Release Notes For Versant Object Database

Release 7.0.1.4

VERSANT

Versant History, Innovating for Excellence

In 1988, Versant's visionaries began building solutions based on a highly scalable and distributed object-oriented architecture and a patented caching algorithm that proved to be prescient.

Versant's initial flagship product, the Versant Object Database Management System (ODBMS), was viewed by the industry as the one truly enterprise-scalable object database.

Leading telecommunications, financial services, defense and transportation companies have all depended on Versant to solve some of the most complex data management applications in the world. Applications such as fraud detection, risk analysis, simulation, yield management and real-time data collection and analysis have benefited from Versant's unique object-oriented architecture.

For more Information please visit **www.versant.com**

Versant US

Versant Corporation

255 Shoreline Drive, Suite 450, Redwood City, CA 94065

Ph +1 650-232-2400, Fx +1 650-232-2401

Versant Europe

Versant GmbH

Wiesenkamp 22b, 22359 Hamburg, Germany

Ph +49.40.60990-0, Fx +49.40.60990-113

Table of Contents

CHAPTER 1

System Description

This Chapter gives a brief overview of the Versant Object Database.

The Chapter covers the following:

- Versant Object Database Overview
- Versant Object Database Composition
- Versant Object Database Feature List 7.0
- Changes and Improvements in 7.0.1.1
- Changes and Improvements in 7.0.1.2
- Changes and Improvements in 7.0.1.3
- Changes and Improvements in 7.0.1.4
- Bugs Fixed since 7.0.1.1
- Bugs Fixed since 7.0.1.2
- Bugs Fixed since 7.0.1.3
- 7.0.1.4 Supported Platforms
- Compatibility Issues
- General Restrictions and Suggestions

VERSANT OBJECT DATABASE OVERVIEW

The Versant Object Database is an Object Database Management System (ODBMS). It has been designed to ease development and enhance performance in complex, distributed, heterogeneous environments. It is very useful where applications are written in Java and/or C++.

Versant Object Database bridges the complexity of developing object-oriented applications by offering transparent access from the application code, to the distributed transaction management of the data. It provides optimistic locking, balanced client-side and server-side caching and direct object navigation.

VERSANT OBJECT DATABASE COMPOSITION

Versant Object Database 7.0.1.4 consists of the following:

Standard components:

COMPONENT	DETAILS
Versant ODBMS	Versant Object Database Management System
GUI Tools	Versant GUI Tools
JDO Interface	JDO Interface for Versant ODBMS
Java Versant Interface	Java Language Interface for Versant ODBMS
C and C++/Versant	C and C++ Language Interface for Versant ODBMS

Add-on components:

COMPONENT	DETAILS
VAR	Versant Asynchronous Replication for Versant ODBMS.
Vedding	Fault Tolerant Versant ODBMS Server (FTS)
HABACKUP	Backup Solution for use with High Availability Server
Vorkout	Versant Online Database Reorganization Tool (vcompactdb)
ReVind	Structured Query Language Interface for Versant ODBMS(VSQL)
VODBC	Versant ODBC driver
Warm Standby	Used for an Incremental Rollforward recovery
Vitness	Database Monitoring Tool - Vitness comes with an extra installer

NOTE:- All the Versant Add-on components require a separate licence.

VERSANT OBJECT DATABASE FEATURE LIST 7.0

Query Enhancements

A New Query Engine has been introduced in Versant 7.0, for increased efficiency and improved performance.

From 7.0, querying has been made simple and transparent to the user. For this, New Query APIs are introduced, and some existing APIs are enhanced with complete backward compatibility. In addition, the Versant Query Language is extended to support several new functionalities.

Versant Object Database 7.0 now supports the following main features:

Complex Expressions in Query

Support for attributes on LHS & RHS

```
age > spouse.age
30 > age
age <= 30
```

Support for mathematical operations

```
salary * 1.2 > 100000
salary >= age * 1000
salary + spouse.salary <= 200000
```

Support for toupper() / tolower()

The following query methods are introduced that can be used in the query predicate with attributes of string type:

```
toupper(city) = "SAN FRANCISCO"
tolower(name) like "mar*"
```

Support for o_list

Support for IN operator

```
age IN {16, 18, 19, 21, 30, 40, 50, 60, 65}
“chuck” IN nicknames
```

Support for EXISTS & FOR ALL operators

```
EXISTS child IN children: (child.age > 18)
FOR ALL child IN children: (child.age > 18)
```

Server-side sorting

```
ORDER BY name ASC, age DESC
```

SET Operators

Support for SET operators on multi-valued attributes (except for strings type).

Improved indexing capabilities

Btree indexing for multi-valued attribute of any Versant elementary type (except for string type and 1b/u1b type).

Improved stability of cursor query

New Query processing support and cursor changes.

Querying on Candidate Collection

Instead of querying on an entire class, it will now be possible to restrict the query to only a subset of objects from the class.

Security Enhancements

Independence from OS passwords for DBA's

Additional password check for authentication of DBA's**.**

Utilities supporting password

DBA authentication for utilities introduced.

Support for External User Authentication (plugins)

Third party software (plug-ins) can be used to authenticate users.

Reliability, Availability & Supportability Enhancements

Utility Enhancements

vbackup

- New API to determine the progress of Backup/Restore
- The backup media now contains the server profile which can be extracted using the new vbackup options:

  ```
  vbackup –list -getprofile
  ```

dbtool

- New look and feel for `dbtool`
- Component wise classification of options
- New options:
  - Prints information about schema evolution
  - Prints Loids for a particular dbid

- Replication related operations

removedb

- Now prompts for a confirmation before deleting the database.

makedb

- Options `-beprofile` and `-feprofile` which allow copying of server/application profile files from a different location.

Reliability Enhancements

- Versant can now be used with Windows Terminal Server.
- Indexes with key size greater than 2048 bytes, will return an error and avoid a database crash.
- Enhancement to the Versant database server on Windows to avoid collisions with third party products (such as Novell or Windows XP SP2)

Vedding(FTS) Enhancements

Network partitioning Detection and Avoidance

- Enhancements to `setreplicareporting`
- Behavior changes to `COMMIT` and `ROLLBACK`

Inducing a fail-over for maintenance

Earlier Versant did not provide any tool or API to break the replication once it is set. From 7.0 release, the ftstool utility supports two new options "`-disable`" and "`-forcedisable`" to break replication.

In order to restart the down database and re-sync a new option "`-enable`" will also be provided to the `ftstool` utility. API `o_ftstool` will also support the new options.

Getting advanced status of Vedding

`Ftstool` utility has a new option `–status` that will display the state of the database.
In addition, `polling` process forked /requested will also be displayed for the database.

HABACKUP now Vedding(FTS) Compatible

Vedding re-synchronization is now possible from the database backed up by HABACKUP.

HABACKUP supports Vedding's on-line restore solution which allows re-synchronization of failed replica databases through combinations of on-line HABACKUP restore and polling resynchronization. This solution improves the performance of re-synchronization process significantly if one database of the database pair was down for a prolonged period of time.

For more information please refer to the *Versant High Availability Backup Usage Manual.*

XA Enhancements

- Support for multiple database connections
- Support for transaction time-out by TM
- Support for both local and XA transactions using the same database connection
- Improved error logging

Performance Enhancements

dropinst/dropcls: reverse scalability

Enhanced performance of dropinst, dropclass operations when cardinality is low and database size is high.

B-tree Improvements

Enhanced performance for B-tree maintenance with respect to online compaction of B-trees and issues with cursor position in queries.

B-tree Index Defragmentation

Over time the B-Tree index can get fragmented due to object insertions and deletions.

An online compaction can take place on the B-Tree, if the index has a large number of non-unique keys which spans across multiple pages and is fragmented.

The compaction is designed to happen as part of the B-Tree RID delete when the user object is deleted or updated causing B-Tree maintenance. There is no need to recreate existing B-Tree indices.

Buffer Latch Improvements

Improved fairness in scheduling for reader vs. writer latches.

Scalability improvements

Scalability issues for readers

- No wait times in case of parallel reads.
- Optimistic locking: lock latch contention reduction

Database server scalable

Multi Region Heap (For Windows Platform only)

Versant object database server has a new feature called "Multi region heap". Versant utilizes memory-mapped files to create an internal heap that is used for communication and data sharing between the server processes. In pre-7.0 releases, all these memory-mapped files were loaded in a single region of memory within the process virtual address space and the start of this region was a fixed address. This would lead to conflicts with other third party DLL's that were loaded by the Operating system in the same memory region that was used by the database server to create its heap.

The "Multi region heap" feature allows Versant to address this issue by splitting its heap across multiple regions in the virtual address space of the server process.

Another advantage of this feature is that it allows for the creation of a much larger heap than what was possible earlier.

The "Multi region heap" feature is scalable when 3GB large memory addressing is configured on the system.

For more details on how to do 3GB configuration, please refer to Microsoft Help and Support.

The “Multi region heap” feature can be controlled by a parameter “`heap_multi_region`” in the server process profile file.

For more information please refer to the Chapter “Database Profiles” in the *Database Administration Manual*.

Interface Enhancements

C++ Interface

New ANSI compliant Cxx libraries

Starting with Versant Object Database 7.0, Versant will ship the ANSI compliant Cxx libraries separately. They are available in the directory `<software_root>/lib/ansi` of your installation. These libraries are compliant to the ANSI C++ Standard (ISO/IEC Standard 14882:1998.) guidelines.

Introduction of new query classes to support the new query engine

- Improved usability and functionality
- New class "`VQueryResult`" to obtain results of query execution
- Supports the following features:
  - Complex Expressions in Query
  - Server-side sorting
  - SET operators
  - Improved indexing capabilities
  - Querying on Candidate Collection

Please refer to the Query section above for more details.

Security Enhancements

- DBA authentication for embedded utility APIs

Vedding(FTS) Improvements

- Network partitioning Detection and Avoidance
- Getting advanced status of Vedding

Java Versant Interface (JVI)

64-bit JVI Support

JVI is built with J2SE1.4.

As J2SE1.4 supports loading of 64-bit libraries, now JVI libraries are supported for 64-bit Versant ODBMS.

Query Processing Enhancements

Introduction of new query classes to support the new query engine.

Security Enhancements

To provide DBA level security for a database, JVI enhanced all embedded utility APIs to create a password-protected database and also to accept DBA password for any operations on such password-protected database.

Vedding(FTS) Improvements

- Network partitioning Detection and Avoidance: New APIs at fundamental layer by which user applications can avoid network partitioning and enable replica error reporting at commit time.
- Getting the proper status of Vedding: New APIs to `DBUtility` class to display the state of the database. In addition, polling process forked /requested will also be displayed for the database.

Embedded vbackup supported

- New classes introduced to the `com.versant.util.backup` package to provide embedded vbackup utility support.
- New API to get the status of backup/restore progress.

New API's to improve performance and correctness of JVI

- Two new APIs "`getNotFoundObjects()`", and "`getTimeStampFailedObjects()`" are added to Fundamental layer at class VException.
- Two new APIs "`getSessionOfCurrentThread()`" and "`isDirty()`" are added to Transparent layer at class TransSession
- Performance of `groupReadObjects()` API has been improved.

C/Versant data type o_list supported

JVI supports the C/Versant data type, `o_list` that has the repetition factor as -2.

JDO Interface

Key features of Versant JDO Interface:

- Versant's JDO interface is a high performance persistence engine for the Versant Object Database. It is a standard compliant implementation of the JDO specification.
  This interface has been developed from the ground up using 100% in Java so it can run anywhere Java can.
- Versant JDO Interface implements all the required features and most of the optional features of version 1.0.1 of the JDO specification.
- It implements all the JDO APIs and supports all Java 2 collection interfaces and classes. It also supports non-transactional read and write, retain values, optimistic transactions and transient transactional JDO optional features.
- The Versant JDO Interface also has a Workbench which is a powerful IDE for monitoring Versant JDO Interface servers and running JDOQL queries.
- Conversion Tool for VOA 3.2.x customers

  With this tool, the VOA 3.2.x databases will have proper inheritance hierarchy. Before invoking this tool, please make sure you have used `convertdb` utility to convert VOA 3.2 x databases to VOD 7.0 x databases.

  The usage for this tool is as follows:

  ```
  <software_root>\bin\updatevoa32schema.bat
  <software_root>/bin/updatevoa32schema.sh
  ```

  or

  ```
  java com.versant.core.vds.tools.jdo.UpdateVOA32Schema [options]
  ```

  Options are:

  `-cp|-classpath`

  Specifies classpath to load domain classes

  `-p|-properties`

  Specifies jdo configuration file. Defaults to "versant.properties". Here, -p is a mandatory option.

`-h|-?|-help`

Prints this help message

GUI Tools

Object Inspector Enhancements

- Progress bar to map the progress of various operations, which take considerable time.
- Performance and cache coherency: Improved performance by maintaining an application level cache.
- Object graph management: Accessing and viewing all objects referenced from specific object for a certain level or for all levels.
- Root object management: Now allows the users to view the root objects of the database and set/unset root objects to be regular objects.
- Class locator: New Class Locator helps the user to search a class in the database schema.
- Object locator: New Object Locator helps the user to locate a particular class instance with respect to its LOID(s) and opens and Instance Inspector to inspect it.
- Query Builder enhancements: Added Key value options Null value, Ignore case and No option, to run a query for objects with null attribute values and on string attribute values without the need to specify the key value using correct letter case.
- VAT support: Added two main capabilities which can be used as guidelines for VAT Query support
  - Use existing defined VAT Indexes
  - Define VAT on the fly to allow queries on this VAT's.

Security enhancements

To provide DBA level security for a database, GUI interface has been enhanced to create a password-protected database and also to accept DBA password for any operations on such password-protected database.

C/Versant data type o_list supported

GUI supports the C/Versant data type, o_list that has the repetition factor as -2.

Administration Console Enhancements

- Backup/restore wizard

  The newly added Database Backup and Restore Wizard guides you through a step by step process to define and set the parameters for the database backup and restore process.

- Schema management enhancements

  A new Load Schema dialog box is provided which allows defining database schema for a selected database in many ways.

CHANGES AND IMPROVEMENTS IN VERSANT OBJECT DATABASE 7.0.1.1

New methods added in Dictionary class

Following methods have been added in the Dictionary class:

`VEEDictionary::release()`

`VEIDictionary::release()`

`VIIDictionary::release()`

`VVIDictionary::release()`

`VVVDictionary::release()`

`VVEDictionary::release()`

`VIEDictionary::release()`

`VEEDictionary::get_key_vstr()`

`VIEDictionary::get_key_vstr()`

`VEIDictionary::remove()`

For more information, refer to *C++ Reference Manual*, "Chapter 14: Collection Classes".

New variable VERSANT_SERVER_PORTS introduced

In order to the fix the problems related to VERSANT security and limitations related to `obe_port_begin` and `obe_port_end` for utilities which are not attached to DB, `VERSANT_SERVER_PORTS` variable is introduced.

For more information, refer to *Database Administration Manual*, "Chapter 2: Tuning Parameters".

New Versant JDO Interface introduced

Versant JDO Interface is a high performance persistence engine for Versant Object Database. Versant JDO Interface implements all the required features and most of the optional features of version 1.0.1 of the JDO specification.

For more information, refer to *Versant JDO Interface Usage Manual*.

Deprecated Features

- `IS_EMPTY` and `NOT_IS_EMPTY` set operators not supported in VQL 7.0.
- Server Process profile parameters - `obe_port_begin` and `obe_port_end` are deprecated and replaced with an environment variable `VERSANT_SERVER_PORTS`.

CHANGES AND IMPROVEMENTS IN VERSANT OBJECT DATABASE 7.0.1.2

Warm Standby

Warm Standby (Incremental Restore) is introduced as part of the vbackup utility to strengthen Versant's high availability capabilities.

This feature is an extension to the rollforward recovery. It is designed to minimize the downtime in an emergency event, which requires a database recovery.

This utility requires a separate license.

For more information, please refer to the *Warm Standby Usage Guide* and *Versant Database Administration manual*, "Chapter 3: Database utility-Vbackup".

Startup Script

It is now possible to invoke customized scripts at database startup time. To facilitate this, a new server profile parameter called "startup_script" has been introduced. This feature can be used by the DBA to assist in Database Management.

For more information please refer to the parameter "startup_script" in "Chapter 2: Database Profiles" in the *Versant Database Administration Manual*.

Vedding(FTS) Parallel Write

In a normal Vedding scenario, all "Write" operations are done one after the other, on the primary and replica database. In order to improve the "Write" operation performance, it is now possible to write on both the databases in parallel, rather than one after the other.

This behavior can be controlled by using the new application process profile parameter called "parallel_write".

For more information please refer to the parameter "parallel_write" in "Chapter 2: Database Profiles" in the *Versant Database Administration Manual*.

Deferred Delete

Prior to 7.0.1.2 release, object deletions were immediate. This caused problems in transaction rollback where unique indices were present. To fix these types of problems, it is now possible to defer the deletion till transaction commit time.

A new server profile parameter called "commit_delete" is introduced. It specifies whether the physical deletion of objects should be delayed till commit. If "commit_delete" is enabled, the objects are physically deleted at commit time. This feature is only applicable for non-schema objects i.e., normal objects.

For more information please refer to the parameter "commit_delete" in "Chapter 2: Database Profiles" in the *Versant Database Administration Manual.*

Trace File Version

The trace file version has been upgraded in this release. The older trace files will be incompatible with the utilities in this release. If an older database is started using the new 7.0.1.2 release, the old trace file will be automatically renamed and a new one created. The renamed trace file will be of the form ".systrace_<day>_<month>_<date>_<year> and can be examined using a pre-7.0.1.2 dbtool utility.

Eg. File created on 27th April will be renamed as "`.systrace_Thu_Apr_27_2006`".

The renamed trace file cannot be read by 7.0.1.2 dbtool utility and an attempt to do so will result into a version mismatch error. It is safe to delete the old trace file since Versant Object Database will no longer use it.

Hash Index Improvement

Improvements have been made in the areas of hash index maintenance and access. Index fragmentation is now reduced thereby improving retrieval time for hash index access.

This improvement will not affect the fragmentation present in existing databases with hash indices. However, any new insertions/deletions will use the new algorithm thus reducing the fragmentation.

JVI JCA Adapter

The JVI JCA Connector architecture provides the functionality by which the Versant enterprise application can plug into application servers.

The VOD product is packaged with the JVI JCA resource adapters vodLOCAL.rar , vodNOTX.rar and vodXA.rar that support all the three types of transactions i.e Local Transaction , No Transaction and XA Transaction specified in the JCA specification.

For more information please refer to "Chapter 7: Java Connector Architecture (JCA)" in the *Java Versant Interface Usage Manual*.

JDO 2 Fetch Plan

The Versant JDO Interface Fetch Group functionality is upgraded to support JDO 2 style Fetch Plan that enhances user control on the field fetching behavior of many JDO APIs. A fetch plan defines rules for instantiating the loaded state for an object graph. It specifies fields to be loaded for all of the instances in the graph. A fetch plan can be associated with a PersistenceManager and independently, with a Query and with an Extent.

For more information, please refer to the Fetch Groups section in "Chapter 11: Cache Management" in the *Versant JDO Interface Usage Manual.*

Index Definition in JDO Metadata File

Automatic index maintenance is now supported by Versant JDO Interface.

Supported values of the indexed attribute are:

- true: generate a b-tree index on the attribute.
- false: do not generate any index on the attribute.
- unique: generate a unique b-tree index on the attribute.
- hash: generate a hash index on the attribute.
- hash-unique: generate a unique hash index on the attribute.

For more information, please refer to "Chapter 21: Meta Data Extension Reference" in the *Versant JDO Interface Usage Manual*.

Lazy Loading of Collection Elements in JDO

The deferred loading of second-class object (SCO) collection elements in Versant JDO Interface, or lazy loading, is a mechanism that delays reading the collection element or the element contents until the element or contents are actually needed. Initializing lazy loading for a collection/map defers the fetch operation of the SCO's data until the first method is executed that requires data. This allows more control over when and what data to fetch.

For more information, please refer to the Deferred Loading of Collection Elements section in "Chapter 8: Map, Collection and Array Mapping" in the *Versant JDO Interface Usage Manual.*

New Procedure to Install and Update all Eclipse Plugins

Eclipse plugins for Versant JDO Interface and Java Versant Interface now follow the normal Eclipse software update procedures (from the Eclipse Help | Software Updates | Find and Install... menu).

For more information, please refer to “Chapter 4: Eclipse Plugins” in the Versant JDO Interface Usage Manual.

Deprecated Features

- Nested transactions are not supported and will be deprecated in future releases.

CHANGES AND IMPROVEMENTS IN VERSANT OBJECT DATABASE 7.0.1.3

System Table Enhancement

Versant uses an internal hash table to map object loids to their physical locations within the data volumes. For large databases, the directory of this hash table can grow significantly large and can have an impact on the buffer cache and log manager thereby affecting the performance of the database server.

A new directory structure has been introduced in Release 7.0.1.3 to optimize the memory utilization by the directory pages. With this new structure there is improved memory usage for this hash table within the buffer cache and some performance improvements to the log manager as well thereby improving the database system performance.

All new databases created with Release 7.0.1.3 and beyond will be created with the new directory structure for this hash table. Databases that were created before Release 7.0.1.3 will continue to work with the old hash directory structure. For these databases., you can upgrade the version of the hash table explicitly by using "dbtool -AT -version -update" option. The old hash table version was "3" and the new version introduced in this release is "4".

For more information, please refer to the dbtool utility in *Database Administration Manual*.

Vorkout Improvements

From 7.0.1.3, Versant has improved the Vorkout utility, to pack the objects in better way than before. It will also have improved buffer management

Changes have been done to improve the handling of large objects (known internally as medium objects).

Starting from 7.0.1.3.3, Vorkout has been enhanced to be compatible with Vedding. In case Vedding is used, Vorkout will not replicate its changes to the replica database (if any). It will work on one and only one database that is provided on the command line.

Starting with 7.0.1.3.5 a new implementation of vorkout offers a faster and better compaction algorithm. This new vcompactdb considerably reduces network overhead, avoids unnecessary index maintenance and does logging optimization besides other optimizations. The new implementation also provides a progress information about the ongoing compaction.

Along with the improved compaction tool, Versant has also introduced a new type of class fragmentation analysis tool. This new implementation uses random sampling to check for unused space and then extrapolates this result over the entire class. This results in a much faster space analysis options than the default one which takes time because it examines all the allo-

cated pages to the class. The sampling size can be tuned by the user but the default should be good enough for most cases.

For more information, please refer to the *Vorkout Usage Manual*.

Automatic Add Volume

The automatic add volume feature allows the DBA to automate the addition of the new data volumes at run time.

Using this feature, the DBA can specify details about the location (when and where) of the new volumes to be added. This feature removes the necessity to constantly monitor the free space on the database for the production database systems and adds a volume manually.

For more information, please refer to the new server process profile parameter "auto_addvol" in the *Database Administration Manual*.

JDO 2.0 persistence

Starting with 7.0.1.3 the Versant JDO Interface implements the JDO 2.0 specification.

Please refer to the section “Limitations in Versant JDO Interface” for more detail.

64 bit support on x86_64 Architecture for Linux

Starting 7.0.1.3, Versant is supported on x86_64 architecture with Linux operating system version RHEL 4.0 and compiler version gcc 3.4.3.

The product works with the following machine configurations:

- AMD64 machine with Opteron processor on which the product is built.
- EM64T hardware with the Xeon processor on which the product has been certified.

For more information, please refer to “Platform Notes for Linux” on page 87.

64 bit support on x86_64 Architecture for Windows

Starting 7.0.1.3.7, Versant is supported on x86_64 architecture for Windows

operating system with Microsoft Visual Studio 2005, the compiler version 8.0 and Microsoft .NET framework 2.0.

NOTE:- Visual Studio 2005 is strict ANSI compliance, so we have not shipped any non-ansi libraries with this release.

The product works with the following machine configurations:

- EM64T machine with Opteron processor and OS Windows 2003 Server Standard x64 edition with Service Pack 2, on which the product is built.
- AMD64 hardware with the Xeon processor on which the product has been certified.

For more information, please refer to "Platform Notes for Windows" on page 71.

Read-Only Database

In 7.0.1.3, Versant introduces a new database mode called read-only mode.

This mode restricts all the writes or updates to the database volumes (log volumes as well as data volumes). This restriction is applicable even to the DBA.

A database can be made read-only for different reasons. One of the reasons could be using a database for pure query processing purposes. Making the database read-only guarantees that nothing can be modified in the database even by accident and that the data will never change till the mode is read-only.

A read-only Versant database can also be written to a CD/DVD and be directly accessed from there. This functionality was not supported in earlier releases.

For more information, please refer to the dbinfo utility in the *Database Administration Manual*.

Error Tracing Enhancements

In 7.0.1.3, improvements have been made to the output of error messages. From now, some error messages contain additional information about the problem cause, e.g. the loid of an object, that couldn't be found or the name of a class in case of a bad class name.

The following are the affected error messages:

Error Number	Error Code
E133	VSL_FILE_OPEN
E1009	SM_E_KEY_NOT_FOUND
E2725	SM_E_DUPLICATEKEY
E2731	SM_E_KEYALREADYEXISTS
E2762	SM_E_INVALIDPID
E2902	SM_LOCK_DEADLOCK
E2903	SM_LOCK_TIMEDOUT

E2995	SM_E_SAMEKEYFOUND
E5006	OB_NO_SUCH_OBJECT
E6002	SCH_CLASS_UNDEFINED
E6004	SCH_BAD_SUPERCLASS
E6005	SCH_BAD_ATTRNAME
E6007	SCH_BAD_DOMAIN
E6012	SCH_BAD_CLSOBJ
E7001	UT_DB_NOT_FOUND

NOTE:- There are still some cases where the additional information couldn't be provided due to the restriction of a service pack release, to not change the network layer. This is in particular the case if group operations fail. Then the substitutes, e.g. "$(loid)", will be printed rather than the actual data.

ANSI Compliance for VAR/VED Libraries

From this release, Versant will ship the ANSI compliant VAR/VED libraries separately.

They are available in the directory <software_root>/lib/ansi of your installation. These libraries are compliant to the ANSI C++ Standard (ISO/IEC Standard 14882:1998.) guidelines.

Integration of ReVind(VSQL), VODBC in VOD Installer

From 7.0.1.3, Versant is introducing new components to the Versant Object Database installer.

- ReVind (Versant SQL server) (For Windows and Solaris 32-bit platforms)
- Versant ODBC Driver (For Windows 32-bit platforms)

NOTE:- All these Add-on components require an additional license.

For more information, refer to the *ReVind Release Notes and VODBC Release Notes*.

Introduction to CPU core based licensing

From 7.0.1.3, Versant has a improved licensing mechanism where it will check for the total number of licensed CPU cores.

Versant will check for total number of CPU cores available on the machine and total number of licensed cores. If the total licensed cores are less than the total available cores on the machine, Versant will log a warning message in LOGFILE.

Event Daemon Notification to Clients

The database clients might want to know if the event daemon is not running. Depending upon the application requirements the database can be configured to notify the clients/DBA about event daemon's existence.

For more information, refer to Chapter Database Profiles of the *Database Administration Manual* and Chapter Versant Event Notification of the *Database Fundamentals Manual*.

Deprecated Features

The following C and C++ APIs are deprecated and so going forward you are requested to use the APIs in the "Use API" column instead of the deprecated APIs.

	Deprecated API	Use API
C API	• o_readbeprofile • o_writebeprofile • o_readfeprofile • o_writefeprofile	• o_readprofile() • o_writeprofile()
	• o_setcacheparams() • o_replenishcache()	
	• o_utility	
C++ API	• PDOM::readbeprofile • PDOM::writebeprofile • PDOM::readfeprofile • PDOM::writefeprofile	• PDOM::readprofile() • PDOM::writeprofile()
	• PDOM::setcacheparams() • PDOM::replenishcache()	
	• PDOM::utility	

CHANGES AND IMPROVEMENTS IN VERSANT OBJECT DATABASE 7.0.1.4

Latch performance improvements

VOD 7.0.1.4, introduces a newer latch implementation which guarantees that no one can "steal" the latch from the waiting execution units and it therefore guarantees a fairness in scheduling.

VOD latches are used to protect critical sections of code / data. It is a complex implementation that allows multiple execution units (threads /processes) to enter the critical section simultaneously (in case of shared access) or allows only one execution unit (in case of exclusive access). A latch also maintains a wait queue that keeps a track of waiting execution units that need to get the latch.When a latch is released by an execution unit, then in an ideal situation the waiter(s) in the queue should get access to the latch.

Our pre-7.0.1.4 implementation does not guarantee a fairness in the latch acquisition i.e. it is not guaranteed that the waiter will always get the latch once it is released by the owning execution unit and it is possible that the latch could be acquired by a newly requesting execution unit. In high latch contention situations this problem becomes severe since the waiting execution unit has to retry and go repeatedly into the wait queue. This can waste unnecessary CPU cycles on the server machine and can lead to slower responsiveness for the client applications.

Additionally the new implementation also reduces some of the lower level synchronization primitives which are used within the latch thereby reducing unnecessary contention hot-spots. This also adds to the overall performance improvement.

Query Projection

In VOD 7.0.1.4 Query Projection has been introduced. Without projection, the class instances that meet the query criteria are returned. And this may be what you need. But for some queries this can be a lot of data. And if all you need is the value of a particular field from each of the objects returned, without knowing which object is associated with the value, then reading the entire object is not really desirable. Projection provides the more desirable behavior in such cases. By using projection, just the data is returned by the query.

Projection is very useful for many types of reporting.

When using query projection in VQL, the result is returned as an instance of VQueryResult-Any. This class presents the result in a table "column" and "row" comparable format.

For more information on Query Projection, please refer to *Versant C++ and Java Manuals*.

Index selection Hint

The rules used to select an index for term evaluation will, in most cases provide the best choice. However, there are situations where it is desirable to override the default index selection. For such situations, VQL 7.0 provides a “Hint” feature that allows you to override the default index selection.

For more information on this new feature, please refer to the Chapter Versant Queries in the *Database Fundamentals Manual*.

Performance Enhancements for Distributed Databases

Parallel Commit: In this release, Versant has improved the performance of the commit process, by making this sequential procedure more parallel. Pre 7.0.1.4 commit messages were sequentially sent to all the participant databases and for an overall commit success, the commit in all the participating databases needed to succeed.

In order to disable this feature you can add the ‘`parallel_commit off`’ statement in the front end profile of the co-ordinator database.

Navigation: Starting this release, the loid including the dbid is acquired from the database where the new object is stored.

This will avoid unnecessary lookups into the participant databases and therefore improve performance and reduce network utilization.

ICU/TCL/EDG/Recursion Toolkit Upgrades

Prior to VOD 7.0.1.4 the International Components for Unicode(ICU), TCL, Edison Design group C++ Front End(EDG) and Recursion C++ Toolkit tools were supplied in different versions for the various platforms.

With VOD 7.0.1.4 all the above listed tools are supplied with the same version on all platforms.

ICU - 3.8.1, TCL - 8.4.16, EDG - 3.7, TOOLKIT - 5.0.6.

VMWare Support

VOD 7.0.1.4 has introduced distinct license checking for Native and VM hosts. Appropriate license will be needed to run VOD 7.0.1.4 on VM machines.

Support for JDK 5.0 and JDK 6.0 class file format

The enhancer now supports JDK 5.0 and JDK 6.0 class file formats. This permits customers to use enumerations, generics and annotations in their classes.

For using generics and enumerations as types for persistent fields please read below.

NOTE:- Inner classes are not supported as persistent classes. However, it is supported, to have inner classes in persistent as well as persistent-aware classes.

Support for enum typed fields in persistent classes

Persistent classes can now declare enum-typed fields, which are automatically mapped to String fields in the VOD database. Enum-typed fields can be indexed and queried.

Support for java.util. collections / maps

Most java.util collection and map interfaces or implementations (within java.util) thereof are supported as field types of persistent classes. Following is a comprehensive list of the supported types:

```
java.util.Collection
java.util.List
java.util.Map
java.util.Set
java.util.SortedMap
java.util.SortedSet
java.util.ArrayList
java.util.BitSet
java.util.HashMap
java.util.HashSet
java.util.Hashtable
java.util.IdentityHashMap
java.util.LinkedList
java.util.LinkedHashMap
java.util.LinkedHashSet
java.util.Properties
java.util.Stack
java.util.TreeMap
java.util.TreeSet
java.util.Vector
```

All supported java.util types are stored as SCO. These supported types can also be used as generics, however in the VOD database these collections are stored untyped via dual serialization.

New parameter “auto_addvol_threshold”

The usage and semantics of `auto_addvol_threshold` parameter used in the `auto addvol` feature has changed starting from this release.

The default value of this parameter will no longer be 90 percent as in the older releases, but will be 10%, meaning 10% space free.

If you are using the `auto_addvol` feature you are likely to receive UT_ER_PARSE_BEPROFILE for databases prior to 7.0.1.4 if you have used the `auto_addvol_threshold` parameter.

`auto_addvol_threshold` can now accept a size in KBs or MBs or in "%", meaning " KBs/MBs free" OR meaning "%" "space free" and not "space full" as in the previous releases.

For the correct usage and default values accepted, please refer to the description of the auto_addvol_threshold parameter in Chapter Database Profiles, *Database Administration Manual*.

New profile parameter “be_syslog_level”

A new server process profile parameter be_syslog_level has been introduced. This parameter allows to switch on logging to the system event log.

For more information on this parameter, please refer to Chapter Database Profiles in the *Database Administration Manual*.

New profile parameter “lock_batch_size”

A new server process profile parameter lock_batch_size has been introduced. This parameter signifies the number of lock requests to be batched together and processed under one acquisition of the lock latch during acquire/backout locks.

For more information on this parameter, please refer to Chapter Database Profiles in the *Database Administration Manual*.

New profile parameter “spin_count”

A new server process profile parameter spin_count has been introduced. This parameter allows you to tune the count associated with the busy wait portion of the spin lock.

For more information on this parameter, please refer to Chapter Database Profiles in the *Database Administration Manual*.

Schema Objects are now retained in the object cache

Change in behavior for "O_COMMIT"

From this release, `O_COMMIT` option has changed its behavior and now by default the schema objects will be retained in the object cache after a TX commit.

This new behavior, will therefore improve performance and reduce network utilization.

New option "O_RELEASE_SCHEMA_OBJ"

A new option "`O_RELEASE_SCHEMA_OBJ`" is introduced to re-enable the old behavior: releasing the schema objects from the objects cache at the TX commit.

The `O_RELEASE_SCHEMA_OBJ` option needs to be set during beginsession() in C++ or as a session properties in JVI.

This change of behavior applies to C++ and JVI applications only.

Change in behavior for vstr<char> attribute types

For 7.0.1.4 the default behavior of the property '`treat_vstr_of_1b_as_string_in_query`' in profile.be has been changed from '`on`' to '`off`'.

This property determines how strings coded as `vstr<o_1b>` and `vstr<o_u1b>` are interpreted. Previously, strings of type `vstr<o_1b>` and `vstr<o_u1b>` were treated in the same way as vstr<char> by default. For strings of type `vstr<o_1b>` and `vstr<o_u1b>` to be treated as strings of one-byte integer values the property '`treat_vstr_of_1b_as_string_in_query`' was set to 'off'.

This is now the default.

