Installing ObjectStore for Windows

Release 6.1
February 2003

Installing ObjectStore for Windows

ObjectStore Release 6.1 for all platforms, February 2003

Contents

Chapter 1 **Overview 3**
ObjectStore Components 3
ObjectStore DBMS 4
C++ Interface 4
Java Interface 4
Javlin 5
Dynamic Data Modeling Language 5
ObjectStore Platform Configurations 5
Types of Installation Procedures 6
New installation 6
Upgrade from Release 6.0 6
Upgrade from Release 5.1 7
Upgrading from 6.1 to a Later Service Pack 7
Adding a Component 8
Configuration 8
Installation Concepts 8
Development Configuration 13
Runtime Configuration 13
Types of Databases 13
What is the Transaction Log? 14
What is a RAWFS? 15
Overview of the Installation Process 15
Getting Help from Technical Support 17

Chapter 2 **System Requirements for ObjectStore 19**
Supported Platforms 19
Hardware Requirements 19
CPU 19
Memory 19
Disk Space 20
Software requirements 21

Installation Media for ObjectStore 22
Obtaining Installation Media 22
Description of ObjectStore Releases 22

Chapter 3 Installing and Configuring ObjectStore 25
Installation Procedure 25
Changing the ObjectStore Configuration 31
ObjectStore Environment Variables 31
Verify Installation 32
Configuration Tasks 36
Changes to the Transaction Log 36
Changes to RAWFS 37
Auto-starting ObjectStore Services 38
Modifying ObjectStore Server Parameters 39
Upgrading from ObjectStore 6.0 39
Upgrading from Release 5.1 40

Chapter 4 Miscellaneous Information 43
Uninstalling ObjectStore 43
Adding ObjectStore Components 44
Resolving Client-Only Schema Database Issues 44
Troubleshooting Installation Problems 45

Chapter 1
Overview

ObjectStore is an industrial-strength Object Database Management System (DBMS) that enables applications to store and manage any type of data, no matter what the complexity. It features a distributed process architecture that allows for applications to scale and enables in-memory access to data of any type (C++, Java, XML). The ObjectStore DBMS supports any variety of distributed processing through its flexible process architecture which ensures transactional consistency in a wide variety of network configurations. The ObjectStore server process supports clients on many different platforms. For a list of supported platforms, see the *Support Matrix* (www.objectstore.net/support/matrix) on the Technical Support web site.

ObjectStore applications running on different platforms can all access the same data. ObjectStore has all of the necessary features for system management which include on-line backup and restore, archive logging and recovery, replication, transparent failover and many other system management operations necessary in an enterprise DBMS.

ObjectStore Components

ObjectStore contains several components that can be used in a development or in a deployment environment. This document refers to the latter as a *runtime environment*. The type of ObjectStore installation depends on the type of environment required. Here are the major components of an ObjectStore environment.

ObjectStore DBMS

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. Along with the ObjectStore server, the DBMS also includes a number of utilities and libraries that can be used to administer databases in an ObjectStore environment. Which utilities and libraries are installed will depend upon whether the environment is for C++ or Java.

C++ Interface

C++ developers write client applications using the ObjectStore C++ Application Programming Interface (API) to 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, but use the ObjectStore C++ API for controlling placement of their objects in the database and for transactional control. ObjectStore includes a collections library that provides application developers the ability to 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++ API also includes the C++ Middle-tier Library (CMTL) which enables developers to implement transactionally-consistent distributed caches.

Java Interface

Java developers write client applications using the ObjectStore Java API to access ObjectStore databases in a Java Virtual Machine (JVM) which communicates with the ObjectStore server process.

Developers access data in their databases as they would any Java object in program memory, but use the ObjectStore Java API for controlling placement of their objects in the database and for transactional control. The JDK Collections and ObjectStore Collections classes provide application developers the ability to 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.

Javlin

The Javlin component is used to integrate with several application servers for either EJB or J2EE JTA transaction support (or both). Javlin includes the Java Middle-tier Library (JMTL) that developers use when they want to write middle-tier applications that utilize the power of transactionally-consistent distributed data caches. The Javlin component utilizes the Java interface to ObjectStore, along with a specialized set of classes which 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) is a class library which allows for 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.

ObjectStore Platform Configurations

ObjectStore is a product that may be installed on multiple types of hardware, operating systems, and in conjunction with a variety of compilers, language-related libraries, JDKs and JREs. The product is built to run on a set of platforms.

ObjectStore supports a number of platforms including Windows, Solaris, and Linux. For each platform, there are one or more compilers that ObjectStore is built with in order support applications built with those compilers. In this document, the combination of compiler/operating system is referred to as a *platform configuration*. In order to install ObjectStore, it is necessary to understand what platform configuration you require for your application.

For a list of platform configurations supported by this release of ObjectStore, see the *Support Matrix* (www.objectstore.net/support/matrix) on the Technical Support web site. This information is also available in the *Release Notes*.

Types of Installation Procedures

The procedures that you follow in order to install this release of ObjectStore or any of its components will depend on what your goal is. Here is a summary of the ways in which the ObjectStore installation procedures may be used on a given platform.

New installation

This is the most straightforward type of installation because there is no existing application or set of databases that need to be migrated. These procedures should be followed when ObjectStore is being installed for the first time for new development or for setting up a system for deployment.

Upgrade from Release 6.0

Upgrading from Release 6.0 is very straight-forward, unless your application requires a different compiler than the one used in 6.0. You should first make sure that the platform configuration is supported for this release and service pack. For a list of supported platforms, see the *Support Matrix* (www.objectstore.net/support/matrix) on the Technical Support web site.

For an upgrade from 6.0 to 6.1, application developers need to recompile and relink their applications. The rules concerning compatibility of ObjectStore processes in a mixed-release environment will hold. That is, a 6.0 ObjectStore server process will not be able to communicate with a 6.1 ObjectStore client process but a 6.1 ObjectStore server process will be able to communicate with a 6.0 or 6.1 ObjectStore client process. This means that if you are upgrading a number of host systems in a distributed environment, it is important that you first upgrade the hosts where the ObjectStore server process reside. In general, databases created in 6.0 will be compatible for 6.1 applications. The exceptions to this will be if there is a major compiler change (see below).

An upgrade requires that you first ensure that all applications running on a given host that use the ObjectStore server process are shut down, the ObjectStore server process has checkpointed the databases it has open (propagated the transaction log to the physical databases) and that the necessary ObjectStore processes have been cleanly shut down. The checkpointing and shutting down of the ObjectStore server process is only necessary if you are upgrading the host where the ObjectStore server process is running.

When 6.1 is installed, the installation process will ask the user if they want to overwrite the existing ObjectStore binaries, delete and install them, or install them in a new location. It will be the responsibility of the user doing the installation to make sure that all databases are properly backed up prior to the upgrade.

In addition to database compatibility between releases, there are some code changes that users need to make if they are using deprecated APIs. The deprecated APIs are detailed in the *Release Notes* for this release and in the *Migration Guide* available on the support Web site at `www.objectstore.net/documentation/migration`.

