusion JacORB at the same time as the CORBA Services: refer to Section 4.2, *Installation Procedure*.
 - A default JacORB configuration properties file, *jacorb.properties*, is included in the *classes* subdirectory of the OpenFusion installation. No additional orb configuration is required when OpenFusion CORBA Services and JacORB is installed using this default configuration.

When installed as a shared distribution, each user should either copy the *jacorb.properties* file to their home directory or add it to the CLASSPATH, then edit the user-specific properties listed in the file.

- Visibroker v5.2, v6.0 with JDK v1.4

In addition, the CLASSPATH environment variable will need to include the location of the `class` and `jar` files associated with the selected ORB: see *CLASSPATH and ORBs* on page 19 under *System Variables*.

Please refer to the ORB documentation for specific ORB configuration requirements.

System Variables

The PATH and CLASSPATH environment variables must be set as described below.

The PATH must include the directory where the Java interpreter is located (i.e. `<JDK_install_path>/bin`, where `<JDK_install_path>` is the directory where the JDK is installed).

The CLASSPATH is used by the Java Virtual Machine (JVM) to locate required Java class files, including those located in the current directory. Accordingly, ensure that the CLASSPATH includes the current directory reference, indicated by the full-stop or period character (`"."`).

UNIX

Set these variables for the environment from which OpenFusion will be run. For example:

UNIX

```
% PATH=/usr/local/jdk1.4.1/bin:$PATH; export PATH
% CLASSPATH=.:$CLASSPATH; export CLASSPATH
```

Windows

Set these variables using Window's **Control Panel | System | Environment** dialogue (recommended) or from the command line. For example:

WIN

```
> SET PATH=c:\jdk1.4.1\bin;%PATH%;
> SET CLASSPATH=.;%CLASSPATH%;
```

It is also recommended that the `<install_dir>/bin` (where `<install_dir>` is the OpenFusion installation directory) is added to the `PATH` environment variable.

CLASSPATH and ORBs

The `CLASSPATH` must also include the location of the `class` and `jar` files used by the ORB which OpenFusion will be used with. The `CLASSPATH` must be set with this location *before* OpenFusion is run. The `jar` files for the ORBs supported by OpenFusion include:

- **JacORB 1.4, 2.1** - `jacorb.jar`
- **VisiBroker 5.2** - `vbjorb.jar`, `lm.jar` and `vbsec.jar`
- **VisiBroker 6.0** - `vbjorb.jar`, `lm.jar`, `vbsec.jar`, `sanctuary.jar` and `sanct2.jar`

C++ Client Development

A C++ ORB and compiler must be installed if C++ client development is to be undertaken.

4.2 Installation Procedure

General

All installed OpenFusion files are placed in the OpenFusion installation directory specified during installation - no files are stored in any of the UNIX or Windows system directories.



The OpenFusion CORBA Services and OpenFusion JacORB can optionally be run as a Windows Service, which automatically starts OpenFusion or JacORB when Windows starts: follow the instructions in Section 4.5, *Running as a Windows Service*, **after** OpenFusion has been installed.

OpenFusion is installed using a Java-based *Setup* program. This program can be run using a Graphical User Interface (GUI Mode) or from the command line (Command Line Mode) which enables the installation to be run from a script.

Further, there is a distinct *Setup* program for each ORB that OpenFusion supports. It is therefore necessary to run the particular *Setup* program located in the *distributions* sub-directory having the same name as the ORB which will be used.

Preparation

It is recommended that any existing OpenFusion V4 installation is removed before installing the current version (see *Uninstalling* on page 25). Please note the following warning.



Uninstalling OpenFusion removes all OpenFusion files, including the executables, licence, configuration, and data files located in the OpenFusion sub-directories. If these files are required, then they should be backed-up prior to uninstalling.

Installation

As mention above, OpenFusion can be installed using an interactive graphical user interface (GUI Mode) or using commands entered on the command line (Command Line Mode). Using GUI Mode is generally the more popular method, however the Command Line Mode is useful when automating the installation with a script.

Installing Using GUI Mode

Step 1: If you are installing from the OpenFusion CORBA Services CD, then change to the *distributions/* sub-directory on the CD with the same name and version number as the ORB you are using. For example, if you are using the Visibroker 5.2 ORB, then change to the directory called */distributions/VisiBroker-52*.