For existing databases, if you are using a string coding of `vstr<o_1b>` or `vstr<o_u1b>` rather than `vstr<char>` and you use a ReVind (VSQL) query with arithmetic string expressions such as `{toupper(x) or x + "abc"}` then the query will return the error `SL_EXPR_BAD_OPERAND_TYPE`.

You can return to the previous behavior by setting '`treat_vstr_of_1b_as_string_in_query`' '`on`' in profile.be but you will not be able to use query projection for attributes of type `vstr<o_1b>` and `vstr<o_u1b>`.

New command “-list” for “Vstats”

From this release, a new command -list has been provided for the “vstats” utility. This command will get the list of enabled server statistics (backend and database statistics).

The "vstats" command -timestamp has been deprecated. The timestamp will always be displayed when viewing the statistics.

For more information, please refer to “vstats” utility in the *Database Administration Manual*.

Changed behavior for VAR

The behavior of varchadmin create {required parameters} -class all has been corrected:

- VAR now replicates VHashtable and VAssociate classes's objects
- User classes having a substring "VAR" will also get replicated

New VAR parameter

From this release, a new parameter RequestRetryInterval has been introduced for basic replication. This parameter determines how frequently the request processor polls the database to get the next transaction replication request. It is specified in milliseconds.

For more information, please refer to the *Versant Asynchronous Replication Manual*.

JDO Improvements

Cursor queries

JDO supports cursor queries which can be enabled by VersantQuery.setCursorQuery() and VersantQuery.setFetchSize() methods.

Activity info

You can use the VersantPersistenceManager.getActivityInfo() method to get information about locks, transactions and connections associated with the database.

The JDO demos contain an example about how to use this functionality.

Memory usage improvement

The memory usage for count star on size queries by setting VersantQuery.setCountStarOnSize(), has been significantly reduced with 7.0.1.4.

Configurable read lock mode

The default lock mode for reading objects from the server can be configured to either NO_LOCK or READ_LOCK. This can be set as a global property in the versant.properties file by versant.defaultReadLockMode where valid options are NO_LOCK or READ_LOCK.

For instance versant.defaultReadLockMode = NO_LOCK, will cause to read objects without a READ_LOCK.

The lock mode can also be modified at runtime by using the VersantPersistenceManager.set-DefaultReadLockMode() method.

The default is READ_LOCK.

New Log event

The startup of the PersistenceManagerFactory now creates a new log event.

Detach of interface references

Detachment of an object with a reference to a persistent interface is now supported.

Group read performance

The RPC for group reading of objects is improved and it now uses an optimized call on the server side. Furthermore, in optimistic transaction there will no more be an additional RPC call to release the locks after objects have been read, as it is done by the server as a part of the group read operation.

Deprecated Features

The following features will be deprecated in future releases:

- Shared sessions
- Object versioning and Configuration management
- C++ Automatic tracking and Tracking API
- Class partitioning

BUGS FIXED SINCE 7.0.1.1

Following is a list of fixed bugs for this release:

Bug ID	Description
	ODBMS Bugs
19928	OBE crashes in case SDA memory is exhausted in the moment when creating internally a lock structure.
20197	Addressing hash index fragmentation.
20479	`SM_E_SDA_CRASHED` after running query involving sub-classes with fetchsize specified.
20504	Obe crash and crash outside VM.
20552	`db_timeout` > 0 causes FTS database to be unstartable.
20555	Unnecessary `VSL_ENOVERSANTPORTS` message in `LOGFILE`.
20584	Index creation can make a database unstartable if it runs out of volume space.
20587	ODMG - setting reference causes d_Iterator to cause core dump.
20597	ODMG - In some cases we get error `d_Error_PositionOutOfRange` when using remove_element function of d_List.
20607	Crash with `E969: SL_AREA_ALREADY_FREE`: Area is marked as already free.
20608	Error message is logged in `VERSANT.LOG` each time versantd is stopped and/or started.
20610	Restart of versantd is causing E3009 and `db2tty` hang.
20622	FTS manual sync using vbackup reports invalid timestamp.
20638	`VERSANT_HOST_NAME` is not supported on Red Hat ES 3.0 with VDS 7.0.1.2.
20639	Addvol deletes be_permit_pre70_clients from profile.be.
20655	`Vstr::swap()` and `LinkVstr::swap()` leak memory.
20692	Database crash on Windows with exception 0xc0000008.
20711	Addvol can crash if system volume's absolute path is > 29 characters.
20724	`o_vbackup` doesn't accept device size > 2GB.

JVI Bugs	
20606	Java.lang.ArithmeticException: / by zero occurs when an empty object array is passed to TransSession `zapObjectExcept()` and `zapObjects()`.
20686	Different behavior of TransSession.deleteObject() method in optimistic locking between JVI 6.0.5 and JVI 7.0.1.

VAR Bug	
20298	VARTxMessages accumulating in database.

JDO Bugs	
OA-261	Several VOD Backend property settings are not being taken into account with remote PM.
OA-278	Support to control if a readlock will be acquired for queries/fetches.
OA-280	Expose XAResource interface via method on `VersantPersistenceManager`.
OA-287	Exceptions were thrown when assertions are enabled for JVM.
OA-290	Using inner classes of more than one level deep in JDOQL resulted in parser error.
OA-295	Updated support to handle broken/dangling references as null.

BUGS FIXED SINCE 7.0.1.2

Following are the bugs fixed since 7.0.1.2:

Bug ID	Description
	ODBMS Bugs
17905	`E133: VSL_FILE_OPEN if addvol is call with path without driveletter`
18918	`cleanbe` causes server crash with error 1099 on Windows
19460	`dbuser -add -P -m r <dbname>` removes write privileges from non-DBA users
19813	`E1005: SM_E_BAD_KEY_LENGTH` error on VCursor at indexed `O_INTERSECT` predicate
20162	`schcomp` generates wrong code for #pragma pack when using with `typedef` structs
20268	`dbinfo -p` returns `SL_PANIC` from invalid PID
20345	Log version and patch number in the database LOGFILE when the database is started
20526	ODMG - Database alias names not working correctly with some ODMG methods
20531	obe can crash while doing an automatic setdbid in a clustered environment
20547	Windows `o_getclientconnectinfo()` is case-sensitive
20590	High CPU consumption of VEDDriver processes
20638	`"VERSANT_HOST_NAME"` is not supported on Redhat ES 3.0
20746	Customer reports deprecated API message from linker
20768	Cursor queries can get `SM_E_INVALIDPID` while traversing schema chain
20794	Control C on `dbtool -locks -table` crashes the database
20802	`vstats` usage doesn't show the `-timestamp` option
20834	`schcomp` throws error if cmath library is included in imp file on Linux RHEL4
20838	`cnvrtdb.exe` requires versantd to be running despite it being linked in 1-p mode

Bug ID	Description
	ODBMS Bugs
20839	`obe` can crash when using select with vstr type of query.
20859	Group Read Error Handler hangs client.
20860	convertdb does not work if VAT indexes are defined.
20862	`obe` can crash if 1p processes exit without cleaning up their resources.
20874	VQL 6.0 query on VOD 7 can return wrong number of instances in some cases.
20879	`vstream` enhancement to recognize the escape character '\' to allow you to specify template classes with multiple parameters.
20884	No core and no stack in `systrace` in certain cases.
20889	JVM Crash inside Native C code due to unhandled NULL pointer.
20893	VEDDriver dumps core.
20903	`popen()` implementation change for Solaris.
20905	schcomp segmentation fault if not with C locale.
20919	`dbtool -sys -info -resource` output does not show correct client process ID's.
20924	If server crashes on Windows due to an error/exception then in some cases the mini crash dump file might not be generated correctly.
20928	Certain 7.0 query on super class with index, and fetchsize specified can crash the server.
20929	`schcomp` emulates gcc 3.0, not gcc 3.4, breaking version-dependent headers at customer.
20939	`schcomp` generates wrong code for internal structs.
20947	`schcomp` cannot handle transient union types.
20996	New VQL7.0 query with 'LIKE <loid>' crashes database.
21010	`obe` can hang if there are many concurrent queries.
21024	Removing last element of VVSet creates problem in subsequent iterator.
21030	`connect` parameter in front-end profile does not work.
21034	`O_ALL_CONNECTED_DBS` doesn't work when connected to only one database.

Bug ID	Description
ODBMS Bugs	
21036	Hide command window for synchronously-launched programs in Windows OS.
21058	`vbackup` fails with `E7127` if btree index is defined
21060	Database can crash if optimistic locking is being used and the number of user defined classes is greater than the number specified in the server process profile.
21087	64-bit `obe` process can crash in certain cases when sending error stack to the client.
21101	When using XA, the database can crash when trying to retrieve a Versant transaction from an external transaction.
21119	Client crashes with E3016, if 1024 file descriptors are allocated before starting VSession.
21150	Transaction rollback using utility dumps core.
21207	Database hung in signal handler.
21218	`Systrace` file having garbled data in HP-UX 64 bit.
21219	Database crash while freeing a buffer page.
21230	Leftover latch issue.

JVI Bugs	
21171	JVI does not store the null array correctly.
21227	JVI DBUtility.dbInfo() support for read-only option.

JDO Bugs	
OA 327	Issue when the subclass in the db is not the same as for client side model.
OA 334	Implement JDO JNDI PersistenceManagerFactory Binding
OA 356	Add an Explicit Locking API for locking Objects
OA 381	Vds internal connection pool not in sync with managed connection pool

OA 383	Serious problems with derived classes and schema evolution
OA 391	NPE with SCOLazyXXXMap when creating the diff
OA 394	Vds SchemaEditor may produce a scenario where the DataStoreSchema-Classes and the UserSchemaClasses is not properly sync'd
OA 396	Schema evolution alters memory layout information because of which back-end tools cannot be used to read data.
OA 397	Running 'tut1' example with 'enableassertions' JVM parameter causes exception.
OA 398	If upper limit in query.setRange is greater than the number of objects in DB, it causes ArrayIndexOutOfBoundsException when running query.
OA 399	NPE after 'clear' on lazy collection/map and then obtaining an iterator.
OA 400	Elements of Lazy Collection not correctly deleted after a 'clear' and update.
OA 401	FetchField calculation can give exception when subclass has fewer fields than base class.
OA 404	The 'fetch group' element is not being included in the defining fetch-group.
OA 405	Add 'set/getName' to api to allow to associate a custom string with eventlogs and user transactions where possible.
OA 406	Pooled connections (datastoremanager) instances have a open connection with an active transaction once put back in the pool.
OA 409	Updated the 'retrieve' process to limit the depth traversel according to fetch-plan
OA 411	Fields on the pc instance was not properly cleared/hollowed on tx boudaries.
OA 414	Clearing a Lazy Map caused a the collection to be nulled instead of cleared in vds backend.
OA 415	Fetching of data for vds backend does not correctly honour the fetchdepth setting on the fetchplan.
OA 416	Update lock timeout exception to include extra info
OA 417	Improve delete perf for 'deletePersistentAll' operation.
OA 419	Misspelled property for versant.scoLazyDataFetch and misspelled property for versant.scoLazyElements / Vantive Case 20857 & 21066
OA 420	Improve startup time for pmf
OA 421	NPE with flush after a 'lazy' collection was cleared

OA 422	Querying using a date field from an already read object causes UnsupportedOperationException
OA 424	Change enhancer add a private 'default constuctor' instead of a 'public' when the class does not have a default constructor
OA 425	Overwriting of collection reference during 'flush(retainState=true)' in same transaction.
OA 426	NPE's when connection url is null. This could happen when using a jndi datasource.
OA 427	Exception thrown on detach when objects have a broken reference to other objects
OA 429	DatastoreException when commiting a pm with open connection to vds backend when there is no changes on the pm.
OA 432	JDOUserException when adding a 'null' pc value, to a 'lazy' map
OA 438	Bug with 'replaced' object in a map.put and set.add call for lazy collections/ map
OA 440	Memory leak when trying to free memory with 'evict' calls
OA 441	Null Pointer Exception if size() or isEmpty() is called on a null initialized collection
OA 442	Double call of afterCompletion

BUGS FIXED SINCE 7.0.1.3

Following are the bugs fixed since 7.0.1.3:

Bug ID	Description
	ODBMS Bugs
16461	Can't have spaces in username on Windows
17607	logvolmaxsize does not accept 'g' or 'G' (e.g. '2g' rather than '2048M')
18404	FE stats cannot be viewed by vstats on 64-bit solaris
18617	PDOM methods should use 'const char*' parameters instead of 'char*'
19373	`vstats` enhancement: option to get a list of enabled statistics
20632	Numeric volume name/path entries in profile.be cannot be parsed
20637	`sch2db` causes invalid long TR name with >15-character username
20847	Retain Schema as default for commit
20945	`vbackup -restore` takes 98% of CPU if some backup device is entered again
21043	Log stack trace if memory allocation fails
21059	`setfeprofile()` of VSession cannot be called outside a session as there is no empty VSesssion contructor
21123	Lingering shared memory segment cleanup
21158	deleted Vstr from `checktimestamps` shows incorrect data under FTS
21163	Application crashes due to swapping out of attribute objects.
21186	client crash during `o_releaseObj`
21207	database hung in signal handler
21221	optional printing of E3004/3003 errors in Server LOGFILE
21238	`schcomp` cannot handle wchar_t as native type on Windows
21242	`vbackup` does not restore the last chunk in an archive leading to out of sequence
21263	7012/7013 `dbtool -sys -info -act` keeps 6053 databases on same machine from being started/stopped

Bug ID	Description
	ODBMS Bugs
21288	`o_getclassinfo()/db2tty` slow because of incorrect segment owner counts.
21291	V??Dictionary.select_on_value() is non-constant method
21292	Cleanup of dead units to prevent obe crash
21294	E7034 during `createdb` if shmmax is set to a large value
21295	`ss.d` closing invalid file descriptors.
21308	`rev_iter.h` has nonstandard declaration for friend function
21310	read-only database not shown as active when active
21314	`connect` parameter in fe profile does not handle databases with hyphens in name
21331	Entering non-existing LOID in \`dbtool -object -info <LOID> <db>\` will cause strange errors.
21338	`schcomp` on 64 bit linux can not handle asssembly code properly.
21360	`dbtool` uses improper format specifier for segment and class names
21372	Logging of crashes to system event log
21375	downgradelocks API does not accept NULL as first parameter value
21379	new API which allows to set a transaction name in JDO. A new RPC will be introduced to set the session name
21384	`dbtool -locks -table` exhibits a mismatch in the latching behaviour
21389	`vstream` crash with Default exception handler!!!errno = 0
21402	leftover latch in transaction rollback in one error case
21409	‘`dbtool -lock -info`' & '`dbtool -sys -info -resource`' need to print txname column
21420	`setdbid` stops reporting errors after executing `dblist` using UT API in 1p mode
21430	`dbtool -index -check` very slow on unbalanced btree indexes
21442	Enhancement for de-fragmentation of system hash tables
21447	Report growth and shrinkage of physical.log
21470	Access violation when adding hash index on string attribute
21486	FTS vbackup synch

Bug ID	Description
	ODBMS Bugs
21513	SAMEKEYFOUND not reported to client under FTS, also causes unnecessary `polling`
21542	Serialize C++ activation in shared sessions
21547	Many utility `obe/dbid` processes spawned because VERSANT_HOST_NAME was set in cluster environment
21552	Indexed query hangs after buffer contention with writer thread
21583	Container restart due to SIGSEGV while executing Versant native code
21623	`dbtool -trans -info <db>` shows PID obe instead of PID of JDO client process
21624	hang in `refreshobjs()`
21641	VOD 7.x cursor query not using subclass index properly
21653	VA index term demoted below non-VA predicate term slows query
21693	Segmentation fault when calling dbtool: `-sys -info -resource <dbname>`
21699	profile.be parameter `auto_addvol_threshold` that can take value in absolute size
21704	`VQuery::set_param()` crashes when passed a NULL_LINK for a param
21710	Avoid crash when stopping database forcefully
21723	Add back detailed error reporting for remote utilities
21725	`VQuery::execute()` leaks memory in heap
21730	`obe` can crash for null transaction in `getacivityinfo`
21735	E2734 SM_E_ILLEGALCURSOR
21760	`setdbid` does not succeed if `dbinfo -c` is called before committing new object to new AT
21761	Mutexes get reinitialized multiple times while executing startup code.
21807	Client memory leak in groupread/groupwrite for FTS clients
21814	`dbtool -index -info -list -short <db>` cuts off class name
21815	VERSANT_ROOT/LOGFILE is filled with shared memory related messages very quickly

Bug ID	Description
ODBMS Bugs	
21851	`dbtool -sys -info -resource <db>` does not format client process id correctly on 64 bit platforms
21912	space in username causes makedb@localhost to fail
21942	`o_writeprofile` duplicates datavol entries
21949	Freeze in multi-thread C++ session after error in PLink::getObject()
22063	Leftover BF latch during query involving medium object
22100	Error 1003 during checkdb -class -ownercounts -fix
22173	Security hole

JVI Bugs	
17322	BEProfile get/set methods for new profile.be parameters added
20074	FEProfil.getEstimatedConnections() throws exception of type "Throwable"
20104	VOD server crash if JVI uses wrong value : Constants.OPT_LK for property lockmode while creating sessions.
20852	VException causes JBoss 4.0 server crash.
20865	Enhancement of class com.versant.trans.Reflection to improve JVI performance.
21080	Enhancer does not need to call static init() method during class enhancement.
21381	Object still accessible even after leaving session
21522	Cannot bind a boolean parameter in VQL 7 JVI
21569	FundQueryResult.close () throws NullPointerException
21727	NPE when calling VStatistics.getConnectionInfo() outside a session
21786	JVI crash in native code in putString
21880	Implement : int TransSession.getCachedObjectsCount (int state)
21881	Implement : int FundSession.getCodCount(int state)

JVI Bugs	
21902	JVI JCA throwing EJC_DESTROY_FAIL/EJC_CLEANUP_FAIL because of E9621 error
21936	NPE when calling TransSession.zapObjects() with null or empty handle in the array

VAR Bugs	
1464	VAR does not replicate all changes to 2nd replication site
21465	varcheckmeta throws NullPointerException
21466	varsiteadmin stop -database <db> throws error in logfile
21468	VAR does not replicate VAssociate and VHashtable etc.
21524	Restart of veddriver causes running VDBRepllicator to replicate no objects
21811	VDBReplicator process does not terminate and does not reconnect

GUI Bug	
19943	vdbinspector.bat script overwrites CLASSPATH

VXML Bug	
20946	vdb2xml -r throws E9609:EVJ_BAD_ATTR_TYPE if a class contains List<String> attribute

VSQL Bugs	
21313	dhserver problem in VSQL 7.0 (crash)
21336	schload does not work if a DBA password is defined
21417	dhserver hangs on Solaris 10

JDO Bugs	
OA-443	CacheStrategy=No is not honoured by level2cache
OA-451	PersistenceManager XAResource does not always begin a tx on 'XAResource.start()'
OA-452	NPE's LRUStorageCache has inconsistent references
OA-455	pm.retrieve does not push superclass fields to pc instance
OA-456	Pessimistical concurrency control does not always lock
OA-457	Memory leak in VdsCompileQuery
OA-464	dbtool -locks -table <db> returns PID of obe instead of client PID
OA-479	Using wrong code nextFetchgroup during detachment
OA-500	AsValueNode exception
OA-501	javax.resource.spi.LocalTransactionException: Transaction not running, possibly caused by premature Versant rollback
OA-505	attachcopy does not work for class hierarchies with references in the base class
OA-506	VersantTransaction.setName, VersantPersistenceManager.setName crashes if RemotestorageManager is used
OA-508	javax.jdo.JDOFatalInternalException: The transaction is not active: 1
OA-509	obe crash with p3 for VOD 7.0.1.3 on Windows (does not occur with p2) if transaction or session name is greater than 31 chars
OA-511	log level none should not write logs
OA-514	Remote Server not accessibly
OA-515	Workbench can not contact remote server
OA-516	loids for class objects returned from getActivityInfo(null,null,null) cannot be resolved
OA-518	The lock rpc does not provide the loid of the class
OA-522	Creating a ConnectionFactory is not synchronized
OA-523	Serialized Single field identity problem
OA-524	Update failures after transaction flush
OA-528	Object from Query are fetched from server even if they are already in client
OA-537	Consistency check for empty collections does not work

JDO Bugs	
OA-538	VOD Using a fetch plan going over a field with an oid reference set to null crashes in StateFetchQueue
OA-540	Workbench: pm server connect should not be enabled if no project is loaded
OA-543	Running some of the JDO tools on VOD DB throws ClassCastException
OA-544	Evolving an existing class AND defining a new class in the same step throws FIELD_ALREADY_EXISTS exception
OA-549	States are getting garbage collected from LocalPMCache if using weak references
OA-553	RPC implementation does not read/write all data from/to the socket
OA-557	wrong defaultReadLockMode when pm is reused from pool
OA-560	JDO cursor query return wrong instance count if object in result set is deleted

7.0.1.4 SUPPORTED PLATFORMS

VOD

Windows 32-bit

Windows (X86)

W2000 (2), XP (3), W2003, Vista (1)

- C++: VS.NET 2002 (7.0), VS.NET 2003 (7.1), VS.NET 2005 (8.0)
- JVI: JDK 1.4, 1.5, 1.6
- JDO: JDK 1.5

NOTES:-

1. Vista requires VS .NET 2005 SP1 and SP1 for Vista.
2. W2000 requires Service Pack 4 or above.
3. XP requires Windows XP Professional Edition, SP2 or above.

Windows 64-bit

Windows (X86-64 AMD64 Opteron & EM64T Xeon)

XP (2), W2003, Vista (1)

- C++ VS.NET 2005 (8.0)
- JVI: JDK 1.5, 1.6
- JDO: JDK 1.5

NOTES:-

1. Vista requires VS .NET 2005 SP1 and SP1 for Vista.
2. XP requires Windows XP Professional with x64 Edition Service Pack 2 or above.

Solaris 32-bit & 64-bit

Solaris (SPARC)

2.9

- C++: CC 5.6, CC 5.7(Sun Studio 10), CC 5.8 (Sun Studio 11)
- JVI: JDK 1.4, 1.5, 1.6
- JDO: JDK 1.5

2.10

- C++: CC 5.8 (Sun Studio 11), CC 5.9 (Sun Studio 12)
- JVI: JDK 1.4, 1.5, 1.6
- JDO: JDK 1.5

RedHat Enterprise Linux 32-bit & 64-bit

RedHat Enterprise Linux 32-bit (X86)

RedHat Enterprise Linux 64-bit (X86-64 AMD64 Opteron & EM64T Xeon)

RHEL 3.0 (32-bit only)

- C++ : gcc 3.2 (starting from 3.2.3)
- JVI: JDK 1.4, 1.5, 1.6
- JDO: JDK 1.5

RHEL 4.0

- C++ : gcc 3.4 (starting from 3.4.3)
- JVI: JDK 1.4, 1.5, 1.6
- JDO: JDK 1.5

RHEL 5.0

- C++ : gcc 4.2 (starting from 4.2.1)
- JVI: 1.5, 1.6

- JDO: JDK 1.5

HP-UX 32-bit & 64-bit

HP-UX (PA-RISC)

11.iv2

- C++: aCC 3.57, aCC 3.70
- JVI: JDK 1.4 / 1.5
- JDO: JDK 1.5

AIX 32-bit

AIX (PowerPC)

5.2

- C++ . xlC 6.0
- JVI: JDK 1.4,
- JDO: not supported

5.3

- C++: xlC 7.0, xlC 8.0, xlC 9.0,
- C++ 5.3: xlC 8.0
- JVI: JDK 1.4, 1.5
- JDO: JDK 1.5

ReVind

Windows 32-bit

Windows (X86)

W2000 (2), XP, W2003, Vista

NOTES:-

1. Vista requires VS .NET 2005 SP1 and SP1 for Vista.
2. W2000 requires Service Pack 4 or above.
3. XP requires Windows XP Professional Edition, SP2 or above.

Solaris 32-bit

Solaris (SPARC)

2.9

2.10

COMPATIBILITY ISSUES

Version Compatibility

In general, pre 7.0 clients are binary compatible with 7.0 server. But there could be some incompatibility between 6.0.0.x clients (6.0.0.1 or 6.0.0.2) and the new 7.0.1 server.

Customers are strongly advised:

- To upgrade both the client and the server to 7.0.1 if they are on one of the 6.0.0.x FCS releases. Customers using the new security features introduced as part of 6.0.0.x are especially prone to seeing some errors in this environment.
- Not to use newer version clients e.g. 7.0.1.2 with older version servers e.g. 7.0.1.1, as this type of configuration is not supported.

The following table lists the version compatibility:

Client Version	**7.0.1 Server**
6.0.0.0 (Beta)	NOT compatible
6.0.0.1 (FCS)	NOT compatible
6.0.0.2(FCS)	NOT compatible
6.0.1 (GA)	Compatible
6.0.5 (GA)	Compatible

Backward Compatibility

As a default behavior, pre 7.0 clients (session connections and utility connections) will not be able to connect to 7.0 server.

The DBA can enable to a certain extent the pre 7.0 clients to connect by using the backend profile parameter `be_permit_pre70_clients (on/off)`. Note that even if this parameter is set, pre 7.0-DBA-specific utility connections will not be permitted.

Compatibility matrix for Versant Add-ons

The compatibility matrix for Versant add-ons is as follows:

	VAR	Vedding	Warm Standby	HABACKUP	Vork-out	ReVind	VODBC
VAR	N/A	No	No	Yes	No	Yes	Yes
Vedding	No	N/A	No	Yes	Yes	Yes	Yes
Warm Standby	No	No	N/A	Yes	Yes	Yes	Yes
HABACKUP	Yes	Yes	Yes	N/A	No	Yes	Yes
Vorkout	No	Yes	Yes	No	N/A	Yes	Yes
ReVind	Yes	Yes	Yes	Yes	Yes	N/A	Yes
VODBC	Yes	Yes	Yes	Yes	Yes	Yes	N/A

VQL

The behavior of using `O_ISA_EXACT/O_NOT_ISA_EXACT` query operators on non-link type has changed. Using pre 7.0 server, these operators return false on attribute of non-link type. Using 7.0 server, error `SM_E_BAD_DOMAIN_TYPE` will be returned by the server and the query will stop.

The behavior of Set Operators on multi-valued link attribute has changed with respect to the flag `O_EMPTY_TRUE`. If pre 7.0 server is used and `O_EMPTY_TRUE` is set for a query, objects will be selected if the attribute is empty or contains NULL links. With 7.0 server, objects will be selected only if the multi-valued attribute fulfills the criteria.

For example, suppose we have class A and class B showed below:

```
class A  {
 vstr<B> attrib;
};
class B  {
};
```

We want to select objects from class A whose attribute `attrib` contains at least one of the give objects “b1” and “b2” of class B. The query can be formulated as below:

```
select selfoid from A where attrib intersect { b1, b2 };
```

In the database, there is an object “a1” of class A whose attribute attrib is empty. With pre 7.0 server, if O_EMPTY_TRUE flag is used, object a1 will be selected even though it does not contain b1 or b2. But with 7.0 server, object a1 will not be selected.

JDO

If the user already has classes compiled with an earlier version of JDO and he tries to run the application with the new JDO jars, then the following error may occur:

```
[java] Exception in thread "main" java.lang.NoSuchMethodError:
javax.jdo.JDOHelper.getPersistenceManagerFactory(Ljava/util/
Properties;)Ljavax/jdo/PersistenceManagerFactory;
```

To resolve this problem, you need to rebuild the domain classes with the new JDO jars.

32/64 bit DB format Compatibility

For similar Operating Systems, VOD/32bit and VOD/64bit database formats are identical. A VOD/32bit populated DB can be migrated to a VOD/64bit server and vice versa.

For instance, you may take a VOD/32 database, stop it, save your VOD/32bit Windows installation, then install VOD/64bit Windows and run `startDB` on the same database using your VOD/64bit installation.

Before you switch from 32bit to 64bit or vice versa, make sure that the `logical.log` of the database is completely empty.

In order to empty the `logical.log`, perform following steps:

1. `dbinfo -1 <db>` (to ensure that no user will use the database after you stop it in the next step)
2. `stopdb <db>`
3. `startdb <db>` (this will run through the physical and the logical recovery and thus will empty the `logical.log`)
4. `stopdb <db>`
5. Switch from 32bit --> 64bit or from 64bit --> 32bit
6. `dbinfo -m <db>`
7. `startdb <db>`

GENERAL RESTRICTIONS AND SUGGESTIONS

Compilation Considerations

Starting 7.0.1.4, the VOD core is compiled as C++. Please note the following while compiling your applications with VOD 7.0.1.4:

- Pure C applications need to link additionally against the C++ runtime library. In case that internal VOD functions were used, you will get 'unresolved externals' linking errors.
- For C++ applications, in the case that functions from C-API or internal C functions were called and you have 'extern "C" ' around the VOD headerfile includes, you have to remove the 'extern "C" '. All C-functions are classified as 'extern "C" ' within the VOD headerfiles.

Cannot detect object deletion, in Optimistic Locking session

In an optimistic locking session, there is no way to find out if an object in your cache has been deleted after it was read. If you call a check timestamp routine, such as `o_checktimestamps()`, you will get a timestamp check failure, if you are still in the same transaction as when the object was read.

Do not set timestamp to values less than zero

Do not set a timestamp of type `o_timestamp` or `d_Timestamp` to a value less than `0`, because values less than zero cannot be properly handled. This means that in this release you can only use timestamp values later than `00:00:00 UTC, Jan 1, 1970`.

Embedabbility

Server process profile parameter `stat` cannot be updated or added using this API. If you use `writeprofile` to write `stat` parameter then it is ignored and not written to the `profile.be` file. Also, if it was already present in the `profile.be`, then after you `o_writeprofile()`, `stat` parameter is deleted from `profile.be` file. To workaround this problem, you can use `vstats` statistics collection utility as an alternate.

API `o_writeprofile()` does not check for duplicate entries of `datavol` parameter (only `datavol` name and `datavol` path part of the `datavol` entry) while writing to the server process profile file `profile.be`.

VOD XA has not been implemented to work with Vedding

Read-only database restrictions

- The utility vcopydb does not work when the source database is in read-only mode.
- You cannot use a read-only database as a coordinator database in a distributed transaction.

Known limitation in 7.0 query

Currently, set operators are supported only for simple expressions (meaning an expression which will end up as "attr op value". They are not supported for complex expressions (which cannot be simplified to "attr op value").

For example:

```
"attribute1 intersect attribute2" or "for all x in attribute1:(x i
ntersect some_constant_collection)”.
```

Limitations in Versant JDO Interface

Schema Limitations

- Application identity is not supported.
- Managed relationship is not supported.
- Null element support is limited to Collection and Map (does not work with `HashSet` and `TreeSet`).
- Object/interface support is limited to PCs (does not work with wrapper classes or non-PCs).
- No one-to-one relationships can be embedded - Versant does not have support for `embedded="true"` feature for embedding `PersistenceCapable` objects.
- If a field is mapped as an Object, List<Object>, Map<Object,Object> in the schema , only elements of PC classes can be persisted. Elements of wrapper classes and non PC classes cannot be persisted in this case.
- Multi-dimensional arrays are not supported.

JDOQL Limitations

- Comparisons between non-PCs are not supported.
- `BigInteger`/`BigDecimal` can only be compared with equal and not-equal. No arithmetic operation is allowed.
- Equal and Not-Equal to null value is only supported for primitive wrappers and `java.util.Date`.
- Nested contains is supported, but only collection of PCs is allowed.
- String concatenation for attribute is not supported.
- No type promotion when binding a value.
- Transient PC cannot be passed to Query as parameter.
- `isEmpty` on a collection field where the element-type is a String and `isEmpty` on a map field where the key-type/value-type is a String are not supported.
- Instance of operators is not supported.

JDO Metadata Loading Limitations

- Dynamic loading of JDO metadata files is not supported.
- No support for automatic resolving of classes in package "`java.lang.*`" for element-type, key-type etc. ('`java.lang.Object`' can't be specified as 'Object').

JDO 2.0 Features Not Supported in this Release

- The following collection/map types are not supported:

  ```
  java.util.LinkedHashMap
  java.util.LinkedHashSet
  ```

- Data type `java.util.Currency` is not supported.
- Query features aggregates, `setGrouping`, `setResult`, `setUnique`, `setResultClass`, `setRange` are not supported.
- Query methods `Math.abs`(numeric), `Math.sqrt`(numeric), `size()`, `substring()`, `indexOf()`, `JDOHelper.getObjectId()` are not supported.

Vedding(FTS) Not Supported in JDO

- Vedding, the Fault Tolerant Versant ODBMS Server is not supported in JDO.

VERSANT RELEASE IDENTIFIER

Each Versant release can be identified by a five-digit number like:

`w.x.y.z.p` where

The first three digits `w.x.y` indicate the release number

The fourth digit `z` indicates the service release number and

The fifth digit `p` indicates the patch release number.

For example, the release 7.0.1.3.6 is the patch 6 of the Service release 3 of the release 7.0.1

You can find this release number by typing in the name of a utility and the `-l` option.

For example, if you want to know the patch release number of the `oscp` utility, type

```
oscp –l
```

CHAPTER 2

Platform Notes for Windows

This Chapter gives detailed explanation on the system requirement and platform restrictions for Windows Platform.

The Chapter covers the following in detail:

- System Requirements
- Certification Notes
- Usage Notes for Windows
- Platform Restrictions

SYSTEM REQUIREMENTS

These Platform Notes contain information specific to Release 7.0.1.4 for Windows.

Please check your architecture, operating system release, and compiler version before installing Versant Object Database.

32 bit support on Windows

Following information is specific to Windows platform for 32bit:

Hardware

Versant Object Database, Release 7.0.1.4 for Windows 32 bit has the following hardware requirements:

- A machine capable of running Windows, per the requirements listed by Microsoft.
- A minimum of 24 MB on the system with 2 MB of real memory available to Versant if a client-only and 4 MB of real memory available if a personal database is used.
- Approximately 170 MB of disk space
- Ethernet card supported by Windows

Software

Versant Object Database, Release 7.0.1.4 for Windows 32 bit has the following software requirements:

- Windows 2000 Server with Service Pack 4, or
- Windows 2000 Advanced Server with Service Pack 4, or
- Windows XP Professional with Service Pack 2, or
- Windows Server 2003, or
- Vista
- Microsoft TCP/IP for Windows must be installed.

To use Versant C++

- Visual Studio .NET 2002 or 2003. Vista requires VS .NET 2005 SP1 and SP1 for Vista.

To use JVI 7.0

- Sun JDK 1.4 or JDK 5.0

To use VXML 7.0

- JSWDK 1.0.1 (You may use other web server too.)
- Sun JDK 1.4 or JDK 5.0

To use Versant JDO

- Sun JDK 1.5

64 bit support on Windows

Following information is specific to Windows platform for 64bit:

Hardware

Versant Object Database, Release 7.0.1.4 for Windows 64 bit has the following hardware requirements:

- Windows AMD64 machine (x86_64) with Opteron processor, or
- EM64T machine (x86_64) with Xeon processor, as per the requirements listed by Microsoft.
- Approximately 250 MB of disk space
- Ethernet card supported by Windows

Software

Versant Object Database, Release 7.0.1.4 for Windows 64 bit has the following software requirements:

- Vista, or
- Windows 2003 Server with Service Pack 2, or
- Windows XP Professional with x64 Edition Service Pack 2

- Microsoft TCP/IP for Windows must be installed

To use Versant C/C++

- Visual Studio .NET 2005 with the x64 compiler. Vista requires VS .NET 2005 SP1 and SP1 for Vista.

