ObjectStore®

Installing ObjectStore for Windows

Release 6.3

PROGRESS SOFTWARE | Real Time Division

Installing ObjectStore for Windows

ObjectStore, Release 6.3 for Windows, October 2005

September 2005

Contents

Preface

Purpose — *Installing ObjectStore for Windows* provides information and instructions for installing ObjectStore Release 6.3 on Windows machines.

Audience — This book is for system administrators responsible for ObjectStore installation.

How This Book Is Organized

Chapter 1, System Requirements for ObjectStore, on page 9 describes the software and hardware prerequisites, and points to the list of supported platforms.

Chapter 2, Installing and Configuring ObjectStore, on page 13 provides step-by-step instructions for installing ObjectStore, upgrading ObjectStore, modifying your ObjectStore configuration, and uninstalling ObjectStore.

Chapter 3, Miscellaneous Information, on page 33 includes background information about ObjectStore components, processes, and configurations. The information in this chapter can help you make decisions about how to configure ObjectStore.

Notation Conventions

This document uses the following notation conventions

Convention	*Meaning*
`Courier`	`Courier` font indicates code, syntax, file names, API names, system output, and the like.
`Bold Courier`	**`Bold Courier`** font is used to emphasize particular code.
`Italic Courier`	*`Italic Courier`* font indicates the name of an argument or variable for which you must supply a value.
Sans serif	Sans serif typeface indicates the names of user interface elements such as dialog boxes, buttons, and fields.
Italic serif	In text, *italic serif* typeface indicates the first use of an important term.
[ ]	Brackets enclose optional arguments.
{ a | b | c }	Braces enclose two or more items. You can specify only one of the enclosed items. Vertical bars represent OR separators. For example, you can specify *a* or *b* or *c*.
...	Three consecutive periods indicate that you can repeat the immediately previous item. In examples, they also indicate omissions.

Progress Software Real Time Division on the World Wide Web

The Progress Software Real Time Division Web site (www.progress.com/realtime) provides a variety of useful information about products, news and events, special programs, support, and training opportunities.

Technical Support

To obtain information about purchasing technical support, contact your local sales office listed at www.progress.com/realtime/techsupport/contact, or in North America call 1-781-280-4833. When you purchase technical support, the following services are available to you:

- You can send questions to realtime-support@progress.com. Remember to include your serial number in the subject of the electronic mail message.
- You can call the Technical Support organization to get help resolving problems. If you are in North America, call 1-781-280-4005. If you are outside North America, refer to the Technical Support Web site at www.progress.com/realtime/techsupport/contact.
- You can file a report or question with Technical Support by going to www.progress.com/realtime/techsupport/techsupport_direct.
- You can access the Technical Support Web site, which includes
  - A template for submitting a support request. This helps you provide the necessary details, which speeds response time.
  - Solution Knowledge Base that you can browse and query.
  - Online documentation for all products.
  - White papers and short articles about using Real Time Division products.
  - Sample code and examples.
  - The latest versions of products, service packs, and publicly available patches that you can download.
  - Access to a support matrix that lists platform configurations supported by this release.
  - Support policies.
  - Local phone numbers and hours when support personnel can be reached.

Education Services

To learn about standard course offerings and custom workshops, use the Real Time Division education services site (www.progress.com/realtime/services).

If you are in North America, you can call 1-800-477-6473 x4452 to register for classes. If you are outside North America, refer to the Technical Support Web site. For information on current course offerings or pricing, send e-mail to classes@progress.com.

Searchable Documents

In addition to the online documentation that is included with your software distribution, the full set of product documentation is available on the Technical Support Web site at www.progress.com/realtime/techsupport/documentation. The site provides documentation for the most recent release and the previous supported release. Service Pack README files are also included to provide

historical context for specific issues. Be sure to check this site for new information or documentation clarifications posted between releases.

Your Comments

Real Time Division product development welcomes your comments about its documentation. Send any product feedback to realtime-support@progress.com. To expedite your documentation feedback, begin the subject with Doc:. For example:

```
Subject: Doc: Incorrect message on page 76 of reference manual
```

Chapter 1
System Requirements for ObjectStore

Before you begin ObjectStore installation, ensure that you have a supported platform, along with the required hardware and software.

Supported Platforms

ObjectStore is a product that can be installed on multiple types of hardware, operating systems, and in conjunction with a variety of compilers, language-related libraries, JDKs and JREs.

ObjectStore supports a number of platforms, including Windows, Solaris, and Linux. On each of these platforms, there are a number of compilers or JDKs that you can use to build an application. In this document, *platform configuration* refers to a combination of compiler and operating system. To install ObjectStore, it is necessary to understand what platform configuration you require for your application.

The ObjectStore Technical Support web site lists the platforms supported by this release. Go to `http://www.progress.com/realtime/techsupport/support_matrix/objectstore`. When you consult the support matrix, it is important to be aware of the following:

- A *supported* platform configuration has been thoroughly tested prior to product release.
- A *maintained* platform configuration has been known to work and has received some testing, but has not been tested thoroughly.

For details concerning product support, see the ObjectStore Technical Support site at `http://www.progress.com/realtime/techsupport`.

Hardware Requirements

Physical memory and disk space requirements for ObjectStore are as follows. Before you install the ObjectStore software, make sure that your machine environment meets these requirements.

Physical Memory

The amount of physical memory you need depends on the installation configuration required by your development or deployment (run-time) environment.

For ObjectStore server processes, a minimum of 128 MB is recommended.

ObjectStore client processes need more physical memory. Most ObjectStore processing is done in the client. If the application is written to require large transactions, then more physical memory is an advantage. There are some client applications that are written with the expected performance of in-memory access speeds for very large datasets. These configurations require a great deal of physical memory.

Disk Space for ObjectStore and for Your Application

An ObjectStore environment requires disk space for the following:

- Application binaries and related files — all files required to run the application (excluding ObjectStore databases and client caches). This might be on a shared device used by many clients.
- ObjectStore installation files — disk space requirements depend on which components you install.)

ObjectStore DBMS (server) and C++ interface	105 MB
Java interface	17 MB
JMTL	9 MB
DDML	9 MB
Documentation	70 MB

- ObjectStore client cache space — client cache requirements depend on the application. On Windows, you add the client cache space requirements (cache size * number of clients on that host) to the required system swap space. The default client cache size is 8MB. You can adjust this with the `OS_CACHE_SIZE` environment variable. See Chapter 3, in *Managing ObjectStore*.
- ObjectStore transaction log file — the maximum size depends on the application and whether the application has large transactions and whether it is using Multi-Version Concurrency Control (MVCC). During installation, you indicate where you want ObjectStore to store this file. Specify a physical disk that is both
  - Local to the host running the ObjectStore server process
  - Not a physical disk that holds any of the databases that will be served by this ObjectStore server process

- Application databases — ensure that there is enough physical disk space for all databases the application might need. Account for expected growth. Also, allocate disk space for on-line backup, archive logging, and replication, according to your application's needs.

  It is expected that databases served by a particular server will be on disks that are local to that server's host. If this is a problem, contact ObjectStore technical support to determine if there are any alternatives available for your configuration.

Disk Space for Installation Processing

When installing ObjectStore, you also need to allocate temporary disk space for:

- Installation package — 303 MB (including debug libraries)
- Unpackaging the installation — 200 MB

The amount of disk space for the above steps, depends on the components to be installed. You do not need additional disk space for the installation package if you are installing from CDROM. However, if you download the package from Technical Support, you need as much as 303 MB of extra disk space, depending on which components you download.