ObjectStore does not automatically support an upgrade of an application whereby the compiler is changing. Please refer to the *Migration Guide* for instructions for how to upgrade to a different compiler. A common compiler change as customers upgrade to this release is Visual C++ 6 to Visual Studio .NET on Windows.

As more platforms/compilers are available for ObjectStore 6.1, more options will be available for developers to upgrade their ObjectStore environments.

For more information on upgrading Release 6.0 databases and applications, see Upgrading from ObjectStore 6.0 on page 39.

Upgrade from Release 5.1

If you have an existing ObjectStore Release 5.1 installation and want to upgrade to Release 6.1, you need to perform a database migration and modify your application source code in order to use your databases and applications with Release 6.1 Consult the *Support Matrix* (www.objectstore.net/support/matrix) on the Technical Support web site which lists the platform configurations supported for this release of ObjectStore. Some Release 5.1 platform configurations that are no longer supported.

For more information on upgrading Release 5.1.x databases and applications, see Upgrading from Release 5.1 on page 40.

Upgrading from 6.1 to a Later Service Pack

This procedure is very simple because it involves the installation of the latest release of ObjectStore over your existing release. It does not involve any application source code changes or database migrations. It is generally drop-in or link compatible, depending on the nature of the service pack.

As with any ObjectStore upgrade, you must ensure that all databases have been backed up, all ObjectStore applications have been shut down and all ObjectStore services are no longer running. This type of upgrade should be done to overwrite the existing ObjectStore installation.

Adding a Component

If, during installation, you choose not to install the entire ObjectStore product suite, you can use the ObjectStore installation program to add additional components at a later date.

An example of this would be if you installed the Java Interface and now want to install the Javlin component. For more information, see Adding ObjectStore Components on page 44.

Configuration

On Windows, you can use the installation process to do some configuration tasks such as initializing the transaction log file or setting up ObjectStore server process parameters.

See Configuration Tasks on page 36 for more details.

Installation Concepts

The main processes in an ObjectStore environment depend upon the components installed and whether the application is a Java application or a C++ application. The core ObjectStore environment is C++. If your application is written in Java, it will execute some C++ code at the storage management level automatically. This is why Visual Studio C++ is required in an ObjectStore environment.

All access to ObjectStore databases is done via the ObjectStore server process (`osserver`). The ObjectStore server process must reside on the same host where the physical databases reside. This eliminates the need for the ObjectStore server process to access databases over the network.

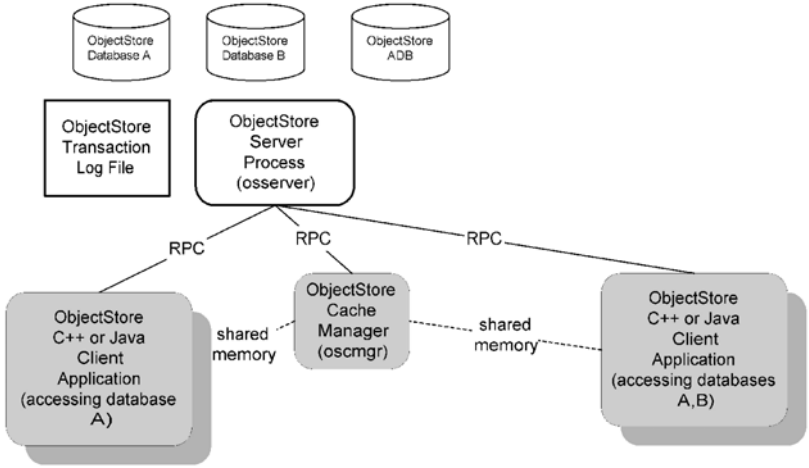
On the *server* side, the resources are the ObjectStore server process, its databases that it is accessing, and the ObjectStore transaction log file which also must reside on the same host as the ObjectStore server process. There is

always one and only one ObjectStore transaction log file, no matter how many databases the ObjectStore server process is managing. A database can only be managed by a single ObjectStore server process. The ObjectStore transaction log is a special file that holds all transaction data that needs 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.

Any number of ObjectStore clients can access databases via one or more ObjectStore server processes. (each ObjectStore server process needs to be running on a separate host). In order for an ObjectStore client to access a database, it communicates with two processes: the ObjectStore server process and the ObjectStore cache manager process (`oscmgr6`). The ObjectStore server process grants access to clients and the ObjectStore cache manager process keeps track of locks that all clients have on a particular host with any number of ObjectStore server processes they are communicating with which could be on multiple hosts. The ObjectStore client can either be a C++ process that is linked with the ObjectStore libraries or a JVM that utilizes the ObjectStore classes in the ObjectStore `.jar` files.

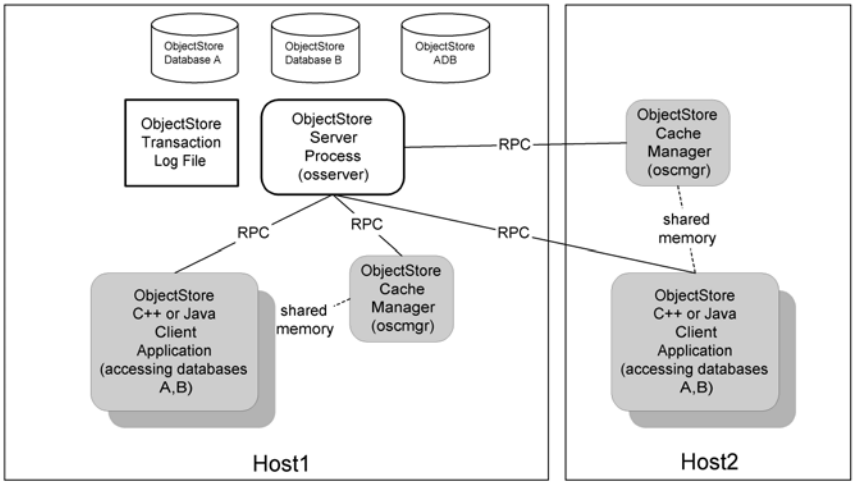
When Javlin is installed and configured, either for development or runtime, the process architecture involves the use of an application server. There are many design choices that can be made in the use of Javlin and how it will utilize an application server. A typical configuration would be one where multiple JVMs are instantiated, each of which could be an ObjectStore client process.

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 typical installation if you need both C++ and Java environments. If you need only a C++ environment, then perform a custom installation, installing the C++ interface and ObjectStore DBMS, and selecting both client and server.