To use Versant JVI/JDO

- Sun JDK 5.0 for x64bit release
- Runtime components from the Microsoft Visual C++ 2005 SPI Redistributable Package (x64) (This can be downloaded from the Microsoft web-site.) or Visual Studio .NET 2005 (x64) installed.

CERTIFICATION NOTES

Visual Studio .NET 2003/.NET 2005 for Windows 32 bit

Versant Object Database 7.0.1.4 is certified on Windows Server 2003 with VS.NET 2003 compiler and is also certified on Windows 2000 Server with VS.NET 2005 compiler. The exception to this is the ReVind(VSQL component) which has not been certified on VS.NET 2003 and VS.NET 2005.

Owing to the strict ANSI compliance of this compiler, it is necessary to observe the following guidelines:

- Applications should be built with the Versant flag "`-DVERSANT_ANSI`".
- A linker flag "`-force:multiple`" needs to be specified while linking the program with `schcomp` generated files.
- During runtime it will be necessary to add `%VERSANT_ROOT%\bin\ansi` to your `PATH` setting before `%VERSANT_ROOT%\bin`
- Versant ANSI C++ libraries should be picked from `%VERSANT_ROOT%\lib\ansi`, while the Versant C libraries can be picked up from `%VERSANT_ROOT%\lib`

NOTE:- For .NET 2005, when you create a new Versant application, you should include `typeinfo.h` as the first header file in your application.

CLARiiON CX & EMC PowerPath

Versant 7.0.1.4 is certified to run HABACKUP with CLARiiON CX family & EMC PowerPath.

AMD64 hardware Certification

The 64 bit VOD release is certified to work with AMD64 hardware.

Windows VISTA Certification

Versant Object Database 7.0.1.4 - patch 9 is certified on Windows VISTA SP 1 with the following software requirements and prerequisites:

Software Requirements:

- Visual Studio 2005 SP 1 or Visual Studio 2008
- Sun JDK 5.0 (for 64bit release)
- Sun JDK 4.2 (for 32bit release)

Owing to the strict ANSI compliance of this compiler, it is necessary to observe the following guidelines:

- Applications should be built with the Versant flag "`-DVERSANT_ANSI`"
- A linker flag "`-force:multiple`" needs to be specified while linking the program with `schcomp` generated files
- During runtime it will be necessary to add `%VERSANT_ROOT%\bin\ansi` to your `PATH` setting before `%VERSANT_ROOT%\bin`
- Versant ANSI C++ libraries should be picked from `%VERSANT_ROOT%\lib\ansi`, while the Versant C libraries can be picked up from `%VERSANT_ROOT%\lib`

Prerequisites:

- Due to Windows UAC restrictions starting VISTA, DBA commands of Versant Object Database can only be run under "Administrator" account
- On a UAC enabled machine even having explicit administrative privileges is not enough to execute DBA commands for a user who is not logged in as an "Administrator"

NOTE:- User Authentication using Third Party Plug-ins and ReVind (VSQL component), are not certified on Windows VISTA.

Windows 2008 Certification

Versant Object Database 7.0.1.4 - patch 9 is certified on Windows 2008 SP 1 with the following software requirements and prerequisites:

Software Requirements:

- Visual Studio 2008
- Sun JDK 5.0 (for 64bit release)
- Sun JDK 4.2 (for 32bit release)

Owing to the strict ANSI compliance of this compiler, it is necessary to observe the following guidelines:

- Applications should be built with the Versant flag "`-DVERSANT_ANSI`"
- A linker flag "`-force:multiple`" needs to be specified while linking the program with `schcomp` generated files

- During runtime it will be necessary to add `%VERSANT_ROOT%\bin\ansi` to your `PATH` setting before `%VERSANT_ROOT%\bin`
- Versant ANSI C++ libraries should be picked from `%VERSANT_ROOT%\lib\ansi`, while the Versant C libraries can be picked up from `%VERSANT_ROOT%\lib`

Prerequisites:

- Due to Windows UAC restrictions starting VISTA, DBA commands of Versant Object Database can only be run under "Administrator" account
- On a UAC enabled machine even having explicit administrative privileges is not enough to execute DBA commands for a user who is not logged in as an "Administrator"

NOTE:- User Authentication using Third Party Plug-ins and ReVind (VSQL component), are not certified on Windows 2008.

USAGE NOTES FOR WINDOWS

Running an Application as a Service

You cannot run Versant utilities with the user names root/SYSTEM/Administrator.

You can run Versant applications with the SYSTEM login id if you set a different, valid user name in the environment variable VERSANT_USER. This is relevant if you want to write an application, which runs as a Windows service and uses a Versant database.

A Windows service generally runs under the login id SYSTEM. When an Windows service application begins a Versant session, the Versant code within your application detects the login ID SYSTEM, which probably will not be a valid database user name.

Whenever Versant detects the login id SYSTEM, it internally checks the environment variable VERSANT_USER to find out which Versant user name it should use when accessing a database. If VERSANT_USER is not set at this point, the user id SYSTEM will be used, and if SYSTEM is not valid then the error NET_INVUSRNM or UT_DB_NO_ACCESS will be returned.

This feature allows you to run an application as an Windows service under the login id SYSTEM, while overriding the SYSTEM login id by another user id which is a valid database user name. Note, however, that setting the VERSANT_USER environment variable to override the Windows login name works only if you login as SYSTEM. In all other cases Versant uses the Windows login name as the database user name.

You can set VERSANT_USER with a program statement in the Windows service application before starting a Versant session. For example:

```
SetEnvironmentVariable("VERSANT_USER", "john");
putenv("VERSANT_USER=john");
```

In the above, note that you need to execute both SetEnvironmentVariable(), in order to set the variable in the Windows process header, and putenv(), in order to set the variable in the C runtime environment.

In order to decide what database user name to set in VERSANT_USER, you will probably have to pass the user name from your client application to your Windows service application by some means, or else use a standard Versant user name always.

Typically, you will pass the login name obtained by the GetUserName() call from your client Windows application to your Windows server application (running as an Windows service) via a TCP/IP socket connection or a named pipe connection between the client and server.

Versant does not check whether a VERSANT_USER setting is a valid Windows user or not, as long as the database being used allows access by a user of that name.

Difference between Versant utilities for Unix Workstations and Windows.

If you also use UNIX workstations, you should note the following differences between Versant utilities for UNIX and personal computers.

Windows uses dispatch programs.

On UNIX, most Versant utility programs are links to the same program to save space.

On Windows, Versant uses a dispatch program to call a generic program. For example, the `oscp.exe` and `startdb.exe` utilities are dispatching programs that call `startut.exe`.

You will normally see no difference between linked files and dispatching programs. However, if you try to use a debugger or tracer on Versant utilities, you might experience different behaviors.

Names are limited to eight characters.

Versant utilities for personal computers function in exactly the same way as Versant utilities for UNIX workstations.

However, if you also use versions of Versant for workstations, you will note that some utilities are renamed to conform to eight character name limits. For example, the workstation `dropclass` utility has been renamed to dropcls.exe for personal computers.

Utilities have a suffix of .exe

Although all Versant utilities necessarily have a .exe suffix, they are referenced in this document as they are invoked, i.e. without the .exe suffix. For example, the `dbuser.exe` utility is referenced as `dbuser`.

PLATFORM RESTRICTION

Do not dynamically load and unload Versant DLL's.

Do not dynamically load and unload Versant DLL's with Win32 API's such as `LoadLibrary` and `FreeLibrary`.

If you dynamically load and unload Versant DLL's, you may encounter problems with resource leaking, such as running out of TLS indexes. The workaround is to keep the Versant DLL's loaded for the lifetime of the application process.

CHAPTER 3

Platform Notes for Sun Solaris

This Chapter gives detailed explanation on the system requirement and certification notes for Sun Solaris Platform.

The Chapter covers the following in detail:

- System Requirements
- Using C++/Versant with Sun C++ Compiler
- Certification Notes

SYSTEM REQUIREMENTS

These Platform Notes contain information specific to Release 7.0.1.4 for Solaris.

Please check the architecture, operating system release and compiler version before installing Versant Object Database.

Hardware

Versant Object Database, Release 7.0.1.4 has the following hardware requirements:

- Sun for Sparc compatible hardware
- Minimum disk space of approximately `300` megabytes
- Although there is no absolute minimum system memory requirement, we strongly recommend a minimum of `32` megabytes.
- A recommended swap space to RAM ratio of `3` to `1`

For other requirements, such as the minimum configuration for your UNIX kernel- See also “Pre-Installation Checklist” on page 180 in "Chapter 8 - Installation on Solaris/ Linux/ HP-UX/AIX".

Software

Versant Object Database, Release 7.0.1.4 has the following software requirements:

- Solaris OS Version 9
- Compilers:

  The following is the list of minimum patch levels needed for the compilers supported on Solaris:

  For Sun C++ 5.6 Patch 117549-02 2005/02/08 or later must be installed.
  The Compiler Library (libC) Versions are:

  - "libCrun.so.1": Sun C++ 5.6 Patch 111711-12 2004/10/25 or later must be installed
    "libCstd.so.1": Sun C++ 5.6 Patch 111711-12 2004/10/25 or later must be installed
- The multi-threaded memory allocator library libmtmalloc

- TCP/IP network protocol (if you want to connect to a remote database or make more than one connection to a local group database.)
- To use C++/Versant and Sun C++ 5.6 Compiler please see section "Using C++/Versant with Sun C++ 5.6 Compiler"
- This release requires use of compiler independent programming, as described in the C++/Versant Reference Manual.
- To use Java Versant Interface, an installation of JDK 1.4.2_07 is required.
- To use Versant Asynchronous Replication (VAR) Java APIs, you need Sun JDK 1.4.2_07 for Solaris and for using VAR C++ APIs and ODBMS Release 7.0.1.4 you need Sun C++ 5.6 Compiler.
- To use Versant XML, you need Sun JDK 1.4.2_07 for Solaris and JSWDK 1.0.1 (You may use other webserver too.)
- To use Versant JDO, you will need Sun JDK 1.5

USING C++/VERSANT WITH SUN C++ COMPILER

Directories and Files

Changes in Library Directories

`LD_LIBRARY_PATH`

After installation, set the environment variable `LD_LIBRARY_PATH` to point to the directory: `<SOFTWARE_ROOT>/lib.`

The `<VERSANT_ROOT>/lib` directory contains ODBMS libraries as well C++ libraries for use with compiler. If you are using C++ checking libraries, set the environment variable `LD_LIBRARY_PATH to point` to the two directories:

```
<SOFTWARE_ROOT>/lib/cxxchk
<SOFTWARE_ROOT>/lib/cxxchk/ansi
```

The non-checking libraries are:

```
<SOFTWARE_ROOT>/lib/ansi
<SOFTWARE_ROOT>/lib
```

The CC5.6 compiler ships some of the CC4.2 libraries under the `lib/CC4` directory. Versant does not support the use of these backward compatibility libraries.

Linking

Link the new C++ libraries `libCstd.a` and `libCrun.a` instead of the `libC.a` to your C++ program, i.e. change the `-lC` to `-lCstd -lCrun` in your link line.

CERTIFICATION NOTES

OS Certification

Versant 7.0.1.4 for Sun Solaris is certified with “Solaris 10”. The exception to this is the ReVind(VSQL component) which has not been certified on “Solaris 10”.

Compiler Certification

Versant 7.0.1.4 for Sun Solaris is certified with “Sun C++ 5.7 2005/01/07”, with “Sun C++ 5.8 Patch 121017-10 2007/02/21" and with “Sun C++ 5.9 SunOS_sparc 2007/05/03”. The exception to this is the ReVind(VSQL component) which has not been certified for these compilers.

Java Development Kit

Versant 7.0.1.4 is certified with “JDK 5.0”, Java version "1.5".

CLARiiON CX & EMC PowerPath

Versant 7.0.1.4 is certified to run HABACKUP with CLARiiON CX family & EMC PowerPath.

CHAPTER 4

Platform Notes for Linux

This Chapter gives detailed explanation on the system requirement, certification notes and platform restrictions for Linux Platform.

The Chapter covers the following:

- System Requirements
- Using C++/VERSANT with gcc
- Certification Notes
- C++ Schema Compiler Notes
- Compiling and Linking
- Platform Restrictions

SYSTEM REQUIREMENTS

Please check your architecture, operating system release and compiler version before installing Versant Object Database.

32 bit Support on Linux

These contain information specific to Red Hat Enterprise Linux (RHEL) versions 3.0 (Update 4), 4.0 and 5.0 for 32 bit Linux.

Hardware

Versant Object Database, for RHEL 3.0 (Update 4), 4.0 and 5.0 has the following hardware requirements:

- Intel x86
- Minimum disk space of 250 megabytes.
- Although there is no absolute minimum system memory requirement, we strongly recommend a minimum of `32` megabytes.
- A recommended swap space to RAM ratio of `3` to `1`.
- If your application will connect to multiple databases, you may need to increase the number of shared memory segments over a standard hardware configuration. To do this, run the `sysctl` utility and change the following under the kernel subsystem:

  ```
  shmmni 4096
  ```

  You will need to reboot the system for this change to take effect.

  The values shown above for `shmmni` are examples and the defaults on your system may be different.
- If you use Versant event notification you will probably need to increase the size of the IPC message queues, otherwise your application may lose events because the queues overflow. You can increase the size of the message queues by using the `sysctl` utility and change the following under the kernel subsystem:

  ```
  msgmnb 16384
  ```

  You will need to reboot the system for this change to take effect.

  The values shown above for `msgmnb` are examples and the defaults on your system may be different.

Software

Versant Object Database, for RHEL has the following software requirements:

- Red Hat Enterprise Linux versions 3.0 (Update 4) or 4.0 or 5.0
- Recommended GNU C/C++ compiler must be present in the PATH during the installation time.
- Native Posix Thread Library (NPTL) should be installed
- TCP/IP network protocol (if you want to connect to a remote database or make more than one connection to a local group database).

To use C++/Versant and gcc

- For Red Hat Enterprise Linux, you must use native compilers, i.e., for RHEL 3.0 use gcc 3.2.3, for RHEL 4.0 use gcc 3.4.3 and for RHEL 5.0 use gcc 4.2.1.

  To find the current version and patch level on your C++ compiler, use the option `-v`. The compiler will then list your version and patch level.

  ```
  gcc --version
  ```
- This release requires use of compiler independent programming, as described in the *C++/Versant Reference Manual*.
- To use Java Versant Interface, an installation of Sun JDK 1.4.2_07 is needed.
- To use Versant Asynchronous Replication Java APIs, you need Sun JDK 1.4.2_07.
- To use Versant Asynchronous replication C++ APIs and ODBMS Release 7.0, you need native gcc.
- To use Versant XML Toolkit (VXML), you need Sun JDK 1.4.2_07 and JSWDK 1.0.1. (You may use other web server also.)
- To use Versant JDO, you will need Sun JDK 1.5.

64 bit Support on Linux

VOD on Linux 64-bit RHEL4.0 configuration is now supported with two different compiler suites, GNU gcc/g++ 3.4.3 and Intel icc/icpc 10.0 Package ID: l_cc_p_10.0.023, available in 2 different installer packages. Please choose the appropriate installer as per your compiler installation.

Following is the information specific to Red Hat Enterprise Linux ES release 4.0 (Nahant) and RHEL 5.0 for 64 bit Linux.

Hardware

- AMD64 machine (x86_64) with Opteron processor OR EM64T machine (x86_64) with Xeon processor.
- Minimum disk space of 250 megabytes.
- Although there is no absolute minimum system memory requirement, we strongly recommend a minimum of `32` megabytes.
- A recommended swap space to RAM ratio of `3` to `1`.
- If your application will connect to multiple databases, you may need to increase the number of shared memory segments over a standard hardware configuration. To do this, run the `sysctl` utility and change the following under the kernel subsystem:

  ```
  shmmni 4096
  ```

  You will need to reboot the system for this change to take effect.

  The values shown above for `shmmni` are examples and the defaults on your system may be different.
- If you use Versant event notification you will probably need to increase the size of the IPC message queues, otherwise your application may lose events because the queues overflow. You can increase the size of the message queues by using the `sysctl` utility and change the following under the kernel subsystem:

  ```
  msgmnb 16384
  ```

  You will need to reboot the system for this change to take effect.

  The values shown above for `msgmnb` are examples and the defaults on your system may be different.

For other requirements, such as the minimum configuration for your UNIX kernel- See also “Pre-Installation Checklist” on page 180 in "Chapter 8 - Installation on Solaris/ Linux/ HP-UX/AIX".

Software

Versant Object Database, for Red Hat Enterprise Linux ES release 4.0 (Nahant) for 64 bit has the following software requirements:

- Red Hat Enterprise Linux ES release 4.0 (Nahant) for 64 bit.
- Recommended GNU/INTEL C/C++ compiler must be present in the PATH during the installation time.
- Native Posix Thread Library (NPTL) should be installed
- TCP/IP network protocol (if you want to connect to a remote database or make more than one connection to a local group database).

To use C++/Versant and g++/icpc

- For Red Hat Enterprise Linux, you must use native compilers, i.e.,for RHEL 4.0 use g++ 3.4.3 and for RHEL 5.0 use g++ 4.2.1.

  Additionally for this release, VOD is supported on RHEL 4.0 64-bit configuration with

  Intel compiler icc/icpc support, Version: 10.0 Package ID: l_cc_p_10.0.023.

  To find the current version and patch level on your C++ compiler, use the option `-v`. The compiler will then list your version and patch level.

  `g++ --version` for GNU compiler g++

  `icpc -V` for Intel compiler icpc
- This release requires use of compiler independent programming, as described in the *C++/Versant Reference Manual*.
- To use Java Versant Interface, an installation of Sun JDK "1.5.0_06" for 64 bit is needed. More information on Java version is:
  - Java(TM) 2 Runtime Environment, Standard Edition (build 1.5.0_06-b05)
    Java HotSpot(TM) 64-Bit Server VM (build 1.5.0_06-b05, mixed mode)
- To use Versant Asynchronous Replication Java APIs, you need Sun JDK "1.5.0_06" for 64 bit.
- To use Versant Asynchronous replication C++ APIs and ODBMS Release 7.0, you need native gcc.
- To use Versant XML Toolkit (VXML), you need Sun JDK "1.5.0_06" for 64 bit and JSWDK 1.0.1. (You may use other web server also.)
- To use Versant JDO, you will need Sun JDK "1.5.0_03" for 64 bit.

USING C++/VERSANT WITH G++/ICPC

Directories and Files

Changes in Library Directories

LD_LIBRARY_PATH

After installation, set the environment variable LD_LIBRARY_PATH to point to the directory:

<VERSANT_ROOT>/lib

The <VERSANT_ROOT>/lib directory contains ODBMS libraries as well C++ libraries for use with gcc. If you are using C++ checking libraries, set the environment variable LD_LIBRARY_PATH to point to the two directories:

<VERSANT_ROOT>/lib/cxxchk

<VERSANT_ROOT>/lib

If you are using ANSI libraries, set the environment variable LD_LIBRARY_PATH to point to the two directories:

<VERSANT_ROOT>/lib/ansi
<VERSANT_ROOT>/lib

If you are using ANSI checking libraries, set the environment variable LD_LIBRARY_PATH to point to the two directories:
<VERSANT_ROOT>/lib/cxxchk/ansi
<VERSANT_ROOT>/lib

This environment variable tells Versant where to find the shared libraries at runtime. This variable must be set, because some of the Versant utilities use shared libraries.

If you are using shared libraries and set LD_LIBRARY_PATH to the wrong set of libraries, your application program will crash.

C++ Schema Compiler Note

To support multiple versions of GNU C/C++ compiler, Versant Object Database ships an additional shell script "`make_predef_macro_table`" in its `<software_root>/bin` directory for Red Hat Enterprise Linux. This script creates a compiler specific file in the `<software_root>/lib` directory which is needed by the utility "`schcomp`". The file is created by Versant at the time of installation using the gcc compiler present in the PATH. Please refer to software requirement section of Release notes for further details.

If you are using gcc compiler version that is different from the compiler used at the time of Versant installation, then utility "`schcomp`" will invoke this script internally to create the compiler specific file. The script thus requires a write permission to the `<software_root>/lib` directory for the user invoking `` `schcomp` ``.

Versant support for -ansi flag for gcc

All the code compiled with `-ansi` option will confirm to strict ANSI standard. The macro `__STRICT_ANSI__` is predefined when the `-ansi` option is used. Some header files may notice this macro and refrain from declaring certain functions or defining certain macros that the ISO standard doesn't call for. This is to avoid interfering with any programs that might use these names for other things e.g. linux macro is not available while compiling with `-ansi` option so it is recommended to use `__linux` or `__linux__` macros.

Currently we provide the (`-ansi`) support to C++ applications linked with Cxx libraries. Hence, customers will be required to make the following changes:

1. New set of libraries have been introduced in the ansi directory of `<VERSANT_ROOT>/lib`. These are in addition to the normal ones at `<VERSANT_ROOT>/lib`. In case of internationalization libraries (e.g. `libicuuc.so`) and ODBMS libraries (e.g. `liboscfe.so`), the same copy could be used irrespective of the use of `-ansi` option.
2. Applications compiled with `-ansi` option should use the ANSI version of library. This can be done by specifying `-L<VERSANT_ROOT>/lib/ansi` before `-L<VERSANT_ROOT>/lib` while linking. Optionally, it can also be done by setting the `LD_LIBRARY_PATH` to `<VERSANT_ROOT>/lib/ansi:<VERSANT_ROOT>/lib`. Linking the application compiled with `-ansi` option to the default Cxx libraries may result in unpredictable behavior. Applications which are not compiled with this option of gcc should be linked with the default libraries in `<VERSANT_ROOT>/lib directory`.
3. If the existing applications are to be made ANSI compliant, they will have to be re-compiled with the `-ansi` option and then be linked to the libraries in `<VERSANT_ROOT>/lib/ansi`.
4. Variable `LD_LIBRAY_PATH` must be set to `<VERSANT_ROOT>/lib/ansi:<VERSANT_ROOT>/lib` while executing the Versant C++ ANSI compliant applications.

5. The schema compiler (`schcomp`) has been updated to include the `-ansi` option. Using this option we generate strict ANSI code in `schema.cxx` file. The resulting `schema.cxx` file should also be compiled with the `-ansi` option and linked by specifying `-L<VERSANT_ROOT>/lib/ansi` before `-L<VERSANT_ROOT>/lib`. Optionally, it can also be done by setting the `LD_LIBRARY_PATH` to `<VERSANT_ROOT>/lib/ansi:<VERSANT_ROOT>/lib`.
6. Demos and tutorials provided with the product can also be run in `-ansi` mode by setting the variables `ANSILIB` to `ansi` and `ANSIFLAG` to `-ansi` in the environment. Other details are mentioned in the readme file provided with the demos.
7. Support to VED: While building custom drivers the `-ansi` compilation option should not be used.
8. Any application using the VAR C++ APIs should not use the `-ansi` compilation option.

CERTIFICATION NOTES

Compiler Certification

Versant 7.0.1.4 for Red Hat Enterprise Linux 4.0 (32-bit) is certified with “GNU C/C++ Complier 3.4.4”.

Java Development Kit

Versant 7.0.1.4 for Red Hat Enterprise Linux (32-bit) is certified with “JDK 5.0”, java version "1.5".

EM64T hardware Certification

The 64 bit VOD release is certified to work with Red Hat Enterprise Linux ES release 4.0 on EM64T hardware on the Xeon processor with compiler version gcc 3.4.3.

COMPILING AND LINKING

Native Posix Thread Library (NPTL) in your applications.

For multi-threaded applications -

1. The `-D_REENTRANT` flag should be passed during compilation of multi-threaded applications.
2. The `-lpthread` flag must be included while linking applications with the Native Posix Thread Library (NPTL).

The NPTL version can be verified using the following command:

```
getconf GNU_LIBPTHREAD_VERSION
```

NOTE:- To see the installed NPTL version, use the following command:

```
getconf GNU_LIBPTHREAD_VERSION
```

The output on RHEL 3.0 should be: `NPTL 0.60`

The output on RHEL 4.0 should be: `NPTL 2.3.4`

PLATFORM RESTRICTIONS

Multiple levels of virtual inheritance may cause problems.

Using VPP_CLASS_CONSTRUCTOR and VPP_CLASS_CONSTRUCTOR2 in classes that have more than one level of virtual inheritance may cause memory corruption.

For example,

suppose that you have the following schema:

```
class A : public PObject    { VPP_CLASS_CONSTRUCTOR1(A);  ... };
class B : virtual public A  { VPP_CLASS_CONSTRUCTOR1(B);  ... };
class C : virtual public B  { VPP_CLASS_CONSTRUCTOR1(C);  ... };
```

This may cause memory corruption when a transient instance of C is constructed.

The workaround is to call PClass<T>::vppConstruct() manually, after creating transient instances of C.

For example:

```
C *c1 = new C(...);
PClassObject<C>::Pointer()->vppConstruct(c1);
C c2(...);
PClassObject<C>::Pointer()->vppConstruct(&c2);
```

The C++/VERSANT checking libraries may experience problems.

The C++ checking libraries may experience problems with some applications. If you experience problems, please retry the same operation with the corresponding non-checking libraries. Please report all problems to Versant Customer Support.

The checking libraries for the fundamental classes are:

archive — <VERSANT_ROOT>/lib/cxxchk/libcxxcls.a

shared — <VERSANT_ROOT>/lib/cxxchk/libcxxcls.so

The checking libraries for the collection classes are:

archive — <VERSANT_ROOT>/lib/cxxchk/libvcoll.a

shared — <VERSANT_ROOT>/lib/cxxchk/libvcoll.so

Explicit casting

Casting needs to be done explicitly when overloaded functions are called with the exact data-type of the parameters. For example:

```
d_oql_execute((d_VQL_Query &)query, (LinkVstr<Person> &)personVstr)
```

Initialization of global pointers restricted

The compiler does not allow initialization of global pointers at the time of declaration with constants. For example:

```
FILE *psr_in = {stdin}; //Gives error
```

Shared Memory Limits for 32 bit

The shared memory size should not exceed 1.2 GB for the database to function properly.

The client and Versant's obe share same process and memory space for 1p applications, while in a 2p application both have different process and memory space. In case of 1p, if the shared library size exceeds 256MB, then it is recommended to use static linking. Implementing static linking ensures that the libraries don't get dynamically loaded in the shared libraries region.

Please note that static linking of libraries will increase the size of the application executable.

Raw devices on Linux

Versant Object Database supports block raw devices and not character raw devices.

VAT restriction for RHEL

To run internationalized applications using JVI, set the environment variable `LANG` to `en_US`.

For C shell the command is:

```
% setenv LANG en_US
```

For Korn shell the command is:

```
export LANG=en_US
```

Recommended Usage for GUI tools

When running any java GUI tool or application with a Sun JDK 1.4.2_07 you may get some font warnings like,

```
Warning: Cannot convert string
"-b&h-lucida-medium-r-normal-sans-*-140-*-*-p-*-iso8859-1"
to type FontStruct
```

This is because the JVM cannot find a particular symbol written in its `font.properties` file on the system.

Solution for this, is to make a change in the file:
`<JDK installation directory>/jre/lib/font.properities.`

Change it from:

```
fontset.default=\ -b&h-lucida-medium-r-normal-sans-*-%d-*-*-p-*-iso88591
```

to:

```
fontset.default=\ -b&h-lucida-medium-r-normal-sans-*-140-*-*-p-*-
iso10646-1
```

Read-Only Database not supported on RHEL 3.0

The Read-Only database mode is not supported on Red Hat Linux 3.0.

CHAPTER 5

Platform Notes for HP-UX

This Chapter gives detailed explanation on the system requirement and platform restrictions for HP-UX Platform.

The Chapter covers the following in detail:

- System Requirements
- Using C++/Versant with HP ANSI Compiler aC++
- Certification Notes
- Compile and Link Options
- Platform Restrictions

SYSTEM REQUIREMENTS

These Platform Notes contain information specific to Release 7.0.1.4 for HP-UX 11i.

Please check your architecture, operating system release, and compiler version before installing Versant Object Database.

Hardware

Versant Object Database, Release 7.0.1.4 for HP-UX 11i has the following hardware requirements:

- HP 9000 Series 700 or Series 800workstation.
- Minimum disk space of 225 megabytes.
- System memory 32 megabytes.
- A recommended swap space to RAM ratio of 3 to 1.
- Ethernet card supported by HP.
- Optionally there should be raw partitions available for log and database volumes. Use of raw partitions can improve performance dramatically, but your disk must be pre-configured to support them.
- If your application connects to multiple databases, you may need to increase the number of shared memory segments over a standard configuration. To accomplish this, you must add the following lines to your configuration file using SAM.

  ```
  options SHMMNI=200
  ```

  The value displayed above for SHMMNI is only an example. The default for SHMMNI is 100 segments with a maximum of one megabyte per segment.
- Kernel Patches to be installed:
  1. PHCO_30544
  2. PHNE_31247
  3. PHCO_30397
  4. PHKL_32927

See HP documentation for details on how to rebuild the kernel. For other requirements, such as the minimum configuration for your UNIX kernel- See also “Pre-Installation Checklist” on page 180 in "Chapter 8 - Installation on Solaris/Linux/ HP-UX/AIX".

Software

Versant Object Database, Release 7.0.1.4 for HP-UX 11i has the following software requirements:

- TCP/IP network protocol (if you want to connect to a remote database or make more than one connection to a local group database).

To use C++/Versant you must have:

- HP ANSI Compiler aCC A.03.57 along with the following patches/bundled-components:

Component	Component synopsis
`libCsup.a/sl`	Bundled run-time support libraries
`dde 4.25`	Bundled debugger upgrade

To find the current version of the compiler/library, type the following at the command line in the directory containing aCC:

For example:

```
what aCC or what /lib.libCsup.a
```

To use C/Versant, you must have:

- HP ANSI C++ B3910B A.03.57
- To use C++/Versant, please see section "Using C++/Versant with HP ANSI Compiler aC++ A.03.57"
- This release requires use of compiler independent programming, as described in the *C++ /Versant Reference Manual*.
- To use Java Versant Interface, an installation of JDK 1.4.2.07 is required.
- To use Versant Asynchronous Replication (VAR) Java APIs, you need JDK 1.4.2.07 for HP.
- To use Versant Asynchronous replication C++ APIs and ODBMS Release 7.0.1.4, you need HP ANSI Compiler aCC A.03.57.
- To use Versant XML Toolkit (VXML), you need IBM JDK 1.4.2.07 and JSWDK 1.0.1. (You may use other webserver also.)
- To use Versant JDO, you will need Sun JDK 1.5.

USING C++/VERSANT WITH HP ANSI COMPILER aC++

Directories and Files

Changes in Library Directories

SHLIB_PATH

After installation, set the environment variable SHLIB_PATH to point to the directory:

<VERSANT_ROOT>/lib

The lib directory contains system libraries as well as C++ libraries for use with aCC A.03.57. In case of HP-UX 11i , the lib directory also contains C++ ANSI libraries to support the -AA option.

If you are using C++ checking libraries, set the environment variable SHLIB_PATH to point to the two directories:

<VERSANT_ROOT>/lib/cxxchk

<VERSANT_ROOT>/lib

If you are using ANSI libraries, set the environment variable SHLIB_PATH to point to the two directories:

<VERSANT_ROOT>/lib/ansi

<VERSANT_ROOT>/lib

If you are using ANSI checking libraries, set the environment variable SHLIB_PATH to point to the two directories:

<VERSANT_ROOT>/lib/cxxchk/ansi

<VERSANT_ROOT>/lib

Versant support for –AA flag for aCC A.03.57 on HP-UX 11i

The –AA option enables use of the new 2.0 Standard C++ library, which is the new standard (ISO C++) conforming to ("templatized") iostream library. All code compiled with -AA option will conform to strict ANSI standards. If this option is used, it must be used consistently through out all translation units. Mixing object files within an executable is not supported.

Currently we provide the (–AA) support to C++ applications linked with cxx libraries. Hence, customers will be required to make the following changes:

1. New set of libraries in the ansi directory of <VERSANT_ROOT>/lib. These are in addition to the normal ones at <VERSANT_ROOT>/lib. In case of internationalization libraries (e.g. libicuuc.sl) and ODBMS libraries (e.g. liboscfe.sl), the same copy could be used irrespective of the use of –AA option.
2. Applications compiled with -AA option should use the ANSI version of library. This can be done by specifying –L<VERSANT_ROOT>/lib/ansi before –L<VERSANT_ROOT>/lib while linking. Optionally, it can also be done by setting SHLIB_PATH to <VERSANT_ROOT>/lib/ansi:<VERSANT_ROOT>/lib.

   Linking the application compiled with -AA option to the default cxx libraries may result in unpredictable behaviour. Applications which are not compiled with this option of aCC should be linked with the default libraries in <VERSANT_ROOT>/lib directory as was done previously.
3. If existing applications are to be made ANSI compliant, they will have to be re-compiled with the -AA option and then linked with libraries in <VERSANT_ROOT>/lib/ansi.
4. Variable SHLIB_PATH must be set to <VERSANT_ROOT>/lib/ansi:<VERSANT_ROOT>/lib when executing Versant C++ ANSI compliant applications.
5. It is important and advisable to use the -mt flag rather than any specific multi-threading options, when using –AA option for multi-threaded applications. -mt option in aCC compiler will automatically enable the thread flags according to the application and mode of use. Moreover it is advisable to use the same multi-threading flags while compiling and linking. Refer to aCC release notes for more details.
6. The schema compiler (schcomp) has been updated to include the –AA option. Using this option will generate strict ANSI code in schema.cxx file. The resulting schema.cxx file should also be compiled with the -AA option and linked by specifying –L<VERSANT_ROOT>/lib/ansi before –L<VERSANT_ROOT>/lib. Optionally, it can also be done by setting the SHLIB_PATH to <VERSANT_ROOT>/lib/ansi:<VERSANT_ROOT>/lib.
7. Demos and tutorials provided with the product can also be run in -AA mode by setting the variables ANSILIB to ansi and ANSIFLAG to -AA in the environment. Other details are mentioned in the readme file provided with the demos.

8. The 1.x standard C++ library provided with the aCC compiler is binary incompatible with the 2.x version. Using the -AA version will enable use of the 2.x version. Customers should note that the 1.x version is being deprecated and will be eventually replaced by the 2.x version. Thus it is recommended to have new applications compliant to ANSI standards.
9. Support to VED: While building custom drivers the –AA compilation option should not be used.
10. Any application using the VAR C++ APIs should not use the -AA compilation option.

CERTIFICATION NOTES

HP ANSI Compiler aC++

Versant 7.0.1.4 is certified with "HP ANSI C++ B3910B A.03.70"

Java Development Kit

Versant 7.0.1.4 is certified with “JDK 5.0”, Java version "1.5".

CLARiiON CX & EMC PowerPath

Versant 7.0.1.4 is certified to run HABACKUP with CLARiiON CX family & EMC PowerPath.

COMPILE AND LINK OPTIONS

While compiling and linking you must pass the following options:

```
-D_REENTRANT -D_THREAD_SAFE -DHP11 -D_POSIX_C_SOURCE=199506L
-D_HPUX_SOURCE
```

NOTE:- `-DHP11` is necessary while compiling with `schcomp`.