If you are downloading the installation files from the PrismTech Web site, then follow the instructions on the Web site to select the appropriate *Setup.class* file.

Step 2: Run *java Setup* (**without** any options) from the command line, as follows:

- if installing from the installation CD, run from the directory selected above, or

- if you downloaded the *Setup.class* file from the Web, then change to the directory containing *Setup.class*

```
% java Setup
```

This will display the OpenFusion installation's graphical user interface.

- Step 3:** Follow the instructions displayed in the GUI, selecting the services and components you want to install.

Installing Using Command Line Mode

- Step 1:** Follow the instructions in *Step 1* under *Installing Using GUI Mode* above.

- Step 2:** Run *java Setup* from the directory selected above **with** the options shown below. *Setup* can be run directly on the command line or from a script:

```
% java Setup <-list | [<install_dir> [components]]>
```

where

- *-list* will list all available services and components, without performing the installation
- *<install_dir>* is the directory where OpenFusion is to be installed
- *[components]* is the list of components and services to be installed; if no components are specified, then all components will be installed

Example 1 List all available services and components

```
% java Setup -list
```

Example 2 Install all services and components to /prismtech/myOF

```
% java Setup /prismtech/myOF
```

Example 3 Install the Notification and Naming Services to /prismtech/myOF

```
% java Setup /prismtech/myOF Notification Naming
```

Install the Licence File

A valid OpenFusion license file must be placed into the *<install_dir>/etc* directory after OpenFusion has been installed. Please note that the OpenFusion services and components will not run without a valid licence file.

Licence files are provided by PrismTech for services or products which have been purchased. Contact PrismTech for purchasing details (see *Contacts* on page vi).

Evaluation licences are provided when an OpenFusion service or product is downloaded from the PrismTech Web site (at www.prismtechnologies.com.)

4.3 Running and Configuration

Running OpenFusion is a simple two-step process:

- Step 1:** Ensure that an ORB daemon has been started and is running properly. Running an ORB daemon is described under *Running the ORB Daemons* below.
- Step 2:** Start OpenFusion using either the (GUI-based) *Administration Manager* or the *server* command (from the command line), described under *Starting the Administration Manager* on page 23 and *Running Servers* on page 23, respectively.

It is likely that configuration tasks will need to be performed for OpenFusion, itself, or for the installed services. Configuration can be conveniently performed using the *Administration Manager* (see *Starting the Administration Manager* on page 23).

Details on configuring the services - and the effects of the various configuration properties - are given in the *System Guide* and the individual OpenFusion service guides.

Running the ORB Daemons

The commands for starting the supported ORB daemons are described below. Note that an ORB daemon may not be necessary when servers are run on fixed ports. The daemons can be started from the command line in one of the following ways, depending on the ORB you have installed.

JacORB 1.4

The JacORB implementation repository daemon is run using the *imr* utility (*imr* and the other OpenFusion JacORB utilities are located in the *bin* directory of the OpenFusion installation)¹:

```
% imr
```

1. The *bin* directory must be in the *PATH* or the user will need to change to the *bin* directory before running the utility.

JacORB 2.1

The JacORB implementation repository daemon is run using the *jac_imr* utility (*jac_imr* and the other OpenFusion JacORB utilities are located in the *bin* directory of the OpenFusion installation)¹:

```
% jac_imr
```

VisiBroker

The VisiBroker daemon is run using the *osagent* command:

```
% osagent
```

Starting the Administration Manager

The Administration Manager is a powerful, GUI-based management tool which can be used for running, configuring, managing, and testing the services.

Before running the Administration Manager you should:

- ensure that the `CLASSPATH` environment variable includes the appropriate ORB *jar* files (see *CLASSPATH and ORBs* on page 19)
- ensure that the `UNIX DISPLAY` environment variable is set to the *X Window* display you will be using, for example, *host_name:0*

UNIX

The Administration Manager can be started from the command line using the OpenFusion *manager* command located in the *bin* directory of the OpenFusion installation:

```
% <install_dir>/bin/manager
```

WIN

Windows users can also start the Administration Manager by selecting **Start | Programs | OpenFusion | Administration Manager**.

Running Servers

The OpenFusion CORBA Services can be run as servers from the Administration Manager or directly from the command line using the *server* command.

The Administration Manager can also be used to manage the configured servers.