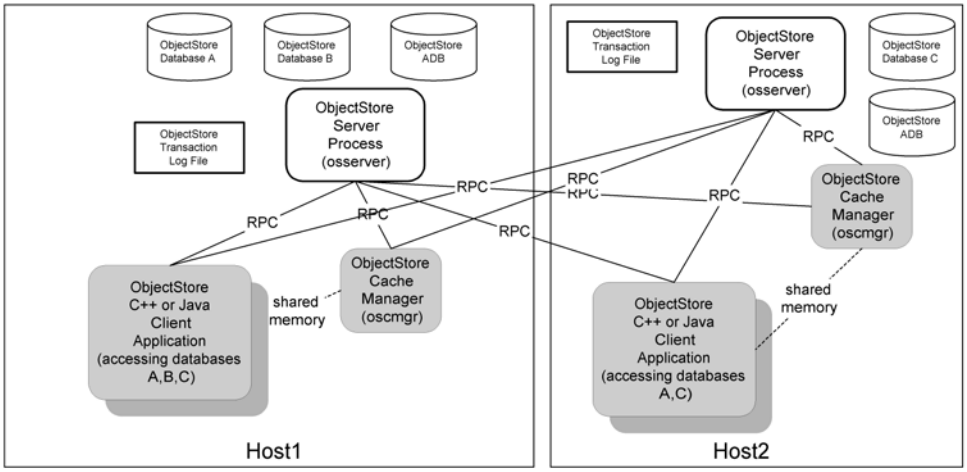
The following figure illustrates a multi-host ObjectStore process architecture involving a single ObjectStore server process:



In this environment, there is a single ObjectStore server process and multiple ObjectStore client processes. The ObjectStore server process has been 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. This environment requires a typical installation if you want to install both C++ and Java environments on Host 1.

If you require only a C++ environment, then perform a custom installation deselecting the Java interface and installing both client and server on Host 1. For Host 2, perform a custom installation and deselect the Java interface, installing the client only.

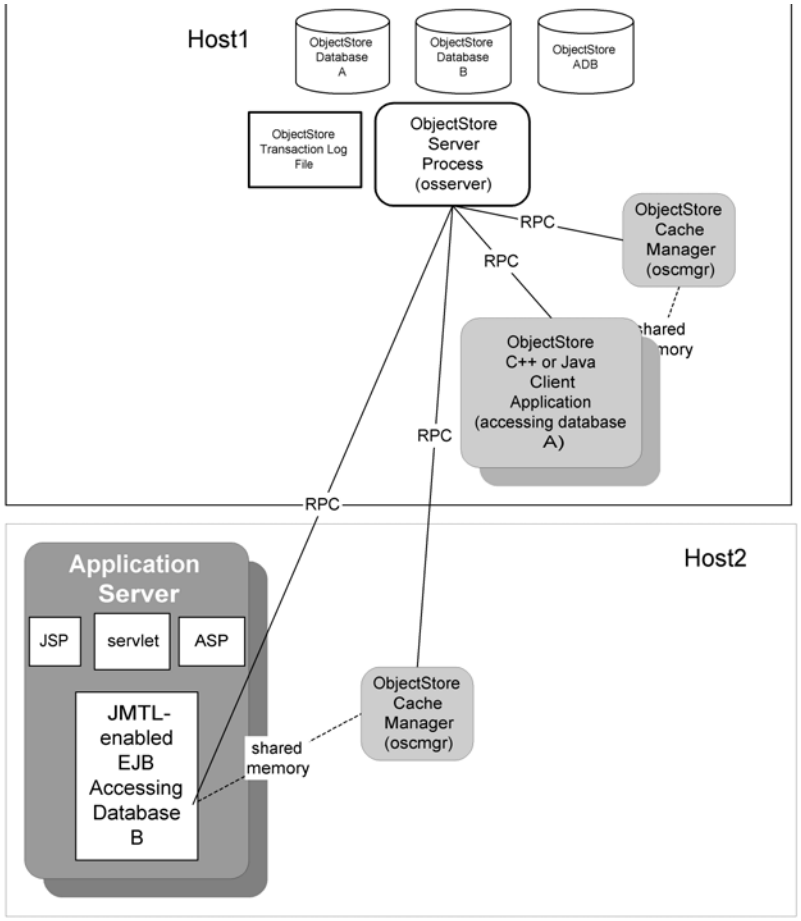
The following figure illustrates a multi-host ObjectStore process architecture involving multiple ObjectStore server processes:



In this environment, there are a multiple ObjectStore server processes running on multiple hosts and multiple ObjectStore client processes. The ObjectStore server process has been 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.

This environment requires a typical installation if you want to install both C++ and Java environments on Host 1 or Host2. If you require only a C++ environment, then perform a custom installation, deselecting the Java interface and installing both client and server on Host 1 and Host2

The following figure illustrates a very simple process architecture where Javlin 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 and on Host 2, there is an application server which is using Javlin. The ObjectStore clients can run on either Host1 or Host2.

This environment requires a typical installation if you want to install both C++ and Java environments on Host 1. If you require only a C++ environment, then you would perform a custom installation, but would deselect the Java interface and install both client and server on Host 1. For Host2, perform a custom installation where you select the Java Interface and Javlin, installing the client only.

Development Configuration

A development configuration typically involves the installation of all ObjectStore components for development and runtime along with the appropriate compilers or JDK. C++ developers need access to an ObjectStore server process in order to build the application. This is because the C++ build process involves the creation of an application schema database which must be created using the ObjectStore server process. A developer needs access to the C++ include files and libraries or the Java `.jar` files in order to build and test the application. Any number of developers can share the resources of an ObjectStore server process on a *shared* host, provided that the developers who need to access databases (during build or testing) have access to a host with an ObjectStore server process and the databases used by the application for building or testing will be local to that host where the ObjectStore server process is running.

Runtime Configuration

A runtime or deployment configuration will be one that should have a pre-determined configuration, based upon the application needs. In many cases, it will involve a dedicated host that runs only the ObjectStore server process. All clients are on different hosts. It may involve a pool of hosts that will have ObjectStore server processes on them. The ObjectStore server process hosts will need to be capable of having very large disk storage capabilities. Memory requirements are not as important for the ObjectStore server process.

The ObjectStore client processes will require the most memory. Each client will need a minimum amount of memory and backing disk store for paging. How many clients one configures to run per host will depend on the application design.

It is certainly possible to have both the ObjectStore server process and the ObjectStore client process(es) running 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. The exception to this is a RAWFS database which will be described later in a later section. The only ObjectStore process which physically accesses the database is the ObjectStore server process. A database contains an application schema which describes the types of objects that this database can contain. Only instances of objects of the defined types may be stored in the database. An

ObjectStore database can only be accessed or view by an application or specialized tools and utilities that are developed to access the data in the database. An ObjectStore database can only be served by a single ObjectStore server process. An application can access any number of ObjectStore databases at the same time. The physical database files must reside on the local file system of the host where the ObjectStore server process is running. A special type database with the suffix of `.adb` and `.ldb` is used by applications in order to ensure that the schema of objects being used by the client applications match the schema of the database containing the data. The special databases are called application schema databases and library schema databases. Since these special files are databases, they can only be accessed by the ObjectStore server process. These databases must be placed on a host file system where an ObjectStore server process is running.

What is the Transaction Log?