To resolve thread library symbols use following linker option:

```
-Wl -E -l:libdld.sl -lpthread
```

To make the binaries compatible for the following architectures, use the said linker options :

```
HP-UX 11i (32 bit) PA-RISC architectures: +DA1.1 +DS2.0
HP-UX 11i (64 bit) PA-RISC 2.0 architectures: +DA2.0W +DS2.0
```

Use following linker option for 64 bit applications using "i18N libraries":

```
-lcl
```

The default stack size for posix threads is 64 Kilobytes. This may not be adequate. Hence, it is recommended that depending on the application, the stack size for threads be increased to 512 Kilobytes or more. The `pthread_attr_setstacksize()` call should be used before creating the thread. Please refer to the HP-UX 11i documentation for more details.

If you use the C++/Versant Check Facility

The C++/Versant checking feature does not work correctly with shared libraries.

If you want to use the checking feature (by compiling with `-DCHECKING` and using the libraries under `../lib/cxxchk`) you must link your application statically, using the following `aCC` option:

```
-Wl,-a, archive
```

Expect "unsafe cast" warnings

Expect "unsafe cast" warnings when using C++/Versant header files.

Since we can't filter out specific warnings, you can use the `-w` option to turn off warnings altogether once your source code is stable.

Warnings observed during compilation of C++, ODMG demos and Tutorials

The following warnings observed during compilation are harmless and should be ignored:

1. If compilation is done without the flags specifying the architecture of the machine (e.g.,`+DA1.1, +DA2.0W,+DAportable` etc) then warnings indicating the behavior of the linked output may be thrown.
2. Warnings about implementation of placement delete operators.
3. Future errors/warnings.
4. Deprecation warnings of implicit conversion of string literal to `char*`.

PLATFORM RESTRICTIONS

The C++/Versant checking libraries may experience problems.

The C++ checking libraries may experience problems with some applications. If you experience problems, please retry the same operation with the corresponding non-checking libraries. Please report all problems to Versant Customer Support.

The checking libraries for the fundamental classes are:

```
archive – /lib/cxxchk/libcxxcls.a
shared – /lib/cxxchk/libcxxcls.sl
```

The checking libraries for the collection classes are:

```
archive – /lib/cxxchk/libvcoll.a
shared – /lib/cxxchk/libvcoll.sl
```

The parent object must exist in the database for object level clustering.

If the parent object is newly created, you must explicitly write it out before using it in a group write clustered operation.

Clustering more objects than will fit on a page will cause objects to be stored on a new page.

The parent attribute is not stored in the object cache.

You must explicitly call group write clustered.

Clustering works only for objects in the same storage file.

If you want to cluster objects of different classes, use the `o_cluster` API to put the classes in the same storage file. This API must be called before any instances are created.

Embeddability library.

```
libembed.sl
libembed.a
```

Applications linking with these libraries need to be linked with the HP ANSI Compiler aCC A.03.57 in order to resolve all the symbols present in this library. This is necessary since the embeddability library is a C++ library.

VAR Utilities

VAR executables (`varinit, varchadmin` etc) will not work under HP ANSI environment.

To run VAR demos, use: `make run_nonansi`

To build VAR demos, use: `make nonansi`

CHAPTER 6

Platform Notes for AIX

This Chapter gives detailed explanation on the system requirement and platform restrictions for AIX Platform.

The Chapter covers the following in detail:

- System Requirements
- Using C++/Versant with AIX Compiler xlC_r
- Certification Notes
- Compiling and Linking
- Platform Restrictions

SYSTEM REQUIREMENTS

These Platform Notes contain information specific to Release 7.0.1.4 for IBM AIX 5.2.

Please check your architecture, operating system release, and compiler version before installing Versant Object Database.

Hardware

Versant Object Database, Release 7.0.1.4 for IBM AIX 5.2 has the following hardware requirements:

- Homogeneous IBM RS/6000 environment.
- Either a Token Ring or a High Performance Ethernet network controller using TCP/IP network protocol.
- Minimum disk space of `270` megabytes.
- Minimum system memory of `8` megabytes. System memory of `16` megabytes is strongly recommended.
- A recommended paging space to RAM ratio of `3` to `1`.
- Optionally there should be raw partitions available for log and database volumes. Use of raw partitions can improve performance dramatically, but your disk must be pre-configured to support them.

For other requirements, such as the minimum configuration for your UNIX kernel- See also “Pre-Installation Checklist” on page 180 in "Chapter 8 - Installation on Solaris/ Linux/ HP-UX/AIX".

Software

Versant Object Database, Release 7.0.1.4 for IBM AIX 5.2 has the following software requirements:

- You need IBM AIX Version 5.2.
- AIX Version 5.2 or greater includes all of the necessary system software and optional program products to support Versant. You must install the following optional program products:

  - Network Support Facilities
  - Base Applications Development Toolkit
  - BOS Extensions 1
  - BOS Extensions 2
- `libc` version: 5.2.0.14 or higher
- `libpthread` version: 5.2.0.12 or higher
- TCP/IP network protocol (if you want to connect to a remote database or make more than one connection to a local group database).
- To use C++/Versant Interface

  You should have VisualAge xlC_r compiler Version 6.0.0.0.

  To check the compiler version, run

  ```
  lslpp –la | grep –i compiler
  ```

  Following output should be displayed:

`vac.C`	6.0.0.0 COMMITTED C for AIX Compiler COMMITTED C for AIX Compiler 6.0.0.10 COMMITTED C for AIX Compiler
`vacpp.cmp.core`	6.0.0.0 COMMITTED VisualAge C++ Compiler COMMITTED VisualAge C++ Compiler 6.0.0.11 COMMITTED VisualAge C++ Compiler
`vacpp.cmp.include`	6.0.0.0 COMMITTED VisualAge C++ Compiler Include COMMITTED VisualAge C++ Compiler Include 6.0.0.9 COMMITTED VisualAge C++ Compiler Include

- To use graphical utilities, you must install IBM AIX windows Environment/6000 1.2 (included with AIX 5.2).
- This release requires use of compiler independent programming, as described in the *C++/Versant Reference Manual*.
- To use Java Versant Interface, an installation of IBM JDK 1.4.2 is required.
- To use Versant Asynchronous Replication (VAR) Java APIs, you need IBM JDK 1.4.2 for AIX.
- To use Versant Asynchronous Replication C++ APIs and ODBMS Release 7.0.1.4, you need IBM Compiler xlC_r 6.0.0.0
- To use Versant XML Toolkit (VXML), you need IBM JDK 1.4.2 and JSWDK 1.0.1. (You may use other webserver also.)

- To use Versant JDO, you will need Sun JDK 1.5

Environment and Application Requirements

Shared memory

Due to the way AIX shared memory works, if you link Versant libraries with an application that uses shared memory, you should begin your Versant database session with a "begin session" command before the application attaches to its shared memory.

Threads

Before running the applications that use `pthread` library, you should set the environment variable `AIXTHREAD_MNRATIO` to the value 1:1.

Utility Requirements

`vcopydb` – To ensure enough heap size for `vcopydb`, set and export the environment variable `LDR_CNTRL` with the following value : `LDR_CNTRL=MAXDATA=0x30000000`.

USING C++/VERSANT WITH AIX COMPILER xlC_r

Versant support for ansi flag for xlC_r 6.0.0.0 on AIX 5.2

All the code compiled with `-qlanglvl=ansi` option will confirm to the C89 standard for C programs and the ANSI C++ standard for C++ programs. Currently we provide the (`-qlanglvl=ansi`) support to C++ applications linked with Cxx libraries. Hence, customers will be required to make the following changes:

1. New set of libraries have been introduced in the ansi directory of `<VERSANT_ROOT>/lib`. These are in addition to the normal ones at `<VERSANT_ROOT>/lib`. In case of internationalization libraries (e.g. `libicuuc.so`) and ODBMS libraries (e.g. `liboscfe.so`), the same copy could be used irrespective of the use of `-qlanglvl=ansi` option.
2. Applications compiled with `-qlanglvl=ansi` option should use the ANSI version of library. This can be done by specifying `-L<VERSANT_ROOT>/lib/ansi` before `-L<VERSANT_ROOT>/lib` while linking. Optionally, it can also be done by setting the environment variable `LIBPATH` to `<VERSANT_ROOT>/lib/ansi:<VERSANT_ROOT>/lib`. Linking the application compiled with the `-qlanglvl=ansi` option to the default cxx libraries may result in unpredictable behavior. Applications which are not compiled with this option of `xlC_r` should be linked with the default libraries in `<VERSANT_ROOT>/lib` directory.
3. If the existing applications are to be made ANSI compliant, they will have to be re-compiled with the `-qlanglvl=ansi` option and then be linked to the libraries in `<VERSANT_ROOT>/lib/ansi`.
4. Variable `LIBPATH` must be set to `<VERSANT_ROOT>/lib/ansi:<VERSANT_ROOT>/lib` while executing the Versant C++ ANSI compliant applications.
5. The schema compiler (`schcomp`) has been updated to include the `-qlanglvl=ansi` option. Using this option we generate strict ANSI code in `schema.cxx` file. The resulting `schema.cxx` file should also be compiled with the `-qlanglvl=ansi` option and linked by specifying `-L<VERSANT_ROOT>/lib/ansi` before `-L<VERSANT_ROOT>/lib`. Optionally, it can also be done by setting the `LIBPATH` to `<VERSANT_ROOT>/lib/ansi:<VERSANT_ROOT>/lib`.
6. Demos and tutorials provided with the product can also be run in ANSI mode by setting the variables `ANSILIB` to ansi and `ANSIFLAG` to `-qlanglvl=ansi` in the environment. Other details are mentioned in the readme file provided with the demos.
7. Support to VED: While building custom drivers the `-qlanglvl=ansi` compilation option should not be used.
8. Any application using the VAR C++ APIs should not use the `-qlanglvl=ansi` compilation option.

Directories and Files

Changes in Library Directories

LIBPATH

After installation, set the environment variable LIBPATH to point to directory:

<VERSANT_ROOT>/lib

The /lib directory contains system libraries. The <VERSANT_ROOT>/lib directory contains C++ libraries for use with xlC_r 6.0.0.0.

If you are using ANSI libraries, set the environment variable LIBPATH to point to the two directories:
<VERSANT_ROOT>/lib/ansi
<VERSANT_ROOT>/lib

If you are using C++ checking libraries, set the environment variable LIBPATH to point to the directory:

<VERSANT_ROOT>/lib/cxxchk

<VERSANT_ROOT>/lib

If you are using ANSI checking libraries, set the environment variable LIBPATH to point to the two directories:
<VERSANT_ROOT>/lib/cxxchk/ansi
<VERSANT_ROOT>/lib

The environment variable LIBPATH tells Versant where to find the shared libraries at runtime. This variable must be set, because some of the Versant utilities use shared libraries. If you are using shared libraries and set LIBPATH to the wrong set of libraries, your application program will crash.

CERTIFICATION NOTES

AIX 5.3, xlC_r 7.0, 8.0 and 9.0 Certification

Versant Object Database 7.0.1.4 is certified on AIX 5.3 with Visual Age Compiler 7.0, 8.0 and 9.0

- OS level: AIX 5.3.0.0
- The Visual Age Compiler version is 7.0.0.0 or 8.0.0.0 or 9.0.0.0 for C and C++
- JDK version: 1.4.2
- `libc` version: 5.3.0.11 or higher
- `libpthread` version: 5.3.0.10 or higher
- To use CXX checking libraries set the environment variable `LIBPATH` to `<VERSANT_ROOT>/lib/cxxchk`
- To use ANSI libraries set the environment variable `LIBPATH` to `<VERSANT_ROOT>/lib/ansi` and to use ANSI CXX checking libraries set `LIBPATH` to `<VERSANT_ROOT>/lib/cxxchk/ansi.`
- The new identifier "__ExternStaticData" introduced in xlC 9.0, is treated as the storage class "extern" by the schema compiler.

CLARiiON CX & EMC PowerPath

Versant 7.0.1.4 is certified to run HABACKUP with CLARiiON CX family & EMC PowerPath.

COMPILING AND LINKING

Shared Libraries

This release is shipped with both static and shared libraries. You can use either the static or shared libraries for any application.

The static libraries have the suffix `.a`; for example `libosc.a`. The shared libraries have the suffix `.so;` for example, `libosc.so`.

Compile and Link Options

While compiling and linking you must pass the following options:

```
-D_REENTRANT -DTHREAD_SAFE -D_THREAD_SAFE -q32 –qnotempinc
```

If the `-q32` option is not specified and the `OBJECT_MODE` environment variable is not set, the compiler defaults to 32-bit output mode.

All C++ applications must be compiled with the `-qnotempinc` option.

While compiling you must pass the following option:

```
-qalign=natural
```

To resolve thread library symbols use following linker option:

```
-lpthreads
-qmkshrobj[ = priority]
```

This option together with the related options should be used instead of `makeC++SharedLib` command.

PLATFORM RESTRICTIONS

Memory limits.

If you are using large databases, you may have to change the values for the `data`, `rss` and `stack` parameters in the file `/etc/security/limits` to -1 (to indicate unlimited size).

You can catch a signal indicating low memory.

When process memory is low, the AIX kernel will send the signal `SIGDANGER`. If process memory runs out entirely, the kernel will send `SIGKILL` and kill the process.

To avoid having your process killed when you run out of memory, you may want to write code that traps the `SIGDANGER` signal and then deals with the situation, perhaps by exiting gracefully.

Restrictions on memory usage.

Shared memory segments from 8 to F are not available to 1P linked applications as it is used internally by Versant.

While running applications set the environment variable `EXTSHM` to ON (and not to “1”) to enable the use of the extended shared memory capability.

To ensure that a process is not killed due to low paging space condition, set the environment variable `PSALLOC=early` and export it.

Shared Memory Limit

Versant reserves not more than 1 GB of shared memory area for its applications.

Default Constructor needs to be provided for persistent classes

For a persistent class, if the user does not provide any constructors then the compiler will generate a default constructor. However, on AIX the compiler generated code is incorrect and results in a Versant runtime exception, hence the user must supply at least one constructor (which could be a default constructor) and not rely on the compiler generated default constructor.

The C++/Versant checking libraries may experience problems.

The C++ checking libraries may experience problems with some applications. If you experience problems, please retry the same operation with the corresponding non-checking libraries. Please report all problems to Versant Customer Support.

The checking libraries for the fundamental classes are:

archive — `<VERSANT_ROOT>/lib/cxxchk/libcxxcls.a`

shared — <VERSANT_ROOT>/lib/cxxchk/libcxxcls.so

The checking libraries for the collection classes are:

archive — <VERSANT_ROOT>/lib/cxxchk/libvcoll.a

shared — <VERSANT_ROOT>/lib/cxxchk/libvcoll.so

Linker warnings when building C++/Versant executables

If a C++/Versant application uses one of the Versant template classes such as `BiLink`, `BiLinkVstr` or any of the C++/Versant collection classes, the linker may generate warnings for duplicate symbols at the time of linking the application executables. The reason for this is that the C++ compiler generates template member functions in each object file that references a given template class. This results in multiple instantiations of the same template member function in different object files.

The linker detects these multiple definitions for the template member functions and displays several warning messages resembling the following at the time of linking the executable:

```
ld: 0711-224 WARNING:
Duplicate symbol: .BiLinkVstr<Document>::add(Document*)

ld: 0711-224 WARNING:
Duplicate symbol: .BiLink<Document>::operator=(Document*)
ld: 0711-224 WARNING:
Duplicate symbol: .BiLink<Document>::BiLink(Document*)
```

C++/Versant users should note that these warnings do not indicate an error, and are just a side effect of the way IBM's C++ compiler handles template classes.

Include header files using `extern "C"`

Some of the header files on AIX 5.2 are not C++ aware, so the `extern "C"` linkage specifier must be used when including the system header files. For example:

```
extern "C" {
#include <ctype.h>
#include <stdio.h>
#include <unistd.h>
}
```

If you do not use `extern "C"` you could get compiler or linker errors when building your application. IBM will fix this problem in a later release of AIX.

Template Instantiation warning in xlC_r 6.0.0.0

The users might encounter the following warning when `rev_iter.h, bag.h` are included during compilation:

The friend function declaration "`operator+`" will cause an error when the enclosing template class is instantiated with arguments that declare a friend function that does not match an existing definition. The function declares only one function because it is not a template but the function type depends on one or more template parameters.

The above warning can be neglected as it is generated due to the friend declarations in the templates, as the `xlC_r 6.0.0.0` compiler needs explicit template arguments. For more information see the AIX xlc documentation.

Use numerical positioning with vbackup.

The on-line incremental backup utility `vbackup` has 3 options for positioning the tape:

`Number`	A non-negative number. The first position on the tape is `0`, the second is `1`, and so on. If you are using a file for the backup, the only valid position is `0`.
`Current`	Use the keyword `current` to use the current position of the tape.
`Append`	Use the keyword `append` to write after the last file on the tape.

Due to differences in the IBM tape driver, **only tape position `0`** can be used by `vbackup`. You will have to use separate media for the first volume of each on-line incremental backup.

Tape drive settings

If you have problems reading the release tape, please configure your tape drive to `2.3 GB` mode with a block size of `1024`.

CHAPTER 7

Installation on Windows Platform

This Chapter gives detailed explanation on how to Install Versant Object Database on “Windows Machine”.

The Chapter covers the following in detail:

- Installing Versant Object Database
- Pre Installation Checklist
- Installing Versant Using GUI
- Installing Versant in Silent mode
- Installed Directories and Files
- Steps Post Installation
- Known issue with UnInstaller

INSTALLING VERSANT OBJECT DATABASE

This chapter explains the installation of the Versant Object Database on a Windows machine. It explains the directory structure, install panels, directories and files during installation. Some of these may/may-not apply to your installation as this depends on the installation components you choose and the mode you install.

Among other things, this chapter also discusses the locations for your software root directory, database root directory and database system identifier file.

For more information, on what these directories and files are, please refer to Chapter "Directories and Files" in the *Versant Database Administration Manual*.

Versant Object Database comes with a common, integrated installer which offers a single GUI interface to install various components. This integrated installation eliminates version conflicts when installing and deploying components of Versant Object Database. Some of the components within Versant Object Database are licensed individually and are node locked.

Versant Object Database 7.0.1.4 consists of the following components:

Standard components:

COMPONENT	DETAILS
Versant ODBMS	Versant Object Database Management System
GUI Tools	Versant GUI Tools
JDO Interface	JDO Interface for Versant ODBMS
Java Versant Interface	Java Language Interface for Versant ODBMS
C and C++/Versant	C and C++ Language Interface for Versant ODBMS

Add-on components:

COMPONENT	DETAILS
VAR	Versant Asynchronous Replication for Versant ODBMS.
Vedding	Fault Tolerant Versant ODBMS Server.
HABACKUP	Backup Solution for use with High Availability Server
Vorkout	Versant Online Database Reorganization Tool
ReVind	Structured Query Language Interface for Versant ODBMS
VODBC	Versant ODBC driver

Warm Standby	Used for an Incremental Rollforward recovery

NOTE:- All the Add-on components require an additional separate license.

Versant installation allows you to choose between Typical and Custom installation.

- **Typical install** - Installs all components of Versant Object Database with Demos, Tutorials and Documentation.
- **Custom Install** - Allows user to select the required components for installation.

The installer can be run in Super user mode as well as Non-super user mode.

For users with 'Administrator' privileges, the mode is super-user mode and for all other users the installation mode is selected as non-super user mode.

Users running applications in which Versant is embedded are likely to choose the Silent mode of installation. In this mode, the user presets a number of configuration parameters and directs the installation procedure to use those parameters.

For more information, See also “Installing Versant in Silent mode” on page 145.

PRE-INSTALLATION CHECKLIST

If you are installing Versant for the first time, you need to know and decide a few important things before the installation process begins.

Following is a checklist for machines running on Windows OS:

Review your hardware and software requirements

Before starting the installation, you should view the list of hardware and software requirements and confirm that it follows the required specification.

Refer to chapter “Platform Notes" for platform specific information on installation.

Decide the software root directory

Versant will be installed under the selected software root directory in a pre-determined set of platform, release and software subdirectories.

During installation, you need to decide the software root directory else it will be installed at the default location.

The default location is: `<system drive root>\versant\7_0_1`

NOTE:- In the examples below, system drive root is assumed as `C:\` i.e. default location for software root directory is `C:\versant\7_0_1`

From this point onwards, we will refer to the installation directory as '`software_root`'.

Decide the database root directory

The installer will create database directories and information files under the database root directory (called `<db_root>`).

During installation, you need to decide the database root directory else it will be installed at the default location.

The default location is: `c:\versant\db`

After installation, all the information about the newly created database will be placed in a subdirectory created under the database root directory. (Although the database volumes themselves can be anywhere.)

You can specify either a new database root directory or use an existing database root directory created from a previous installation of Versant.

NOTE:- If you are going to be the sole user of the Versant installation and the databases created by this installation, then you can choose the software root and database root directory of your own choice.

If others are going to use this installation, then you should choose a directory visible to others on a network.

The software root directory and database root directory can be local or remote. If it is remote, you must be able to access it with the same user name that you use on your local machine.

Decide the database system identifier file (osc-dbid)

The name of the database system identifier file is `osc-dbid`.

The database system identifier file, `osc-dbid -` must exist and be accessible to your machine before you can create a database. This file contains information about all databases that you and others might conceivably access during a particular database session.

During installation, you will be asked for the location of the `osc-dbid` file.

For new installation, you will need to create an `osc-dbid` file.

For old installations, you need to associate to the existing `osc-dbid` file.

To Get information about a existing `osc-dbid` file

If you are joining an existing system of databases on a network, then specify the location of the `osc-dbid` file already in use.

To find the location of an existing file, you can go to the machine containing it and run the `oscp` utility as:

```
oscp -i
```

To Get the machine name

To find the name of the machine containing the `osc-dbid` file, run the `oscp` utility:

```
oscp -n
```

To Get the directory name

To find the name of the directory containing the `osc-dbid` file, run the `oscp` utility:

```
oscp -o
```

Decide the osc-dbid node and path

If you are creating a standalone installation, then you need to specify the node name of your local machine.

During installation, you will be asked for the node name and path of the `osc-dbid` file.

If you are creating a new network of databases or joining an existing network of databases then you can specify a remote machine name. In this case, the installer will not create a new `osc-dbid` file. The user will have to manually specify the dbid node on which the osc dbid file is residing. However, it will record it in the configuration files generated during installation.

Normally, the `osc-dbid` file will be on a machine acting as a server and will be already created by the time you finish the installation on a Windows machine.

The purpose of the `osc-dbid` file is to ensure that each database has a unique identifier number in a network system of databases. This is important, because the object model requires that each object has an identifier number unique among all other objects, regardless of database. When you create an object, the object is given an identifier number composed partly of the database identifier and partly of a number unique to the database.

You will not need to access the database system file when you create an object or connect to a database. You will need to access the database system file when you create a database or ask for information about any or all databases in a network system of databases.

If you are creating a standalone installation that will never use objects in databases on other machines, the installer will by default create an `osc-dbid` file in your Versant database directory.

Specify the future location of the database system file

If you want to be a part of a system of databases that has not yet been created, then specify the future location of the database system file. (However, you will not be able to create a database until the database system file exists.)

The default location for an `osc-dbid` file is database directory.

If you use the default for database directory then it will be `<db_root>, i.e. <software_root>\..\db.`

The database system file can be local or remote.

If it is local, it must be visible to all others who will create databases in a network system of databases.

If it is remote, you must be able to access it with the same user name that you use on your local machine.

Since there is only one `osc-dbid` file for a system of databases, typically for most users it will be on a remote machine.

The installer will not check your choice of a location for a database system identifier file to see if it exists. Thus, if you want to create an `osc-dbid` file, you will have to do so after installation.

During installation, your environment will be altered so that this installation of Versant knows where to look for the `osc-dbid` database system identifier file.

If you specify an incorrect location for the `osc-dbid` file during installation or later move the `osc-dbid` file, you can correct it later by modifying a system file or environment variable.

CAUTION:- Please take care while creating the `osc-dbid` file as a mistake can cause a lot of rework! You will have to go back to each and every installation and re-specify the path to the `osc-dbid` file.

For more information, on specifying the location of `osc-dbid` after installation, please refer to Chapter "Configuration Parameters" in the *Versant Database Administration Manual*.

Recommendations

- If you are making multiple, new installations, we strongly recommend that you first install on the machine that will act as your database system host. This will ensure that you specify a correct location for the database system file when you make each installation.
- Under some circumstances, you may want to create multiple, separate systems of databases and then change your environment when you connect to one system or the other. However,

since the `osc-dbid` file is not read each time that you make a database connection, you could conceivably create simultaneous connections to databases belonging to differing systems.

CAUTION:- Please take precautions while doing this, because databases in differing systems could have the same identifier number, and thus there could be an object in the differing systems with the same logical object identifier.

Decide on an Installation Mode

The installer can be run in two modes: One with Administrative privileges and other without Administrative privileges.

These modes will depend on the privileges of the user installing Versant. These privileges can be set by the Administrator. If the installation mode is not set, the installer will run with the Administrative privileges mode. The installation panel displayed and the tasks performed by the installer will depend on this installation mode.

If you are installing Versant as user with Administrator privileges, please refer to "Update System Services File " and "Install Windows Service VersantD " on page 142.

For Reinstallation

If you want to re-install on top of the existing installation, stop all databases by using `stopdb` utility. On Windows, you must stop the versantd/VersantAgentService(Vitness)/Versant SQL Listener(VSQL) services if running. Then proceed as if you were installing a new system.

KNOWN ISSUE WITH THE INSTALLER

OS Compatibility

An exception "`ArrayIndexOutOfBoundsException`" is thrown while installing Versant 7.0 on Windows Server 2003, which stops the installation process.

For successful installation on Windows Server 2003, set the compatibility mode as Windows 2000 for the installer before running it.

Screen Resolution

On a computer with higher screen resolution, it might occur in rare circumstances that the installer hangs. This is a known bug with the InstallAnywhere software. In such a case, change the screen resolution to a smaller value.

INSTALLING VERSANT USING GUI

This section explains how to install Versant on a Windows machine using the Versant GUI.

For an existing user -

1. **Stop all databases on the installation machine**.

   If you are updating an existing installation of Versant, then before installing Versant Object Database, use your existing `stopdb` utility to stop all databases on the installation machine and then proceed for installation as below.

   For example, for a database named `mydb`, do the following:

   ```
   stopdb mydb
   ```

   For more information, on `stopdb`, please refer to the Chapter "Database Utilities" in the *Versant Database Administration Manual*.

For a new user -

2. **Invoke the installer**.

   Place the Versant installation CD in the CD-ROM drive on the installation machine and click on the `install.exe` to initiate a GUI-based procedure for Versant installation.

   NOTE:- If the `install.properties` file is also located in the same directory, then the installation will be performed in Silent mode. This file must be removed to install Versant in the GUI mode.

   For further details on silent installation, please refer to the section "Installing Versant Object Database in Silent Mode".

   The installer uses your system's temporary directory to extract the system files. If temporary directory on your machine does not have the required space, then the installer will ask for another directory where it can temporarily extract the files.

3. **Go through the navigation panels**

   The installer consists of several panels. Most of the panels will ask you to provide input. For each question, a default choice is presented before you.

   To terminate the installation at any time, you can press '`Cancel`' button on any of the panels. On some of the panels, you can press the '`Help`' button to get help on that topic. Pressing

the 'Help' button pops up a new window with the help text. To go to the next panel and the previous panel, press the 'Next' button and the 'Previous' button on the panel.

The following panels are displayed during Installation:

Installer Panels

Splash Panel and Introduction

The Installer starts with a Splash panel. Splash panel disappears automatically without any user interaction. This is followed by an Introduction panel which introduces you to the Versant Object Database installation. Choose 'Next' to continue the installation.

License Agreement

The license agreement panel describes the terms and conditions of the use of the software being installed. You can proceed with the installation only if you accept the license agreement.

Important Information

The important information panel gives an overview of all the components that come along with this release. Click 'Next' to continue the installation.

Choose Install Folder

Specify the location of your installation root directory (called <software_root>).

If you specify a directory that does not exist, the installer will create it. Ensure that you have write permissions for the directory path you specify.

The default location for the installation directory is

<system drive root>\versant\7_0_1

For example c:\versant\7_0_1

Choose Shortcut Location

Choose an appropriate option or specify the location where the shortcuts for the product will be created. Click 'Next' to continue the installation.

Choose Install Set

Versant Object Database Installer provides users with two installation sets: Typical and Custom. Select the required installation and click 'Next' to continue the installation. More information about these installations is given in the following sections.

Typical Installation

Choosing this, the user will be able to install the most commonly used installation choices. Some of the components are implicitly installed, depending upon the users choice. For example, Versant Asynchronous Replication will also install Versant ODBMS. The reason for this is that for a specified installation option, the installer automatically determines the components on which the specified component depends, and so all the dependent components are also installed by the installer.

Under Typical installation, Versant Object Database currently offers following installations:

Versant C/C++ Interface: This includes the development environment for Versant C/C++ Interfaces. This is the complete development environment for applications using Versant C/C++. It necessarily includes Versant ODBMS. It also includes demos, tutorials and documentation for both.

Versant GUI Tools: This includes the Versant GUI tools (Versant Administration Console, Versant Object Inspector, Monitoring Console and Versant Developer Console). It also includes the documentation for all the installed components.

Versant JDO Interface: This includes the JDO Interface for Versant ODBMS. This is the complete development environment for application using JDO. It also includes demos and documentation.

Versant Java Interface: This includes the Java Language Interface for Versant ODBMS. This is the complete development environment for application using Java Versant Interface. It includes tutorials, demos and documentation.

Versant Asynchronous Replication (VAR): This includes the Versant Asynchronous Replication component for ODBMS. This is the complete development environment for application using VAR. It includes Versant ODBMS. It also includes demos and documentation.

Versant Vedding (FTS): This includes the Fault Tolerant Server (FTS) and its documentation.

Versant High Availability Backup: This includes Versant's High Availability Backup and its documentation.

Versant Vorkout: This includes Versant's Online Database Reorganization Tool and its documentation.

Versant Warm Standby: This includes the Versant's Warm Standby and its documentation.

Versant ReVind: This includes Structured Query Language Interface for Versant ODBMS and its documentation for Win-32 only.

Versant ODBC: This includes Versant ODBC driver and its documentation for Win-32 only.

NOTE:- Versant ReVind & VODBC is currently not supported on windows 64bit platform.

Custom Installation

If your installation needs are different than what is provided in the Typical installation set, then you can customize your installation.

For customizing, choose '`Custom Install`' on the 'Choose Install Set' Panel and click "`next`" to bring up 'Custom Installation' Panel.

Custom Installation Panel gives you the flexibility to choose components at finer granularity. In the top panel, you will find a 'choice box'. This lists all the installation sets described above in the Typical Installation. When you click on any one of the options in the top pane, its description is displayed in the bottom pane.

In case you want to modify the previously installed components, you can add/remove these components. For adding/removing a component, check on box before the component name. Only the components, which are checked, at the time you press '`Next`' button, will be installed.

Here, the user needs to select the components individually for installation under the "Runtime/ Developer" category.

Versant ODBMS

- Runtime Only - It installs Versant ODBMS Server, Database Tools and Utilities. This installation is useful for deploying Versant Server on a machine.
- Developer - It installs C/Versant language interface development environment. It includes header files and static libraries. It also includes documentation, demos and tutorials.

Versant GUI Tools

- This installs the Versant GUI tools (Versant Administration Console, Versant Object Inspector, Monitoring Console and Versant Developer Console). It also includes the documentation for all the installed components.

Versant JDO Interface

- Runtime Only - It installs elements required to run applications using JDO interface for Versant ODBMS. This installation can be considered as the JDO/Versant client only installation since it does not include Versant ODBMS Server component. It also includes Database Tools and Utilities.
- Developer - It installs developer environment for JDO Interface for Versant ODBMS. It also includes documentation, demos and tutorials.

Java Versant Interface

- Runtime Only - It installs elements required to run applications using Java language interface for Versant ODBMS. This installation can be considered as the Java Versant client only installation since it does not include Versant ODBMS Server component. However, it includes Database Tools and Utilities.
- Developer - It installs developer environment for Java Language Interface for Versant ODBMS. It also includes documentation, demos and tutorials.

Versant C/C++

- Runtime Only - It installs elements required to run applications using C/C++ language interface for Versant ODBMS. This installation can be considered as the C/C++/Versant client only installation since it does not include Versant ODBMS Server component. However, it includes Database Tools and Utilities.
- Developer - It installs developer environment for C/C++ Language Interface for Versant ODBMS. It also includes documentation, demos and tutorials for the same.

Versant Asynchronous Replication (VAR)

- Runtime Only - It installs elements required to run applications using VAR component for Versant ODBMS. This installation can be considered as the Versant client only installation since it does not include Versant ODBMS Server component. However, it includes Database Tools and Utilities.
- Developer - It installs developer environment of VAR component for Versant ODBMS. It also includes its documentation and demos.

Versant Vedding (FTS)

It installs the tools required for replication and its documentation.

Versant High Availability Backup (Habackup):

It installs the elements, which can be used to take backup on high availability server and its documentation.

Vorkout:

It installs the elements required for reorganizing the data in order to reduce fragmentation and to enhance performance along with its documentation.

Warm Standby:

It installs Versant's Warm Standby and its demos, tutorials and documentation.

Versant ReVind:

It installs the Structured Query Language Interface for Versant ODBMS and its documentation.

Versant ODBC:

It installs Versant OBC driver and its documentation.

Pre-installation Summary

This panel lists the summary of the installation with information about the root directory and the component which will be installed. The files and directory structure created in your system will depend on the components you have selected.

If you need to change anything in the installer, you can do so now by clicking on the "Previous" button, else click "Install" to proceed with the installation. Now the installer actually starts copying the files to your system.

The following Runtime system panel can be seen for installer with Windows 64bit support.

Installing RunTime System

Versant Object database applications will require runtime system of VC++ 2005 SP1 and higher to run. Versant Object database installer is bundled with the redistributable executable of VC++2005 SP1. If you do not have required compiler version installed on your machine, you should install the runtime system.

Database Root Directory Folder

The installer asks where the database root directory (called `<db_root>`) should be located.

The default location is the parent directory of the installation directory. That is: `<software_root>\.\db, i.e. c:\versant\db.`

If the directory you specify does not exist, then installation will create it. Make sure that you have the `WRITE` permission where you want this directory to be created. If Installation cannot create this directory, then it will display an error.

Even in the case of error, you can proceed with the rest of the installation. However, you need to create the database directory manually after the installation, set appropriate directory permissions, and tell Versant where it is by updating your system configuration files such as `sysinfo` or by setting appropriate environment variables.

Create osc-dbid File

Before proceeding, you need to decide whether this installation would create a new `osc-dbid` file or work with the existing one.

If yes and you want to create a new network of databases then you may opt to create a new `osc-dbid` file. But, if you want your installation to be part of existing network of databases then please select '`No`' as your choice to create a new `osc-dbid` file.

The purpose of the database system file is to ensure that each database in a network system of databases has a unique identifier number. This is important, because the object model requires that each object have an identifier number unique among all other objects, regardless of database. When you create an object, the object is given an identifier number composed partly of the database identifier and partly of a number unique to the database. You will not need to access the database system file when you create an object or connect to a database.

You will need to access the database system file when you create a database or ask for information about any or all databases in a network system of databases.