Software Requirements

To install this release of ObjectStore, ensure that you have the appropriate version for the following:

- Operating system
- C++ compiler/libraries if C++ client applications will be developed or run
- JDK if Java client applications will be developed or run
- JRE application server if JMTL is to be used with an application server

Also, be sure you have installed the appropriate patches for the above software. Information about specific versions and patch levels for software that supports this release of ObjectStore is documented in the *ObjectStore Release Notes*. See the `ReleaseNotes.pdf` file in the ObjectStore installation package.

To read product documentation in a help file format, you must have a Web browser. Netscape 4.7 or greater or Internet Explorer 5.0 or greater is required. To read product documentation from PDF files, Adobe Acrobat Reader 4.0 or greater is required.

Chapter 2
Installing and Configuring ObjectStore

To install and configure ObjectStore:

1. Ensure that you have a supported platform along with the required hardware and software. See Chapter 1, System Requirements for ObjectStore, on page 9.
2. Read the README file and *ObjectStore Release Notes* in the installation package.
3. Read the instructions, from beginning to end, for the installation procedure you will follow. This lets you make decisions about any installation options before you begin installation. For information to help you make these decisions, see Miscellaneous Information on page 33.

   If you are installing ObjectStore for the first time, read through and then perform the steps described in Installing ObjectStore on page 14.

   If you are upgrading, go to Upgrading from ObjectStore 6.x on page 19 or Upgrading from ObjectStore 5.1 on page 22. Read through the instructions and then perform the steps.
4. Follow the instructions for Verifying ObjectStore Installations on page 22.

This section also provides instructions for

Uninstalling ObjectStore on page 27

Updating ObjectStore Configurations on page 29

Installing ObjectStore

Before you install ObjectStore, you must make the decisions described in the next topic, Preparing for Installation. Then you can peform a typical or custom installation.

- Typical Installation on page 14 assumes that you want to install all of the ObjectStore components and that you want the installation program to do most of the configuring. The typical installation requires minimal responses from you.
- Custom Installation on page 16 gives you greater control over the installation process and accordingly asks you to make more decisions about the process than does the typical installation.

Preparing for Installation

Before you install ObjectStore, determine the answers to the following questions:

- What is the path of the destination directory for the ObjectStore installation? That is, where do you want to install ObjectStore?
- Which ObjectStore components do you want to install? See About ObjectStore Installation Components on page 35.
- Is there adequate disk space for installation? See Disk Space for Installation Processing on page 11.
- Will the host be used for developing ObjectStore applications? See About Development Configurations on page 41.
- Will this host will be running an ObjectStore server process? See ObjectStore DBMS - The Server on page 35. If so, you should also determine the following:
  - Where will the transaction log will reside? Note that the transaction log must reside on the host where the ObjectStore server process is running. See About the Transaction Log on page 42.
  - Is the amount of disk space adequate for the databases and the transaction log? See Disk Space for ObjectStore and for Your Application on page 10.
- If you are doing a client-only installation, which host will be used for the ObjectStore server process that serves the clients on this host? Will the clients have access to a file system local to that ObjectStore server process? See About ObjectStore Processes and Configurations on page 37.
- Where will the physical databases reside? See Types of Databases on page 42.

Typical Installation

In a typical installation, the installation program installs all ObjectStore components and does most of the configuring. You can always modify the configuration later.

Note

If you are upgrading, you must first uninstall the currently installed release of ObjectStore, as described in Uninstalling ObjectStore on page 27. The following procedure assumes that you do not have ObjectStore currently installed on the target machine

To perform a typical ObjectStore installation:

1. Run the `setup` program that is included in the installation package.
2. Read the License Agreement and click Accept to indicate your acceptance of the License Agreement.
3. Click OK to acknowledge the copyright notice.
4. Supply or confirm your user information. If the `setup` utility can retrieve this information from the Windows registry, it displays it.
5. Accept the default, or enter the path of the directory where you want to install ObjectStore.
6. Indicate that you want to perform a typical installation. If you want to perform a custom installation, see Custom Installation on page 16.
7. Click Next to confirm that the setup program has the correct information and that you want installation to proceed.

   At this point, the `setup` program creates the ObjectStore directory structure and then installs the ObjectStore components in this directory structure.
8. Indicate whether you want to change any ObjectStore server parameters. Unless you know from previous experience that you need to change any of the settings, you should not need to change them at this time.
9. Indicate whether you want to create rawfs partitions. For more information, see About the rawfs on page 43.
10. Indicate whether you want to initialize the server log file (also known as the transaction log). You should initialize the transaction log only if you have never before run an ObjectStore server process on this host. The transaction log must be initialized before you can start the server for the first time. For more information, see About the Transaction Log on page 42.

    If you do not want to initialize the transaction log, skip to step 12.
11. If you answered `Yes` in the previous step, `indicate` the name and path of the transaction log. If you do not want to use the default, specify a location where the transaction log is not likely to be moved or deleted accidentally. For example, do not put it in a `temp` directory. Also, if you have multiple physical disk drives and need maximum performance, put the transaction log on a disk that will not or does not contain databases. Locating the transaction log on a different disk results in faster propagation of the transaction log.
12. Indicate whether you want the ObjectStore server process to start automatically at system startup. Technical Support recommends configuring the server for automatic startup. You can always shut down the server and restart it manually.
13. Indicate whether you want to start the ObjectStore cache manager and (if you specified automatic startup for the server in the previous step) the ObjectStore server at this time.
14. Indicate whether you want to view the `README` file.

Typical installation is complete. See Verifying ObjectStore Installations on page 22.

Custom Installation

In a custom installation, the setup program lets you select the ObjectStore components to install and the configuration options to implement.

Note

If you are upgrading, you must first uninstall the currently installed release of ObjectStore, as described in Uninstalling ObjectStore on page 27. The following procedure assumes that you do not have ObjectStore installed on the target machine.

To perform a custom installation:

1. Run the `setup` program that is included in the installation package.
2. Read the License Agreement and click Yes to indicate your acceptance of the License Agreement.
3. Click OK to acknowledge the copyright notice.
4. Supply or confirm your user information. If the `setup` utility can retrieve this information from the Windows registry, it displays it.
5. Accept the default, or enter the path of the directory where you want to install ObjectStore.
6. Indicate that you want to perform a custom installation. If you want to perform a typical installation, see Typical Installation on page 14.
7. In the left pane, deselect the components that you do not want to install. A check next to a component's name means that it will be installed.
   - If you have only a C++ application environment, uncheck Java Interface and JMTL, and leave DDML checked if your application might require it.
   - If you have only a Java application environment, leave *both* the C++ and Java Interfaces checked and leave JMTL or DDML checked if your application might require them.
   - If you have a Java and C++ application environment, you should uncheck JMTL and DDML if your application does not need these components.

   To display a brief description of a component, click the component name.

   Do not check or uncheck anything in the right pane. Click Next.
8. Indicate which of the following you want to install on this machine:
   - Client run time only.
   - Sever run time only.
   - Client and server run time.