The ObjectStore *transaction log* is a special file that is only accessed by the ObjectStore server process. It should be configured to reside on the same file system where the ObjectStore server process is running. In addition, for performance reasons, it is recommended that the ObjectStore transaction log is configured to be located on a different physical disk drive than the disk drive which will hold the databases that the ObjectStore server process will access. The ObjectStore transaction log is the file that contains the *active* transactions for all databases that the ObjectStore server process is serving to clients. At pre-configured intervals, the contents of the log file for committed transactions is propagated to the databases. For this reason it is extremely important that the ObjectStore transaction log not be removed unless a special configuration is being made and the system administrator has ensured that all transaction data has been forced to be propagated to the databases. This forcing of the propagation can be achieved by checkpointing the ObjectStore server process and by shutting down the ObjectStore server process. In order for this effective checkpointing or shutdown to occur, it is also necessary that the system administrator can ensure that no ObjectStore client applications access the databases served by this ObjectStore server process. Once the ObjectStore server process is completely shut down, the ObjectStore transaction log can be safely removed. This enables the system administrator to initialize a new ObjectStore transaction log file either in the same location or in a different location.

What is a RAWFS?

RAWFS is the term for the ObjectStore RAW File System. This is a special area or partition on a disk that is reserved for use only by the ObjectStore server process. Only one ObjectStore server process may 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 only visible via the ObjectStore commands or by an ObjectStore client process.

A RAWFS is expandable, but it cannot be made smaller. If you want a RAWFS to be smaller, you have to back up the databases in the RAWFS, remove the RAWFS and recreate a smaller one.

If you are planning on using a RAWFS partition, you should make sure the partition has been already created prior to the configuration of the partition to be used by ObjectStore. If you are planning on using a file for a RAWFS, you should make sure that there is adequate disk space for the RAWFS that you want it to contain. Finally, a RAWFS for a host must be local to that host.

Overview of the Installation Process

1. Determine which platform configuration that you need to install and obtain the installation media either on CD or by downloading from the ObjectStore Technical Support Web site.
2. Read the `README.htm` for any last minute information that might affect installation choices.
3. Read the `ReleaseNotes.pdf` for an overview of what this release contains that may affect an application that was developed using a prior release of ObjectStore.
4. Read this Installation Guide completely before you perform any installation steps.
5. Ensure that you have Administrator or root privileges in order to do the installation.
6. Ensure that you have the correct hardware (memory and disk space), operating system, compiler, application server (if applicable) as described

in System Requirements for ObjectStore on page 19 and the Support Information in the *Release Notes*.

7 Determine the ObjectStore components that will be installed.

8 Determine if you will be installing a client-only or server-only configuration.

9 Determine where the physical databases will reside for development or server-only and ensure that there is adequate disk space.

10 Determine where the ObjectStore transaction log file will reside. Ideally, the transaction log should be located on a different disk drive from the drive where the databases are located.

If you are planning to use ObjectStore RAWFS databases, ensure that you have raw file space that is available to be overwritten. If you are at all unsure about whether to use file databases or RAWFS databases, you can discuss the decision with your system administrator or Technical Support. You can defer the creation of an ObjectStore RAWFS partition until later.

11 Determine whether developers will need to modify any existing application source code (deprecated APIs or migrating from Release 5.1) based upon the information in the *Release Notes* for this release. The upgrade cannot occur until the software has been modified and rebuilt.

12 Determine whether the databases will need to be either schema evolved or dumped/loaded based upon the type of upgrade of existing database that needs to occur. Developers will need to perform this step. If the previous release of ObjectStore is 5.1, then a full migration must occur of the application software and the databases. For more information see the *ObjectStore Migration Guide* on the Technical Support web site at `www.objectstore.net/documentation/migration`.

13 If you are running an earlier release of ObjectStore, shut down any ObjectStore applications and back up your databases, ObjectStore installation binaries and your application software.

14 Perform the desired type of installation. See Installation Procedure on page 25.

15 Verify the installation.

Getting Help from Technical Support

When you purchase technical support, the following services are available to you:

- You have access to the latest revision of the ObjectStore documentation at www.objectstore.net/documentation.
- You can send questions to support@objectstore.net. Remember to include your site ID in the body of the electronic mail message.
- You can call the Technical Support organization to get help resolving problems. If you are in North America, call 781.280.4005.
- You can access the Technical Support Web site at www.objectstore.net/support., which includes
  - Information about contacting ObjectStore Technical Support outside of North America.
  - A template for submitting a support request. This helps you provide the necessary details, which speeds response time.
  - Frequently asked questions (FAQs) that you can browse and query.
  - White papers and short articles about using ObjectStore products.
  - Sample code and examples.
  - The latest versions of ObjectStore products, service packs, and publicly available patches that you can download.
  - Access to an ObjectStore product Support Matrix.
  - A migration guide for developers who need to migrate from ObjectStore 5.1 to ObjectStore 6.1 at www.objectstore.net/documentation/migration.

Chapter 2
System Requirements for ObjectStore

Supported Platforms

The Support Matrix (`www.objectstore.net/support/matrix)` contains information about which platform configurations are supported for this release. A supported platform configuration means that it has been thoroughly tested prior to product release. A maintained platform configuration means that it has been known to work and has received some testing, but it has not been tested thoroughly. See the Technical Support site for more details concerning product support.

Hardware Requirements

Before you install the ObjectStore software, make sure that your machine environment meets the following requirements.

CPU

Your CPU must be capable of running the operating system installed. Technical Support recommends the fastest possible CPU for increased performance.

Memory

The amount of physical memory required will depend on the installation configuration required by the development or deployment environment.

The ObjectStore server process does not require much physical memory, but a minimum of 128 MB is recommended. If you are running ObjectStore client processes on a host, you will need more physical memory. Most of the ObjectStore processing is done in the client. If the application is written to require large transactions, then more physical memory will be a benefit. There are some applications that are written with the expected performance of in-memory access speeds of very large datasets. These configurations require a great deal of physical memory.

Disk Space

Disk Space for the Application and ObjectStore

There are several considerations for estimating the amount of disk space required. One consideration is the types of files required in an ObjectStore environment:

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

ObjectStore DBMS and C++ interface	105 MB
Java interface	17 MB
Javlin	9 MB
DDML	9 MB
Documentation	70 MB

- ObjectStore client cache space — client cache requirements depend on the application. On Windows, the cache requirements (cache size * number of clients on that host) need to be added to the system swap space.
- ObjectStore server
- ObjectStore transaction log file — the maximum size will depend on the application and whether it has large transactions or is using Multi-version Concurrency Control (MVCC). Create this file on a physical disk that is local to the host running the ObjectStore server process and is a different physical disk than all of the databases that will be served by this ObjectStore server process.