1. The *bin* directory must be in the `PATH` or the user will need to change to the *bin* directory before running the utility.

**Note**

- No servers can be started from the command line until the XML configuration files have been fully populated using either the Administration Manager or the *adminMgrTool* tool (located in the *bin* directory). For details, see the *System Guide*.



- If a service uses the OpenFusion Naming Service then the **NameService** server must be started first. A service can use the Naming Services by completing the *Name Service Entry* field for the service by entering the server's *resolve name*.

Using the Administration Manager

A server for a service can be started by right-clicking on the service name and selecting **Start** from the pop-up menu.

Using the Server Command

Servers can also be controlled from the command line using the *server* command (located in the *bin* directory of the OpenFusion installation).

The *server* command uses the following switch options. Each option is followed by one or more server names:

-start - runs the service(s) in the background

-run - runs the service(s) in the foreground

-restart - restarts the service(s)

-stop - stops the service(s)

-status - displays the operational status of the service(s), for example if it is running or not; using the *-status* command without specifying a server name will display the status of all supported servers.

-exec - starts the service(s) and your call will block until the server starts

For example, the Naming Service can be started from the command line using:

```
% <install_dir>\bin\server -start NameService
```

Alternatively, the service can be run in the foreground, rather than the background, with:

```
% <install_dir>\bin\server -run NameService
```

It is also possible to start a service so that it is automatically restarted after any abnormal event:

```
% <install_dir>\bin\server -restart NameService
```

(Note that this is a blocking operation, so the Administration Manager is required to stop the service.)

In order to stop a running server, use:

```
% <install_dir>\bin\server -stop NameService
```

To check whether a server is already running, use:

```
% <install_directory>\bin\server -status NameService
```

4.4 Uninstalling

This section describes the procedure for uninstalling the OpenFusion CORBA Services.



Uninstalling OpenFusion removes all OpenFusion files, including the executables, licence, configuration, and data files located in the OpenFusion sub-directories. If these files are required, then they should be backed-up prior to uninstalling.

Step 1: Stop any running OpenFusion Services.

Step 2: Backup any data, licence or other required files which are in the OpenFusion directories.

Step 3: Run the *uninstall* utility (located in the *bin* directory):

```
% <install_dir>/bin/uninstall
```



Windows users can also use **Start | Programs | OpenFusion | Uninstall OpenFusion** to start the utility.

The utility displays a confirmation dialogue box which asks if you wish to proceed with uninstallation. Clicking the **Yes** button will uninstall OpenFusion.

4.5 Running as a Windows Service



The OpenFusion CORBA Services and OpenFusion JacORB can be optionally run as Windows Services. An application which is run as a Windows Service can be set to automatically start when Windows starts.



The installation process makes changes to Windows' Registry. You must ensure your login has adequate privileges (i.e. *Administrator* privileges) to allow these changes to be made.

Instructions for running OpenFusion and JacORB as Windows Services are given below.

OpenFusion

Installing

Step 1: After installing and configuring OpenFusion, run the `node` command (located in the `bin` directory) as follows:

```
% <install_dir>/bin/node -ntsvc
```

This will create the `service_install.bat` and `service_uninstall.bat` files in the `bin` directory of the OpenFusion installation.

Step 2: Run the `service_install` command as follows:

```
% <install_dir>/bin/service_install
```

A successful installation will list the installed service names.

The services will be installed with the start-up mode set to *Automatic* (so that the system will attempt to start OpenFusion when Windows subsequently reboots). If automatic starting is not required, change the service to start *Manually* in the Windows *Services* console.

Configuration and Service Information

1. In the `service_install.bat` file, the last JDK installed prior to installing OpenFusion is used as a command line argument. This parameter can be edited, if required, to specify the version of Java that OpenFusion is to run from as a Windows service.
2. The node specified in the `OF.Node.URL` environment property will be started (the default is `localhost`) when the Windows service is run.

3. Starting and stopping the Windows service in quick succession can cause problems. It is recommended that the *Administration Manager* is used to check the status of and to stop the OpenFusion services, if required.
4. Windows uses the environment variables set at a *system level only* when running OpenFusion as a Windows service. It is essential that all environment variables required by the ORB installation are set at the *system level* rather than at the *user level*. If the environment variables are not set at the system level, then some OpenFusion services may be prevented from starting successfully.