The remaining installation steps depend on your choice:

- Client-Only Configuration
- Server-Only or Server-and-Client Configuration

Client-Only Configuration

At the end of the steps in Custom Installation on page 16, if you chose to install client run time only then follow these steps:

1 Specify the name of the remote directory that contains the ObjectStore application schema database files. If you do not know the location of this directory, leave the path-information field blank and click Next. The `setup` program displays a message saying that it cannot resolve the path. Click `Yes` to indicate that you will resolve the path at a later time. For information about relocating and resolving the application schema database files, see Resolving Client-Only Schema Database Issues on page 31.

2 Indicate the name of the folder where you want the `setup` program to add shortcuts. The default folder is `ObjectStore` Win32.

3 The `setup` program displays (among other things) a list of the components it will install. Click the `Back` button if you want to change any of the listed information. Otherwise, click the `Next` button to continue the installation.

The `setup` program installs the ObjectStore components that you selected.

4 Indicate if you want the cache manager process to start automatically when the system starts and, if so, whether you want it to start now. Technical Support recommends indicating `Yes` for both questions so that you can verify the operation of the cache manager.

5 Indicate if you want to view the `README` file.

At this point, the custom installation of a client-only configuration is complete.

Server-Only or Server-and-Client Configuration

At the end of the steps in Custom Installation on page 16, if you choose to install the server run time only, or you chose to install client and server run time, then follow these steps:

1 Indicate the name of the folder where you want the `setup` program to add shortcuts. The default folder is `ObjectStore`.

2 The setup program displays (among other things) a list of the components it will install. Click the `Back` button if you want to change any of the listed information. Otherwise, click the `Next` button to continue the installation.

3 After installing the components, the `setup` program prompts you to indicate if you want to change any of the ObjectStore server parameters. Unless you know from previous experience that you will need to change any of the server parameter values, you should not need to change them at this time.

4 Indicate if you want to create rawfs partitions. For more information, see About the rawfs on page 43.

5 Indicate if you want to initialize the server log file (also known as the transaction log). Initialize the transaction log only if you have never run an ObjectStore server process on this host before. You must initialize the transaction log before you start the ObjectStore server process for the first time.

If you do *not* want to initialize the transaction log, skip to step 7.

6 To initialize the transaction log, indicate the name and path of the transaction log. If you do not accept the default, specify a location where the transaction log is not likely to be moved or deleted accidentally. For example, do not put the transaction log in a temporary directory. Also, if you have multiple physical disk drives and need maximum performance, put the transaction log on a disk that will not

contain any ObjectStore databases. Locating the transaction log on a different disk results in faster propagation of the transaction log.

7 Indicate if you want the ObjectStore server process to start automatically at system startup. Technical Support recommends configuring the server for automatic startup. You can always shut down the server and restart it manually.

8 If you choose automatic server startup, indicate if you want to start the server now. It is a good idea to start the server now so that you can verify its operation when installation is complete.

9 Indicate if you want the cache manager process to start automatically when the system starts and, if so, whether you want it to start now. Technical Support recommends indicating `Yes` for both questions so that you can verify the operation of the cache manager.

   If you installing only a server, the `setup` program does not prompt you about the cache manager.

10 Indicate if you want to view the `README` file.

11 If you installed the DDML component, you need to add the DDML information to the `INCLUDE`, `LIB`, and `PATH` environment variables as described in ObjectStore Environment Variables on page 44.

At this point, the custom installation is complete.

Upgrading from ObjectStore 6.*x*

Upgrading from ObjectStore 6.*x* is straight-forward, unless your application requires a different compiler for 6.3 than the one used in the earlier release. This section provides the procedure for upgrading from ObjectStore 6.*x* to ObjectStore 6.3. After the upgrade procedure, there is a discussion about compatibility among 6.*x* releases.

Preparing to Upgrade from ObjectStore 6.*x*

Before you upgrade to Release 6.3:

- Ensure that your platform configuration is supported for this release and service pack. For a list of supported platforms, see the *Support Matrix* (http://www.progress.com/realtime/techsupport/support_matrix/objectstore) on the Technical Support web site.
- Read the `README.htm` and the *Release Notes* to ensure that no source code changes to the application are required prior to the upgrade.
- Read the *ObjectStore Migration Guide* to determine if any steps are required prior to upgrading (http://www.progress.com/realtime/techsupport/documentation/objectstore).
- Read through all the instructions for upgrading. This lets you carefully plan your upgrade to minimize the interruption in service.

After an upgrade from Release 6.*x*, you need to re-compile and re-link your applications with the Relase 6.3 libraries.

If you are upgrading a number of host systems in a distributed environment, your first step is to upgrade the host (or hosts) where the ObjectStore server process resides.

Procedure for Upgrading from Release 6.*x*

To upgrade from ObjectStore 6.*x* to ObjectStore 6.3:

1. Shut down all ObjectStore applications.
2. If an ObjectStore server process is running on the host you are upgrading
   a. Ensure that ObjectStore client processes are not accessing this server.
   b. Checkpoint any databases that the ObjectStore server has open (*checkpointing* propagates information in the transaction log to the database):
      ```
      ossvrchkpt host_name
      ```
   c. Back up all databases, application files, and the ObjectStore installation directory.
   d. Select Start | Settings | Control Panel | Administrative Tools | Services and then shut down the two ObjectStore services — ObjectStore server and ObjectStore cache manager.
3. If you want the upgraded ObjectStore installation to be in the same location as your current ObjectStore installation, follow the instructions in Uninstalling

ObjectStore on page 27. To move your current installation instead of uninstalling it, contact ObjectStore technical support for instructions for doing this.

4 Follow the instructions in Installing ObjectStore on page 14.

5 Follow the instructions in Verifying ObjectStore Installations on page 22.

6 Re-compile, re-link, and verify that your application runs as expected.

Compatibility Among 6.*x* Releases

If your ObjectStore environment includes more than one release version, you need to be aware of the following compatibilities.

6.3 Servers

ObjectStore 6.3 servers can fulfill requests from 6.1.2, 6.2, 6.2.1, or 6.3 clients.

ObjectStore 6.3 servers cannot fulfill requests from 6.0.10 clients. Upgrade 6.0.10 clients to at least Release 6.1.2.

6.3 Clients

Only ObjectStore 6.3 servers can fulfill requests from 6.3 clients. ObjectStore 6.0.*x*, 6.1.*x*. and 6.2.*x* servers cannot fulfill requests from 6.3 clients.

ObjectStore 6.3 clients can use 6.0.*x*, 6.1.*x*. and 6.2.*x* databases that are managed by 6.3 servers if the schema of the 6.3 client application is compatible with the schema of the code that generated the 6.0.*x*, 6.1.*x*, or 6.2.*x* database. The schemas are compatible if at least one of the following is true:

- The client application and the code that generated the database were compiled by the same compiler version.
- When the code that generated the database was compiled, it was neutralized for the machine on which the client application was compiled.

Databases

In general, databases created with 6.0.*x*, 6.1.*x*, or 6.2.*x* are compatible with 6.3 applications. However, ObjectStore does not automatically support an upgrade of an application for which the compiler is changing. Please refer to the *Migration Guide* for instructions for how to upgrade to a different compiler (http://www.progress.com/realtime/techsupport/documentation/objectstore).

A common compiler change as customers upgrade to this release is Visual C++ 6 or 7 to Visual Studio .NET 2003 on Windows.