- Application Databases — make sure that there is enough physical disk space for all databases the application may need (even for future growth). All databases served by this ObjectStore server process must be on a disk that is local to the host where the ObjectStore server process is running. Consider disk space for possible on-line backup, archive logging, replication if the application requires it

Disk Space for Installation Processing

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

- Installation package — 303 MB (including debug libraries)
- Temporary space required to do the installation (unpackaging)

The amount of disk space for the above steps, will depend upon 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'll need as much as 303 MB of extra disk space, depending on which components you download.

Software requirements

In order to install this release of ObjectStore, you need to make sure that the appropriate versions of operating systems, C++ compilers/libraries (if C++ client applications will be developed or run), JDKs or JREs, application servers (if Javlin is to be used with an application server), and the appropriate patches for the above software is installed. Information about specific versions and patch-levels for software that supports this release of ObjectStore is documented in the `README.htm` for this release. In addition, current information about supported and maintained configurations can be found in the Support Matrix (`www.objectstore.net/support/matrix`).

In addition to the platform configuration information for your installation, it is recommended that you read the product documentation using a Web browser (Netscape 4.7 or greater or Internet Explorer 5.0 or greater) to view the Webhelp or Adobe Acrobat Reader (4.0 or greater) to view the PDFs.

Installation Media for ObjectStore

Obtaining Installation Media

The files necessary for installing ObjectStore can be obtained either from a CDROM or from the Technical Support download area. A CDROM can be obtained by contacting your sales representative or Technical Support (`www.objectstore.net/support`. or 781-280-4005).

In order to gain access to the Technical Support FTP server, you must contact Technical Support (`www.objectstore.net/support`. or 781-280-4005) to obtain the necessary username and password.

Each platform configuration for the installation of the ObjectStore components is in its own directory. In addition, on the download site, there will be a directory name which reflects the name of the release of ObjectStore. It is important to understand which platform configuration you need to install.

For a particular platform configuration, the files necessary to do the installation are:

`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

Description of ObjectStore Releases

ObjectStore releases are given numbers followed by a Service Pack number. For example, ObjectStore 6.1 is the first release and subsequent releases will be named 6.1 Service Pack 1, 6.1 Service Pack 2, etc. 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, Technical Support sometimes releases *patches* or *quick drops*. These are very specific libraries that are intended to address a software problem that needs to be fixed before a full Service Pack can be tested and released. The installation of Service Packs are full installations or upgrades. The installation of patches or quick drops are 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.

Chapter 3
Installing and Configuring ObjectStore

Review Chapter 1, Overview, on page 3 before you begin the installation and make sure you know the following information:

- Location of ObjectStore installation directory
- If there is adequate disk space for installation. See Disk Space on page 20 for more information on estimating disk space requirements.
- What ObjectStore components need to be installed
- The type of installation you want. The choices are
  - Client and server
  - Client-only
  - Server-only
- If you are doing a typical or server-only installation:
  - Where the ObjectStore transaction log will reside
  - Adequate disk space for databases
- If you a doing a client-only installation, the name of the host that will be used for the ObjectStore server process for the clients (make sure the client applications will have access to a file system accessible by that ObjectStore server process)

Installation Procedure

This section describes the steps you need to perform to install the ObjectStore Release 6.1 software. The actual steps you perform will vary depending on whether you are creating a brand new installation or if you are installing

over an existing version of the ObjectStore software. The actual steps you perform will also vary depending on whether you are performing a Typical or a Custom installation.

If you are installing Release 6.1 over an existing release, it is better to uninstall the existing software before installing the new software. You can do this before you start the installation process or you can do it as part of the installation process.

If you are going to uninstall a previous version of the software, you need to decide on an upgrade procedure to minimize interruptions in service. See Upgrading from ObjectStore 6.0 on page 39 or Upgrading from Release 5.1 on page 40 for more information on this.

Note

If you do not have an up-to-date version of the Windows Installer, the first thing that the installation process will do is update the Installer. This may require you to restart Windows.

To install the ObjectStore software:

1. Run `setup.exe` from the CDROM or from the directory where you downloaded the installation package.
2. If you have an updated version of InstallShield, you should then acknowledge the License Agreement and ObjectStore Copyright. Click Yes to accept the License Agreement. If you are installing ObjectStore on a machine that does not have an existing version of ObjectStore installed, go to step 9.
3. If you are installing ObjectStore over an existing version, the ObjectStore Setup dialog box asks what you want to do with the existing software:
   - Install ObjectStore 6.1 in a different location from that of the existing software.
   - Uninstall the existing software and install the 6.1 software. This is the default and recommended setting.

   Make your selection and click Next.
4. If you selected to uninstall the existing software, a message is displayed asking for confirmation to delete the existing software. Click Yes to continue.
5. The ObjectStore Setup dialog box asks whether you want to delete the RAWFS file partition. After you have made your choice in the check box, click Next and click OK at the confirmation message.

6 The ObjectStore Setup dialog box asks whether you want to remove information from the Windows registry. Click Yes and click OK at the confirmation message

The uninstall operation deletes the existing software.

7 A message is displayed when the uninstall operation is complete. Click OK to continue. At this point the actual installation begins.

8 The ObjectStore Setup dialog box displays a list of information you have specified. If you do not need to make any changes, click Next.

9 At the Customer Information dialog box, enter your User Name and Company Name which will become part of the registry entries for ObjectStore and click OK.

10 At the ObjectStore Setup dialog box, specify the location where ObjectStore will be installed. You can accept the default location or click Browse to specify a different location. Make sure you have enough disk space available. Click Next.

11 At the ObjectStore Setup dialog box, click Typical or Custom installation.

- Typical installation installs everything except Javlin and DDML (all APIs, all libraries, documentation, and ObjectStore DBMS (server) software).
- Custom installation allows you to pick which ObjectStore components you want to install and whether you are setting up a client-only environment, a server-only environment, or a client and server environment.

If you selected Typical installation, go to step 15.

12 If you selected a Custom installation, the ObjectStore Setup dialog box will be displayed. A check next to a component's name means it will be installed. By default, all the ObjectStore components are checked. Uncheck the components in the left pane that you do not want to install.

- If you have only a C++ application environment, uncheck Java Interface and Javlin and leave DDML checked if your application may require it.
- If you have only a Java application environment, leave the *both* the C++ and Java Interfaces checked and leave Javlin or DDML checked if your application may require it.
- If you have a Java and C++ application environment, you should uncheck Javlin and DDML if your application does not need these components.

Do not check or uncheck anything in the right pane and click Next.

13 The ObjectStore Setup dialog box states that you have chosen to install the ObjectStore DBMS. Depending on the application requirements for the current host, select the ObjectStore client only, the ObjectStore server only, or both client and server. Click Next.

14 If you specified client only, the ObjectStore Setup dialog box asks you to specify the remote directory containing the ObjectStore ADB files. Leave the path information field blank and click Next, and then click Yes at the message that the installation process cannot resolve the ADB file directory path.

After ObjectStore is installed, you need to use the ObjectStore `ossetasp` utility to resolve this path information. See Resolving Client-Only Schema Database Issues on page 44 for a description of how to do this.