Choose osc-dbid Node and osc-dbid PATH

The installer needs to get:

a. the name of the node where the `osc-dbid` file will reside, and
b. the path name where the `osc-dbid` file will reside.

If you had decided to create new `osc-dbid` file, then installer will use the above information to create a new `osc-dbid` file at the path you specify. If the path you specify does not exist then installation will create whole path before creating `osc-dbid` file. You should have write permission if you specify a remote host name as node name, then Installation will not create `osc-dbid` file, even if you had selected the option to create it. Also, if the specified path for the `osc-dbid` file is NFS mounted then Installation might fail to create `osc-dbid` file. If `osc-dbid` file already exists at the path specified then installer does not create it.

Even in case of failure in creating `osc-dbid` file, dbid node name and path will be recorded in the configuration files generated during installation. After installation, you can run '`dbid -N`' (after setting environment variables) to create the `osc-dbid` file yourself. In cases, where `oscssd` entry could not be made in the services file, you may need to set environment variable `VERSANT_SERVICE_PORT` to some port before executing '`dbid -N`'. This is especially useful when you are installing Versant as user who does not have Administrator privileges.

Even if you did not choose to create an `osc-dbid` file from the earlier panel, installation will still ask for the `osc-dbid` node and `osc-dbid` path. By doing this, you are essentially asking installation to put this information in the configuration file that it generates.

Installation does not check whether `osc-dbid` file exists at the path you specified, if you did not choose to create it.

Default values for `osc-dbid` node and `osc-dbid` path are local machine name and `<software_root>\db` respectively.

Update System Services File

If you are installing Versant as user with Administrator privileges, installation procedure prompts you whether you want it to update services of your system to include the port number for the Versant ss daemon.

The services file on a Windows system is located under:

```
<system drive root>\WinNT\system32\drivers\etc\services
```

Installation can add the following entry to the TCP/IP Services file:

```
oscssd 5019\tcp# versant connect service
```

Versant uses TCP/IP protocol to communicate between a Versant client and Versant Server. 5019 is the port number used for standard configuration. If the above entry is already present in the TCP/IP services file then Installation will not update the file.

To test the connection between client and server machine for Versant configuration, run the `itest` utility:

```
itest -v <hostname>
```

Install Windows Service VersantD

If you are installing Versant with Administrator privileges and you have conformed to update the system services file, then the installer will ask whether you want installer to install Windows service `VersantD` on your system. Installer will install `VersantD` service if you say yes to this question.

An MS-DOS command prompt window will pop up briefly and then disappear. This happens because the installation procedure generates and runs a batch file to install a new Windows service on your machine.

For more information please See also “Addition of new windows service "Versantd" ” on page 165.

Install Complete

This the last panel that confirms your installation to be complete. This also lists a brief summary describing the changes done to your system during the installation.

In case you are upgrading the product such that there is no change in the first two digits of the version, then the existing licenses can be used and no new licenses are needed.

The installer will get the machine details (`machine name, OS type` and identification numbers [`hostid, IPv4 address and IPv6 address`]) and print them to `VERSANT_ROOT\InstallSummary` file. These values are required to obtain the license records. These details can also be obtained by invoking the utility “`vinstinf`” present in the “`bin`” directory of the installation.

Versant executables and libraries need the C/C++ run-time libraries `MSVCR70.dll`, `MSVCP70.dll`, `MSVCI70.dll` and `dbghelp.dll`. If your system does not have these DLL's, the Versant installation procedure will copy them to your machine and in the summary tell you what has happened.

To start the `versantd` service, you can either reboot or start the `versantd` service with the following command issued from the command line:

```
net start versantd
```

Alternately, you can start the `versantd` service by opening Control Panel / Services, highlighting the `versantd` entry, and clicking on `Start`.

Restart the Computer

The installer changes the environment variables and registry entries of your system. A message window appears informing the user that after installing Versant, you can logoff and logon to allow changes to the system registry and environment variables settings take into effect. It is advisable to restart the computer after installation.

Configuration Done by the installer

Depending upon what installation options you choose, the installer generates configuration files, modifies environment variables and updates registry entries. These files are:

```
<software_root>\lib\sysinfo
<root>\WINNT\vr070001.ini
```

It generates a system information file.

It updates environment variables such as `PATH`, `VERSANT_ROOT`, `VERSANT_PATH` and `CLASSPATH`. These variables could be modified for the system or just for the user installing Versant depending upon the privileges of the user.

Installation also updates system's registry entries. These entries can be found under

`HKEY_LOCAL_MACHINE→SOFTWARE→Versant`

Post Installation

After installation, you may need to perform additional steps, such as creating a database (if you do not already have access to one) to run Versant.

For more information see “Steps Post Installation” section for steps to perform after installation.

INSTALLING VERSANT IN SILENT MODE

During silent installation, the installer is enabled to run without any user interaction.

This can be achieved by defining certain parameters in a configuration file (`install.properties`). The values of the parameters in the configuration file need to be correct or the installer will display error messages during installation.

To run the installer in silent mode, do the following:

```
install.exe -f install.properties
```

If `install.exe` is not in the same directory as the `install.properties` file, you need to specify the absolute path for the property file.

E.g,. install.exe -f c:\versant\install.properties

NOTE:- On Windows machine, two windows pop up even during silent installation.

Below is an example of the contents of `install.properties` file if the user selects Typical install and Additional Components options:

```
INSTALLER_UI=silent
USER_INSTALL_DIR=C\:\\Versant\\7_0_1

#Standard Components
ODBMS_DEV=TRUE
ODBMS_RT=TRUE
JAVA_RT=TRUE
JAVA_DEV=TRUE
JDO_DEV=TRUE
JDO_RT=TRUE
CPP_RT=TRUE
CPP_DEV=TRUE
GUI_TOOL=TRUE

#Add-ons
HABACKUP=TRUE
FTS=TRUE
VAR_RT=TRUE
VAR_DEV=TRUE
WARM_STANDBY=TRUE
```

```
VSQL=TRUE
VODBC=TRUE
VCOMPACTDB=TRUE
#To Install VC++ 2005 SP1 redistributables for Windows 64bit
#installer.
INSTALL_REDIST=TRUE
#DB

#set CREATE_OSCDBID_ACTION to true if you install ODBMS, CXX, or VAR
#components and you want to create osc-dbid file
CREATE_OSCDBID_ACTION=TRUE
DB_FOLDER=c:\\versant\\db

#set OSCDBID_NODE and OSCDBID_PATH
# if CREATE_OSCDBID_ACTION=TRUE
OSCDBID_NODE=machinename
OSCDBID_PATH=c:\\versant\\db

#following parameters are used for Windows if you want to install
#versantd service.
#UPDATE_SERVICES_ACTION=TRUE
INSTALL_NT_SERVICE_ACTION=TRUE

# for UNIX Systems: DBSA_GROUP_ID and DBSA_LOGIN_ID are set when the
#user is root.
#DBSA_GROUP_ID=group
#DBSA_LOGIN_ID=name

# for UNIX Systems: only set the following parameters if the user is
#root and want to update services and inetd
#UPDATE_SERVICES_ACTION=TRUE
#UPDATE_INETD_ACTION=TRUE
#NOTIFY_INETD_ACTION=TRUE
#UPDATE_VSQL_SERVICES_ACTION=TRUE
```

INSTALLED DIRECTORIES AND FILES

This section provides an overview of the installed files and directories by the installer.

Root directory structure

The installer will create a software root directory (if it does not exist), other directories, such as `bin` and `lib`, and copy system files to the installation machine. The actual directory structure and the files copied into your system will depend on the installation options you choose during installing Versant. The details given below are based on a Typical Installation done by user with Administrative privileges.

After installation, you can move this directory, if you modify your environment to point to the new location. However, you should not change the names or contents of the subdirectories under the software root directory.

```
<software_root>
   |---ant
   |   |---bin
   |   |---lib
   |---bin
   |   |---ansi
   |   |---jvi
   |   |   |---1p
   |   |---memdir
   |---demo
   |   |---addons
   |   |   |---fts
   |   |   |   |---demo
   |   |   |---habackup
   |   |   |   |---demo
   |   |   |---var
   |   |   |   |---demo
   |   |   |---vcompactdb
   |   |   |   |---demo
   |   |   |---vsql
   |   |   |   |---demo
   |   |   |---warmstandby
   |   |   |   |---demo
```

```
|   |---c
|   |   |---demo
|   |   |---tutorial
|   |---cxx
|   |   |---advanced
|   |   |---hello
|   |   |---hellomfc
|   |   |---optlock
|   |   |---pntline
|   |   |---query
|   |   |---stl
|   |   |---tutorial
|   |---jdo
|   |   |---advanced
|   |   |---graph
|   |   |---inheritance
|   |   |---parentchild
|   |   |---queries
|   |   |---tut1
|   |   |---tutorial
|   |---jvi
|   |   |---advanced
|   |   |---IDE
|   |   |---odmg-trans
|   |   |---query
|   |   |---tutorial
|   |---vxml
|   |   |---demo
|---doc
|   |---addons
|   |   |---var
|   |---c
|   |---cxx
|   |---jdo
|   |---jvi
|---h
|   |---cxxcls
|   |---ilang
|   |---odmg
|   |---var
```

```
|---jre
|   |---bin
|   |---lib
|---lib
|   |---ansi
|   |---data
|   |---jdo
|   |   |---jca
|   |   |---templates
|   |---jvi
|   |   |---jca
|   |   |   |---jboss
|   |---vxml
|   |   |---dtds
|   |   |   |---v1.0
|   |---eclipse
|   |---schemas
|   |---help_isql
|---uninstaller
```

DIRECTORIES AND FILES AFTER INSTALLATION

There are two directories created after installation:

- Software root
- Database root

Software root directory

`<software_root>`

You can choose to place the Versant software root directory anywhere, but the default software root directory is:

`c:\versant\7_0_1`

Once you have specified a location for the software root directory, all other Versant software directories are built from it in a way that cannot vary.

ant directory

`<software_root>\ant`

This directory contains the stripped down Apache Ant 1.5 (minimal configuration).

bin directory (General executables)

`<software_root>\bin`

The executables directory contains command line utilities for administering Versant and invoking tools. Executables and dll's from all the components of Versant Object Database are copied here. It also includes dynamic link libraries for Java Versant Interface (JVI) and C++ debugging libraries for your compiler.

<software_root>\bin also contains the Batch files and script files for JDO as follows:

File name	Description
jdosetup.bat	To setup the JDO environment
enhance.bat	To enhance the JDO persistence capable files
workbench.bat	To launch the Workbench
schematool.bat	To invoke the schema tool
updatevoa32schema.bat	To invoke the Conversion Tool for VOA 3.2.x customers. Before invoking this tool, please make sure you have used convertdb utility to convert VOA 3.2 x databases to VOD 7.0 x databases.

The following is the list of launch scripts to be used to invoke the various GUI Tools:

GUI Tool Name	Launch Scripts
Administration Console	vdbadmin.bat
Object Inspector	vdbinspector.bat
Monitoring Console	vdbmonitor.bat
Versant Developer Console	vconsole.bat

<software_root>\bin\ansi

This directory contains the ANSI dynamic link libraries.

<software_root>\bin\memdir

This directory contains ReVind(Versant SQL) related files. The ReVind(VSQL) server and utilities use this directory to creates some temporary files.

demo directory

<software_root>\demo\addons

A directory containing subdirectories for all Versant add-on components like FTS, HABACKUP, VAR, VCompactdb, VSQL and Warm Standby and which contain the respective demos.

<software_root>\demo\C

A directory containing demos and tutorial for C interface.

`<software_root>\demo\CXX`

A directory containing demos and tutorial for C++ interface.

`<software_root>\demo\JDO`

A directory containing demos and tutorial for JDO interface.

`<software_root>\demo\JVI`

A directory containing demos and tutorial for Java Versant interface.

`<software_root>\demo\VXML`

A directory containing demo programs for VXML.

doc directory

`<software_root>\doc`

A directory containing all the documentation for Versant Object Database. This contains the following manuals:

- Versant Database Administration Manual
- Versant Database Fundamentals Manual
- Release Notes for Versant Object Database for 7.0.1.4

`<software_root>\doc\addons`

A directory containing documentation for all Versant Add-on components. This directory contains the following Versant Manuals in PDF format:

- Versant Vedding Usage Guide
- Versant High Availability Backup Usage Manual
- Versant Vorkout Usage Guide
- Versant/ReVind Reference Guide + Release Notes
- Versant/ODBC Reference Guide + Release Notes
- Versant Warm Standby Usage Guide

`<software_root>\doc\addons\var`
This directory contains the following Versant Manuals:

- Versant Asynchronous Replication (VAR) Manual
- VAR C++ Reference Manual (CXXdoc – HTML format)
- VAR Java Reference Manual (Javadoc – HTML format)

`<software_root>\doc\C`
A directory containing documentation for C Versant manuals in PDF format.

- C/Versant Reference Manual

`<software_root>\doc\CXX`
A directory containing documentation for C++ interface. This also contains an introductory demo for C++ interface. This directory contains the following Versant Manuals in PDF format:

- C++/Versant Reference Manual
- C++/Versant Usage Manual

`<software_root>\doc\JDO`
A directory containing documentation for JDO interface. This directory contains the following Versant Manuals:

- Versant JDO Interface Usage Manual (PDF)
- Versant JDO Interface Reference Manual (Javadoc – HTML format)

`<software_root>\doc\JVI`
A directory containing documentation for Java Versant interface. This also contains an introductory demo for JVI interface. This directory contains the following Versant Manuals:

- Java/Versant Interface Usage Manual (PDF)
- Java/Versant Interface Reference Manual (Javadoc – HTML format)

h directory (header files)

`<software_root>\h`

A directory containing the system header files such as `omapi.h` (which contains C/Versant functions).

`<software_root>\h\cxxcls`

A directory containing the C++ header files which contain the definitions of the database and library classes that comprise the C++/Versant interface.

`<software_root>\h\cxxcls\template`

A template directory containing definitions of C++/Versant template classes, such as the dictionary and array types.

`<software_root>\h\ilang`

A language information directory containing files with information about the native language in which you are programming.

`<software_root>\h\odmg`

A ODMG header files directory containing definitions of the database and library classes that comprise the ODMG C++/Versant interface.

`<software_root>\h\var`

A VAR header files directory containing definitions of VAR/C++ API classes.

jre directory

`<software_root>\jre`

This directory contains Java Runtime Environment (JRE) required by the GUI tools, VAR and the uninstaller.

`<software_root>\jre\bin`

This directory contains the dlls required for Java application.

`<software_root>\jre\lib`

This directory contains libraries required for Java application.

NOTE:- The JRE directory is installed even if you do not select any Java component while installing the Versant Object Database. This is because the installer needs these files to run itself.

lib directory

`<software_root>\lib`

This library directory contains system libraries and error message files for Versant ODBMS. It also contains the Versant JDO Interface runtime jars.

`<software_root>\lib\ansi`

This directory contains ANSI production libraries for Visual Studio .NET 2003 compiler on Windows.

`<software_root>\lib\data`

This directory contains data files required for Internationalization support.

`<software_root>\lib\JDO`

This directory contains data files required for JDO. It also contains the JCA adapter for EJB applications.

`<software_root>\lib\JVI`

This directory contains data files required for Java Versant Interface and JCA.

`<software_root>\lib\VXML`

This directory contains data files required for Versant XML Toolkit.

NOTE:- Although the ODBMS library files have the same name for all releases, the C++ library files have different names for each compiler.

`<software_root>\lib\eclipse`

This directory contains the eclipse plugins for editing the mapping in Eclipse 3.0 and 3.1.

`<software_root>\lib\schemas`

This directory contains Versant ReVind(VSQL) related files. This directory contains 3 schema files which are needed by the schload utility of Versant ReVind.

`<software_root>\lib\help_isql`

This directory contains the help files required by the ReVind(VSQL) utilities.

uninstaller directory

`<software_root>\uninstaller`

This directory contains the uninstaller and its related files.

Database root directory

`<db_root>`

This is the directory under which directories for individual databases will be created.

The default location for the `<db_root>` is the same level as the `<software_root>` directory:

`\usr\local\versant\db`
Although you can create database volumes anywhere, for each database that you create, Versant creates a database subdirectory under the database root directory. In the database subdirectories, Versant places information files that describe the database and set its operating parameters.

During installation, your environment will be altered so that this installation of Versant can find its database root directory.

After installation, you can move the database root directory, if you modify your environment to point to the new location. However, you should not change the names or contents of the subdirectories under the database root directory.

See also “Pre-Installation Checklist” on page 128, and the Chapter "Configuration Parameters" in the *Versant Database Administration Manual*.

Individual database directories

`<db_root>\dbname`

The name of the directory for a database will be the same as the created database and will be placed under the database root directory.

Individual database directories can be created with either the `makeprof` or `makedb` utilities.

For example, for a database named `mydb` and a database root directory of `c:\versant\db`, the database directory will be:

```
c:\versant\db\mydb
```

For more information on Creating a database, refer to chapter 1 of *Database Administration Manual*.

Database volumes

A database consists of a system log, physical log and logical log volume. These volumes are created with the `createdb` utility. You can add additional storage volumes to a database by using the `addvol` utility.

For more information on database utilities, refer to chapter 3 of *Database Administration Manual*.

When you create a database, you can place your database volumes anywhere. For example, if you have multiple disk drives, it would improve performance if you placed your volumes on a different disk than the one used by your application.

System volume

```
<db_root>\dbname\system.log
```

There is one system volume, named `system`, per database. It is a file that contains database information, class descriptions and object instances. The default location is the individual database directory.

For example, for a database named `mydb` and a database root directory of `c:\versant\7_0_1\db`, the name of the system volume file would be:

```
c:\versant\7_0_1\db\mydb\system
```

Physical log volume

```
<db_root>\dbname\physical.log
```

There is one physical log volume, named `physical.log`, per database. The physical log volume is a file that, along with the logical log volume, records information about transactions and provides information for roll back and recovery. The default location is the individual database directory.

For example, for a database named `mydb` and a database root directory of `c:\versant\7_0_1\db`, the name of the physical log volume file would be:

```
c:\versant\7_0_1\db\mydb\physical.log
```

Logical log volume

```
<db_root>\dbname\logical.log
```

There is one logical log volume named `logical.log`, per database. The logical log volume is a file that, along with the physical log volume, records information about transactions and provides information for roll back and recovery. The default location is the individual database directory.

For example, for a database named `mydb` and a database root directory of `c:\versant\7_0_1\db`, the name of the logical log volume file would be:

```
c:\versant\7_0_1\db\mydb\logical.log
```

Storage volumes

```
<db_root>\dbname\storage
```

You can add storage files to a database by using the `addvol` utility. There is no default name for a storage volume. The default location for a storage volume is the individual database directory.

Database Files

When you create a database, Versant will create the following database files.

Server process profile

```
<db_root>\dbname\profile.be
```

The server process profile file is named `profile.be` and is located in the database directory for the database, which must branch from the database root directory.

For example, for a database named `mydb` and a database root directory of `c:\versant\7_0_1\db`, the location of the server profile will be:

```
c:\versant\7_0_1\db\mydb\profile.be
```

The server process profile is created with the `makedb` utility.

When a database starts up, the database server process reads the server process profile to determine the location of the database volumes and to set the database operating environment. If a server profile does not exist for a database, you cannot start that database.

The server process profile contains several kinds of parameters. Database creation parameters are used only when a database is created with the `createdb` utility. Database functional parameters and database tuning parameters are used each time a database is started. You can edit the functional and tuning parameters to change the operating environment of a database.

For more information on the Server Profile file, please refer to the *Versant Database Administration Manual*.

Lock file

```
<db_root>\dbname\lock
```

When you create an individual database directory with the `makeprof` or `makedb` utility, Versant will create a file named `lock` in your individual database directory.

Versant will use the `lock` file to determine whether a database has already been started. If the `lock` file is destroyed for some reason, you cannot start or recreate a database until you create a new file.

Database type file

`<db_root>\dbname\PERSONAL.FLG`, *or*
`<db_root>\dbname\GROUP.FLG`

When you create an individual database directory with the `makeprof` or `makedb` utility, Versant will create a hidden file with a `.FLG` extension. This file is used to indicate whether it is a personal or group database, and also to store the name of the database owner.

For a personal database, the file created is called PERSONAL.FLG. For a group database, the file is called GROUP.FLG.

Shared memory file

<db_root>\dbname\sharemem

When you create a database with the createdb utility, Versant will create a file named sharemem in your individual database directory. The contents of the sharemem file are used as a key to get a unique id for the shared memory of the database server process. On Windows, this file is created but not used.

Password file

<db_root>\dbname\pw

A password file is created only if the DBA authentication is done using a password-based mechanism. The specified password is stored in this file and is accessed every time a utility that requires DBA authentication is invoked like createdb, dbinfo, vmovedb etc. If DBA forgets the password, the DBA can log as OS user into the machine, delete the password file and recreate a new password using the dbuser utility.

Application Process Profiles

%HOMEDRIVE%%HOMEPATH%\osc*dbname*

When you start a session, you must name a session database. The operating environment for the application will be set per the specifications in the application process profile file corresponding to the session database. If no application profile is found, the application will use default environment settings.

You can edit the application process profile to add aliases and various application process parameters.

For more information on the Application Process Profile parameters, refer to chapter 2 - Database Profiles of the *Versant Database Administration Manual*.

When a database is created, Versant will create an initial application process profile for the database. The name of the file will be the same as the database name, and it will be placed in a hidden directory under the home directory of the user who created the database.

Home directories are not set as system environment variables, instead they are associated with user profiles.

To see the values of your HOMEDRIVE and HOMEPATH variables for your environment, you can go to your command prompt and enter the following:

```
set <return>
```

To see the contents of your home directory, you can enter:

```
dir \a %HOMEDRIVE%%HOMEPATH% <return>
```

Since the variables HOMEDRIVE and HOMEPATH are associated with a profile and are not set as part of the system environment, users other than the user who created a database will need to invoke the makeprof utility in order to create and use an application profile file.

For example, to create an application profile for the database db1, login as a user other than the DBA and run the following from the command prompt:

```
makeprof db1
```

The above creates a profile file named db1 in an osc directory under that user's home directory.

Advanced Usage of Application Profiles

The following information is useful only under very special circumstances.

If you want multiple users to use a single application profile file for a particular database, you can modify the location of their home directory as set in their User Profile for each user.

To do this, open your Windows User Manager (usually located under Start/Programs/Administrative Tools/User Manager) and modify the Home Directory entry under the Properties area for each user.

If you want to use multiple application profiles, you can login with different names and create differing profiles for each user name.

Configuration Files

During installation, Versant will create two configuration files: a machine configuration file (.oscxxyyzz or vrxxxyyzz.ini) and a system information file (sysinfo) depending on your Operating System. The contents of the machine and system information files will vary depending upon the type of installation you perform.

Machine configuration file

.oscxxyyzz
vrxxyyzz.ini
%HOMEDRIVE%\WINNT\vrxxyyzz.ini

If you install using the installer, the installer will create a machine configuration file in your /etc directory.

The machine configuration file, where xx is the major release number, yy is the maintenance release number, and zz is the minor release number.

The contents of the file have the information of the following parameters:

VERSANT_ROOT
VERSANT_DBID_NODE
VERSANT_DBID
VERSANT_DB

For example, for Release 7.0.1.3 the machine configuration file is:

c:\WINNT\vr070001.ini

System information file

<software_root>\lib\sysinfo

During installation, Versant will create a system information file named sysinfo and place it in the <software_root> lib directory under your <software _root> directory.

If you use the default location for the software root directory, the location of this file will be:

c:\versant\lib\sysinfo

<software_root>\bin\jdosetup.bat

Used to setup the JDO runtime environment.

Statistics files

<software_root>\lib\vstats.ini

When it starts, the vstats Statistics Utility will read a configuration file in which derived statistics can be predefined and named.

A system-wide file named `vstats.ini` in the `<software_root> lib` directory will always be read.

For more information, please refer to the Section “Configuration Files” in Chapter “Directories and Files” in the *Versant Database Administration Manual*.

Environment Parameters

During installation, Versant will create and store values for the following environment parameters:

Parameter	Description
VERSANT_ROOT	The location of the Versant software root directory is set as a system environment variable in the registry.
VERSANT_PATH	The location of the Versant `bin` directory containing executables is set as a system environment variable in the registry.
PATH	The entry "`%VERSANT_PATH%`" will be appended to the string that sets your `PATH` system environment variable. On Windows, this operation will modify the registry.

For more information, on these variables, please refer to the Chapter "Configuration Parameters" in the *Versant Database Administration Manual*.

The values for the following environment parameters are stored at <software_root>`\lib\sysinfo` file where information about the database is stored:

Parameter	Description
VERSANT_DB	The location of your database root directory is set with an entry in the Versant `sysinfo` file.
VERSANT_DBID_NODE	The name of the machine containing your `osc-dbid` database system identifier file is set with an entry in the Versant `sysinfo` file.
VERSANT_DBID	The directory containing your `osc-dbid` database system identifier file is set with an entry in the Versant `sysinfo` file.

Post Installation Changes to your Environment

Creation of a new program group

The installation procedure will create a new program group called "Versant Object Database" with an Uninstall Versant Object Database icon in it. If you want to remove Versant from your machine, go to this program group and click on Uninstall Versant Object Database.

Modification to TCP/IP services file

The installation procedure can add an entry to the TCP/IP services file.

The TCP/IP services file will be under your Windows directory. For example, if Windows OS is installed in the directory `c:\winnt`, then the TCP/IP services file path name is `c:\Winnt\System32\Drivers\etc\Services`. The installation procedure will add a new Versant TCP/IP service to your machine by adding the following line to your services file:

```
oscssd 5019\tcp# Versant connect service
```

This indicates that the Versant daemon service name is `oscssd`, and it listens for connect requests on port 5019.

Addition of new windows service "Versantd"

The installation procedure can add a new Windows service to your machine named `versantd`.

If you look under Control Panel\Services, you will see `versantd` installed as an automatically started Windows service. The `versantd` service listens for Versant connection requests and spawns Versant server processes to satisfy these requests.

To start the `versantd` service, you can either reboot or start the `versantd` service with the following command issued from the command line:

```
net start versantd
```

Alternately, you can start the `versantd` service by opening Control Panel/ Services, highlighting the `versantd` entry and clicking on `Start`.

Addition of Microsoft Visual C++ libraries

Versant executables and libraries need the C/C++ run-time libraries `MSVCR70.dll`, `MSVCP70.dll`, `MSVCI70.dll` and `dbghelp.dll`. The installation will copy these DLLs into the `bin` directory.

If your system does not have these DLL's, then you need to copy these to your Windows system directory.

Addition of the Uninstaller icon

The installation procedure adds a new “Versant Object Database“ program group with an Uninstall icon in it. Clicking on the Uninstaller will remove all Versant files from your machine, and also undo Versant modifications to your environment.

You may find after the Uninstall procedure that some of the Versant directories have been left behind. This will happen if some new files have been added to the Versant directory structure after installation. For example, if you have created your database directory under the Versant software root directory, this will not be deleted. If you have compiled some of the demos or tutorials and the binaries are left behind, then these directories will not be deleted.

Some files may also be left behind if some process is still accessing one of the Versant directories or files. For example, if you have not stopped all databases, or if some command prompt window is accessing one of the Versant directories, the directories will not be deleted. In such situations, you will have to remove the remaining Versant directories and files manually.

Also, after installing Versant Object Database once, if you run the installer again to add more components to the existing Versant Object Database installation, then the uninstaller program may not remove the previously installed components. It is recommended that you choose all the components that you might require at the first time itself. This is because the uninstaller program gets overwritten each time.

For more information, please contact Versant Customer Support.

If You Re-install an OS

If you reinstall your operating system after installing Versant, you will need to reinstall Versant to reconstruct your system information and configuration files. Reinstalling will not destroy existing databases.

STEPS POST INSTALLATION

Following are steps that you may need to take after running the Versant installer.

Step 1: Obtain and install licenses

Some of the standard and all of the Add-on components in the Versant Object Database are licensed.

A license file must be installed with proper license records obtained from Versant.

The utility "`vlicchk`" can be used to identify whether a particular component is already licensed. For example, to check whether ODBMS component version 7.0 is already licensed:

```
vlicchk –c ODBMS –v 7_0_1
```

A new license is required if there is a change in the first two digits of the version of that component.

To obtain a license record:

- Visit http://www.versant.com. If you are not a registered user, you will need to register.
- Browse to the link for requesting a license.
- In the form complete all the details. In the machine identification field supply the hostid value reported by the utility "`vinstinf`".
- A license record will be mailed to you at the email address specified.
- Save the attached XML data in the email. This is the license record. If you do not have a license file <software_root>`\license.xml`, or if the present licenses are not required, then save the XML data as this file. In case there is an existing license file, then append the data to the file. The licenses are matched on the basis of the component name, version and the machine identification. In case two or more records have the same information for the component name, version and machine identification, then the first one to occur in the file will be picked up. If you are replacing a previously expired license, then it is important that the new data supplied is inserted at the beginning of the license file.

For more information on the license file, refer to Chapter "Directories and Files" in the *Versant Database Administration Manual*.

Step 2: Create a Database System File

As described in the "Pre-Installation Checklist" section, a database system identifier file named `osc-dbid` must exist and be accessible to your machine before you can create a database.

If a database system file does not exist at the location you specified during installation, you must create it before creating any databases. Since only one `osc-dbid` file should exist for a system of databases, if an `osc-dbid` file already exists anywhere in a network of databases, you should not create another one.

If during installation, you specified that you wanted a stand-alone installation, you must create a database system identifier file on your local machine.

To create an `osc-dbid` file, use the `dbid` utility on the machine that will contain it. If the machine is remote, first login to it. If the machine is local, go to the command line. Then run the following:

```
dbid -N
```

The file `osc-dbid` will then be created in the directory specified by the environment on the machine on which you invoked the command.

For more information about the `dbid` utility, please refer to Chapter "Database Utilities" in the *Versant Database Administration Manual*.

Step 3: Create a Database

If it is a new installation i.e. if you have installed for the first time, you will need to create at least one database, because to start a Versant session, you need a session database. The session database may be on your local machine or on a remote machine.

If you want to use the default configuration for your database, the steps are simple. (Assume, here, that the name of the database is to be `mydb`.)

For complete instructions on Creating a database and Resource requirements per database i.e disk space/memory/ processes, please refer to Chapter “Create a Database” in the *Versant Database Administration Manual*.

Create a local database

For a database on your local machine, first, from the command line, create the database directories and support files:

```
makedb mydb
```

Then, create the database:

```
createdb mydb
```

Create a remote database

For a database on a machine named `alpha` (assuming you have set your environment per the above instructions,) first, from the command line, create the database directories and support files:

```
makedb mydb@alpha
```

Then, create the database:

```
createdb mydb@alpha
```

Versant uses the local parameters while creating a database:

Parameters	Description
VERSANT_DBID_NODE	To find the machine containing the `osc-dbid` database system file.
VERSANT_DBID	To find the directory containing `osc-dbid`.
VERSANT_DB	To find the local database root directory.
`local login name`	To access the remote machine containing `osc-dbid` file.

In the above case for a remote database, Versant uses the local login name to connect to `alpha` and then sends the `makedb` and `createdb` commands to it. Versant location parameters set on `alpha` will be used to find the `osc-dbid` database system identifier file and the database root directory used by `alpha`. If `osc-dbid` is on a machine other than `alpha`, Versant will also use the local login name to access the machine containing the `osc-dbid` file.

Step 4: Convert and Backup Existing Databases

If you are upgrading from old version:

Convert existing databases to Release 7.0.1.4 format

To convert existing databases of Versant to the current release (7.0.1.4):

- Use the conversion utility (`convertdb`) of the new release, to convert the databases on the installation machine to 7.0.1.4 format.

The following table briefly describes the conversion procedure to Release 7.0.1.4 database from the previous release:

FROM (prior release)	TO (current release)	PROCEDURE
6.x (32-bit platform)	7.x (32-bit platform)	Run `cnvrtdb` from 7.x (32-bit) Release.
6.x (32-bit platform)	7.x (64-bit platform)	Run `cnvrtdb` from 7.x (64-bit) Release.
6.x (64-bit platform)	7.x (32-bit platform)	Not supported.
6.x (64-bit platform)	7.x (64-bit platform)	Run `cnvrtdb` from 7.x (64-bit) Release.

After a database has been converted to a 7.0.1.4 database, you can still access it with applications prior to 7.0.1.4 release.

For more information related to release compatibility, please refer to the "Pre-Installation Checklist" on page 128.

Convert the Database

For a database named `mydb`, first stop it with the following (do not use `stopdb -f`):

```
stopdb mydb
```

Then convert it with:

```
cnvrtdb mydb
```

For more information, on the cnvrtdb utility, please refer to the Chapter "Database Utilities" in the *Versant Database Administration Manual*.

RESTRICTION:- After conversion of a database to this release, the users in the database have no passwords. The authentication mechanism for such users is described in the chapter "Controlling Access To Database" in the *Versant Database Fundamentals Manual*.

Backup existing databases

Backup files belonging to older releases cannot be used to restore databases in the major releases. If you have backup files belonging to older releases, restore the database to the corresponding older release first. Then use convertdb to migrate the database to the 7.0.1.4 release.

If a database is part of a FTS replica pair needs to be converted to the 7.0.1.4 release, the client environment needs to ensure that an entry for the replica pair is present in the replica file in the VERSANT_ROOT directory of the 7.0.1.4 client environment.

After a database has been converted to Release 7.0.1.4 format, you will not be able to use backup files or tapes created with releases prior to Release 7.0.1.4 to restore it. Accordingly, before proceeding, you should backup your existing databases.

Backups can be created with the `vbackup` utility.

For example, to backup a database named `mydb` to the file `c:\tmp\level0`:

```
vbackup -device c:\tmp\level0 -backup mydb
```

For more information, on the `vbackup` utility, please refer to the Chapter “Database Utilities” in the *Versant Database Administration Manual*.

Step 5: Licensing the Add-on Components

Versant Object Database has following Add-ons which require separate licensing.

Add-on components:

COMPONENT	DETAILS
VAR	Versant Asynchronous Replication for Versant ODBMS.
Vedding	Fault Tolerant Versant ODBMS Server.
HABACKUP	Backup Solution for use with High Availability Server
Vorkout	Versant Online Database Reorganization Tool
ReVind	Structured Query Language Interface for Versant ODBMS

VODBC	Versant ODBC driver
Warm Standby	Used for an Incremental Rollforward recovery
Vitness	Versant Monitoring Console

Licensing Vedding

Vedding(FTS) is distributed within the Versant Object Database installation. However as other licensed components, it needs a separate license to operate.

Please contact Versant Customer Support for a license. You need to provide the same information as that for the ODBMS license. The license is similar to the ODBMS license and a separate license is required for each participating servers. Append the XML data provided by Versant Customer Support to the license file `<software_root>\license.xml`.

Once this is done, this option is licensed.

To test, setup a FTS pair of databases (`db1` and `db2`) on the server (`server`):

1. Edit the file `<software_root>\replica` and add the line "`db1@server db2@server`".
2. Execute the commands on the server:

```
makedb db1
createdb db1
makedb db2
createreplica db1 db2
```

3. The "`createreplica`" command above should complete successfully.

You can also check whether the Fault Tolerant Server option is licensed by executing

`vlicchk –c FTS –v 7_0_1` -- which should report that it is licensed.