ObjectStore C++ databases created with 6.1.*x* or 6.2.*x* are compatible with Release 6.3 if their schemas were generated with Visual C++ 7. Release 6.1.*x* databases generated with Visual C++ 6, are compatible with Release 6.3 applications if both of the following are true:

- The applications do not use packing pragmas with persistently stored classes.
- The applications do not use integral extension types such as `_int16` and `_int32` in any context that appears in the schema.

If your Release 6.1.x databases use schemas generated with Visual C++ 6 from code that uses packing pragmas or integral extension types, see the *ObjectStore Release Notes* for information about how to handle these incompatibilies.

Obsolete and Deprecated APIs

If your applications use obsolete or deprecated APIs, there are some code changes that you need to make. Obsolete and deprecated APIs are listed in the *Migration Guide* available on the Technical Support Web site at http://www.progress.com/realtime/techsupport/documentation/objectstore.

Upgrading from ObjectStore 5.1

The internal database format changed between 5.1 and 6.3. This change was implemented with 6.0. This means that 5.1 ObjectStore clients and 6.3 ObjectStore clients cannot access the same database. In order for a 6.3 ObjectStore client to access a database created by a 5.1 client, the database needs to be dumped and loaded with the ObjectStore utilities `osdump` and `osload`.

The following steps summarize this type of upgrade. For detailed instructions for upgrading this release of ObjectStore, consult the *Migration Guide* at (http://www.progress.com/realtime/techsupport/documentation/objectstore).

1. Install the 5.1 `osdump` utility that is compatible with ObjectStore 6.3.
2. Install ObjectStore 6.3 in a directory different from that of 5.1. See Installing ObjectStore on page 14.
3. Read the *Migration Guide* and migrate your application accordingly (http://www.progress.com/realtime/techsupport/documentation/objectstore).
4. Completely rebuild your application (C++ migrated code or Java code) with ObjectStore 6.3. For C++ applications, use an ObjectStore 6.3 server process to build your application.
5. Back up your 5.1 application and database(s).
6. Use the 5.1 `osdump` utility from step 1 to dump your ObjectStore 5.1 database(s).
7. Load your database(s) using the 6.3 `osload` utility.
8. Test your 6.3 application completely.
9. Uninstall ObjectStore 5.1.

Verifying ObjectStore Installations

After you install ObjectStore software, you can verify that the installation process properly configured your environment:

- Verifying Installation Directory Contents on page 23
- Verifying ObjectStore Services on page 24
- Verifying a C++ Development Client on page 24
- Verifying a C++ Run-time Client on page 24
- Verifying a Java Client on page 26
- Modifying Configuration Settings on page 29

Verifying Installation Directory Contents

ObjectStore installation directory contents depend on what you selected to install. Some directories are always installed. You should verify that the following directories or a subset of them have been installed

`ostore`	Directory for the ObjectStore DBMS, and C++ environment; also contains `setup.exe` for performing configuration changes
`ostore\bin`	All ObjectStore utilities (`.exe` files) and DLLs
`ostore\binsngl`	Utilities and DLLs for ObjectStore Single
`ostore\etc`	Configuration files
`ostore\examples`	ObjectStore C++ API code examples
`ostore\include`	ObjectStore C++ API include files
`ostore\lib`	ObjectStore libraries and schema databases
`ostore\unsupported`	ObjectStore performance measurement tools that are available for customers to use, but are not a supported component of ObjectStore
`doc`	Documentation in HTML and PDF for all ObjectStore components
`osji`	This directory is for the Java interface to ObjectStore `osji.jar` — ObjectStore Java interface (OSJI) classes `tools.jar` — OSJI development tools classes `osjdo.jar` — ObjectStore JDO (OSJDO) implementation classes `jdo.jar` — Standard JDO interface `jdd.jar` — ObjectStore's Java Dynamic Data classes `browser.jar` — ObjectStore Java Browser `stublib.jar` — Stubs of OSJI classes that allow user-defined persistence-capable classes to be used in a pure transient manner. For details, see Using Persistence-Capable Classes in a Transient Manner in Chapter 14, (Miscellaneous Information) in the *Java API User Guide*. `osji_g.jar` `tools_g.jar` `jdd_g.jar` These are the debug versions of the OSJI jar files. They are compiled with the `-g` flag and have symbolic information that is useful to ObjectStore engineers. If you are having a problem, you might be asked to use these jar files to obtain a stack trace.
`osji\lib`	Directory of libraries and library schema databases
`osji\bin`	Directory of executables, DLLs, and batch files
`osji\include`	Directory of include files needed for C++ interoperability

`osji\com\odi\demo`	Demonstration hierarchy of examples of ObjectStore programs
`osji\com\odi\tutorial`	Instructions and sample program for getting started
`osji\setup.bat`	Script that initializes OSJI after it is installed
`osji\Setup.class`	Run by the setup script for client-only installations
`osji\run_person.bat`	Script to verify the installation by running the `Person` demo
`jmtl`	Installation directory for the Java Middle Tier Library; contains the `.jar` files for using JMTL
`jmtl\bin`	JMTL utilities
`jmtl\examples`	JMTL examples
`ddml`	Installation directory for the Dynamic Data Modeling Library

Verifying ObjectStore Services

For a typical installation or a custom client-and-server installation, verify that the ObjectStore server process and the ObjectStore cache manager process are listed in the collections of services for your machine.

Select Start | Settings | Control Panel | Administrative Tools | Services. You should see both ObjectStore Cache Manager and ObjectStore Server.

Ensure that you can start and stop these services. By default, the services are started automatically if you chose this option during installation.

Verifying a C++ Development Client

If you have a development environment on this host with a VC++ compiler, build and run the `hello2` example. In a .NET 2003 command window, go to `ostore\examples\hello2` and enter:

```
doall.bat
```

Building and running this example verifies that your development environment is set up and that you can create an ObjectStore database with the C++ API.

Verifying a C++ Run-time Client

If you set up a C++ run-time environment, you can run a C++ ObjectStore application on this host but you cannot build one. A simple way to verify that a C++ run-time client is properly installed is to run one of the utilities that comes with ObjectStore. The following verification procedure runs the ObjectStore `ossize` utility:

1. In a default installation, the `setup` program sets the necessary environment variables. However, you might want to make sure that the `OS_ROOTDIR, LIB`, and `PATH` environment variables are set to the values shown in ObjectStore Environment Variables on page 44.

2 Run the `ossize` utility, which is in the `ostore\lib` directory. You run it on an ObjectStore database. You can use any of the library schema databases that are in the `ostore\lib` directory. These are the `*.adb`, `*.db`, or `*.ldb` files.

The actual location of these database files depends on

- The host on which the ObjectStore server process is running
- Whether the ObjectStore server process has been configured for rawfs database access

For example, if the ObjectStore server process host is configured for rawfs databases and the library schema databases are in the default rawfs, then you can verify a client (whether a UNIX or Windows host) as follows:

```
ossize RemoteHostName::/ostore/schema/6.3/metaschm.db
```

If the ObjectStore server process host has a different location for the library schemas, then enter that path, rather than the default path shown above.

If the remote host is not configured for rawfs databases, then you must find out where ObjectStore has been installed on that host.