15 If you are installing ObjectStore over an existing version, the ObjectStore Setup dialog box asks you to specify how you want to treat the existing software. Click one of the following choices and then click Next.

- Overwrite the existing software
- Rename the directory where the existing software is located
- Install the software in the same directory after deleting all the existing software in the specified directory

16 The ObjectStore Setup dialog box gives you the opportunity to review and change anything that you have specified. If you are satisfied with the settings, click Next.

The installation process installs the specified ObjectStore components.

17 If you specified a client-only installation, the ObjectStore Setup dialog box will ask you if you want to start the ObjectStore cache manager process immediately. If you do not start it immediately, it will be automatically started when an ObjectStore client initializes ObjectStore. It is recommended that you select Yes, because this will make it easier to verify that the software is installed correctly. Click Next.

18 If you specified a server-only or client and server installation, the ObjectStore Setup dialog box will ask if you want to change any ObjectStore server parameters. In general, you do not want to change any of these parameters at installation time. If there are specific parameters that you know you will need to change because of past development or runtime experiences, then you can change them at this time. ObjectStore server parameters can be changed at any time by running Setup at a later time.

Any parameter changes will take effect the next time the ObjectStore server process is started; normally at the end of the installation process. Click Next.

19 If you specified a server-only or client and server installation, the ObjectStore Setup dialog box asks if you want to create RAWFS partitions. If you know that you will need RAWFS partitions, then you can set them up at this time. You can run Setup at a later time to create a RAWFS partition if you decide you want to add one. Click Next.

20 If you specified a server-only or client and server installation, the ObjectStore Setup dialog box asks whether you want to initialize the server log file. If you have never run an ObjectStore server process on this host before, then you should initialize the server log file. This is the same as the ObjectStore transaction log file and it needs to be initialized before you can start up the ObjectStore server process.

Note: If you are upgrading an existing installation that uses a RAWFS under no circumstances should you initialize the server log file.

If you have an existing transaction log file because you are running Setup to change a server parameter, then you should not initialize the server log file because it may have some previous application transaction data that has yet to be propagated to the databases. There are two instances where you might want to initialize the server log file. One is when you are setting up the ObjectStore server process for the first time on this host and the other is when you need to re-create it because it has grown too large due to a long-running transaction. If you decide to re-create the server log file, you should make sure that you have checkpointed the ObjectStore server process to ensure that all transactional data is propagated to the databases prior to re-creating the server log file.

21 If you specified a server-only or client and server installation, and you chose the initialize the server log file, ObjectStore Setup dialog box asks where you want to locate the server log file `osserver.log`. Put this file in a secure place that won't be removed (for example, do not put the file in the `temp` directory). In addition, if you have multiple physical disk drives and need maximum performance, put this file on a different disk from where the physical databases will be placed. This performance optimization fosters faster propagation of the ObjectStore transaction log file. Click Next.

If you specified a server-only or client and server installation, a message box tells you that the server log file has been initialized if you chose to initialize it. Click OK

22 If you specified a server-only or client and server installation, the ObjectStore Setup dialog box asks whether you want the ObjectStore server process to be configured to start automatically when the system is booted. The ObjectStore server process runs as a service. Most users chose to have this service started automatically upon system startup. Even though it is configured to start up automatically, you can always shut down the ObjectStore server process at any time and start it up again manually. Click Next.

23 If you specified a server-only or client and server installation, the ObjectStore Setup dialog box informs you that the ObjectStore server and cache manager processes have been configured to start automatically upon system reboot and the installation process asks you if you want them to be started immediately. Normally, you should choose Yes because this will make it easier to verify that ObjectStore typical, server-only, or client and server installations have been done correctly. Click Next.

24 Finally, the ObjectStore Setup dialog box asks if you want to view the `README.htm` file. Make your choice in the check box and click Finish.

At this point the ObjectStore installation is complete. See Verify Installation on page 32 for information on verifying that the ObjectStore software is properly configured for your environment.

Configuring Visual C++ Developer Studio Release 7.0

If you are using ObjectStore with Visual C++ Developer Studio Release 7, you need to update the Visual Studio environment manually so that knows where to find the ObjectStore `include`, `lib`, and `bin` directories. Here is the procedure:

1 Start up Developer Studio.

2 Open the Tools -> Options dialog box.

3 Scroll down and select the Projects folder.

4 Select VC++ Directories.

5 Select Executable files under Show directories for:.

6 Add `$(OS_ROOTDIR)\bin`.

7 Select Include files under Show directories for:.

8 Add `$(OS_ROOTDIR)\include`.

9 Select Library files under Show directories for:.

10 Add `$(OS_ROOTDIR)\lib`.

11 Click on OK.

12 Shut down Developer Studio.

Changing the ObjectStore Configuration

At any time in the future, you can use the ObjectStore installation program (`%OS_ROOTDIR%\setup.exe`) to do any of the following. However, in order to perform the operation you need to first shut down ObjectStore services. The installation program prompts you to do this.

- Re-create the ObjectStore transaction log file
- Upgrade ObjectStore to a new Service Pack or new release
- Make an ObjectStore server parameter change

To uninstall the ObjectStore Release 6.1 software, see Uninstalling ObjectStore on page 43.

ObjectStore Environment Variables

The ObjectStore installation assigns default values to several environment variables. 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` (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` (if DDML is used)
`OS_ROOTDIR`	Installation directory for ObjectStore (ends with `ostore`)
`OS_TMPDIR`	Location of log files used by ObjectStore services

PATH	Path to ObjectStore executables (services and utilities): %OS_ROOTDIR%\bin %OS_ROOTDIR%\..\osji\bin (if Java Interface used) %OS_ROOTDIR%\..\ddml\bin (if DDML used) %OS_ROOTDIR%\..\javlin\bin (if Javlin used)
CLASSPATH	Path to OSJI and Javlin classes: %OS_ROOTDIR%\..\osji\osji.jar (if Java Interface used) %OS_ROOTDIR%\..\osji\tools.jar (if Java Interface used) %OS_ROOTDIR%\..\javlin\jmtl.jar (if Javlin used) %OS_ROOTDIR%\..\javlin\xerces.jar (if Javlin used)

Verify Installation

After you install the ObjectStore software, you can verify that the installation process properly configured your environment. Select the processes that are appropriate for the type of installation you specified:

- Verify Installation Directory Contents on page 32
- Verify ObjectStore Services on page 34
- Verify a C++ Development Client on page 34
- Verify a C++ Runtime Client on page 34
- Verify a Java Client on page 35
- Configuration Tasks on page 36

Verify Installation Directory Contents

Your installation directory contents will depend on what you have selected to install. Some directories will always be installed. You should verify that the following directories or 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	All utilities and DLLs for an ObjectStore Single environment
ostore\etc	Configuration files
ostore\examples	ObjectStore C++ API code examples
ostore\include	ObjectStore C++ API include files
ostore\lib	all ObjectStore libraries and schema databases
ostore\unsupported	ObjectStore performance measurement tools which are available for customers to use, but are not a supported component of ObjectStore
doc	Documentation in HTML/PDF for all ObjectStore components
osji	This directory is for the ObjectStore interface.
	osji.jar ObjectStore Java interface (OSJI) classes.
	tools.jar OSJI development tools classes.
	jdd.jar
	browser.jar
	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 Persistent Classes in a Transient Manner" in Chapter 13 (Miscellaneous Information) in the *Java API User Guide*.
	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_g.jar
	tools_g.jar
	jdd_g.jar
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.
`javlin`	Installation directory for Javlin/JMTL; contains the `.jar` files for using Javlin/JMTL.
`javlin\bin`	Utilities.
`javlin\examples`	Directory of Javlin/JMTL examples.
`ddml`	Installation directory for the Dynamic Data Modeling Library.