Stopping

Run *service_uninstall.bat* (located in the *bin* directory of the OpenFusion installation) to stop OpenFusion running as an automatic start-up service.

Troubleshooting

If the service does not start after installation and either displays an error to the screen or writes the error to the *Event Viewer* log, then check that the installation file and/or the registry setting show below is pointing to a valid *jvm.dll* within a JDK. This *dll* should **not** be copied from a JDK to a location other than where the JDK was installed:

```
HKEY_LOCAL_MACHINE\\SYSTEM\\CurrentControlSet\\Services\\
<OpenFusion Version>\\Parameters - key: JVMLibrary
```

JacORB

Systems which use the OpenFusion CORBA Services with JacORB can install the JacORB Implementation Repository as a Windows Service.

The installation program for OpenFusion products that include JacORB include an option for installing JacORB. If this option is selected and it is desired that JacORB is installed as a Windows Service, then complete the OpenFusion installation as normal, then run *service_install.bat* from the command line or by double-clicking on its icon in Windows Explorer.

The Windows service for OpenFusion will be installed as dependent on the JacORB ImR service: both will be assigned the Windows start-up type of *Automatic*.

The JacORB ImR will be registered with the service name ***JacORB 1.4 IMR*** or ***JacORB 2.1 IMR*** (the version number will depend on the current JacORB version). If necessary (for example to resolve a service name conflict), then these values can be changed by editing the *service_install.bat* file: this file is located in the *<install_dir>\bin* directory, where *<install_dir>* is the OpenFusion installation directory.

5 Running Applications

This section provides basic instructions for running client applications and OpenFusion examples with OpenFusion. The information provided here is intended to help developers learn to run applications with OpenFusion quickly and easily. This section is not intended to be a guide for writing code or applications.

The topics covered here include:

- basic CLASSPATH and PATH configuration
- basic steps needed to run client applications
- running the pre-compiled OpenFusion examples
- running user-compiled applications and examples
- general advice on running applications with the OpenFusion CORBA Services

i

Note

- The directory paths shown in this section use the UNIX forward-slash (/) separator convention; Windows® users should substitute these separators with the standard DOS back-slash (\) separator.
- `<install_dir>` is used as a convention in this section to refer to the directory where OpenFusion is installed
- The terms *OpenFusion example*, *client application*, and *application components* are referred to here simply as the *application*, for brevity.

5.1 General

The following information applies to all applications using OpenFusion.

The *run* Script and PATH

Applications generally are started by the OpenFusion *run* script, located in the `<install_dir>/bin` directory. The *run* script can be called by either including the absolute or relative pathname to it on the command line or by adding `<install_dir>/bin` to the PATH environment variable. Setting the PATH variable is generally the best method to use since it saves time and effort in the long term, i.e. *run* can then be called from any directory without worrying about its path location.

Including `<install_dir>/bin` in the PATH will simplify calling the OpenFusion scripts and utilities in general, plus the (commonly used) *run* and *manager* scripts in particular.

CLASSPATH

The CLASSPATH must always include `<install_dir>/lib/fusion.jar`. The CLASSPATH can be set during the system's start-up initialisation, manually from the command line, or using a script.



The `run` script automatically adds `fusion.jar` to the CLASSPATH for the current shell or window.

The CLASSPATH must also include the root directory of any other classes or jars used by the application. Adding these directories or jars to the CLASSPATH is can be done manually or (preferably) by using a script (see *Examples of Running Applications* on page 33).

5.2 Configuration and Running

Basic Steps

The following basic steps are required when running any client-server application (including the OpenFusion services examples) with OpenFusion.

These steps assume that the application has been compiled, is ready to run, and that OpenFusion and an ORB are correctly installed and running (refer to Section 4, *Installing OpenFusion*, on page 17).

The specific details for running the pre-compiled OpenFusion examples are listed under *OpenFusion Examples Specifics* on page 32.

The specific details for running user-compiled examples are listed under *User-compiled Application Specifics* on page 32.

Step 1: Determine from the application's instructions if it is executed using the OpenFusion `run` script, an application-specific script, or (atypically) using the `java` command directly. The application may consist of two or more components: each may need to be run from individual shells or windows. If the application is run from an application-specific script, determine if it uses the `run` script to start the application or its components (see the *Information* note under *Step 2*:).