Here is an example of how to access the library schema on a UNIX remote host. This assumes that ObjectStore has been installed in `/usr/local`:

```
ossize RemoteHostName:/usr/local/ODI/ostore/lib/metaschm.db
```

Here is an example of how to access the library schema on a Windows remote host. This assumes that ObjectStore is installed on `C:\`:

```
ossize \\RemoteHostName\c\ODI\ostore\lib\metaschm.db
```

The output should look something like this:

```
C:\ODI\ostore\63\VC71\OStore\lib>ossize metaschm.db
Name: C:\ODI\ostore\63\VC71\OStore\lib\metaschm.db
Size: 335872 bytes (328 Kbytes)
Created: Mon Sep
17 11:21:33 2005

There is 1 root:
Name: __comp_schema   Type: _Comp_schema

There are no affiliated databases.

The schema is local.

There is 1 segment:

Data segment 2:
Size: 248832 bytes (243 Kbytes)
Segment 2 has 1 cluster:
Cluster 0 size: 248832 bytes (243 Kbytes)
```

Verifying a Java Client

To verify the installation of a Java client, enter the following command in the `%OS_ROOTDIR%\..\osji` installation directory:

```
run_person
```

This runs the `Person` demo. It displays diagnostic messages if it detects something wrong in the installation.

If you are not running an ObjectStore server process on the machine on which the `%OS_ROOTDIR%\..\osji` directory is stored, specify a pathname for a database that resides on a machine that is running an ObjectStore server. For example:

```
setup \usr1\bob
run_person \usr1\bob\person.odb
```

As with the `setup` command, you can specify a host-qualified pathname.

If you are running a client with a remote ObjectStore server that requires a user name and password (authentication required), you can specify them like this:

```
setup jackhammer:c:\\bob
run_person jackhammer:c:\\bob\\person.odb bob PassWd
```

Uninstalling ObjectStore

The procedure for uninstalling ObjectStore depends on the release of ObjectStore that you are uninstalling.

- Uninstalling Releases Prior to 6.3
- Uninstalling Release 6.3

Uninstalling Releases Prior to 6.3

To uninstall an ObjectStore release that is earlier than 6.3:

1. Shut down all ObjectStore client applications that are running.
2. Shut down all ObjectStore services that are running.
   a. Select Start > Control Panel > Administrative Tools > Services.
   b. Double-click the ObjectStore server service and click Stop.
   c. Double-click the ObjectStore cache manager service and click Stop.
3. From the Windows Start menu, navigate to the ObjectStore folder, and select the `setup` program.

   The `setup` program detects that ObjectStore is installed, and prompts you to select one of the following options:

   - ObjectStore Setup.
   - Reinstall.
   - Uninstall.
4. Select Uninstall.
5. Confirm that you want to unintsall the ObjectStore components that the `setup` program lists.
6. Indicate whether you want to delete the contents of any rawfs partitions. For more information, see About the rawfs on page 43
7. Confirm that you want to continue with the uninstall process. After you click Yes, the `setup` program uninstalls ObjectStore. That is, it
   - Deletes the directory and all its contents
   - Removes references from the registry
   - Deletes any file-based rawfs partitions if you so indicated
8. If the following components are installed, the setup program prompts you to indicate whether you want to remove them. You should confirm that you want to uninstall each of these components.
   - OSJI
   - JMTL
   - DDML

Uninstalling Release 6.3

To uninstall ObjectStore 6.3:

1. Shut down all ObjectStore client applications that are running.
2. Shut down all ObjectStore services that are running.
   a. Select Start > Control Panel > Administrative Tools > Services.
   b. Double-click the ObjectStore server service and click Stop.
   c. Double-click the ObjectStore cache manager service and click Stop.
3. Select Start > Control Panel > Add or Remove Programs.
4. In the Add or Remove Programs dialog, select ObjectStore 6.3.0, and click Change/Remove.
5. Select Uninstall.
6. Confirm that you want to unintsall the ObjectStore components that the `setup` program lists.
7. Indicate whether you want to delete the contents of any rawfs partitions. For more information, see About the rawfs on page 43
8. Confirm that you want to continue with the uninstall process. After you click Yes, the `setup` program uninstalls ObjectStore. That is, it
   - Deletes the directory and all its contents
   - Removes references from the registry
   - Deletes any file-based rawfs partitions if you so indicated

Updating ObjectStore Configurations

This section provides information about configuration changes that you can make:

- Modifying Configuration Settings
- Resolving Client-Only Schema Database Issues

Modifying Configuration Settings

After you install ObjectStore, you can use the `setup` installation program (`%OS_ROOTDIR%\setup.exe`) to do any of the following tasks. To perform any of these tasks, you must first shut down ObjectStore services. The installation program prompts you to do this.

Here are the configuration tasks that the `setup` program can do:

- Change the settings of ObjectStore server parameters.
- Initialize an existing or new transaction log.
- Enable or disable automatic startup for the ObjectStore server.

Note

You must have successfully installed ObjectStore before you can use the `setup` program in configuration mode.

To use the `setup` program to configure ObjectStore, perform the following procedure:

1. Use the Windows Start menu to navigate to the ObjectStore folder, and select the `setup` program.
2. The `setup` program asks you to select one of the following task items:
   - ObjectStore Setup.
   - Reinstall.
   - Uninstall.

   Select ObjectStore Setup to configure ObjectStore.
3. Indicate if you want to change any server parameters. If you answer `Yes`, `setup` displays a dialog box that lists the server parameters and enables you to edit their settings. For detailed information about all of the server parameters, see *Managing ObjectStore*, Chapter 2, (Server Parameters). Any changes you make to the server parameters do not take effect until you restart the ObjectStore server process.
4. Indicate if you want to create rawfs partitions. For more information, see About the rawfs on page 43.
5. Indicate if you want to initialize the server log file (also known as the *transaction log*). You should initialize the transaction log only if you have never run an ObjectStore server process on this host before. It must be initialized before you can start up the ObjectStore server process for the first time. For more information, see About the Transaction Log on page 42.
6. If you answered `Yes` in the previous step, `setup` asks for the name and path of the transaction log. If you do not accept the default, you should specify a location

where the transaction log is not likely to be moved or deleted accidentally. For example, do not put it in a `temp` directory. Also, if you have multiple physical disk drives and need maximum performance, put the transaction log on a different disk from where you have located the ObjectStore database. Locating the transaction log on a different disk results in faster propagation of the transaction log.

7 Indicate if you want the ObjectStore server to start automatically. If you answer `Yes`, the server will be started automatically on system startup. If you answer `No`, you must start the server manually.

Resolving Client-Only Schema Database Issues

If you created a client-only ObjectStore installation and did not specify the path to the remote directory containing the schema database files, you need to manually add this information to the following ObjectStore DLLs:

- `o6cmpct1.dll`
- `o6_coll1.dll`
- `o6sevol1.dll`
- `o6query1.dll`

Run the `ossetasp` utility, using the following command line to display the pathname of the current location of an application schema database file:

```
ossetasp -p dll_pathname
```

Use the following command line to assign a new location to be used by any of the libraries listed above:

```
ossetasp dll_pathname new_schema_db_location
```

You can find detailed information about the `ossetasp` utility in *Managing ObjectStore*.

Chapter 3
Miscellaneous Information

This section discusses the following topics:

About ObjectStore Releases and Release Numbers

A particular ObjectStore release is identified by a number and can be followed by a service pack number. For example, ObjectStore 6.3 is the first release and subsequent releases would be named 6.3 Service Pack 1, 6.3 Service Pack 2, and so on. Service packs are *drop-in* compatible and it is expected that customers will upgrade to service packs as they are released.