NOTE:- All other Add-on components can be licensed in a similar manner.

SETTING ENVIRONMENT FOR WINDOWS

If you have made a mistake during installation or want to change some installer settings, you will have to set the missing or incorrect environment parameters post installation.

If you answered all questions during installation, then the environment parameters would have been set for you already.

For more information on these environment parameters, please refer to Chapter "Configuration Parameters" in the *Versant Database Administration Manual*.

Mandatory Parameters to Set

Versant needs to set values for the following environment parameters:

```
VERSANT_ROOT
VERSANT_PATH
PATH
VERSANT_DB
VERSANT_DBID_NODE
VERSANT_DBID
```

Steps to Set the Environment Parameters

System Variables

Use your Control Panel to change an environment parameter to set as a system variable.

1. Go to Control Panel->System->Environment
2. Highlight the variable to be changed and enter a new value in the Value box
3. Click on Set and then click on either Apply or OK.

System information file

Use a text editor to change the Versant system information file.

The name of this file is `sysinfo` and is located in your `<software_root> lib` directory.

Values for the environment parameters set in the `sysinfo` file can be overridden with environment variables of the same name.

Application profiles

Use the User Manager to specify the location of database application profile files.

Go to the Windows User Manager (usually located under Start->Programs->Administrative Tools->User Manager) and Modify the Home Directory entry under the Properties area for each user.

For more information, please refer to the Chapter “Database Profiles” in the *Versant Database Administration Manual.*

KNOWN ISSUE WITH THE UNINSTALLER

You might experience some issue with the uninstaller not removing files.

This is an issue with InstallAnywhere's global registry file (.com.zerog.registry.xml).

On Windows, this global registry is located at: Program Files\Zero G Registry\

On UNIX systems, if logged in as root, the global registry is located in the /var directory. If logged in as user, it is in the user's home directory.

In this case, please remove Versant Object Database installation manually.

CHAPTER 8

Installation on Solaris/Linux/ HP-UX/AIX

This Chapter gives detailed explanation on how to Install Versant Object Database on “Solaris/Linux/HP-UX and/or AIX machine”.

The Chapter covers the following in detail:

- Installing Versant Object Database
- Pre-Installation Checklist
- Installing Versant Using GUI
- Installing Versant in Silent mode
- Installed Directories and Files
- Steps Post Installation
- Known Issue with UnInstaller
- Setting Environment for Solaris
- Setting Environment for Linux
- Setting Environment for HP-UX
- Setting Environment for AIX

INSTALLING VERSANT OBJECT DATABASE

This chapter explains how to install Versant Object Database on a Solaris/Linux/HP-UX and/or AIX machine.

It explains the directory structure, install panels, directories and files during installation. Some of these may/may-not apply to your installation as this depends on the installation components you choose and the mode you install.

Among other things, this chapter also discusses the locations for your software root directory, database root directory and database system identifier file.

For more information, on what these directories and files are, please refer to Chapter "Directories and Files" in the *Versant Database Administration Manual*.

Versant Object Database comes with a common, integrated installer which offers a single GUI interface to install various components. This integrated installation eliminates version conflicts when installing and deploying components of Versant Object Database. Some of the components within Versant Object Database are licensed individually and are node locked.

Versant Object Database 7.0.1.4 consists of the following components:

Standard Components:

COMPONENT	DETAILS
Versant ODBMS	Versant Object Database Management System.
GUI Tools	Versant GUI Tools
JDO Interface	JDO Interface for Versant ODBMS.
Java Versant Interface	Java Language Interface for Versant ODBMS.
C and C++/Versant	C and C++ Language Interface for Versant ODBMS.

Add-on Components:

COMPONENT	DETAILS
VAR	Versant Asynchronous Replication for Versant ODBMS.
Vedding	Fault Tolerant Versant ODBMS Server.
HABACKUP	Backup Solution for use with High Availability Server
Vorkout	Versant Online Database Reorganization Tool

Warm Standby	Used for an Incremental Rollforward recovery
ReVind	Structured Query Language Interface for Versant ODBMS

NOTE:- All the Add-on components require an additional separate license.

Installation allows you to choose between Typical and Custom installation.

- **Typical install** - Installs all components of Versant Object Database with Demos, Tutorials and Documentation.
- **Custom Install** - Allows user to select the required components for installation.

The installer can be run in Super user mode as well as Non-super user mode.

For users with 'Administrator' privileges, the mode is super-user mode and for all other users the installation mode is selected as non-super user mode.

Users running applications in which Versant is embedded are likely to choose the Silent mode of installation. In this mode, the user presets a number of configuration parameters and directs the installation procedure to use those parameters.

For more information, See also “Installing Versant in Silent mode” on page 201.

PRE-INSTALLATION CHECKLIST

If you are installing Versant for the first time, you need to know and decide a few important things before the installation process begins.

Set Tuning Parameters

The "shm-" parameters affect the size of shared memory available for database page caches and support tables. If your application connects to multiple databases, you may need to increase the number of shared memory segments over a standard hardware configuration.

For Solaris:

The following system parameters may be tuned to suit your application needs.

For more information on tuning these parameters, please refer the "Solaris Tunable Parameters Reference Manual" on Sun Solaris Official Documentation site.

- **shmsys: shminfo_shmmax**

Description	Maximum size of system V shared memory segment that can be created.
Default	8,388,608
Range	0 - MAXUINT32 on 32-bit systems, 0 - MAXUINT64 on 64-bit systems

- **shmsys: shminfo_shmmni**

Description	System wide limit on number of shared memory segments that can be created.
Default	100
Range	0 to MAXINT

For Solaris-10:

The shared memory settings (project.max-shm-memory, project.max-shm-ids) need to be increased not only in the DBA project but also in the system project(0).

For Linux/HP:

You will need the following resources in your UNIX kernel:

Resources used in Unix Kernel	Linux	HP
The maximum number of shared memory identifiers on the system	`shmmni 4096`	`shmmni 512`
The maximum number of shared memory segments per identifier.	`shmseg 64`	`shmseg 128`

For AIX:

AIX does not require tuning, because these parameters are dynamically adjusted as needed by the AIX kernel.

Usually you will need to modify or insert these specifications in your kernel configuration file `/etc/system`, rebuild your kernel, and then reboot. However, your operating system may allow you to set kernel parameters dynamically. See your Operating System manual for details.

Tuning the file-descriptor limit parameter of the Operating System

Versant uses one file descriptor per TCP/IP connection towards a database. A common bottle neck in the default Operating System configuration, is the lack of file descriptors. If sufficient file descriptors are not made available then, Versant will not be able to accept new connections. Hence it may be necessary to tune the OS specific file descriptor limit according to the estimated number of database connections.

Operating Systems place limits on the number of file descriptors that a process may open. In addition to per-process limits, an OS will also have a global limit on the amount of file descriptors that all its processes together may consume. Configuring file descriptor limits is platform and even distribution-version specific, so please consult your Operating System documentation for the definitive guide.

Here are a few examples:

Solaris

- For Solaris-10

  Setting the per process file descriptor limit to 2048 on Sun Solaris 10:

  ```
  projmod -a -K "process.max-file-descriptor=(basic,2048,deny)" default
  ```

- For other versions of Solaris

  The file /etc/system lists the hard and soft limits for the number of files a process can open.

  `rlim_fd_max` is the hard limit.

  `rlim_fd_cur` is the current (soft) limit.

  A process can have up to `rlim_fd_cur` file descriptors and can increase the number up to `rlim_fd_max`.

  These parameters can be changed system wide by placing/changing entries in the `/etc/system` file:

  ```
  set rlim_fd_max=8192
  set rlim_fd_cur=1024
  ```

  Refer to the Solaris documentation for more information.

HP-UX

To check the current file descriptor limit you can use the system command `sysdef`:

```
|sysdef | egrep "NAME|nfile|maxfiles_lim" |
```

To increase the number of file-descriptors, you have to use the '`sam`' configuration tool provided by HP to adjust some of the system defaults.

The parameters that may need configuration are:

`nfile -` Max number of open files.

`maxfiles_lim` - Hard file limit per process

NOTE:- The values given above for system parameters are examples only, and you may want to select different values for your system requirements.

Review visibility of key files

The following files must be readable by all users:

```
/etc/hosts
/etc/services
```

Decide how you want to install Versant

The Versant installation can be done from a local machine or from a remote machine. If you are installing Versant from remote machine, then you must set your DISPLAY environment variable such that you get the graphical display on your local systems screen.

Decide on a software root directory

Versant will be installed under your choice of a software root directory in a pre-determined set of software subdirectories.

The default location is:

```
/usr/local/versant/7.0
```

If you are going to be the sole user of this Versant installation, then you can choose any directory you want for your software root directory.

If others are going to use this installation, then you should choose a directory visible to others on the network.

Your software directory can be local or remote. If it is remote, you must be able to access it with the same username that you use on your local machine.

Decide on a database root directory

The installer will create database directories and information files under the database root directory (called `<db_root>`).

During installation, you need to decide the database root directory else it will be installed at the default location.

The default location is:

```
/usr/local/versant/db
```

After installation, each time you use this installation of Versant to create a database, information about the new database will be placed in a subdirectory created under the database root directory. (Although the database volumes themselves can be anywhere.)

You can specify either a new database root directory or use an existing database root directory created from a previous installation of Versant.

NOTE:-

If you are going to be the sole user of the Versant installation and the databases created by this installation, then you can choose the software root and database root directory of your own choice.

If others are going to use this installation, then you should choose a directory visible to others on a network.

The software root directory and database root directory can be local or remote. If it is remote, you must be able to access it with the same user name that you use on your local machine.

Decide on a database system identifier file

The name of the database system identifier file is `osc-dbid`.

The database system identifier file, `osc-dbid -` must exist and be accessible to your machine before you can create a database. This file contains information about all databases that you and others might conceivably access during a particular database session.

During installation, you will be asked for the location of the `osc-dbid` file.

For new installation, you will need to create an `osc-dbid` file.

For old installations, you need to associate to the existing `osc-dbid` file.

To Get information about a existing `osc-dbid` file

If you are joining an existing system of databases on a network, then specify the location of the `osc-dbid` file already in use.

To find the location of an existing file, you can go to the machine containing it and run the `oscp` utility as:

```
oscp -i
```

To Get the machine name

To find the name of the machine containing the `osc-dbid` file, run the `oscp` utility:

```
oscp -n
```

To Get the directory name

To find the name of the directory containing the `osc-dbid` file, run the `oscp` utility:

```
oscp -o
```

Decide the osc-dbid node and path

If you are creating a standalone installation, then you need to specify the node name of your local machine.

During installation, you will be asked for the node name and path of the `osc-dbid` file.

If you are creating a new network of databases or joining an existing network of databases then you can specify a remote machine name. In this case, the installer will not create a new `osc-dbid` file. The user will have to manually specify the dbid node on which the osc dbid file is residing. However, it will record it in the configuration files generated during installation.

Normally, the `osc-dbid` file will be on a machine acting as a server and will be already created by the time you finish the installation on a Windows machine.

The purpose of the `osc-dbid` file is to ensure that each database has a unique identifier number in a network system of databases. This is important, because the object model requires that each object has an identifier number unique among all other objects, regardless of database. When you create an object, the object is given an identifier number composed partly of the database identifier and partly of a number unique to the database.

You will not need to access the database system file when you create an object or connect to a database. You will need to access the database system file when you create a database or ask for information about any or all databases in a network system of databases.

If you are creating a standalone installation that will never use objects in databases on other machines, the installer will by default create an `osc-dbid` file in your Versant database directory.

Specify the future location of the database system file

If you want to be a part of a system of databases that has not yet been created, then specify the future location of the database system file. (However, you will not be able to create a database until the database system file exists.)

The default location for an `osc-dbid` file is database directory.

The database system file can be local or remote.

If it is local, it must be visible to all others who will create databases in a network system of databases.

If it is remote, you must be able to access it with the same username that you use on your local machine. If your local machine does not implement usernames, then your local environment must specify a username (you can let the installer set the environment or do it yourself after installation.)

Since there is only one database system file for a system of databases, typically for most users, it will be on a remote machine.

The installer will not check your choice of a location for a database system identifier file to see if it exists. Thus, if you want to create an `osc-dbid` file, you will have to do so after installation.

During installation, your environment will be altered so that this installation of Versant knows where to look for the `osc-dbid` database system identifier file.

If you specify an incorrect location for the database system file during installation or later move the database system file, you can correct your mistake later by modifying a system file or environment variable.

For more information, on specifying the location of `osc-dbid` after installation, please refer to Chapter "Configuration Parameters" in the *Versant Database Administration Manual*.

Recommendations

- If you are making multiple, new installations, we strongly recommend that you install first on the machine that will act as your database system host. This will ensure that you specify a correct location for the database system file when you make each installation.

  Remember that a mistake can cause a lot of work: you will have to go back to each and every installation and re-specify the path to the `osc-dbid` file.
- Under some circumstances, you might want to create multiple, separate systems of databases and then change your environment when you connect to one system or the other. However, since the `osc-dbid` file is not read, each time you make a database connection, you could conceivably create simultaneous connections to databases belonging to differing systems.

CAUTION:- Please take precautions while doing this, because databases in differing systems could have the same identifier number, and thus there could be an object in the differing systems with the same logical object identifier.

Review your permissions

Super user as well as other users can run the installer. You must check the permissions of the installation directory, database directory and `osc-dbid` directory to ensure that you have `WRITE` access.

Decide on an installation owner

If you are installing it in the super-user mode (as user 'root'), you will be asked for the name you want to use as the owner of the installation and the database root directory. The name you supply will be the Versant database system administrator (DBSA). Versant database system administrator (DBSA) name should be not be given as “root”. For non-root installation, the user running the installer will own all the files.

Decide on a user group

If you are installing it in the super-user mode (as user 'root'), you will be asked for the name of a database user group. Only members of the DBSA group will be able to use this installation of Versant and to create databases in this database root directory. For non-root installation, all the files will have group id of the user running the installer.

Decide whether to let Versant modify TCP/IP environment files

If you are installing it in the super-user mode (as user ‘root’), you can let Versant modify your TCP/IP configuration files or modify them yourself after installation. In any case, if the installation machine is not the Yellow Page Master, you will have to update configuration files on the master machine. For non-root installation, the user has to modify them manually.

Specify the Temporary location

Installer uses system’s temporary directory for it’s use during installation. The default temporary location can be overridden by specifying environment variable `IATEMPDIR`. This variable is to be set to any location having enough disk space (approx. 300MB). Please do not provide the same path as that of the software root directory (Versant Root).

If it is a reinstallation

If you want to re-install on top of the existing installation, first stop all databases by using `stopdb` utility. Then proceed as if you were installing a new system.

KNOWN ISSUE WITH THE INSTALLER

Screen Resolution

On a computer with higher screen resolution, it might occur in rare circumstances that the installer hangs. This is a known bug with the Install AnyWhere software. In such a case, change the screen resolution to a smaller value.

INSTALLING VERSANT USING GUI

This section explains how to install Versant on a Unix machine using the Versant GUI.

Versant Object Database installation is a GUI based installation. Installer guides user throughout the installation process. Although, a regular user can perform the Versant installation, it is recommended that super user does the installation.

For an existing user

1. **Stop all databases on the installation machine**.

   If you are updating an existing installation of Versant, then before installing Versant Object Database, use your existing `stopdb` utility to stop all databases on the installation machine and then proceed for installation as below.

   For example, for a database named `mydb`, do the following:

   ```
   stopdb mydb
   ```

   For more information, on `stopdb`, please refer to the Chapter "Database Utilities" in the *Versant Database Administration Manual*.

For a new user

In the description below, the term `<software_root>` or `<root>` is used to denote the software root directory where you choose to install the Versant software.

2. **Mount key directories with read-write privileges**

   Mount the `/` and the `/usr` partitions on the installation machine with read-write privilege. This is necessary for Versant to create or modify necessary environment files. This is essential if you are installing Versant in super-user mode.

3. **Load the distribution CD-ROM and Invoke the installer**

   Place the Versant installation CD in the CD-ROM drive on the installation machine.

   To start the installation program, execute the following command from the directory containing the installer (`install.bin`):

   ```
   ./install.bin
   ```

   This will initiate a GUI-based procedure for Versant installation.

NOTE:- If the `install.properties` file is also located in the same directory, then the installation will be performed in Silent mode. This file must be removed to install Versant in the GUI mode.

For further details on silent installation, please refer to the section "Installing Versant Object Database in Silent Mode".

The installer uses your system's temporary directory to extract the system files. If temporary directory on your machine does not have the required space, then the installer will ask for another directory where it can temporarily extract the files.

4. **Choose the installation mode**

   You can install Versant Object Database either as super-user ('`root`') or as any normal user. During installation, you may be asked only those questions, which are appropriate for that installation mode. Installation cannot perform certain tasks unless you are installing it as super-user 'root'**.**

   Recommendation - Super-user mode is recommended for your installation unless your needs are different (such as using Versant as embedded system).

5. **Go through the navigation panels**

   The installer consists of several panels. Most of the panels will ask you to provide input. For each question, a default choice is presented before you.

   To terminate the installation at any time, you can press '`Cancel`' button on any of the panels. On some of the panels, you can press the '`Help`' button to get help on that topic. Pressing the '`Help`' button pops up a new window with the help text. To go to the next panel and the previous panel, press the '`Next`' button and the '`Previous`' button on the panel.

Installer Panels

The following panels are displayed during Installation:

Splash Panel and Introduction

The Installer starts with a Splash panel. Splash panel disappears automatically without any user interaction. This is followed by an Introduction panel which introduces you to the Versant Object Database installation. Choose '`Next`' to continue the installation.

License Agreement

The license agreement panel describes the terms and conditions of the use of the software being installed. You can proceed with the installation only if you accept the license agreement.

Important Information

The important information panel gives an overview of all the components that come along with this release. Click '`Next`' to continue the installation.

Choose Install Folder

The installer asks you to specify the location of your installation root directory (called `<software_root>`). If you specify a directory that does not exist then the installer will create it. Ensure that you have write permissions for the directory path you specify.

The default location for the installation directory is `/usr/local/versant/7.0`.

The installer also creates a system path file in your `/etc` directory and enters your software root directory in it. The name of this file will depend upon your release number.

For example, for Release 7.0.1 the name of the system file is:

```
/etc/.osc070001
```

Choose Shortcut Location

Choose an appropriate option or specify the location where the shortcuts for the product will be created. Click '`Next`' to continue the installation.

Choose Install Set

Versant Object Database Installer provides users with two installation sets: Typical and Custom. Select the required installation and click '`Next`' to continue the installation. More information about these installations is given in the following sections.

Typical Installation

Choosing this, the user will be able to install the most commonly used installation choices. Some of the components are implicitly installed, depending upon the users choice. For example, Versant Asynchronous Replication will also install Versant ODBMS. The reason for this is that for a speci-

fied installation option, the installer automatically determines the components on which the specified component depends, and so all the dependent components are also installed by the installer.

Under Typical installation, Versant Object Database currently offers following installations:

Versant C/C++ Interface: This includes the development environment for Versant C/C++ Interfaces. This is the complete development environment for applications using Versant C/C++. It necessarily includes Versant ODBMS. It also includes demos, tutorials and documentation for both.

Versant GUI Tools: This includes the Versant GUI tools (Versant Administration Console, Versant Object Inspector, Monitoring Console and Versant Developer Console). It also includes the documentation for all the installed components.

Versant JDO Interface: This includes the JDO Interface for Versant ODBMS. This is the complete development environment for application using JDO. It also includes demos and documentation.

Versant Java Interface: This includes the Java Language Interface for Versant ODBMS. This is the complete development environment for application using Java Versant Interface. It includes tutorials, demos and documentation.

Add-on Components

Versant Asynchronous Replication (VAR): This includes the Versant Asynchronous Replication component for ODBMS. This is the complete development environment for application using VAR. It includes Versant ODBMS. It also includes demos and documentation.

Versant Vedding (FTS): This includes the Fault Tolerant Server (FTS) and its documentation.

Versant High Availability Backup (HABACKUP): This includes Versant's High Availability Backup and its documentation.

Versant ReVind: This includes the Structured Query Language Interface for Versant ODBMS and its documentation.

Versant Vorkout: This includes Versant's Online Database Reorganization Tool and its documentation.

Versant Warm Standby: This includes the Versant's Warm Standby and its documentation.

Custom Installation

If your installation needs are different than what is provided in the Typical installation set, then you can customize your installation.

For customizing, choose '`Custom Install`' on the 'Choose Install Set' Panel and click "`next`" to bring up 'Custom Installation' Panel.

Custom Installation Panel gives you the flexibility to choose components at finer granularity. In the top panel, you will find a 'choice box'. This lists all the installation sets described above in the Typical Installation. When you click on any one of the options in the top pane, its description is displayed in the bottom pane.

You can make your initial selection with the given choices in the choice box above. After that you can add/remove components. For adding/removing a component, check on box before the component name. Only the components, which are checked, at the time you press '`Next`' button, will be installed.

Here, the user needs to select the components individually for installation under the "Runtime/ Developer" category.

Versant ODBMS

- Runtime Only - It installs Versant ODBMS Server, Database Tools and Utilities. This installation is useful for deploying Versant Server on a machine.
- Developer - It installs C/Versant language interface development environment. It includes header files and static libraries. It also includes documentation, demos and tutorials.

JDO/Versant

- Runtime Only - It installs elements required to run applications using JDO interface for Versant ODBMS. This installation can be considered as the JDO/Versant client only installation since it does not include Versant ODBMS Server component. It also includes Database Tools and Utilities.
- Developer - It installs developer environment for JDO Interface for Versant ODBMS. It also includes documentation, demos and tutorials.

Java Versant

- Runtime Only - It installs elements required to run applications using Java language interface for Versant ODBMS. This installation can be considered as the Java Versant client only installa-

tion since it does not include Versant ODBMS Server component. However, it includes Database Tools and Utilities.

- Developer - It installs developer environment for Java Language Interface for Versant ODBMS. It also includes documentation, demos and tutorials.

Versant C/C++

- Runtime Only - It installs Versant ODBMS Server, Database Tools and Utilities. This installation is useful for deploying Versant Server on a machine.
- Developer - It installs C and C++ Versant interface development environment. It includes header files and static libraries. It also includes documentation, demos and tutorials for the same.

Versant Asynchronous Replication (VAR)

- Runtime Only - It installs elements required to run applications using VAR component for Versant ODBMS. This installation can be considered as the Versant client only installation since it does not include Versant ODBMS Server component. However, it includes Database Tools and Utilities.
- Developer - It installs developer environment of VAR component for Versant ODBMS. It also includes its documentation and demos.

Versant Fault Tolerance Server (FTS):

It installs the tools required for replication and its documentation.

Versant High Availability Backup (HABACKUP):

It installs the elements, which can be used to take backup on high availability server and its documentation.

Vorkout:

It installs the elements required for reorganizing the data in order to reduce fragmentation and to enhance performance along with its documentation.

Warm Standby:

It installs Versant's Warm Standby and its demos, tutorials and documentation.

The following Runtime system panel can be seen for Linux 64bit installer with ICC support.

Installing RunTime System

Versant Object database applications will require runtime libraries of Intel compiler version 10.0.023 and higher to run. Versant Object database installer is bundled with the redistributables of Intel compiler version 10.0.023. If you do not have the required compiler version installed on your machine, you should install the redistributables.

If you have the required version installed, you can configure dynamic linker run time bindings for the Intel compiler using ldconfig. Please refer to the man page of ldconfig for further details.

Not having the runtime libraries may result in the following error:

```
error while loading shared libraries: libcxaguard.so.5: cannot open
shared object file: No such file or directory
```

Pre-installation Summary

This panel lists the summary of the installation with information about the root directory and the component which will be installed. The files and directory structure created in your system will depend on the components you have selected.

If you need to change anything in the installer, you can do so now by clicking on the “Previous” button, else click “Install” to proceed with the installation. Now the installer actually starts copying the files to your system.

Specify Installation Owner

If you are installing Versant as super-user “root” then installer asks you to specify the login name and the group name of Database System Administrator (DBSA). In the Versant environment, the term DBSA designates the owner of database root directory as well as all the installed files. Besides the login name of DBSA, installer also asks for the group name of DBSA. Only members of DBSA group will be able to use this installation of Versant.

NOTE:- Versant database system administrator (DBSA) name should be not be given as “root”.

The installer validates the specified login name and group name of the DBSA. An error message will be displayed and you are again prompted for the login name and group name of DBSA:

- If the login name does not exist in the system on which installation is being done
- If the group name does not exist in the system on which installation is being done
- If the login name is given as “root”

The default values for login name and group name of DBSA are 'bin' and 'staff' respectively.

If the Installation is being done on NFS mounted drive, it might encounter some problems while changing ownership of installation directories and files to the specified DBSA.

The Uninstaller program is also owned by DBSA.

If the user installing the software is not “root” this question is not asked and all the directories and files are owned by the user installing Versant.

Database Directory Folder

The installer asks where the database root directory (called <db_root>) should be located.

The default location is the parent directory of the installation directory. It is <software_root>/../db, i.e. /usr/local/versant/db.

If the directory you specify does not exist, then installer will create it. Make sure that you have the WRITE permission where you want this directory to be created. If Installation cannot create this directory, then it will display an error.

Even in the case of error, you can proceed with the rest of the installation. However, you need to create the database directory yourself after the installation, set appropriate directory permissions, and tell Versant where it is by updating your system configuration files such as /etc/.osc070001 or by setting appropriate environment variables.

The Installer sets the ownership of the database root directory to DBSA, and read-write privileges are also given to the DBSA group.

Create osc-dbid File

Before proceeding, you need to decide whether this installation would create a new osc-dbid file or work with the existing one.

If you want to create a new network of databases then you may opt to create a new osc-dbid file by selecting ‘yes’. But, if you want your installation to be part of existing network of databases then please select 'No' as your choice to create a new osc-dbid file.

The purpose of the database system file is to ensure that each database in a network system of databases has a unique identifier number. This is important, because the object model requires that each object have an identifier number unique among all other objects, regardless of database. When you create an object, the object is given an identifier number composed partly of the database identifier and partly of a number unique to the database. You will not

need to access the database system file when you create an object or connect to a database. You will need to access the database system file when you create a database or ask for information about any or all databases in a network system of databases.

Choose osc-dbid Node and osc-dbid PATH

Installer needs to get:

a. The name of the node where the `osc-dbid` file will reside, and

b. The path name where the `osc-dbid` file will reside.

If you had decided to create a new `osc-dbid` file, then the installer will use the above information to create a new `osc-dbid` file at the path you specify. If the path you specify does not exist then installation will create a whole path before creating `osc-dbid` file.

You should have `WRITE` permission if you specify a remote host name as node name, then Installation will not create `osc-dbid` file, even if you had selected the option to create it. Also, if the specified path for the `osc-dbid` file is NFS mounted then Installation might fail to create `osc-dbid` file. If `osc-dbid` file already exists at the path specified then installer does not create it.

Even in case of failure in creating `osc-dbid` file, `dbid` node name and path will be recorded in the configuration files generated during installation. After installation, you can run '`dbid -N`' (after setting environment variables) to create the `osc-dbid` file yourself. In cases, where `oscssd` entry could not be made in the services file, you may need to set environment variable `VERSANT_SERVICE_PORT` to some port before executing '`dbid -N`'. This is especially useful when you are installing Versant as a user other than Administrator.

Even if you did not choose to create an `osc-dbid` file from the earlier panel, the installer will still ask for the `osc-dbid` node and `osc-dbid` path. By doing this, you are essentially asking the installer to put this information in the configuration file that it generates. The installer does not check whether `osc-dbid` file exists at the path you specified if you did not choose to create it.

Default values for `osc-dbid` node and `osc-dbid` path are local machine name and `<software_root>/db` respectively.

Update System Services File

If you are installing Versant as user root (with Administrator privileges), the installer asks you whether you want it to update `/etc/services` of your system to include the port number for the Versant ss daemon.

Installer can add the following entry to the TCP/IP Services file `/etc/services`:

```
oscssd 5019/tcp# versant SS daemon
```

Versant uses TCP/IP protocol to communicate between a Versant client and Versant Server.

5019 is the port number used for standard configuration. If the above entry is already present in the TCP/IP services file then Installation will not update the file. To test the connection between client and server machine for Versant configuration, run the `itest` utility.

```
itest -v <hostname>
```

Port number 5019 is a standard port number used for listening client requests. You can also use some other available ports by modifying `/etc/services` system file. Also you can use service name other than the standard service name oscssd by specifying that service name in the environment variable `VERSANT_SERVICE_NAME`.

Update inetd Configuration File

If you are installing Versant with Administrator privileges and you have conformed to update the system services file, then the installer will ask whether you want installer to update the systems service file on your system. Installer will install system's `inetd` configuration file if you say yes to this question.

Solaris

Installation can add the following entry to the system file `/etc/inetd.conf`

```
oscssd stream tcp nowait root VERSANT_ROOT/bin/ss.d in.oscssd
```

Solaris Operating System has come up with a new feature called “SMF”. This creates a supported unified model for services and service management on each Solaris system. Use the `inetconv`(1M) command to convert the new services to SMF services.

Linux

Converting VERSANT SERVICE to xinetd form

If you are installing Versant as user root and you answered as yes to update your system's `inetd` configuration file, the installation will update the `inetd.conf` file. If the file `inetd.conf` is not present, the installer will create this file in the `/etc` directory and will then update this file. It will then convert the `VERSANT SERVICE` to `xinetd` form. The configuration file for service `oscssd` will be `/etc/xinetd.d/oscssd`. If this file already exists it will be overwritten.

If you are using the Intel compiler suite icc/icpc for your RHEL 4.0 64-bit VOD installation, then you may need to additionally set the following in the VERSANT_SERVICE configuration file. For example:

If the following entry is added to your /etc/inetd.conf file:

```
oscssd stream tcp nowait root /etc/versant_env.sh in.oscssd
```

then in addition to having the path of the VERSANT_SERVICE_DAEMON ss.d in the file /etc/versant_env.sh , the LD_LIBARARY_PATH value having the appropriate libraries must be set in the file: /etc/versant_env.sh.

For example:

```
LD_LIBRARY_PATH=/opt/intel/cce/10.0.023/lib:<VERSANT_ROOT>/
lib:$LD_LIBRARY_PATH
export LD_LIBRARY_PATH
exec <versant_root>/bin/ss.d
```

AIX

Installation can add the following entry to the system file /etc/inetd.conf

```
oscssd stream tcp nowait root /etc/versant_env.sh in.oscssd
```

The file /etc/versant_env.sh contains the entries of the following variables for AIX:

```
PSALLOC=early
EXTSHM=ON
AIXTHREAD_MNRATIO=1:1
```

The path of Versant daemon is:

```
<versant_root>/bin/ss.d
```

NOTE:- Using the environment variable "PSALLOC=early" may cause a performance slowdown. The workaround for this problem is to set an additional environment variable "NODISCLAIM=true" alongwith "PSALLOC=early". By setting the "NODISCLAIM" environment variable the performance in terms of time will improve but this variable causes a steady reduction in the paging slots till the process terminates.

Install Complete

This the last panel that confirms your installation to be complete. This also lists a brief summary describing the changes done to your system during the installation.

In case you are upgrading the product such that there is no change in the first two digits of the version, then the existing licenses can be used and no new licenses are needed.

The installer will get the machine details (`machine name, OS type` and identification numbers [`hostid, IPv4 address and IPv6 address`]) and print them to `VERSANT_ROOT\InstallSummary` file. These values are required to obtain the license records. These details can also be obtained by invoking the utility "`vinstinf`" present in the "`bin`" directory of the installation.

Configuration Files generated by Installation

Installer generates configuration files needed for the kind of installation you choose.

The files are:

```
<software_root>/lib/sysinfo,
/etc/.osc070001,
<software_root>/bin/envsettings.csh
<software_root>/bin/jvisetup.csh
<software_root>/bin/varsetup.csh
```

When you install Versant ODBMS component, first three-file `sysinfo`, `.osc070001` and `envsettings.csh` are generated. These files contain settings for Versant environment variables.

When you install Java Versant Interface, the file `jvisetup.csh` is generated. This is a script, which you can execute to setup the environment required for Java Versant Interface Installation.
When you install Versant Asynchronous Replication, the file `varsetup.csh` is generated. This file contains a script, which users can execute to setup the environment required for Versant Asynchronous Replication Installation.

Update the Yellow Pages Master

You or your system administrator may need to update the Yellow Pages master.
Execute the `ypwhich` command to find the name of the Yellow Pages master host:

```
%ypwhich
osc
```

Now go to the host machine and type in: (`cd /var/yp; make`)

INSTALLING VERSANT IN SILENT MODE

During silent installation, the installer is enabled to run without any user interaction.

This can be achieved by defining certain parameters in a configuration file (`install.properties`). The values of the parameters in the configuration file need to be correct or the installer will display error messages during installation.

To run the installer in silent mode, do the following:

```
% install.bin -f install.properties
```

If `install.bin` is not in the same directory as the `install.properties` file, you need to specify the absolute path for the property file. E.g.:

```
% ./install.bin -f /vol2/install.properties
```

Here is an example of the contents of `install.properties` if the user selects Typical install and Additional Components options:

```
INSTALLER_UI=silent
USER_INSTALL_DIR=/Vol/Versant/7_0_1

#Standard Components
ODBMS_DEV=TRUE
ODBMS_RT=TRUE
JAVA_RT=TRUE
JAVA_DEV=TRUE
JDO_DEV=TRUE
JDO_RT=TRUE
CPP_RT=TRUE
CPP_DEV=TRUE
GUI_TOOL=TRUE

#Add-ons
HABACKUP=TRUE
FTS=TRUE
VAR_RT=TRUE
VAR_DEV=TRUE
WARM_STANDBY=TRUE
VSQL=TRUE
VODBC=TRUE
```

```
VCOMPACTDB=TRUE
#To install intel compiler redistributables for Linux 64bit installer
#with ICC support.
INSTALL_REDIST=TRUE
#DB

#set CREATE_OSCDBID_ACTION to true if you install ODBMS, CXX, or VAR
#components and you want to create osc-dbid file
CREATE_OSCDBID_ACTION=TRUE
DB_FOLDER=/vol/versant/db

#set OSCDBID_NODE and OSCDBID_PATH
# if CREATE_OSCDBID_ACTION=TRUE
OSCDBID_NODE=machinename
OSCDBID_PATH=/vol/versant/db

#following parameters are used for Windows if you want to install
#versantd service.
#UPDATE_SERVICES_ACTION=TRUE
INSTALL_NT_SERVICE_ACTION=TRUE

# for UNIX Systems: DBSA_GROUP_ID and DBSA_LOGIN_ID are set when the
#user is root.
#DBSA_GROUP_ID=group
#DBSA_LOGIN_ID=name

# for UNIX Systems: only set the following parameters if the user is
#root and want to update services and inetd
#UPDATE_SERVICES_ACTION=TRUE
#UPDATE_INETD_ACTION=TRUE
#NOTIFY_INETD_ACTION=TRUE
#UPDATE_VSQL_SERVICES_ACTION=TRUE
```

INSTALLED DIRECTORIES AND FILES

This section provides an overview of the installed files and directories by the installer.

Root Directory structure

The installer will create a software root directory (if it does not exist), other directories, such as `bin` and `lib`, and copy system files to the installation machine. The actual directory structure and the files copied into your system will depend on the installation options you choose during installing Versant. The details given below are based on a Typical Installation done by user with Administrative privileges.