Step 2: If the application is started using an application-specific script, determine if it sets the CLASSPATH to include the directories and jars for required class files. If not, then the CLASSPATH will need to be set by the user (see *Step 5*:). The paths used (relative or absolute) when setting the CLASSPATH will affect which directories the application can be run from.

i

The CLASSPATH must include `<install_dir>/lib/fusion.jar`. If the application is run using the OpenFusion `run` script, then the CLASSPATH is automatically set to include the `fusion.jar` file.

It is recommended (and generally required) that applications use the OpenFusion `run` script since this script sets a variety of configuration settings. The `run` script is located in the `bin` directory of the OpenFusion installation.

Step 3: The ORB and OpenFusion Service server(s) used by the application must be running. If they are not running, then start them.

i

Starting the ORB and services is explained under *Running the ORB Daemons* on page 22 and *Running Servers* on page 23. Also refer to Section 1, *Running Servers*, in the System Guide. The section in the System Guide provides instructions for running servers using the Administration Manager. Also, if the application uses the OpenFusion Naming Service to resolve a service or component, then ensure the Naming Service server is running.

Step 4: Open a separate shell or window for each application component; the current directory of each window should be changed to the directory where the application component is to be run from (refer to the *CLASSPATH and Application Run Directory* topic under *OpenFusion Examples Specifics* or *User-compiled Application Specifics*, as appropriate).

Step 5: If the CLASSPATH is not set by the script(s) which start the application (refer *Step 2:*), then set it to the required directories and jars for each application component's shell.

Step 6: The application's components should be started in the order specified by the application's instructions. If using the `run` script, it is executed as follows (see *The run Script and PATH* on page 29,

```
% run <packagename.classname> [parameters]
```

where `<packagename.classname>` is the application component's fully qualified classname and `[parameters]` is the list of parameters (if any) used by the component.

For example, the `ServerImpl` component from the Notification Service's *Mail* example is started using:

```
% run com.prismt.cos.CosNotification.examples.Mail.Mail
ServerImpl
```

The application should be running at this stage and (hopefully) perform as expected. If it isn't, then check that the `CLASSPATH` has been correctly set in the shell(s), the required services are running properly, and the application's instructions have been correctly followed.

Step 7: Perform the following when the application is to be stopped:

1. Close the application: a running application component can be stopped by selecting its shell or window, then entering *Ctrl-C*.
2. Stop any services which are running (unless being used by other applications).
3. Clean up left-over client or service object references (do with care and **only** if other applications are not using the service). This can be done from the Administration Manager by right-clicking on the service (displayed in the **Object Hierarchy** pane) and selecting **Restore** from the pop-up menu - the **Select Restore Options** panel will be displayed. Ensure the *Restore default properties* option is **unchecked**, then click the *OK* button.

OpenFusion Examples Specifics

The following information is specific to running the pre-compiled OpenFusion examples: all pre-compiled OpenFusion example class files are in the *fusion.jar* file. Follow the instructions under *User-compiled Application Specifics* below for examples which have been compiled by the user (i.e. those whose class files are not in *fusion.jar*).

CLASSPATH and Application Run Directory

No user configuration is required (see *CLASSPATH* on page 30 for additional details).

The examples can be started from any directory when using the *run* script as described under *The run Script and PATH* on page 29.

PATH

The information provided under *The run Script and PATH* on page 29 applies.

User-compiled Application Specifics

The following information is specific to applications or examples which have been compiled by a user and the resulting class files are not in the *fusion.jar* file (refer to *OpenFusion Examples Specifics* above).

Step 1: Set the `CLASSPATH` with the full pathname of the root directory containing the application's class or jar files (see *Examples of Running Applications* below for examples). Ensure that this entry appears **before** the entry to *fusion.jar*.

Step 2: Set the `PATH` with the full pathname of the directory containing the `run` script (`<install_dir>/bin`) or refer to *The run Script and PATH* on page 29.

Step 3: Start the application (from any directory) using the `run` script or from an user-defined/application-specific script which calls the `run` script.

i

Using a user-defined/application-specific script is an easy and efficient method of starting applications since it can automatically perform all the tasks needed to run the application, such as set the `CLASSPATH`, `PATH`, call `run`, then reset the environment when the application is stopped - and it can be used repeatedly.