In addition to service packs, ObjectStore Technical Support sometimes releases *patches* or *quick drops*. A patch or a quick drop is a specific library that addresses a software problem that needs to be fixed before a service pack can be tested and released. The installation of a service pack is a complete installation or upgrade. The installation of a patch or quick drop is typically the replacement of a library. When a service pack is released, it contains all patches and quick drops that were distributed prior to the service pack.

Obtaining ObjectStore Installation Media

You can obtain the files for installing ObjectStore from a CD or from the ObjectStore Technical Support download area.

- To obtain a CD, contact your sales representative or ObjectStore Technical Support.
- To gain access to the ObjectStore Technical Support download area, contact ObjectStore Customer Service to obtain the necessary username and password. You can use email (`objectstore-customerservice@objectstore.com`) or telephone (781 280-4005).

On the download site, each platform configuration for ObjectStore installation is in its own directory. In addition, there is a directory name that reflects the name of the release of ObjectStore. It is important to know which platform configuration you need to install.

The CD for installing ObjectStore on a particular platform configuration contains the following files:

`README.htm`	Important information about this release
`ReleaseNotes.pdf`	New and changed features of this release
`InstallationGuideForWindows.pdf`	Installation instructions
`setup.exe`	Installation program
`*.cab`	Installation contents

If you download the ObjectStore installation package, you get the following files:

`README.htm`	Important information about this release
`InstallationGuideForWindows.pdf`	Installation instructions
`ObjectStore.exe`	Installation program

When you run the `ObjectStore.exe` program, you get the files that are provided on the CD, including `ReleaseNotes.pdf`.

About ObjectStore Installation Components

ObjectStore contains several components that you can use in a development or in a deployment environment. This document refers to the latter as a *run-time environment*. The type of ObjectStore installation depends on the type of environment required. This section describes the major components of an ObjectStore environment:

- ObjectStore DBMS - The Server
- C++ Interface to ObjectStore
- Java Interface to ObjectStore
- Java Middle Tier Library (JMTL)
- Dynamic Data Modeling Language

ObjectStore DBMS - The Server

The ObjectStore DBMS is the database server and associated files that make up the core ObjectStore services that can serve both C++ and Java applications. This document refers to the DBMS as the *server* because in an ObjectStore application, the ObjectStore server communicates with any type of application (C++ or Java) and accesses ObjectStore databases on behalf of the C++ or Java client applications. In addition to the ObjectStore server, the DBMS includes a number of utilities and libraries that you use to manage databases in an ObjectStore environment. The utilities and libraries that are installed depend on whether the environment is for C++ or Java.

C++ Interface to ObjectStore

C++ developers use the ObjectStore C++ Application Programming Interface (API) to write applications that access ObjectStore databases in a client process that communicates with the ObjectStore server process. Developers access data in their databases as they would any C++ object in program memory. However, they use the ObjectStore C++ API for controlling placement of objects in the database and for transactional control. ObjectStore includes a collections library that lets you aggregate, query, and index C++ objects in a database.

ObjectStore supports C++ applications written on different platforms and compilers, all accessing the same data in a distributed environment. The C++ interface includes the C++ Middle-Tier Library (CMTL) that lets the C++ developer write applications with distributed caches.

Java Interface to ObjectStore

Java developers use the ObjectStore Java API to write applications that access ObjectStore databases in a Java Virtual Machine (JVM) that communicates with the ObjectStore server process. Developers access data in their databases as they would any Java object in program memory. However, they use the ObjectStore Java API for controlling placement of objects in the database and for transactional control.

The JDK Collections classes let you aggregate, query, and index Java objects in a database. ObjectStore supports Java applications written on different platforms, all accessing the same data in a distributed environment.

Java Middle Tier Library (JMTL)

The Java Middle Tier Library (JMTL) integrates with several application servers for either EJB or J2EE JTA transaction support (or both). Developers use JMTL to write middle-tier applications that require transactionally-consistent distributed data caches. JMTL uses the Java interface to ObjectStore, along with a specialized set of classes that enable the batching and routing of transactional activity to be sent to the ObjectStore server process.

Dynamic Data Modeling Language

The Dynamic Data Modeling Language (DDML) provides an easy-to-use model for the creation of name/value pairs for flexible schema data access in an ODBMS. With the C++ interface, developers can use DDML to extend schemas at run time.

About ObjectStore Processes and Configurations

This section provides information about ObjectStore processes and configurations. It is intended to help you determine which components to install and configure on particular hosts. The following topics are discussed:

- Description of ObjectStore's Process Architecture
- About Development Configurations
- About Run-time Configurations
- Types of Databases
- About the Transaction Log
- About the rawfs

Description of ObjectStore's Process Architecture

The main processes in an ObjectStore environment depend on the components installed and whether the application is a Java application or a C++ application. The core environment for ObjectStore is C++. If your application is written in Java, you will be executing some C++ code at the storage management level automatically. That is why the C++ library is required in an ObjectStore environment.

Server Processes

All access to ObjectStore databases is done by means of the ObjectStore server process (`osserver`). The ObjectStore server process must reside on the same host as the physical databases that it manages. This eliminates the need for the ObjectStore server process to access databases over the network. On the server host, the resources are:

- The ObjectStore server process.
- Any databases that the server is accessing. A database can be managed/accessed by only one ObjectStore server process.
- The ObjectStore transaction log file. The transaction log must reside on the same host as the ObjectStore server process. For each server, there is always one and only one ObjectStore transaction log file, no matter how many databases that ObjectStore server process is managing. The ObjectStore transaction log is a special file that holds all transaction data that need to be propagated to the databases that the ObjectStore server process is managing. The ObjectStore transaction log is a critical file that should never be removed unless you are sure that all data has been propagated to the databases.

Client Processes

Any number of ObjectStore clients can access databases by means of one or more ObjectStore server processes. For an ObjectStore client to access a database, the client communicates with two processes:

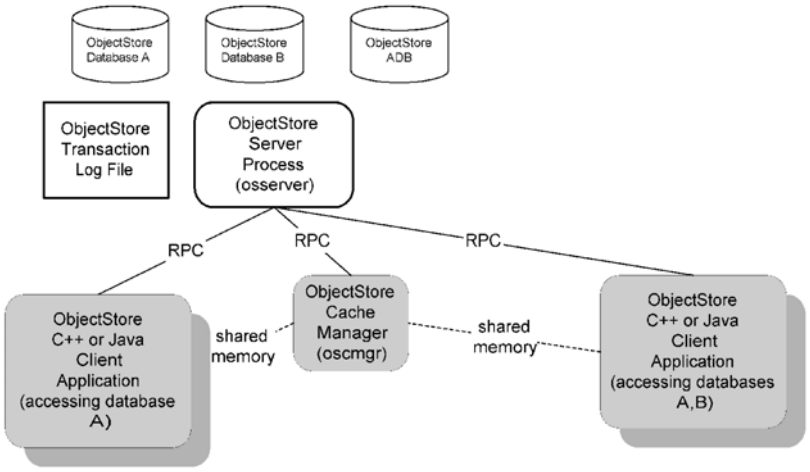
- The ObjectStore server process, which grants access to clients.
- The ObjectStore cache manager process (`oscmgr6`), which must be on the same host as the client process. Each ObjectStore client host has exactly one cache manager process regardless of the number of local clients. The cache manager

keeps track of all pages that are being used by all clients on a particular host. These pages can be from databases that are managed by any number of servers.