Verify ObjectStore Services

For a typical or client and server installation, you should verify that both the ObjectStore server process and the ObjectStore cache manager process are listed in the collections of services for your machine. The list of services is displayed by selecting Control Panel | Administrative Tools | Services (or Control Panel | Services on Windows NT). You should make sure that you can start and stop these services. By default the services are started automatically if you chose this option during installation.

Verify a C++ Development Client

If you have a development environment on this host with a VC++ compiler, you can build and run the `hello2` example by going to ostore`\examples\hello2` and typing:

```
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.

Verify a C++ Runtime Client

If you set up a C++ runtime environment, you will not be able to build a C++ application, but simply run one on this host. A simple way to verify that a

C++ runtime client is properly installed is to run one of the utilities that come with ObjectStore. In the verification procedure shown below, the ObjectStore `ossize` utility is used.

1. In a default installation, the necessary environment variables are set correctly. However, you may 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 31.
2. Run the `ossize` utility that comes with ObjectStore in the `ostore\lib` directory on an ObjectStore database. You can use any of the library schema databases found in the ostore\lib directory. These are the `*.adb`, `*.db`, or `*.ldb` files.

   The actual location of these database files depends on which host the ObjectStore server process is running and if the ObjectStore server process has been configured for RAWFS database access. If the host for the ObjectStore server process has been configured for RAWFS databases and the library schema databases are in the default RAWFS for this host, then you can verify the client on this host (whether a UNIX or Windows host) as follows:

```
ossize <RemoteHostName>::/ostore/schema/6.1/metaschm.db
```

If the host with the ObjectStore server process running has a different location for the library schemas, then you will need to 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 the remote host that is UNIX assuming 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 the remote host that is Windows assuming ObjectStore has been installed on C:\:

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

Verify a Java Client

To test the installation, 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 `osserver` 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 `osserver`.

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
```

Configuration Tasks

Changes to the Transaction Log

There will be times when the ObjectStore transaction log file needs to be reallocated. This reallocation may be required because the file has grown very large due to a long transaction. The ObjectStore transaction log file always grows, but will never shrink in size. Another reason for possible reallocation of the ObjectStore transaction log file is to move it to another physical disk.

Caution

Under no circumstances should you reallocate the ObjectStore transaction log file on a machine where a RAWFS is used.

In order to reallocate the ObjectStore transaction log you must do the following:

1. Make sure that all ObjectStore clients are shut down
2. Perform a checkpointing of the ObjectStore server process to ensure that all committed transactions in the current transaction log file are propagated to the database(s).

   ```
   ossvrchkpt <hostname>
   ```

3. Shut down the ObjectStore Server process which is running as a service

4 Select: Start Menu -> Programs -> ObjectStore Win32 -> ObjectStore Setup

5 Select ObjectStore Setup (this runs the `Installation/Setup` executable)

6 Select No - leave ObjectStore Server parameters as they are

7 Select No - do not create RAWFS partitions

8 Select Yes - reinitialize the server log file

9 Enter the full path to the ObjectStore transaction log file. This file must be on a local disk that has adequate room for the file if it may grow to a large size due to large transactions. It is also recommended that this file be on a disk that is different than the disk or disks containing the ObjectStore databases being accessed by this ObjectStore Server process.

10 If the path you entered is the same as the current transaction log file, you will be asked if you want to overwrite it. Select Yes.

11 Select Yes or No regarding automatic ObjectStore Server startup, depending on how you have done it in the past.

12 The `Installation/Setup` executable exits and you should now start up the ObjectStore Server process as a service

Changes to RAWFS

You can change what RAWFS resources (files or disk partitions) are managed by the ObjectStore server process. This may include adding more RAWFS space available to the ObjectStore server process. If you chose to eliminate a part of the RAWFS (shrink it), then you must back up all of the databases in the RAWFS as you will need to reinitialize you entire RAWFS. The reason for this is because the RAWFS available to the ObjectStore server process is essentially a set of files or disk partitions that contain file space for use by the ObjectStore server process. You have no control over which files or partitions contain specific databases. You can add more RAWFS files or partitions to the RAWFS used by the ObjectStore server, but taking away RAWFS space requires a complete backup, reinitialize and restore of the databases.

In order to add more RAWFS space for use by your ObjectStore server process you should do the following:

1 Make sure that all ObjectStore clients are shut down

2 Perform a checkpointing of the ObjectStore server process to ensure that all committed transactions in the current transaction log file are propagated to the database(s).

```
ossvrchkpt <hostname>
```

3 Shut down the ObjectStore server process which is running as a service

4 Select: Start Menu -> Programs -> ObjectStore Win32 -> ObjectStore Setup

5 Select ObjectStore Setup (this runs the `Installation/Setup` executable)

6 Select No - leave ObjectStore Server parameters as they are

7 Select Yes - create RAWFS partitions

You will now have a Window for Setting Up Server Partitions. The existing RAWFS partitions (if any) will be displayed.

8 Select New to create a add RAWFS capacity for your ObjectStore server process

9 You must now select a RAWFS partition type. If you select File System File, you will be required to give a path and size for this file. You can specify whether this file will be expandable. This will create a file that can be used for the ObjectStore RAWFS space. If you select a Disk Partition, then you should have reserved a disk partition for use exclusively for the RAWFS. The size of the newly created RAWFS partition will be whatever the Disk Partition size is. If you select an Entire Disk Drive, then the entire disk drive will be used exclusively for the ObjectStore Server process' RAWFS. The size of this RAWFS partition will be the size of the disk drive.

10 Continue the dialog for creating the type of RAWFS partition you want to create and click Done when you are done with the changes.

11 Select No to use the existing log as it is.

12 Select Yes or No regarding automatic ObjectStore server startup, depending on how you have done it in the past

13 The `Installation/Setup` executable exits and you should now start up the ObjectStore server process as a service

Auto-starting ObjectStore Services

The ObjectStore Services that can be started automatically upon system startup are the ObjectStore Server process and the ObjectStore cache manager process. You are asked during initial installation if you want them to be started up automatically. If you want to change this, you can do this as follows:

1 Select: Start Menu -> Programs -> ObjectStore Win32 -> ObjectStore Setup

2 Select ObjectStore Setup (this runs the `Installation/Setup` executable)

3 Select No - leave ObjectStore Server parameters as they are

4 Select No - do not create RAWFS partitions

5 Select No to use the existing log as it is.

6 Select Yes or No regarding automatic ObjectStore server startup, depending on how you want it to be

7 The `Installation/Setup` executable exits

If the ObjectStore server process is not running and you want to start it, do so via Services.