After installation, you can move this directory, if you modify your environment to point to the new location. However, you should not change the names or contents of the subdirectories under the software root directory.

```
<software_root>
   |---ant
   |   |---bin
   |   |---lib
   |---bin
   |   |---ansi
   |   |---jvi
   |   |   |---1p
   |   |---memdir
   |---demo
   |   |---addons
   |   |   |---fts
   |   |   |   |---demo
   |   |   |---habackup
   |   |   |   |---demo
   |   |   |---var
   |   |   |   |---demo
   |   |   |---vcompactdb
   |   |   |   |---demo
   |   |   |---vsql
   |   |   |   |---demo
   |   |   |---warmstandby
   |   |   |   |---demo
```

```
|   |---c
|   |   |---demo
|   |   |---tutorial
|   |---cxx
|   |   |---advanced
|   |   |---hello
|   |   |---hellomfc
|   |   |---optlock
|   |   |---pntline
|   |   |---query
|   |   |---stl
|   |   |---tutorial
|   |---jdo
|   |   |---advanced
|   |   |---graph
|   |   |---inheritance
|   |   |---parentchild
|   |   |---queries
|   |   |---tut1
|   |   |---tutorial
|   |---jvi
|   |   |---advanced
|   |   |---IDE
|   |   |---odmg-trans
|   |   |---query
|   |   |---tutorial
|   |---vxml
|   |   |---demo
|---doc
|   |---addons
|   |   |---var
|   |---c
|   |---cxx
|   |---jdo
|   |---jvi
|---h
|   |---cxxcls
|   |---ilang
|   |---odmg
|   |---var
```

```
|---jre
|   |---bin
|   |---lib
|---lib
|   |---ansi
|   |---data
|   |---jdo
|   |   |---jca
|   |   |---templates
|   |---jvi
|   |   |---jca
|   |   |   |---jboss
|   |---vxml
|   |   |---dtds
|   |   |   |---v1.0
|   |---eclipse
|   |---schemas
|   |---help_isql
|---uninstaller
```

DIRECTORIES AND FILES AFTER INSTALLATION

There are two directories created after installation:

- Software root
- Database root

Software root directory

`<software_root>`

You can choose to place the Versant software root directory anywhere but the default software root directory is:

`/usr/local/versant/7.0`

Once you have specified a location for the software root directory, all other Versant software directories are built from it in a way that cannot vary.

Ant directory

`<software_root>/ant`

This directory contains the stripped down Apache Ant 1.5 (minimal configuration).

bin directory (General executables)

`<software_root>/bin`

The executables directory contains command line utilities for administering Versant and invoking tools. Executables and dll's from all the components of Versant Object Database are copied here. It also includes dynamic link libraries for Java Versant Interface (JVI) and C++ debugging libraries for your compiler.

<software_root>/bin also contains the Batch files and script files for JDO as follows:

File name	Description
jdosetup.csh	To setup the JDO environment
enhance.sh	To enhance the JDO persistence capable files
workbench.sh	To launch the Workbench
schematool.sh	To invoke the schema tool
updatevoa32schema.sh	To invoke the Conversion Tool for VOA 3.2.x customers. Before invoking this tool, please make sure you have used convertdb utility to convert VOA 3.2 x databases to VOD 7.0 x databases.

The following is the list of launch scripts to be used to invoke the various GUI Tools:

GUI Tool Name	Launch Scripts
Administration Console	vdbadmin
Object Inspector	vdbinspector
Monitoring Console	vdbmonitor
Versant Developer Console	vconsole

<software_root>/bin/ansi

This directory contains the ANSI dynamic link libraries.

<software_root>/bin/memdir

This directory contains ReVind(Versant SQL) related files. The VSQL server and utilities uses this directory to creates some temporary files.

demo directory

<software_root>/demo/addons

A directory containing subdirectories for all Versant add-on components like FTS, HABACKUP, VAR, VCompactdb, VSQL and Warm Standby and which contain the respective demos.

<software_root>/demo/C

A directory containing demo and tutorial for C interface.

`<software_root>/demo/CXX`

A directory containing demos and tutorial for C++ interface.

`<software_root>/demo/JDO`

A directory containing demos and tutorial for JDO interface.

`<software_root>/demo/JVI`

A directory containing demos and tutorial for Java Versant interface.

`<software_root>/demo/VXML`

A directory containing demo programs for VXML

doc directory

`<software_root>\doc`

A directory containing all the documentation for Versant Object database. This contains the following manuals:

- Versant Database Administration Manual
- Versant Database Fundamentals Manual
- Release Notes for Versant Object Database for 7.0.1.4

`<software_root>\doc\addons`

A directory containing documentation for all Versant Add-on components. This directory contains the following Versant Manuals in PDF format:

- Versant Vedding Usage Guide
- Versant High Availability Backup Usage Manual
- Versant Vorkout Usage Guide
- Versant/ReVind Reference Guide + Release Notes
- Versant Warm Standby Usage Guide

`<software_root>\doc\addons\var`

This directory contains the following Versant Manuals:

- Versant Asynchronous Replication (VAR) Manual
- VAR C++ Reference Manual (CXXdoc – HTML format)
- VAR Java Reference Manual (Javadoc – HTML format)

`<software_root>\doc\C`
A directory containing documentation for C Versant manuals in PDF format.

- C/Versant Reference Manual

`<software_root>\doc\CXX`
A directory containing documentation for C++ interface. This also contains an introductory demo for C++ interface.

This directory contains the following Versant Manuals in PDF format:

- C++/Versant Reference Manual
- C++/Versant Usage Manual

`<software_root>\doc\JDO`
A directory containing documentation for JDO interface. This directory contains the following Versant Manuals:

- Versant JDO Interface Usage Manual (PDF)
- Versant JDO Interface Reference Manual (Javadoc – HTML format)

`<software_root>\doc\JVI`
A directory containing documentation for Java Versant interface. This also contains an introductory demo for JVI interface.

This directory contains the following Versant Manuals:

- Java/Versant Interface Usage Manual (PDF)
- Java/Versant Interface Reference Manual (Javadoc – HTML format)

h directory (header files)

`<software_root>/h`

A directory containing the system header files such as `omapi.h` (which contains C/Versant functions).

`<software_root>/h/cxxcls`

A directory containing the C++ header files which contain the definitions of the database and library classes that comprise the C++/Versant interface.

`<software_root>/h/cxxcls/template`

A template directory containing definitions of C++/Versant template classes, such as the dictionary and array types.

`<software_root>/h/ilang`

A language information directory containing files with information about the native language in which you are programming.

`<software_root>/h/odmg`

A ODMG header files directory containing definitions of the database and library classes that comprise the ODMG C++/Versant interface.

`<software_root>/h/var`

A VAR header files directory containing definitions of VAR/C++ API classes.

JRE directory

`<software_root>/jre`

This directory contains Java Runtime Environment (JRE) required by GUI tools, VAR and the uninstaller.

`<software_root>/jre/bin`

This directory contains the dlls required for Java application.

`<software_root>/jre/lib`

This directory contains libraries required for Java application.

NOTE:- The JRE directory is installed even if you do not select any Java component while installing the Versant Object Database. This is because the installer needs these files to run itself.

lib directory

`<software_root>/lib`

This library directory contains system libraries and error message files for Versant ODBMS. It also contains the Versant JDO Interface runtime jars.

`<software_root>/lib/ansi`

This directory contains ANSI production libraries for Visual Studio .NET 2003 compiler on Windows.

`<software_root>/lib/data`

This directory contains data files required for Internationalization support.

`<software_root>/lib/JDO`

This directory contains data files required for JDO. It also contains the JCA adapter for EJB applications.

`<software_root>/lib/JVI`

This directory contains data files required for Java Versant Interface and JCA.

`<software_root>/lib/VXML`

This directory contains data files required for Versant XML Toolkit.

NOTE:- Although the ODBMS library files have the same name for all releases, the C++ library files have different names for each compiler.

`<software_root>/lib/eclipse`

This directory contains the eclipse plugins for editing the mapping in Eclipse 3.0 and 3.1.

`<software_root>/lib/schemas`

This directory contains ReVind(Versant SQL) related files. This directory contains 3 schema files which are needed by the schload utility of VSQL.

`<software_root>/lib/help_isql`

This directory contains the help files required by the VSQL(ReVind) utilities.

uninstaller directory

`<software_root>/uninstaller`

This directory contains the uninstaller and its related files.

Database root directory

`<software_root>/../db`

This is the directory under which directories for individual databases will be created.

You can use any name and any location for the database root directory (called `<db_root>`).

The default location for the `<db_root>` is the same level as the `<software_root>` directory:

`/usr/local/versant/db`

Although you can create database volumes anywhere, for each database that you create, Versant creates a database subdirectory under the database root directory. In the database subdirectories, Versant places information files that describe the database and set its operating parameters.

During installation, your environment will be altered so that this installation of Versant can find its database root directory.

After installation, you can move the database root directory, if you modify your environment to point to the new location. However, you should not change the names or contents of the subdirectories under the database root directory.

For more information, please refer the Chapter "Configuration Parameters" in the *Versant Database Administration Manual*.

Individual database directories

`<db_root>/dbname`

The name of the directory for a database will be the same as the database and will be placed under the database root directory.

Individual database directories can be created with either the `makeprofile` or `makedb` utilities.

For example, for a database named `mydb` and a database root directory of `/usr/local/versant/db`, the database directory will be:

```
/usr/local/versant/db/mydb
```

Database volumes

A database consists of a system log, physical log and logical log volume. These volumes are created with the `createdb` utility. You can add additional storage volumes to a database by using the `addvol` utility.

For more information on database utilities, refer to "Chapter 3 - Database Utilities" of *Database Administration Manual*.

When you create a database, you can place your database volumes anywhere. For example, if you have multiple disk drives, it would improve performance if you placed your volumes on a different disk than the one used by your application.

System volume

```
<db_root>/dbname/system
```

There is one system volume, named `system`, per database. It is a file or raw device that contains database information files, classes descriptions, and object instances. If a file is used for the system volume, the default location is the individual database directory.

For example, for a database named `mydb` and a database root directory of `/usr/local/versant/db`, the name of the system volume file would be:

```
/usr/local/versant/db/mydb/system
```

Physical log volume

```
<db_root>/dbname/physical.log
```

There is one physical log volume, named `physical.log`, per database. The physical log volume, is a file or raw device which along with the logical log volume, records information about transactions and provides information for roll back and recovery. If a file is used for the physical log volume, the default location is the individual database directory.

For example, for a database named `mydb` and a database root directory of `/usr/local/versant/db`, the name of the physical log volume file would be:

```
/usr/local/versant/db/mydb/physical.log
```

Logical log volume

```
<db_root>/dbname/logical.log
```

There is one logical log volume, named `logical.log`, per database. The logical log volume, is a file or raw device that, along with the physical log volume, records information about transactions and provides information for roll back and recovery. If a file is used for the logical log volume, the default location is the individual database directory.

For example, for a database named `mydb` and a database root directory of `/usr/local/versant/db`, the name of the logical log volume file would be:

```
/usr/local/versant/db/mydb/logical.log
```

Storage volumes

```
<db_root>/dbname/storage_vol
```

You can add storage files or raw devices to a database by using the `addvol` utility. There is no default name for a storage volume. If you use a file, the default location for a storage volume is the individual database directory.

Database files

When you create a database, Versant will create the following database files.

Server process profile

```
<db_root>/dbname/profile.be
```

The server process profile file is named `profile.be` and is located in the database directory for the database, which must branch from the database root directory.

For example, for a database named `mydb` and a database root directory of `/usr/local/versant/db`, the location of the server profile will be:

```
/usr/local/versant/db/mydb/profile.be
```

The server process profile is created with the `makedb` utility.

When a database starts up, the database server process reads the server process profile to determine the location of the database volumes and to set the database operating environment. If a server profile does not exist for a database, you cannot start that database.

The server process profile contains several kinds of parameters. Database creation parameters are used only when a database is created with the `createdb` utility. Database functional parameters and database tuning parameters are used each time a database is started. You can edit the functional and tuning parameters to change the operating environment of a database.

For more information on the Server Profile file, please refer to the *Versant Database Administration Manual*.

Application profile directory

`$HOME/.osc`

When you create a database first time, Versant will create a subdirectory named `.osc` under your home directory.

This directory will contain application process profile files for the databases that you create.

Application profiles

`$HOME/.osc/dbname`

When you start a session, you must name a session database. The operating environment for the application will be set per the specifications in the application process profile file corresponding to the session database. If no application profile is found, the application will use default environment settings.

An application profile has the same name as its corresponding database. For example, for a database named `mydb`, the application process profile will be:

`$HOME/.osc/mydb`

Application profiles are created with the `makeprofile` or `makedb` utilities. When these utilities are invoked, Versant will check first to see if an application profile already exists and create a new one only if there is not an existing one.

You can edit the contents of application profiles to change the operating environment for your application process.

Lock file

`<db_root>/dbname/.lock`

When you create an individual database directory with the `makeprofile` or `makedb` utility, Versant will create a file named `.lock` in your individual database directory. Versant will use the `.lock` file to determine whether a database has already been started. If the `.lock` file is destroyed for some reason, you cannot start or recreate a database until you create a new file.

For more information, please refer to the Chapter "Directories and Files" in the *Versant Database Administration Manual*.

Database type file

`<db_root>/dbname/.dbtype`

When you create an individual database directory with the `makeprofile` or `makedb` utility, Versant will create a file named `.dbtype` in your individual database directory. Versant will use the `.dbtype` file to determine whether the database is a personal or group database. If the `.dbtype` file is missing, you will not be able to start or recreate the database.

For more information, please refer to the *Versant Database Administration Manual*.

Log file

`<db_root>/LOGFILE`

When you install on a local machine, a file named `LOGFILE` is created in the database root directory. In most cases, Versant non-panic system errors are then printed to that file.

When you install to use an installation of Versant on another machine, a file named `LOGFILE` is created in the same directory as the server process profile file `profile.be`. Replication messages are then printed to that file.

Shared memory file

`<db_root>/dbname/.sharemem`

When you create a database with the `createdb` utility, Versant will create a file named `.sharemem` in your individual database directory. The contents of the `.sharemem` file are used

as a key to get a unique id for the shared memory of the database server process. If the `.sharemem` file is missing, you will not be able to start or recreate the database.

For more information, please refer to the Chapter "Directories and Files" in the *Versant Database Administration Manual*.

Password file

`<db_root>/dbname/pw`

A password file is created only if the DBA authentication is done using a password-based mechanism. The specified password is stored in this file and is accessed every time a utility that requires DBA authentication is invoked like `createdb, dbinfo, vmovedb` etc. If DBA forgets the password, the DBA can log as OS user into the machine, delete the password file and recreate a new password using the `dbuser` utility.

Configuration files

Machine configuration file

`/etc/.oscxxyyzz`

If you install using the installer, the installer will create a machine configuration file in your `/etc` directory. The name of this file depends upon the number of your Release: `xx` is the major release number, `yy` is the maintenance release number, and `zz` is the minor release number.

For example, for Release 7.0.1.3 the machine configuration file is:

`/etc/.osc070001`

The contents of the machine configuration file will be:

`VERSANT_ROOT`	The location of your software root directory.
`VERSANT_DB`	The location of your database root directory.
`VERSANT_DBID_NODE`	The machine containing your `osc-dbid` file.
`VERSANT_DBID`	The directory containing your `osc-dbid` file.

System information file

`<software_root>/lib/sysinfo`

The installer will create a system information file under your `<software_root>/lib` directory.

A local `sysinfo` file will specify the following location variables:

`VERSANT_DB`	The database directory you specified.
`VERSANT_DBID_NODE`	The machine containing your `osc-dbid` file.
`VERSANT_DBID`	The directory containing your `osc-dbid` file.

`<software_root>/bin/jdosetup.csh`

Used to setup the JDO runtime environment.

Database system identifier file

`<db_root>/osc-dbid`

The installer asks for both the machine and directory containing the `osc-dbid` file in order to eliminate NFS dependency.

The following explains what happens after the installer asks for a path to an `osc-dbid` database system file.

Same machine

If you specify that the `osc-dbid` machine is the same as the installation machine, the installer will first look for an existing `osc-dbid` file in the specified directory.

If an `osc-dbid` file already exists in the specified directory, you will join the existing system of databases.

If an `osc-dbid` file does not exist in the specified directory, the installer will create an `osc-dbid` file in the specified directory and a new database system will be started.

Remote machine

If you specify that the `osc-dbid` machine is different than the installation machine, the installer will not look to confirm if an `osc-dbid` file exists in the specified directory and will not create a new `osc-dbid` file.

Statistics files

```
<software_root>/lib/.vstatsrc
$HOME/.vstatsrc
```

When it starts, the `vstats` Statistics Utility will read a configuration file in which derived statistics can be pre-defined and named.

A system-wide `.vstatsrc` file in the `<software_root> lib` directory will always be read.

If a file named `.vstatsrc` exists in your home directory, it will also be read.

For more information, please refer to the Chapter "Directories and Files" in the *Versant Database Administration Manual*.

Uninstall

The uninstaller program for the Versant Object Database is located at:

```
<software_root>/uninstaller/UnInstaller
```

Running the UnInstaller will remove all Versant files from your machine. However, it cannot remove entries, if any made into configuration files.

After the Uninstall procedure, You may still find that some of the Versant directories have been left behind. This will happen if some new files have been added to the Versant directory structure after installation.

For example, if you have created your database directory under the Versant software root directory, this will not be deleted. If you have compiled some of the demos or tutorials and the binaries are left behind, then these directories will not be deleted.

Some files may also be left behind if some process is still accessing one of the Versant directories or files. In such situations, you will have to remove the remaining Versant directories and files manually.

Also, after installing Versant Object Database once, if you run the installer again to add more components to the existing Versant Object Database installation, then the uninstaller program may not remove the previously installed components. It is recommended that you choose all the components that you might require at the first time itself. This is because, the uninstaller program gets overwritten each time.

STEPS POST INSTALLATION

This section explains steps that you may need to take after running the Versant installer.

Step 1: Obtain and install licenses

Some of the standard and all of the Add-on components in the Versant Object Database are licensed.

A license file must be installed with proper license records obtained from Versant.

The utility "`vlicchk`" can be used to identify whether a particular component is already licensed. For example, to check whether ODBMS component version 7.0 is already licensed:

```
vlicchk –c ODBMS –v 7_0_1
```

A new license is required if there is a change in the first two digits of the version of that component.

To obtain a license record:

- Visit http://www.versant.com. If you are not a registered user, you will need to register.
- Browse to the link for requesting a license.
- In the form complete all the details. In the machine identification field supply the hostid value reported by the utility "`vinstinf`".
- A license record will be mailed to you at the email address specified.

Save the attached XML data in the email. This is the license record. If you do not have a license file <software_root>`\license.xml`, or if the present licenses are not required, then save the XML data as this file. In case there is an existing license file, then append the data to the file. The licenses are matched on the basis of the component name, version and the machine identification. In case two or more records have the same information for the component name, version and machine identification, then the first one to occur in the file will be picked up. If you are replacing a previously expired license, then it is important that the new data supplied is inserted at the beginning of the license file.

For more information on the license file, refer to Chapter "Directories and Files" in the *Versant Database Administration Manual*.

Step 2: Create a Database System File

As described in the "Pre-Installation Checklist" section, a database system identifier file named `osc-dbid` must exist and be accessible to your machine before you can create a database.

If a database system file does not exist at the location you specified during installation, you must create it before creating any databases. Since only one `osc-dbid` file should exist for a system of databases, if an `osc-dbid` file already exists anywhere in a network of databases, you should not create another one.

If during installation, you specified that you wanted a stand-alone installation, you must create a database system identifier file on your local machine.

To create an `osc-dbid` file, use the `dbid` utility on the machine that will contain it. If the machine is remote, first login to it. If the machine is local, go to the command line. Then run the following:

```
dbid -N
```

The file `osc-dbid` will then be created in the directory specified by the environment on the machine on which you invoked the command.

For more information about the `dbid` utility, please refer to Chapter "Database Utilities" in the *Versant Database Administration Manual*.

Step 3: Create a Database

If it is a new installation i.e. if you have installed for the first time, you will need to create at least one database, because to start a Versant session, you need a session database. The session database may be on your local machine or on a remote machine.

If you want to use the default configuration for your database, the steps are simple. (Assume, here, that the name of the database is to be `mydb`.)

For complete instructions on Creating a database and Resource requirements per database i.e. disk space/memory/ processes, please refer to Chapter “Create a Database” in the *Versant Database Administration Manual*.

Create a local database

For a database on your local machine, first, from the command line, create the database directories and support files.

```
makedb mydb
```

Then, create the database:

```
createdb mydb
```

Create a remote database

For a database on a machine named `alpha` (assuming you have set your environment per the above instructions,) first, from the command line, create the database directories and support files.

```
makedb mydb@alpha
```

Then, create the database:

```
createdb mydb@alpha
```

Versant uses the local parameters while creating a database:

Parameters	Description
VERSANT_DBID_NODE	To find the machine containing the `osc-dbid` database system file.
VERSANT_DBID	To find the directory containing `osc-dbid`.
VERSANT_DB	To find the local database root directory.
`local login name`	To access the remote machine containing `osc-dbid` file.

In the above case for a remote database, Versant uses the local login name to connect to alpha and then sends the makedb and createdb commands to it. Versant location parameters set on alpha will be used to find the osc-dbid database system identifier file and the database root directory used by alpha. If osc-dbid is on a machine other than alpha, Versant will also use the local login name to access the machine containing the osc-dbid file.

Step 4: Convert and Backup Existing Databases

If you are upgrading from old version:

Convert existing databases to Release 7.0.1.4 format

To convert existing databases of Versant to the current release (7.0.1.4):

- Use the conversion utility (`convertdb`) of the new release, to convert the databases on the installation machine to 7.0.1.4 format.

The following table briefly describes the conversion procedure to Release 7.0.1.4 database from the previous release:

FROM (prior release)	TO (current release)	PROCEDURE
6.x (32-bit platform)	7.x (32-bit platform)	Run `convertdb` from 7.x (32-bit) Release.
6.x (32-bit platform)	7.x (64-bit platform)	Run `convertdb` from 7.x (64-bit) Release.
6.x (64-bit platform)	7.x (32-bit platform)	Not supported.
6.x (64-bit platform)	7.x (64-bit platform)	Run `convertdb` from 7.x (64-bit) Release.

After a database has been converted to a 7.0.1.4 database, you can still access it with applications prior to 7.0.1.4 release.

For more information related to release compatibility, please refer to the “Pre-Installation Checklist” on page 180.

Convert the Database

For a database named `mydb`, first stop it with the following (do not use `stopdb -f`):

```
stopdb mydb
```

Then convert it with:

```
convertdb mydb
```

For more information, on the convertdb utility, please refer to Chapter "Database Utilities" in the *Versant Database Administration Manual*.

RESTRICTION:- After conversion of a database to this release, the users in the database have no passwords. The authentication mechanism for such users is described in the chapter "Controlling Access To Database" in the *Versant Database Fundamentals Manual*.

Backup existing databases

Backup files belonging to older releases cannot be used to restore databases in the major releases. If you have backup files belonging to older releases, restore the database to the corresponding older release first. Then use convertdb to migrate the database to the 7.0.1.4 release.

If a database is part of a FTS replica pair needs to be converted to the 7.0.1.4 release, the client environment needs to ensure that an entry for the replica pair is present in the replica file in the VERSANT_ROOT directory of the 7.0.1.4 client environment.

After a database has been converted to Release 7.0.1.4 format, you will not be able to use backup files or tapes created with releases prior to Release 7.0.1.4 to restore it. Accordingly, before proceeding, you should backup your existing databases.

Backups can be created with the `vbackup` utility.

For example, to backup a database named `mydb` to the file `/tmp/level0`:

```
vbackup -device /tmp/level0 -backup mydb
```

For more information, on the `vbackup` utility, please refer to Chapter "Database Utilities" in the *Versant Database Administration Manual*.

Step 5: Licensing the Add-on Components

Versant Object Database has following Add-ons which require separate licensing.

Add-on components:

COMPONENT	DETAILS
VAR	Versant Asynchronous Replication for Versant ODBMS.
Vedding	Fault Tolerant Versant ODBMS Server.
HABACKUP	Backup Solution for use with High Availability Server
Vorkout	Versant Online Database Reorganization Tool

ReVind	Structured Query Language Interface for Versant ODBMS
Warm Standby	Used for an Incremental Rollforward recovery

Licensing Vedding

Vedding(FTS) is distributed within the Versant Object Database installation. However a separate license is needed for it to operate.

Contact Versant Customer Support for a license. You will need to provide the same information as that for the ODBMS license. The license is similar to the ODBMS license and a separate license is required for each participating servers. Append the XML data provided by Versant Customer Support to the license file <software_root>`/license.xml`.

Once this is done, this option is licensed.

To test, setup a FTS pair of databases (`db1` and `db2`) on the server (`server`):

1. Edit the file `<software_root>/replica` and add the line "`db1@server db2@server`".
2. Execute the commands on the server:

```
makedb db1
createdb db1
makedb db2
createreplica db1 db2
```

3. The "`createreplica`" command above should complete successfully.

You can also check whether the Fault Tolerant Server option is licensed by executing

`vlicchk –c FTS –v 7.0` -- which should report that it is licensed.

NOTE:- All other Add-ons can be licensed in a similar manner.

KNOWN ISSUE WITH UNINSTALLER

You might experience some issue with the uninstaller not removing files.

This is an issue with InstallAnywhere's global registry file (.com.zerog.registry.xml).

On Windows, this global registry is located at: Program Files\Zero G Registry\

On UNIX systems, if logged in as root, the global registry is located in the /var directory. If logged in as user, it is in the user's home directory.

In this case, please remove Versant Object Database installation manually.

SETTING ENVIRONMENT FOR SOLARIS

Mandatory Settings

Automated Settings

The script file `envsettings.csh` is located in the directory `<software_root>/bin`.

This script sets the environment variables `PATH`, `LD_LIBRARY_PATH` and the `CLASSPATH`. It also sets the environment variables `VERSANT_ROOT`, `VERSANT_DBID`, `VERSANT_DBID_NODE` and `VERSANT_VERS`. It sets the `JAVA_COMPILER` to `NONE`.

If you carry out the installation as super-user 'root', it will automatically set the Versant Service.

Manual Settings

Add the executables directory to your system path

After installation, you should add the Versant executables directory to your system path if you want to execute Versant utilities without a path specification.

For a standard installation, the Versant executables directory will be named `/bin` and be located below the software root directory:

```
<software_root>/bin
```

For example:

```
/usr/local/versant/7.0/bin
```

C Shell

If you use C Shell, the search path should be in the `.login` in your home directory. To amend it to include the `<software_root>/lib` directory, use the `set path` command.

```
setenv path <software_root>/bin:$path
```

Bourne Shell

If you use Bourne shell, the search path will be in `.profile` in your home directory. To amend it to include the `VERSANT /bin` directory for a default installation:

```
PATH=<software_root>/bin:${PATH}
export PATH
```

Add the library directory to your system path

On systems which support shared libraries, if Versant has provided shared libraries for linking with your applications, you may need to set an environment variable, which tells your application where to find shared libraries at run-time.

The environment variable for shared libraries has different names on different UNIX systems.

For Solaris — `LD_LIBRARY_PATH`

For other systems, please see Setting Environment for respective platform.

The commands you need to use will also vary depending upon your operating system and shell.

For example, for Solaris and C Shell (substitute the name of your actual software root directory):

```
setenv LD_LIBRARY_PATH <software_root>/lib:$LD_LIBRARY_PATH
```

For Solaris and Bourne Shell (substitute the name of your actual software root directory):

```
LD_LIBRARY_PATH=<software_root>/lib:$LD_LIBRARY_PATH
export LD_LIBRARY_PATH
```

Modify the environment variables to run JVI after the installation

The following description assumes Versant ODBMS version to be 7.0.1.4, and process model to be two-process model (2p):

Modify the `CLASSPATH` environment variable.

If you use C shell:

```
setenv CLASSPATH <software_root>/lib/jvi7.0.1-jdk1.4.jar:$CLASSPATH
```

If you are using shell other than the C shell, the syntax for setting environment variables may be different.

NOTE:- If you are using the one-process model, modify the environment variable used to find shared libraries at run time to `<software_root>/lib/jvi/1p` instead of `<software_root>/lib/`.

Modify the environment variables to run VXML after the installation

Modify the `CLASSPATH` environment variable to include the following:

If you use C shell:

```
setenv CLASSPATH\
<software_root>/lib/jvi7.0.1-jdk1.4.jar:\
<software_root>/lib/vxml.jar:\
<software_root>/lib/xerces.jar:\
<software_root>/lib/xalan.jar:\
<software_root>/lib/xml-apis.jar:$CLASSPATH
```

If you are using shell other than the C shell, the syntax for setting environment variables may be different.

Modify the environment variables to run VAR after the installation

Modify the `CLASSPATH` environment variable to include the following:

If you use C shell:

```
setenv CLASSPATH \
<software_root>/lib/jvi7.0.1-jdk1.4.jar:\
<software_root>/lib/var.jar:$CLASSPATH
```

If you are using shell other than the C shell, the syntax for setting environment variables may be different.

Optional Settings

Following should be done, if you have not chosen the installer to modify the environment parameters:

Optionally set the VERSANT_ROOT environment variable

The environment variable VERSANT_ROOT must be set to your software root directory. Setting VERSANT_ROOT after installation is optional, as it will normally be set for you during installation.

C Shell

If you use C Shell, set VERSANT_ROOT in the .login file in your home directory:

```
setenv VERSANT_ROOT <software_root>
```

Bourne Shell

If you use Bourne shell, set VERSANT_ROOT in .profile in your home directory:

```
VERSANT_ROOT=<software_root>
export VERSANT_ROOT
```

Possibly specify the location of the database network system identifier file

If you had installed as super user, during installation you were asked for the location of the osc-dbid database system identifier file. If you specified the location of the osc-dbid file during installation, you do not have to specify it after installation.

To tell Versant where osc-dbid is located, set the VERSANT_DBID_NODE and VERSANT_DBID environment variable by performing the following steps:

1. Optionally refresh your memory.

   Optionally refresh your memory as to the location of osc-dbid that you specified during installation by running the getdbid script:

   ```
   getdbid
   ```

   The getdbid utility might, for example, return:

   ```
   /usr/local/versant/db
   ```

2. Set environment variables.

Set VERSANT_DBID_NODE to the name of the machine containing the osc-dbid file and VERSANT_DBID to the directory containing the osc-dbid file.

C Shell

For C Shell, you can set it directly:

```
setenv VERSANT_DBID_NODE dbid_machine
setenv VERSANT_DBID dbid_directory
```

or you can use the getdbid script as part of the setenv command:

```
setenv VERSANT_DBID 'getdbid'
```

Bourne Shell

For Bourne Shell, to set it directly:

```
VERSANT_DBID_NODE=dbid_machine
export VERSANT_DBID_NODE
VERSANT_DBID=dbid_directory
export VERSANT_DBID
```

or to use the getdbid script:

```
VERSANT_DBID = 'getdbid'
export VERSANT_DBID
```

Possibly update the Yellow Pages

If the machine on which you install Versant is not the Yellow Pages master, you must manually update /etc/services on the master machine by performing the following steps:

1. Add the following line at the end of /etc/services

   ```
   oscssd 5019/tcp #versant SS daemon
   ```

   If the port 5019 is already used, then specify a different port number.
2. Update YP on the Yellow Pages Master

   ```
   cd /var/yp
   make
   ```

   Since the Yellow Pages Master does not distribute the inetd.conf file, the inetd.conf file has to be updated on all machines that allow remote access to their databases. You can do it manually by performing the following steps:

   1. Add the following line at the end of /etc/inetd.conf

      ```
      oscssd stream tcp nowait root /usr/etc/ss.d in.oscssd
      ```

   2. Ask the inet daemon to parse the modified file:

On most machines

If you are running sh:

```
ps xg | grep inet[d] | while read pid junk
do
    kill -1 $pid
done
```

If you are running csh:

```
set p = `ps xg | grep inetd | awk '{ print $2 }'`
foreach i ($p)
    kill -1 $i
end
```

On some machines

If you are running sh:

```
ps -e | grep inet[d] | while read pid junk
do
    kill -1 $pid
done
```

If you are running csh:

```
set p = `ps -e | grep inetd | awk '{ print $2 }'`
foreach i ($p)
    kill -1 $i
end
```

If you are a diskless client with a read-only root

Some systems mount the root file system of diskless clients as read-only, which will prevent the Versant installer from updating necessary system files.

If the root file system of a diskless client has been mounted as read-only, after installation of Versant on the server you will need to update manually the Versant machine configuration file and your machines `inetd.conf` and `services` files.

The name of your machine configuration file will depend upon your Release number. For Release 7.0.1.4, the machine configuration file is `/etc/.osc070001`.

The `inetd.conf` file lists servers, and the `services` file lists services.

In a standard installation, the paths to these files are:

```
/export/<software_root>/client/etc/.osc070001
/export/<software_root>/client/etc/inetd.conf
/export/<software_root>/client/etc/services
```

where `client` is the name of the client machine.

The necessary steps are:

1. Copy the binaries

   Copy the Versant binaries from the Versant server file system to the client file system.

   ```
   cp -p /root/bin/ss.d /export/client/etc
   ```

2. Update machine configuration file

   If you want to share program files and databases, the home directory for system software files must be readable and the home directory for databases must be readable and writable to other users on the network. Also, the path and name of the file containing paths to software and databases must be entered in a system path file.

   The name of this file will depend upon your release number. For example, for Release 7.0.1.3, the name of the system file is:

   ```
   /etc/.osc070001
   ```

   The `/etc/path_file` file should have a line to specify each of the following locations. Each line first states the configuration variable name and then supplies a value.

   `VERSANT_ROOT software_root`

   Location of software root directory.

   `VERSANT_DB database_root`

   Location of database root directory.

   `VERSANT_DBID_NODE` *`dbid_machine`*

   Machine with osc-dbid file.

   `VERSANT_DBID` *`dbid_path`*

   The directory containing the osc-dbid file.

   To assign ownership of the `/etc/path_file` file:

```
chown group_name /etc/path_file
```

For example:

```
chown dbsa.dbgroup /etc/.osc070001
```

For more explanation, of the above configuration variables, please refer to Chapter "Configuration Parameters" in the *Versant Database Administration Manual*.

3. Update `inetd.conf` and `services`.

   You must next update the contents of the Yellow Pages files `/etc/inetd.conf` and `/etc/services`. The locations of these files for a standard installation, where `client` is the name of the client machine, are:

   `/export/<software_root>/client/etc/inetd.conf`

   `/export/<software_root>/client/etc/services`

   To update these files:

   - Add the following line at the end of `/etc/services`:

     ```
     oscssd 5019/tcp #versant SS daemon
     ```

     If the port `5019` is already used, then specify a different port number.

   - Update `YP` on the Yellow Pages Master:

     ```
     cd /var/yp
     make
     ```

   Since the Yellow Pages Master does not distribute the `inetd.conf` file, the `inetd.conf` file has to be updated on all machines that allow remote access to their databases. To do this, perform the following steps:

   - Add the following line at the end of `/etc/inetd.conf`:

     ```
     oscssd stream tcp nowait root /usr/etc/ss.d in.oscssd
     ```

   - As root on the client machine, ask the `inet daemon` to parse the modified file.