The ObjectStore client can either be a C++ process that is linked with the ObjectStore libraries or a JVM (Java Virtual Machine) that utilizes the ObjectStore classes in the ObjectStore `.jar` files.

One-Host Configuration

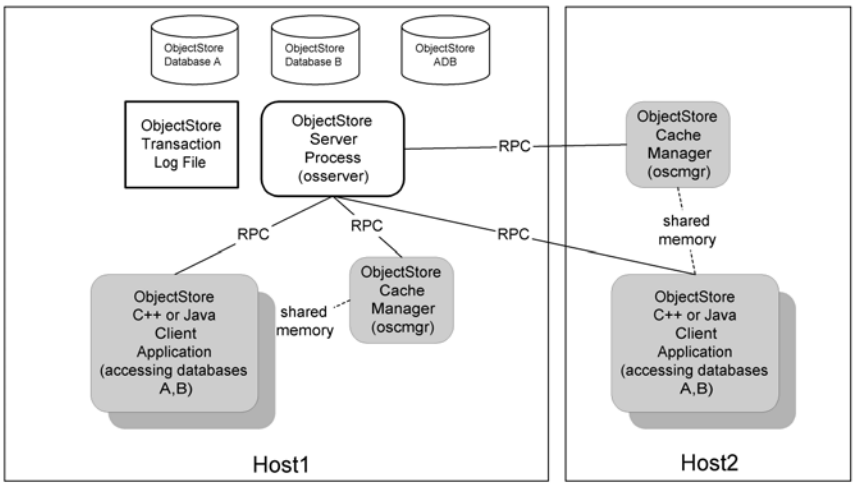
The following figure illustrates a simple ObjectStore process architecture:



In this environment, there is a single ObjectStore server process and multiple ObjectStore client processes, all running on the same host. This environment requires a development or run-time installation, which includes a C++ environment. If you require OSJI, JMTL, or DDML, then you need to install those components. This host is configured as a server.

Two-Hosts One-Server Configuration

The following figure illustrates a multi-host ObjectStore process architecture involving a single ObjectStore server process:

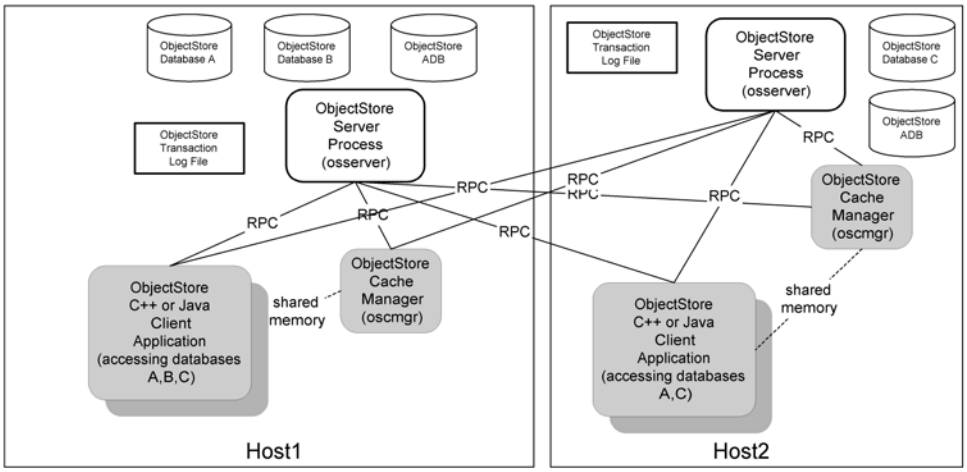


In this environment, there is a single ObjectStore server process and multiple ObjectStore client processes. The ObjectStore server process is set up on `Host1` which

is equipped with perhaps a larger set of disks for databases. The ObjectStore clients can run on either `Host1` or `Host2`. For this configuration, you install ObjectStore (run time or development) on each host. If you require OSJI, JMTL, or DDML, make sure to install them. `Host1` is configured as a server and `Host2` is configured as a client.

Two-Hosts Two-Servers Configuration

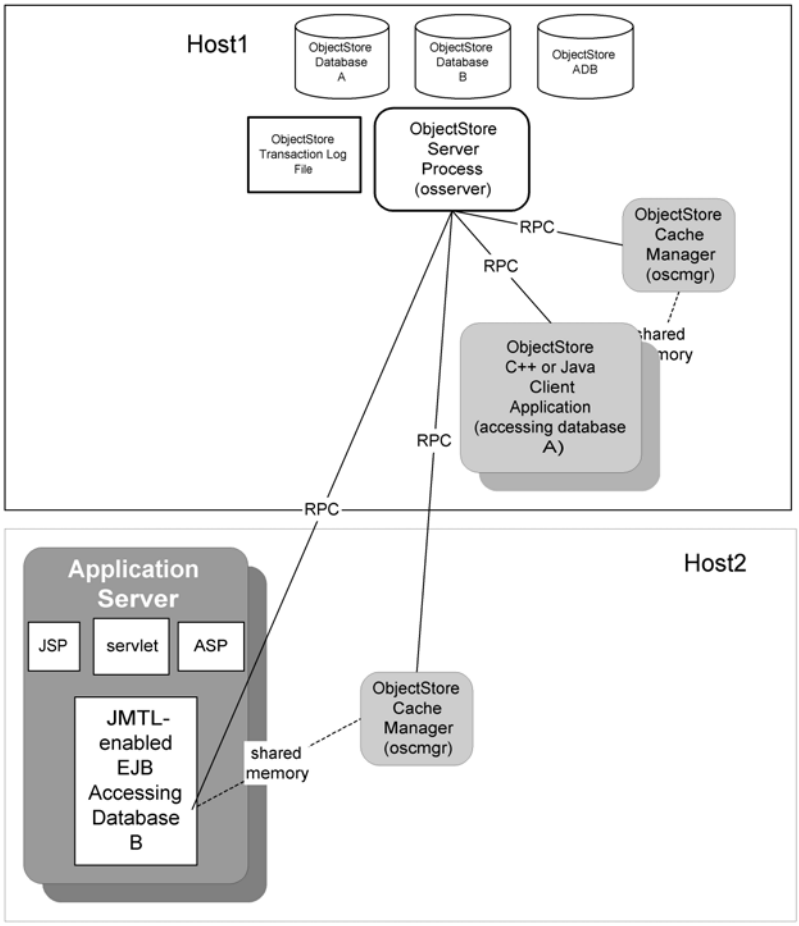
The following figure illustrates a multi-host ObjectStore process architecture involving multiple ObjectStore server processes:



In this environment, there are multiple ObjectStore server processes running on multiple hosts and multiple ObjectStore client processes. An ObjectStore server process is set up on each host. Both `Host1` and `Host2` must have adequate disk space to support locally accessible ObjectStore databases. The ObjectStore clients can run on either `Host1` or `Host2`. This environment requires development or deployjment installation. You can install OSJI, JMTL, or DDML. Both `Host1` and `Host2` are configured as servers.