CLASSPATH and Application Run Directory

The `CLASSPATH` must include the root directory for the classes or jar file containing the application, plus any other classes or jars required to run the application.



If the `CLASSPATH` is set with the **relative** path of the root directory, then the directory where the application is run from must be relative to that (root) directory. For example, if `CLASSPATH` is set to `./classes`, then the application must be run from the parent directory of `classes`.

5.3 Examples of Running Applications

Example 1 Running a pre-compiled OpenFusion example component.

UNIX

```
% export PATH=/prismtech/openfusion/bin:$PATH
% run com.prismt.cos.CosNotification.examples.News.Analyst
```

WIN

```
% set PATH=D:\prismtech\openfusion\bin;%PATH%
% run com.prismt.cos.CosNotification.examples.News.Analyst
```

Where `Analyst.class` and associated classes are in `fusion.jar`.

Example 2 Running an application using a relative CLASSPATH**UNIX**

```
% export PATH=/prismtech/openfusion/bin:$PATH
% export CLASSPATH=./classes:$CLASSPATH
% cd /usr/urs/examples
% run com.company.my.examples.MySupplier
```

WIN

```
% set PATH=D:\prismtech\openfusion\bin;%PATH%
% set CLASSPATH=.\classes;%CLASSPATH%
% cd D:\examples
% run com.company.my.examples.MySupplier
```

Where the application *must* be run from *examples* and *Analyst.class* and its associated classes are in *examples/classes/com/company/my/examples*

Example 3 Running an application using an absolute CLASSPATH**UNIX**

```
% export PATH=/prismtech/openfusion/bin:$PATH
% export CLASSPATH=/usr/urs/examples/classes:$CLASSPATH
% run com.company.my.examples.MySupplier
```

WIN

```
% set PATH=D:\prismtech\openfusion\bin;%PATH%
% set CLASSPATH=D:\examples\classes;%CLASSPATH%
% run com.company.my.examples.MySupplier
```

Where the application can be run from *any* directory and *Analyst.class* and its associated classes are in *examples/classes/com/company/my/examples*

Example 4 Running an example component contained in a jar**UNIX**

```
% export PATH=/prismtech/openfusion/bin:$PATH
% export CLASSPATH=/usr/examples/lib/my.jar:$CLASSPATH
% run com.company.my.examples.MySupplier
```

WIN

```
% set PATH=D:\prismtech\openfusion\bin;%PATH%
% set CLASSPATH=D:\examples\lib\my.jar;%CLASSPATH%
% run com.company.my.examples.MySupplier
```

Where *MySupplier.class* and associated classes are in *my.jar*.

6 Information Sources

6.1 PrismTech Product Information

Links to useful technical information for PrismTech's products, including the OpenFusion CORBA Services and associated components, are listed below.



These links are provided for the reader's convenience and may become out-of-date if changes are made on the PrismTech Web site after publication of this guide. Nonetheless, these links should still be reachable from the main PrismTech Web page located at <http://www.prismtechnologies.com>.

FAQs

Frequently Asked Questions (FAQs) about OpenFusion are listed at:

http://www.prismtechnologies.com/English/Products/CORBA/Resources/faq_index.html

Knowledge Base

The PrismTech Knowledge Base is a collection of documents and resources intended to assist our customers in getting the most out of the OpenFusion products. The Knowledge Base has the most up-to-date information about bug fixes, product issues and technical support for difficulties that you may experience. The Knowledge Base is at:

http://www.prismtechnologies.com/English/Support/KB_index.html

Additional Technical Information

Information provided by independent publishers, newsgroups, Web sites, and organisations, such as the Object Management Group, is listed on the Prismtech *General CORBA Resources* page at:

http://www.prismtechnologies.com/English/Products/CORBA/Resources/corba_resources.html

6.2 PrismTech Product Support

PrismTech provides a range of product support, consultancy and educational programmes for helping you from product evaluation and development, through to deployment of applications using OpenFusion. The support programmes are designed to meet customer's particular needs and range from a basic *Standard* programme to the *Gold* programme, which provides comprehensive, 24 x 7 support.

Detailed information about PrismTech's product support services is provided on the *PrismTech Support* page located at:

<http://www.prismtechnologies.com/English/Support/index.html>

General PrismTech support contacts and enquiries information is located at:

<http://www.prismtechnologies.com/Contacts>