     On most machines

     If you are running sh:

     ```
     ps xg | grep inet[d] | while read pid junk
     do
         kill -1 $pid
     done
     ```

     If you are running csh:

     ```
     set p = `ps xg | grep inetd | awk '{ print $2 }'`
     foreach i ($p)
     ```

```
    kill -1 $i
end
```

On some machines

If you are running sh:

```
ps -e | grep inet[d] | while read pid junk
do
    kill -1 $pid
done
```

If you are running csh:

```
set p = `ps -e | grep inetd | awk '{ print $2 }'`
foreach i ($p)
    kill -1 $i
end
```

If you later want to change the database root directory

If you use the defaults when you install, the database root directory will be placed under your Versant root directory.

After installation and even after databases have been created, you can move the database root directory to a new location as long as you specify the new location in an environment parameter.

For more information, on specifying the database root directory, please refer to Chapter "Configuration Parameters" in the *Versant Database Administration Manual*.

Optionally change permissions on database root directory

If you use the defaults when you install and place the database root directory under your Versant root directory, you will own the database root directory.

If you want to allow others to be able to create databases using this directory, you must change permissions on the database root directory to `777`.

For example:

```
chmod 777 VERSANT_ROOT/db
```

This change is only needed when the database root directory is owned by the DBSA (who installed Versant) and the user that is attempting to create the database is not the DBSA.

SETTING ENVIRONMENT FOR LINUX

Mandatory Settings

Automated Settings

The script file `envsettings.csh` is located in the directory `<software_root>/bin`.

This script sets the environment variables `PATH`, `LD_LIBRARY_PATH` and the `CLASSPATH`. It also sets the environment variables `VERSANT_ROOT`, `VERSANT_DB`, `VERSANT_DBID`, `VERSANT_DBID_NODE` and `VERSANT_VERS`. It sets the `JAVA_COMPILER` to `NONE`.

If you carry out the installation as super-user ‘root’, it will automatically set the Versant Service.

Manual Settings

Add the executables directory to your system path

After installation, you should add the Versant executables directory to your system path if you to execute Versant utilities without a path specification.

For a standard installation, the Versant executables directory will be named `/bin` and be located below the software root directory:

```
<software_root>/bin
```

For example:

```
/usr/local/versant/7.0/bin
```

C Shell

If you use C Shell, the search path should be in the `.login` in your home directory. To amend it to include the `<software_root>/lib` directory, use the `set path` command:

```
setenv path <software_root>/bin:$path
```

Bourne Shell

If you use Bourne shell, the search path will be in `.profile` in your home directory. To amend it to include the `VERSANT /bin` directory for a default installation:

```
PATH=<software_root>/bin:${PATH}
export PATH
```

Add the library directory to your system path

On systems which support shared libraries, if Versant has provided shared libraries for linking with your applications, you may need to set an environment variable, which tells your application where to find shared libraries at run-time.

The environment variable for shared libraries has different names on different UNIX systems.

For Red Hat Enterprise Linux — `LD_LIBRARY_PATH`

For other systems, please see Setting Environment for respective platforms.

The commands you need to use will also vary depending upon your operating system and shell.

For example, for Red Hat Linux and C Shell (substitute the name of your actual software root and compiler directories):

```
setenv LD_LIBRARY_PATH <software_root>/lib:$LD_LIBRARY_PATH
```

For Red Hat Enterprise Linux and Bourne Shell (substitute the name of your actual software root and compiler directories):

```
LD_LIBRARY_PATH=<software_root>/lib:$LD_LIBRARY_PATH
export LD_LIBRARY_PATH
```

Modify the environment variables to run JVI after the installation

The following description assumes Versant ODBMS version to be 7.0.1.4, and process model to be two-process model (2p):

Modify the `CLASSPATH` environment variable.

If you use C shell:

```
setenv CLASSPATH <software_root>/lib/jvi7.0.1-jdk1.4.jar:$CLASSPATH
```

If you are using shell other than the C shell, the syntax for setting environment variables may be different.

NOTE:- If you are using the one-process model, modify the environment variable used to find shared libraries at run time to `<software_root>/lib/jvi/1p` instead of `<software_root>/lib/`.

Modify the environment variables to run VXML after the installation

Modify the `CLASSPATH` environment variable to include the following:

If you use C shell:

```
setenv CLASSPATH\
<software_root>/lib/jvi7.0.1-jdk1.4.jar:\
<software_root>/lib/vxml.jar:\
<software_root>/lib/xerces.jar:\
<software_root>/lib/xalan.jar:\
<software_root>/lib/xml-apis.jar:$CLASSPATH
```

If you are using shell other than the C shell, the syntax for setting environment variables may be different.

Modify the environment variables to run VAR after the installation

Modify the `CLASSPATH` environment variable to include the following:

If you use C shell:

```
setenv CLASSPATH\
<software_root>/lib/jvi7.0.1-jdk1.4.jar:\
<software_root>/lib/var.jar:$CLASSPATH
```

If you are using shell other than the C shell, the syntax for setting environment variables may be different.

Optional Settings

Following should be done, if you have not chosen the installer to modify the environment parameters:

Optionally set the VERSANT_ROOT environment variable

The environment variable VERSANT_ROOT must be set to your software root directory. Setting VERSANT_ROOT after installation is optional, as it will normally be set for you during installation.

C Shell

If you use C Shell, set VERSANT_ROOT in the .login file in your home directory:

```
setenv VERSANT_ROOT <software_root>
```

Bourne Shell

If you use Bourne shell, set VERSANT_ROOT in .profile in your home directory:

```
VERSANT_ROOT=<software_root>
export VERSANT_ROOT
```

Possibly specify the location of the database network system identifier file

If you had installed as super user, during installation you were asked for the location of the osc-dbid database system identifier file. If you specified the location of the osc-dbid file during installation, you do not have to specify it after installation.

To tell Versant where osc-dbid is located, set the VERSANT_DBID_NODE and VERSANT_DBID environment variable by performing the following steps:

1. Optionally refresh your memory.

   Optionally refresh your memory as to the location of osc-dbid that you specified during installation by running the getdbid script:

   ```
   getdbid
   ```

   The getdbid utility might, for example, return:

   ```
   /usr/local/versant/db
   ```

2. Set environment variables.

Set VERSANT_DBID_NODE to the name of the machine containing the osc-dbid file and VERSANT_DBID to the directory containing the osc-dbid file.

C Shell

For C Shell, you can set it directly:

```
setenv VERSANT_DBID_NODE dbid_machine
setenv VERSANT_DBID dbid_directory
```

or you can use the getdbid script as part of the setenv command:

```
setenv VERSANT_DBID 'getdbid'
```

Bourne Shell

For Bourne Shell, to set it directly:

```
VERSANT_DBID_NODE=dbid_machine
export VERSANT_DBID_NODE
VERSANT_DBID=dbid_directory
export VERSANT_DBID
```

or to use the getdbid script:

```
VERSANT_DBID = 'getdbid'
export VERSANT_DBID
```

Possibly update the Yellow Pages

If the machine on which you install Versant is not the Yellow Pages master, you must manually update /etc/services on the master machine by performing the following steps:

1. Add the following line at the end of /etc/services

   ```
   oscssd 5019/tcp #versant SS daemon
   ```

   If the port 5019 is already used, then specify a different port number.
2. Update YP on the Yellow Pages Master

   ```
   cd /var/yp
   make
   ```

   Since the Yellow Pages Master does not distribute the inetd.conf file, the inetd.conf file has to be updated on all machines that allow remote access to their databases. If /etc/inetd.conf is not present create a file inetd.conf in /etc directory. As user root, you can update the file manually by performing the following steps:

   1. Add the following line at the end of /etc/inetd.conf

      ```
      oscssd stream tcp nowait root /usr/etc/ss.d
      ```

2. Convert VERSANT SERVICE to xinetd form

   Create the configuration file oscssd for service oscssd in /etc/xinetd.d.

   The contents of the file should be as follows:

```
service oscssd
{
   socket_type             = stream
   protocol                = tcp
   wait                    = no
   user                    = root
   server                  = /usr/etc/ss.d
   disable                 = no

}
```

NOTE:- You can optionally convert VERSANT SERVICE to xinetd form using inetdconvert command: /usr/sbin/inetdconvert --inetdfile /etc/inetd.conf oscssd

3. Ask the inet daemon to start oscssd service

   On most machines

   If you are running sh:

```
ps xg | grep inet[d] | while read pid junk
do
    kill –USR2 $pid
done
```

   If you are running csh:

```
set p = `ps xg | grep inetd | awk '{ print $2 }'`
foreach i ($p)
    kill –USR2 $i
end
```

   On some machines

   If you are running sh:

```
ps -e | grep inet[d] | while read pid junk
do
    kill –USR2 $pid
```

```
    done
```

If you are running csh:

```
    set p = `ps -e | grep inetd | awk '{ print $2 }'`
    foreach i ($p)
        kill –USR2 $i
    end
```

If you are a diskless client with a read-only root

Some systems mount the root file system of diskless clients as read-only, which will prevent the Versant installer from updating necessary system files.

If the root file system of a diskless client has been mounted as read-only, after installation of Versant on the server you will need to update manually the Versant machine configuration file and your machines `inetd.conf` and `services` files.

The name of your machine configuration file will depend upon your Release number. For example, for Release 7.0.1.3, the machine configuration file is `/etc/.osc070001`. The `/etc/services` file lists services.

The `xinetd.conf` is the configuration file that determines the services provided by `xinetd`. This file includes the `/etc/xinetd.d` directory. Every file inside this directory will be parsed as an `xinetd` configuration file. In Versant installation `inetd.conf` specifies `VERSANT SERVICE`. Hence `VERSANT SERVICE` needs to be converted to `xinetd` form.

In a standard installation, the paths to these files are:

```
/export/<software_root>/client/etc/.osc070001
/export/<software_root>/client/etc/inetd.conf
/export/<software_root>/client/etc/services
```

where `client` is the name of the client machine.

The necessary steps are:

1. Copy the binaries

   Copy the Versant binaries from the Versant server file system to the client file system.

   ```
   cp -p /root/bin/ss.d /export/client/etc
   ```

2. Update machine configuration file

   If you want to share program files and databases, the home directory for system software files must be readable and the home directory for databases must be readable and writable

to other users on the network. Also, the path and name of the file containing paths to software and databases must be entered in a system path file.

The name of this file will depend upon your release number. For example, for Release 7.0.1.3, the name of the system file is:

`/etc/.osc070001`

The `/etc/path_file` file should have a line to specify each of the following locations. Each line first states the configuration variable name and then supplies a value.

`VERSANT_ROOT software_root`

Location of software root directory.

`VERSANT_DB database_root`

Location of database root directory.

`VERSANT_DBID_NODE` *`dbid_machine`*

Machine with `osc-dbid` file.

`VERSANT_DBID` *`dbid_path`*

The directory containing the `osc-dbid` file.

For more explanation, of the above configuration variables, please refer to Chapter "Configuration Parameters" in the *Versant Database Administration Manual*.

To assign ownership of the `/etc/path_file` file:

```
chown group_name /etc/path_file
```

For example:

```
chown dbsa.dbgroup /etc/.osc070001
```

3. Update `inetd.conf` and `services`.

You must next update the contents of the Yellow Pages files `/etc/inetd.conf` and `/etc/services`. The locations of these files for a standard installation, where `client` is the name of the client machine, are:

`/export/<software_root>/`*`client`*`/etc/inetd.conf`

`/export/<software_root>/`*`client`*`/etc/services`

To update these files:

- Add the following line at the end of `/etc/services`:

```
oscssd  5019/tcp  #versant SS daemon
```

If the number 5019 is already used, then specify a different number.

- Update YP on the Yellow Pages Master:

```
cd /var/yp
make
```

Since the Yellow Pages Master does not distribute the inetd.conf file, the inetd.conf file has to be updated on all machines that allow remote access to their databases. If /etc/inetd.conf is not present create a file inetd.conf in /etc directory. To update the file, perform the following steps:

- Add the following line at the end of /etc/inetd.conf:

```
oscssd stream tcp nowait root /usr/etc/ss.d
```

- Convert VERSANT SERVICE to xinetd form

As root on the client machine, create the configuration file oscssd for service oscssd in /etc/xinetd.d.

The contents of the file should be as follows:

```
service oscssd
{
   socket_type             = stream
   protocol                = tcp
   wait                    = no
   user                    = root
   server                  = /usr/etc/ss.d
   disable                 = no
}
```

NOTE:- You can optionally convert VERSANT SERVICE to xinetd form using inetdconvert command:

/usr/sbin/inetdconvert --inetdfile /etc/inetd.conf oscssd

- As root on the client machine, ask the inet daemon to start the VERSANT SERVICE.

On most machines

If you are running sh:

```
ps xg | grep inet[d] | while read pid junk
do
    kill –USR2 $pid
```

```
done
```

If you are running csh:

```
set p = `ps xg | grep inetd | awk '{ print $2 }'`
foreach i ($p)
    kill –USR2 $i
end
```

On some machines

If you are running sh:

```
ps -e | grep inet[d] | while read pid junk
do
    kill –USR2 $pid
done
```

If you are running csh:

```
set p = `ps -e | grep inetd | awk '{ print $2 }'`
foreach i ($p)
    kill –USR2 $i
end
```

If you later want to change the database root directory

If you use the defaults when you install, the database root directory will be placed under your Versant root directory.

After installation and even after databases have been created, you can move the database root directory to a new location as long as you specify the new location in an environment parameter.

For more information, on specifying the database root directory, please refer to Chapter "Configuration Parameters" in the *Versant Database Administration Manual*.

Optionally change permissions on database root directory

If you use the defaults when you install and place the database root directory under your Versant root directory, you will own the database root directory.

If you want to allow others to be able to create databases using this directory, you must change permissions on the database root directory to `777`.

For example:

```
chmod 777 VERSANT_ROOT/db
```

This change is only needed when the database root directory is owned by the DBSA (who installed Versant) and the user that is attempting to create the database is not the DBSA.

SETTING ENVIRONMENT FOR HP-UX

Mandatory Settings

Automated Settings

The script file `envsettings.csh` is located in the directory `<software_root>/bin`.

This script sets the environment variables `PATH`, `SHLIB_PATH` and the `CLASSPATH`. It also sets the environment variables `VERSANT_ROOT`, `VERSANT_DBID`, `VERSANT_DBID_NODE` and `VERSANT_VERS`. It sets the `JAVA_COMPILER` to `NONE`.

If you carry out the installation as super-user 'root', it will automatically set the Versant Service.

Manual Settings

Add the executables directory to your system path

After installation, you should add the Versant executables directory to your system path if you want to execute Versant utilities without a path specification.

For a standard installation, the Versant executables directory will be named `/bin` and be located below the `<software_root>` directory:

```
<software_root>/bin
```

For example:

```
/usr/local/versant/7.0/bin
```

C Shell

If you use C Shell, the search path should be in the `.login` in your home directory. To amend it to include the VERSANT `/bin` directory, use the `set path` command.

```
setenv PATH <software_root>/lib:$PATH
```

Bourne Shell

If you use Bourne shell, the search path will be in `.profile` in your home directory. To amend it to include the `VERSANT /bin` directory for a default installation:

```
PATH=<software_root>/bin:${PATH}
export PATH
```

Add the library directory to your system path

On systems which support shared libraries, if Versant has provided shared libraries for linking with your applications, you may need to set an environment variable, which tells your application where to find shared libraries at run-time.

The environment variable for shared libraries has different names on different UNIX systems.

For HP-UX — `SHLIB_PATH`

For other systems, please see Setting Environment for respective platform.

The commands you need to use will also vary depending upon your operating system and shell.

For example, for HP-UX 11i and C Shell (substitute the name of your actual software root and compiler directories):

```
setenv SHLIB_PATH <software_root>/lib:$SHLIB_PATH
```

For HP-UX 11i and Bourne Shell (substitute the name of your actual software root and compiler directories):

```
SHLIB_PATH=<software_root>/lib:$SHLIB_PATH
export SHLIB_PATH
```

Modify the environment variables to run JVI after the installation

The following description assumes Versant ODBMS version to be 7.0.1.4, and process model to be two-process model (2p):

Modify the `CLASSPATH` environment variable.

If you use C shell:

```
setenv CLASSPATH <software_root>/lib/jvi7.0.1-jdk1.4.jar:$CLASSPATH
```

If you are using shell other than the C shell, the syntax for setting environment variables may be different.

NOTE:- If you are using the one-process model, modify the environment variable used to find shared libraries at run time to <software_root>/lib/jvi/1p instead of <software_root>/lib/

Modify the environment variables to run VXML after the installation

Modify the CLASSPATH environment variable to include the following:

If you use C shell:

```
setenv CLASSPATH \
<software_root>/lib/jvi7.0.1-jdk1.4.jar:\
<software_root>/lib/vxml.jar:\
<software_root>/lib/xerces.jar:\
<software_root>/lib/xalan.jar:\
<software_root>/lib/xml-apis.jar:$CLASSPATH
```

If you are using shell other than the C shell, the syntax for setting environment variables may be different.

Modify the environment variables to run VAR after the installation

Modify the CLASSPATH environment variable to include the following:

If you use C shell:

```
setenv CLASSPATH \
<software_root>/lib/jvi7.0.1-jdk1.4.jar:\
<software_root>/lib/var.jar:$CLASSPATH
```

If you are using shell other than the C shell, the syntax for setting environment variables may be different.

Optional Settings

Following should be done, if you have not chosen the installer to modify the environment parameters:

Optionally set the VERSANT_ROOT environment variable

The environment variable VERSANT_ROOT must be set to your software root directory. Setting VERSANT_ROOT after installation is optional, as it will normally be set for you during installation.

C Shell

If you use C Shell, set VERSANT_ROOT in the .login file in your home directory:

```
setenv VERSANT_ROOT <software_root>
```

Bourne Shell

If you use Bourne shell, set VERSANT_ROOT in .profile in your home directory:

```
VERSANT_ROOT=<software_root>
export VERSANT_ROOT
```

Possibly specify the location of the database network system identifier file

If you have installed as super user, during installation you were asked for the location of the osc-dbid database system identifier file. If you specified the location of the osc-dbid file during installation, you do not have to specify it after installation.

To tell Versant where osc-dbid is located, set the VERSANT_DBID_NODE and VERSANT_DBID environment variable by performing the following steps:

1. Optionally refresh your memory.

   Optionally refresh your memory as to the location of osc-dbid that you specified during installation by running the getdbid script:

   ```
   getdbid
   ```

   The getdbid utility might, for example, return:

   ```
   /usr/local/versant/db
   ```

2. Set environment variables.

Set VERSANT_DBID_NODE to the name of the machine containing the osc-dbid file and VERSANT_DBID to the directory containing the osc-dbid file.

C Shell

For C Shell, you can set it directly:

```
setenv VERSANT_DBID_NODE dbid_machine
setenv VERSANT_DBID dbid_directory
```

or you can use the getdbid script as part of the setenv command:

```
setenv VERSANT_DBID 'getdbid'
```

Bourne Shell

For Bourne Shell, to set it directly:

```
VERSANT_DBID_NODE=dbid_machine
export VERSANT_DBID_NODE
VERSANT_DBID=dbid_directory
export VERSANT_DBID
```

or to use the getdbid script:

```
VERSANT_DBID = 'getdbid'
export VERSANT_DBID
```

Possibly update the Yellow Pages

If the machine on which you install Versant is not the Yellow Pages master, you must manually update /etc/services on the master machine by performing the following steps:

1. Add the following line at the end of /etc/services

   ```
   oscssd  5019/tcp  #versant SS daemon
   ```

   If the port 5019 is already used, then specify a different port number.
2. Update YP on the Yellow Pages Master

   ```
   cd /var/yp
   make
   ```

   Since the Yellow Pages Master does not distribute the inetd.conf file, the inetd.conf file has to be updated on all machines that allow remote access to their databases. You can do it manually by performing the following steps:

   1. Add the following line at the end of /etc/inetd.conf

      ```
      oscssd stream tcp nowait root /usr/etc/ss.d in.oscssd
      ```

   2. Ask the inet daemon to parse the modified file

On most machines

If you are running sh:

```
ps xg | grep inet[d] | while read pid junk
do
    kill -1 $pid
done
```

If you are running csh:

```
set p = `ps xg | grep inetd | awk '{ print $1 }'`
foreach i ($p)
    kill -1 $i
end
```

On some machines

If you are running sh:

```
ps -e | grep inet[d] | while read pid junk
do
    kill -1 $pid
done
```

If you are running csh:

```
set p = `ps -e | grep inetd | awk '{ print $1 }'`
foreach i ($p)
    kill -1 $i
end
```

If you are a diskless client with a read-only root

Some systems mount the root file system of diskless clients as read-only, which will prevent the Versant installer from updating necessary system files.

If the root file system of a diskless client has been mounted as read-only, after installation of Versant on the server you will need to update manually the Versant machine configuration file and your machines `inetd.conf` and `services` files.

The name of your machine configuration file will depend upon your Release number. For example, for Release 7.0.1.3, the machine configuration file is `/etc/.osc070001`.

The `inetd.conf` file lists servers, and the `services` file lists services.

In a standard installation, the paths to these files are:

```
/export/<software_root>/client/etc/.osc070001
/export/<software_root>/client/etc/inetd.conf
/export/<software_root>/client/etc/services
```

.where `client` is the name of the client machine.

The necessary steps are:

1. Copy the binaries

   Copy the Versant binaries from the Versant server file system to the client file system.

   ```
   cp -p /<versant_root>/bin/ss.d /export/client/etc
   ```

2. Update machine configuration file

   If you want to share program files and databases, the home directory for system software files must be readable and the home directory for databases must be readable and writable to other users on the network. Also, the path and name of the file containing paths to software and databases must be entered in a system path file.

   The name of this file will depend upon your release number. For example, for Release 7.0.1.3, the name of the system file is:

   ```
   /etc/.osc070001
   ```

   The `/etc/path_file` file should have a line to specify each of the following locations. Each line first states the configuration variable name and then supplies a value.

   `VERSANT_ROOT software_root`

   Location of software root directory.

   `VERSANT_DB database_root`

   Location of database root directory.

   `VERSANT_DBID_NODE` *`dbid_machine`*

   Machine with `osc-dbid` file.

   `VERSANT_DBID` *`dbid_path`*

   The directory containing the `osc-dbid` file.

For more explanation, of the above configuration variables, please refer to Chapter "Configuration Parameters" in the *Versant Database Administration Manual*.

To assign ownership of the /etc/path_file file:

```
chown group_name /etc/path_file
```

For example:

```
chown dbsa.dbgroup /etc/.osc070001
```

3. Update inetd.conf and services.

You must next update the contents of the Yellow Pages files /etc/inetd.conf and /etc/services. The locations of these files for a standard installation, where client is the name of the client machine, are:

/export/root/*client*/etc/inetd.conf

/export/root/*client*/etc/services

To update these files:

- Add the following line at the end of /etc/services:

  ```
  oscssd  5019/tcp  #versant SS daemon
  ```

  If the port 5019 is already used, then specify a different port number.

- Update YP on the Yellow Pages Master:

  ```
  cd /var/yp
  make
  ```

Since the Yellow Pages Master does not distribute the inetd.conf file, the inetd.conf file has to be updated on all machines that allow remote access to their databases. To do this, perform the following steps:

- Add the following line at the end of /etc/inetd.conf:

  ```
  oscssd stream tcp nowait root /usr/etc/ss.d in.oscssd
  ```

- As root on the client machine, ask the inet daemon to parse the modified file.

On most machines

If you are running sh:

```
ps xg | grep inet[d] | while read pid junk
do
    kill -1 $pid
done
```

If you are running csh:

```
set p = `ps xg | grep inetd | awk '{ print $1 }'`
foreach i ($p)
    kill -1 $i
```

```
end
```

On some machines

If you are running sh:

```
ps -e | grep inet[d] | while read pid junk
do
    kill -1 $pid
done
```

If you are running csh:

```
set p = `ps -e | grep inetd | awk '{ print $1 }'`
foreach i ($p)
    kill -1 $i
end
```

If you later want to change the database root directory

If you use the defaults when you install, the database root directory will be placed under your Versant software directory.

After installation and even after databases have been created, you can move the database root directory to a new location as long as you specify the new location in an environment parameter.

For more information, on specifying the database root directory, please refer to Chapter "Configuration Parameters" in the *Versant Database Administration Manual*.

Optionally change permissions on database root directory

If you use the defaults when you install and place the database root directory under your Versant software directory, you will own the database root directory.

If you want to allow others to be able to create databases using this directory, you must change permissions on the database root directory to `777`.

For example:

```
chmod 777 <VERSANT_ROOT>/db
```

This change is only needed when the database root directory is owned by the DBSA (who installed Versant) and the user that is attempting to create the database is not the DBSA.

SETTING ENVIRONMENT FOR AIX

Mandatory Settings

Automated Settings

The script file `envsettings.csh` is located in the directory `<software_root>/bin`.

This script sets the environment variables `PATH`, `LIBPATH` and the `CLASSPATH`. It also sets the environment variables `VERSANT_ROOT`, `VERSANT_DB`, `VERSANT_DBID`, `VERSANT_DBID_NODE` and `VERSANT_VERS`. It sets the `JAVA_COMPILER` to `NONE`.

If you carry out the installation as super-user ‘root’, it will automatically set the Versant Service.

Manual Settings

Add the executables directory to your system path

After installation, you should add the Versant executables directory to your system path if you want to execute Versant utilities without a path specification.

For a standard installation, the Versant executables directory will be named `/bin` and below the software root directory:

```
<software_root>/bin
```

For example:

```
/usr/local/versant/7.0/bin
```

C Shell

If you use C Shell, the search path should be in the `.login` in your home directory. To amend it to include the `VERSANT /bin` directory, use the `set path` command.

```
setenv path <software_root>/bin:$path
```

Bourne Shell

If you use Bourne shell, the search path will be in `.profile` in your home directory. To amend it to include the `VERSANT /bin` directory for a default installation:

```
PATH=<software_root>/bin:${PATH}
export PATH
```

Add the library directory to your system path

On systems which support shared libraries, if Versant has provided shared libraries for linking with your applications, you may need to set an environment variable, which tells your application where to find shared libraries at run-time.

The environment variable for shared libraries has different names on different UNIX systems.

For AIX — `LIBPATH`

For other systems, please see Setting Environment for respective platform.

The commands you need to use will also vary depending upon your operating system and shell.

For example, for AIX and C Shell (substitute the name of your actual software root and compiler directories):

```
setenv LIBPATH <software_root>/lib:$LIBPATH
```

For AIX and Bourne Shell (substitute the name of your actual software root and compiler directories):

```
LIBPATH=<software_root>/lib:$LIBPATH
export LIBPATH
```

Modify the environment variables to run JVI after the installation

The following description assumes Versant ODBMS version to be 7.0.1.4, and process model to be two-process model (2p):

Modify the `CLASSPATH` environment variable.

If you use C shell:

```
setenv CLASSPATH <software_root>/lib/jvi7.0.1-jdk1.4.jar:$CLASSPATH
```

If you are using shell other than the C shell, the syntax for setting environment variables may be different.

Modify the environment variables to run VXML after the installation

Modify the CLASSPATH environment variable to include the following:

If you use C shell:

```
setenv CLASSPATH\
<software_root>/lib/jvi7.0.1-jdk1.4.jar:\
<software_root>/lib/vxml.jar:\
<software_root>/lib/xerces.jar:\
<software_root>/lib/xalan.jar:\
<software_root>/lib/xml-apis.jar:$CLASSPATH
```

If you are using shell other than the C shell, the syntax for setting environment variables may be different.

Modify the environment variables to run VAR after the installation

Modify the CLASSPATH environment variable to include the following:

If you use C shell:

```
setenv CLASSPATH\
<software_root>/lib/jvi7.0.1-jdk1.4.jar:\
<software_root>/lib/var.jar:$CLASSPATH
```

If you are using shell other than the C shell, the syntax for setting environment variables may be different.

Optional Settings

Following should be done, if you have not chosen the installer to modify the environment parameters:

Optionally set the VERSANT_ROOT environment variable

The environment variable VERSANT_ROOT must be set to your software root directory. Setting VERSANT_ROOT after installation is optional, as it will normally be set for you during installation.

C Shell

If you use C Shell, set VERSANT_ROOT in the .login file in your home directory:

```
setenv VERSANT_ROOT <software_root>
```

Bourne Shell

If you use Bourne shell, set VERSANT_ROOT in .profile in your home directory:

```
VERSANT_ROOT=<software_root>
export VERSANT_ROOT
```

Possibly specify the location of the database network system identifier file

If you had installed as super user, during installation you were asked for the location of the osc-dbid database system identifier file. If you specified the location of the osc-dbid file during installation, you do not have to specify it after installation.

To tell Versant where osc-dbid is located, set the VERSANT_DBID_NODE and VERSANT_DBID environment variable by performing the following steps:

1. Optionally refresh your memory.

   Optionally refresh your memory as to the location of osc-dbid that you specified during installation by running the getdbid script:

   ```
   getdbid
   ```

   The getdbid utility might, for example, return:

   ```
   /usr/local/versant/db
   ```

2. Set environment variables.

Set VERSANT_DBID_NODE to the name of the machine containing the osc-dbid file and VERSANT_DBID to the directory containing the osc-dbid file.

C Shell

For C Shell, you can set it directly:

```
setenv VERSANT_DBID_NODE dbid_machine
setenv VERSANT_DBID dbid_directory
```

or you can use the getdbid script as part of the setenv command:

```
setenv VERSANT_DBID 'getdbid'
```

Bourne Shell

For Bourne Shell, to set it directly:

```
VERSANT_DBID_NODE=dbid_machine
export VERSANT_DBID_NODE
VERSANT_DBID=dbid_directory
export VERSANT_DBID
```

or to use the getdbid script:

```
VERSANT_DBID = 'getdbid'
export VERSANT_DBID
```

Possibly update the Yellow Pages

If the machine on which you install Versant is not the Yellow Pages master, you must manually update /etc/services on the master machine by performing the following steps.

1. Add the following line at the end of /etc/services

   ```
   oscssd  5019/tcp  #versant SS daemon
   ```

   If the port 5019 is already used, then specify a different port number.
2. Update YP on the Yellow Pages Master

   ```
   cd /var/yp
   make
   ```

   Since the Yellow Pages Master does not distribute the inetd.conf file, the inetd.conf file has to be updated on all machines that allow remote access to their databases. You can do it manually by performing the following steps:

   1. Add the following line at the end of /etc/inetd.conf

      ```
      oscssd stream tcp nowait root /usr/etc/ss.d in.oscssd
      ```

2. Ask the `inet` daemon to parse the modified file

   On most machines

   If you are running sh:

```
ps xg | grep inet[d] | while read pid junk
do
    kill -1 $pid
done
```

   If you are running csh:

```
set p = `ps xg | grep inetd | awk '{ print $2 }'`
foreach i ($p)
    kill -1 $i
end
```

   On some machines

   If you are running sh:

```
ps -e | grep inet[d] | while read pid junk
do
    kill -1 $pid
done
```

   If you are running csh:

```
set p = `ps -e | grep inetd | awk '{ print $2 }'`
foreach i ($p)
    kill -1 $i
end
```

If you are a diskless client with a read-only root

Some systems mount the root file system of diskless clients as read-only, which will prevent the Versant installer from updating necessary system files.

If the root file system of a diskless client has been mounted as read-only, after installation of Versant on the server you will need to update manually the Versant machine configuration file and your machines `inetd.conf` and `services` files.

The name of your machine configuration file will depend upon your Release number. For example, for Release 7.0.1.3, the machine configuration file is `/etc/.osc070001`.

The `inetd.conf` file lists servers, and the `services` file lists services.

In a standard installation, the paths to these files are:

```
/export/<software_root>/client/etc/.osc070001
/export/<software_root>/client/etc/inetd.conf
/export/<software_root>/client/etc/services
```

.where `client` is the name of the client machine.

The necessary steps are:

1. Copy the binaries

   Copy the Versant binaries from the Versant server file system to the client file system.

   `cp -p /root/bin/ss.d /export/client/etc`

2. Update machine configuration file

   If you want to share program files and databases, the home directory for system software files must be readable and the home directory for databases must be readable and writable to other users on the network. Also, the path and name of the file containing paths to software and databases must be entered in a system path file.

   The name of this file will depend upon your release number. For example, for Release 7.0.1.3, the name of the system file is:

   `/etc/.osc070001`

   The `/etc/path_file` file should have a line to specify each of the following locations. Each line first states the configuration variable name and then supplies a value.

   `VERSANT_ROOT software_root`
   Location of software root directory.

   `VERSANT_DB database_root`
   Location of database root directory.

   `VERSANT_DBID_NODE dbid_machine`
   Machine with `osc-dbid` file.

   `VERSANT_DBID dbid_path`
   The directory containing the `osc-dbid` file.

For more explanation, of the above configuration variables, please refer to Chapter "Configuration Parameters" in the *Versant Database Administration Manual*.

To assign ownership of the `/etc/path_file` file:

```
chown group_name /etc/path_file
```

For example:

```
chown dbsa.dbgroup /etc/.osc070001
```

3. Update `inetd.conf` and `services`.

You must next update the contents of the Yellow Pages files `/etc/inetd.conf` and `/etc/services`. The locations of these files for a standard installation, where `client` is the name of the client machine are:

`/export/<software_root>/client/etc/inetd.conf`

`/export/<software_root>/client/etc/services`

To update these files:

- Add the following line at the end of `/etc/services`:

  ```
  oscssd  5019/tcp  #versant SS daemon
  ```

  If the port `5019` is already used, then specify a different port number.

- Update `YP` on the Yellow Pages Master:

  ```
  cd /var/yp
  make
  ```

Since the Yellow Pages Master does not distribute the `inetd.conf` file, the `inetd.conf` file has to be updated on all machines that allow remote access to their databases. To do this, perform the following steps:

- Add the following line at the end of `/etc/inetd.conf`:

  ```
  oscssd stream tcp nowait root /usr/etc/ss.d in.oscssd
  ```

- As root on the client machine, ask the `inet daemon` to parse the modified file.

  On most machines

  If you are running sh:

  ```
  ps xg | grep inet[d] | while read pid junk
  do
      kill -1 $pid
  done
  ```

  If you are running csh:

  ```
  set p = `ps xg | grep inetd | awk '{ print $1 }'`
  foreach i ($p)
  ```

```
    kill -1 $i
end
```

On some machines

If you are running sh:

```
ps -e | grep inet[d] | while read pid junk
do
    kill -1 $pid
done
```

If you are running csh:

```
set p = `ps -e | grep inetd | awk '{ print $1 }'`
foreach i ($p)
    kill -1 $i
end
```

If you later want to change the database root directory

If you use the defaults when you install, the database root directory will be placed under your Versant root directory.

After installation and even after databases have been created, you can move the database root directory to a new location as long as you specify the new location in an environment parameter.

For more information, on specifying the database root directory, please refer to Chapter "Configuration Parameters" in the *Versant Database Administration Manual*.

Optionally change permissions on database root directory

If you use the defaults when you install and place the database root directory under your Versant root directory, you will own the database root directory.

If you want to allow others to be able to create databases using this directory, you must change permissions on the database root directory to 777.

For example:

```
chmod 777 VERSANT_ROOT/db
```

This change is only needed when the database root directory is owned by the DBSA (who installed Versant) and the user that is attempting to create the database is not the DBSA.

Index