Application Server Configuration

When JMTL is installed and configured, either for development or deployment, the process architecture involves the use of an application server. There are many design choices that can be made in the use of JMTL and how it utilizes an application server. A typical configuration is one where multiple JVMs are instantiated, each of which can be an ObjectStore client process. The following figure illustrates a very simple process architecture where JMTL is installed, along with an application server



In this environment, there is a single ObjectStore server process and multiple ObjectStore client processes. Some clients are running on `Host1`. On `Host2`, there is an application server that is using JMTL. The ObjectStore clients can run on either `Host1` or `Host2`. This environment requires a development or run-time installation. `Host1` can have OSJI, JMTL, and DDML installed on it. `Host2` must have OSJI and JMTL installed on it (DDML is optional). `Host1` is configured as a server and `Host2` is configured as a client.

About Development Configurations

A development configuration typically involves the installation of all ObjectStore components for development and run time along with the appropriate compilers or JDK.

The C++ build process involves the creation of schema, which requires an ObjectStore server process. Consequently, C++ developers need access to an ObjectStore server process to build an ObjectStore application. Java developers do not need access to an ObjectStore server to build an ObjectStore application.

To build and test an application, a developer needs access to the C++ include files and libraries or the Java `.jar` files. There are special utilities installed in a development environment that allow schema to be created.

Any number of developers can share the resources of an ObjectStore server process on a *shared* host. This assumes that

- Developers who need to access databases (during build or testing) have access to a host with an ObjectStore server process.
- The databases used by the application for building or testing are local to the host where the ObjectStore server process is running.

About Run-time Configurations

A run-time (also called deployment) configuration depends on the application requirements. Run-time configurations are primarily for deployed applications. Consequently, run-time configurations do not contain any of the schema generation utilities needed to build an application.

A run-time configuration might reside on a dedicated host that runs only the ObjectStore server process. All clients can run on different hosts. Alternatively, a deployment configuration can be a pool of hosts that have ObjectStore server processes on them. The ObjectStore server process hosts must be capable of having very large disk storage capabilities. Memory requirements are not as important for the ObjectStore server process as they are for the client processes.

The ObjectStore client processes require the most memory. Each client needs a minimum amount of memory and backing disk store for paging. How many clients you configure to run on each host depends on your application design.

In a run-time configuration, you can run both the ObjectStore server process and one or more ObjectStore client processes on the same host. This is an application design choice.

Types of Databases

An ObjectStore database is a file in the operating system file system or in the rawfs. See About the rawfs on page 43. The only ObjectStore process that physically accesses a database is the ObjectStore server process.

A database contains a schema that describes the types of objects that this database can contain. When you store an object in a database that does not already contain an instance of that object, ObjectStore automatically updates the schema to contain a definition of the new object.

An ObjectStore database can only be accessed or viewed by an ObjectStore application or specialized tools and utilities that are developed to access the data in the database. An ObjectStore database can be served by only a single ObjectStore server process. An application can access any number of ObjectStore databases at the same time.

The physical database files or partitions must reside on the local file system of the host where the ObjectStore server process is running.

Applications use a special type of database, with the suffix of `.adb` or `.ldb`, to ensure that the schema of objects being used by the client applications matches the schema of the database containing the data. These special databases are called application schema databases or library schema databases. Like all databases, only the ObjectStore server process can access application schema databases, and these databases must be on a host file system where an ObjectStore server process is running.

About the Transaction Log

The ObjectStore transaction log is a special file that is accessed by only the ObjectStore server process. It must be located on the same host where the ObjectStore server process is running. In addition, for performance reasons, you should configure the ObjectStore transaction log so that it is located on a different physical disk drive than the disk drive that holds the databases that the ObjectStore server process accesses. Also, the transaction log should be on a dedicated disk drive that is not used for any other purpose.

The ObjectStore transaction log is the file that contains the *active* and some committed transactions for all databases that the ObjectStore server process is managing for clients. At frequent intervals, ObjectStore propagates the committed transaction contents of the log file to the databases. For this reason, it is extremely important that you do not remove the ObjectStore transaction log unless you are making a special configuration change and you have ensured that all transaction data has been propagated to the databases. You force propagation by checkpointing the ObjectStore server process or by shutting down the ObjectStore server process. (You use the `ossvrshtd` utility to ensure a clean shutdown and you wait for the `osserver` process to exit without an error. An unclean shutdown might leave needed data in the transaction log.)

For checkpointing or shutdown to occur, it is also necessary for you to ensure that no ObjectStore client applications access the databases served by this ObjectStore server

process. Once the ObjectStore server process is completely shut down, you can safely remove the ObjectStore transaction log. This enables the system administrator to initialize a new ObjectStore transaction log file either in the same location or in a different location.

About the rawfs

The term rawfs refers to the ObjectStore RAW File System. This is a special area or partition(s) on a disk that is reserved for use by only the ObjectStore server process. A rawfs can occupy any reasonable number of partitions. Only one ObjectStore server process can access a rawfs. One advantage of a rawfs is that it can span multiple disk partitions, allowing for very large databases. Another advantage is that the files (databases) and directory structure that make up the rawfs are not visible from the operating system using normal commands (`dir` or `ls`). This special file system is visible only by means of the ObjectStore commands or an ObjectStore client process.

You can expand a rawfs, but you cannot make it smaller. If you want a rawfs to be smaller, you must back up the databases in the rawfs, remove the rawfs and recreate a smaller one.

If you are planning on using a rawfs, you must create the disk partition(s) before you can configure it(them) for use by ObjectStore. If you are planning on using a file for a rawfs, ensure that there is adequate disk space for the rawfs that you want it to contain.

A rawfs for a host must be local to that host.

ObjectStore Environment Variables

The ObjectStore installation assigns default values to several environment variables (with the exception of the vaules associated with the DDML component — these need to be added manually). Normally, you do not need to change the default settings of these environment variables. The default settings are shown below.

`INCLUDE` Include path for C++ API:

`%OS_ROOTDIR%\include` (if C++ API is used)

`%OS_ROOTDIR%\..\osji\include` (if OSJI Java/C++ interoperability is used)

`%OS_ROOTDIR%\..\ddml\include` ((this must be added manually if DDML is used)

`LIB` Library path for C++ API:

`%OS_ROOTDIR%\lib`

`%OS_ROOTDIR%\..\osji\lib` (if Java Interface is used)

`%OS_ROOTDIR%\..\ddml\lib` (this must be added manually if DDML is used)

`OS_ROOTDIR` Installation directory for ObjectStore (ends with `ostore`)

`OS_TMPDIR` Location of temporary log files used by ObjectStore services. (The transaction log is NOT stored here.)

`PATH` Path to ObjectStore executables (services and utilities):

`%OS_ROOTDIR%\bin`

`%OS_ROOTDIR%\..\osji\bin` (if Java Interface used)

`%OS_ROOTDIR%\..\ddml\bin` ((this must be added manually if DDML is used)

`%OS_ROOTDIR%\..\jmtl\bin` (if JMTL is used)

`CLASSPATH` Path to `OSJI and` JMTL classes:

`%OS_ROOTDIR%\..\osji\osji.jar` (if Java Interface is used)

`%OS_ROOTDIR%\..\osji\tools.jar` (if Java Interface is used)

`%OS_ROOTDIR%\..\osji\osjdo.jar` (if JDO is used)

`%OS_ROOTDIR%\..\osji\jdo.jar` (if JDO is used)

`%OS_ROOTDIR%\..\jmtl\jmtl.jar` (if JMTL is used)