Modifying ObjectStore Server Parameters

There are a number of ObjectStore server parameters that may need to be set in your ObjectStore environment. A complete description of all of the ObjectStore server parameters can be found in the *Managing ObjectStore* book. You can modify these parameters with an ObjectStore server process running. They will not take affect until the ObjectStore server process is restarted.

Here is how you should add or modify any ObjectStore server parameters.

1 Select: Start Menu -> Programs -> ObjectStore Win32 -> ObjectStore Setup

2 Select ObjectStore Setup (this runs the `Installation/Setup` executable)

3 Select Yes - change ObjectStore Server parameters

You will now be in the Customize Server Startup Parameters dialog box Where you can view and edit ObjectStore server parameters. After you have finished editing, click OK.

4 Select No - do not create RAWFS partitions

5 Select No to use the existing log as it is.

6 Select Yes or No regarding automatic ObjectStore server startup, depending on how you have done it in the past

7 The `Installation/Setup` executable exits.

Any modifications that have been made to ObjectStore server parameters will not be used until the ObjectStore Server process is restarted.

Upgrading from ObjectStore 6.0

ObjectStore C++ databases created with 6.0 are compatible with this release provided that the same C++ compiler is used (Visual C++ 6.0). Existing 6.0 ObjectStore databases can be accessed by applications that are either built

with ObjectStore Release 6.0 or Release 6.1 provided that the same C++ compiler is used.

Before you upgrade your ObjectStore installation of 6.1, it is recommended that you read the `README.htm` and the *Release Notes* to make sure that there are no source code changes that must be made to the application prior to the upgrade. Ensure that the desired platform configuration is supported for this release.

Follow these steps to upgrade from ObjectStore 6.0 to ObjectStore 6.1:

1. Back up all databases, application files, and the ObjectStore installation directory.
2. Shut down all ObjectStore applications or services that are running on or accessing the host targeted for installation. The ObjectStore cache manager process must also be shut down via Services Control Panel if it is running.
3. Once you have insured that no applications are accessing the ObjectStore server process, then checkpoint the ObjectStore server process to make sure that all data is propagated from the ObjectStore transaction log to the application databases. You can do this as follows:

   ```
   ossvrchkpt <hostname>
   ```

4. Shut down the ObjectStore server process via the Services Control Panel.
5. Follow the Installation Procedure on page 25.
6. Perform the verification as described in Verify Installation on page 32.
7. Lastly, you should verify that your application runs as expected after the upgrade.

Upgrading from Release 5.1

Besides the new features and functionality in ObjectStore 6.1, the internal format of the database has changed. What this means is that 5.1 ObjectStore clients and 6.1 ObjectStore clients cannot access the same database. In order for a 6.1 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`.

This type of upgrade can be summarized with the following steps. You should consult the *Migration Guide*

(`www.objectstore.net/documentation/migration`.) for detailed instructions on upgrading this release of ObjectStore.

1. Install the 5.1 `osdump` utility that is compatible with ObjectStore 6.1
2. Install ObjectStore 6.1 in a different directory from 5.1. See Installation Procedure on page 25.
3. If the application is written in C++, read the *Migration Guide* and migrate your application accordingly
4. Completely rebuild your application (C++ migrated code or Java code) with ObjectStore 6.1 and using an ObjectStore 6.1 server process
5. Back up your 5.1 application and database(s)
6. Dump your ObjectStore 5.1 database(s) using the 5.1 `osdump` utility from step 1 using the ObjectStore 5.1 server process
7. Load your database(s) using the 6.1 `osload` utility
8. Test your 6.1 application completely
9. Uninstall ObjectStore 5.1

Chapter 4
Miscellaneous Information

Uninstalling ObjectStore

1. Shutdown all ObjectStore client applications that are running.
2. If you need to back up any databases, do so now.
3. Shutdown all ObjectStore Services that are running.
4. To uninstall ObjectStore, go to the Start Menu and select ObjectStore Win32 | ObjectStore Setup.
5. Setup now detects that ObjectStore is running and installed and you are asked if you want to Setup, Reinstall, or Uninstall ObjectStore.
6. Select Uninstall. The process will remove some files and modify the Windows registry so you must acknowledge if that is what you really want to do. If you click Yes, setup asks you once again if you want to delete the contents of any RAWFS partitions. If you think that you may want to save any of this data, it is recommended that you back up these databases in the RAWFS partitions prior to the uninstallation as they will not be accessible after ObjectStore is uninstalled.
7. You are asked one final time if you want to really do the removal. If you click Yes, setup will stop all ObjectStore services and completely uninstall ObjectStore.

Adding ObjectStore Components

You can add ObjectStore 6.1 components to an existing ObjectStore 6.1 installation using the ObjectStore CDROM. Follow this procedure to add a component:

1 Mount the CDROM containing the ObjectStore installation package.

2 Open a command window and change your working directory to the sub-directory on the CDROM containing the component you wish to install. For example:

```
cd javlin
```

3 From the command window, run the `setup.exe` application found in the sub-directory containing the component you want to install. When running `setup.exe`, specify the pathname where you want to install the component. For example:

```
setup.exe /z"path=\"c:\odi\javlin"\"
```

This will install the component in the current working directory to the location specified.

Resolving Client-Only Schema Database Issues

If you create a client-only ObjectStore installation and did not specify the path information to the remote directory containing the ObjectStore schema database files, you need to manually add this information to several of the ObjectStore utilities and DLLs. (See Installation Procedure on page 25 for the installation step where this occurs.) The utilities and DLLs for which you need to add this information are:

- `o6alloc.dll`
- `o6cmpt1.dll`
- `o6query1.dll`
- `o6ossevol1.dll`
- `o6_coll1.dll`

To add this schema database path information, you use the ObjectStore utility `ossetasp` as follows:

1 For each of the above files, run the `ossetasp` utility with the -p option to obtain the name of the associated schema database. For example:

```
ossetasp -p o6alloc.dll
```

You will get back the name of the associated schema database file.

2 For each of the above files, run the `ossetasp` utility, passing in the name of the file and the name of the associated schema database returned from step 1 (*`schema_database_name`* in this example). For example,

```
ossetasp o6alloc.dll schema_database_name
```

Troubleshooting Installation Problems

Sometimes on Windows NT 4.0 SP6a, you may get the following pop-up error message during installation:

This error occurs when there is a problem in the Windows Install Service. In order to correct this problem you should execute `InstMsiW.exe` which is found with the ObjectStore installation files in the same directory as `setup.exe`.

After you run the `InstMsiW.exe` executable, you should be able to install ObjectStore.