HP TransactionVision Software

Software Version: 7.50

Analyzer Installation and Configuration Guide

Software Release Date: May 2008

Legal Notices

Warranty

The only warranties for HP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HP shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HP required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notices

Trademark Notices

TransactionVision® is a registered trademark of the Hewlett-Packard Company.

Java™ is a U.S. trademark of Sun Microsystems, Inc.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

Oracle® is a registered trademark of Oracle Corporation and/or its affiliates.

UNIX® is a registered trademark of The OpenGroup.

Documentation Updates

This guide's title page contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for recent updates, or to verify that you are using the most recent edition of a document, go to:

http://h20230.www2.hp.com/selfsolve/manuals

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HP sales representative for details.

Support

You can visit the HP Software Support web site at:

http://www.hp.com/go/hpsoftwaresupport

HP Software Support Online provides an efficient way to access interactive technical support tools. As a valued support customer, you can benefit by using the support site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Download software patches
- Manage support contracts
- Look up HP support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract.

To find more information about access levels, go to:

http://h20230.www2.hp.com/new_access_levels.jsp

For more information about HP Passport, go to:

http://h20229.www2.hp.com/passport-registration.html

Contents

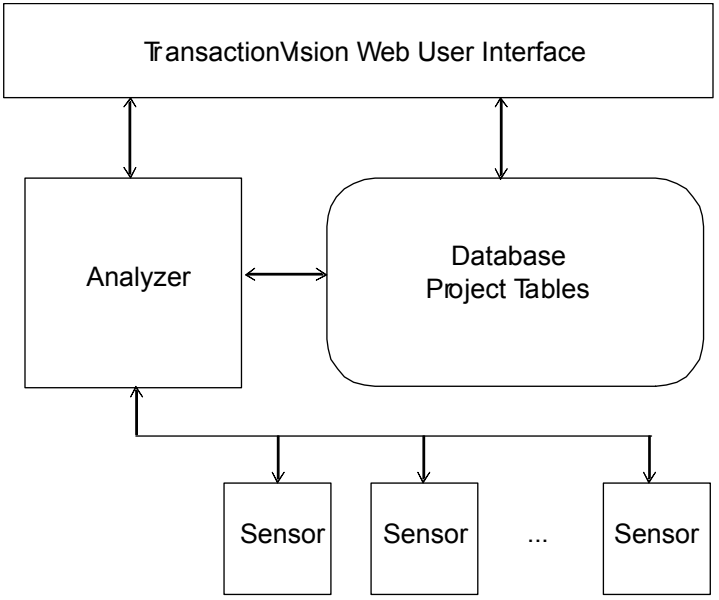
3 TransactionVision Analyzer Overview

The TransactionVision Analyzer is a service that communicates with TransactionVision Sensors via WebSphere MQ or JMS services. It generates and delivers configuration messages to Sensors by placing them on a designated configuration queue. Configuration messages specify Sensor configuration information such as the name of the event queue where the Sensor should place event messages and data collection filter definitions for the project.

The Analyzer also retrieves events placed on an event queue by Sensors and processes them for analysis and display by the web user interface. It performs the unmarshalling, correlation, analysis, and data management functions.

Each TransactionVision project is assigned a single host running the Analyzer. Projects enable you to easily group and manipulate communication links, data collection filters, database schemas, and Analyzers as one entity. When you start a project, the Analyzer on the host assigned to the project is started automatically. You may also start the Analyzer on a host from the Analyzers page in the web user interface, in which case the Analyzer starts processing events on all active projects it is assigned to.

The following diagram shows the relationship of the TransactionVision Analyzer to other TransactionVision components:



Installation Overview

The TransactionVision installation consists of the following three TransactionVision packages:

- The Analyzer binaries and configuration files. Install this package on all hosts where you want the Analyzer service to run. This guide provides instructions for installing the Analyzer.
- The TransactionVision Web User Interface, which consists of WebSphere or WebLogic related files such as the .ear file, JSPs, servlets, etc. Install this package on the hosts that users and administrators will access via web browsers to use and manage TransactionVision. For instructions on installing and configuring the Web User Interface, see the *TransactionVision Web Application Installation and Configuration Guide*.

- TransactionVision Sensors. Install these packages on all hosts running applications that you wish to collect information about. For instructions on installing and configuring Sensors, see the *TransactionVision Sensor Installation and Configuration Guide*.

Installation steps vary for the different TransactionVision components on different platforms.

Before installing Analyzers, make sure the systems you are installing on meet the software requirements for TransactionVision. Requirements vary for each package (Sensors, Analyzer, and Web User Interface). See the *TransactionVision Release Notes* for detailed software requirements.

Upgrading from Previous TransactionVision Releases

The Java Servlet, JMS and EJB Sensors from TransactionVision 5.00 can be used with the TransactionVision 7.50 Analyzer. TransactionVision 7.50 Sensors may not be used with older versions of the TransactionVision Analyzer.

Additional TransactionVision Resources

Documentation

The following documents are provided for TransactionVision:

- The *TransactionVision Planning Guide* provides information to help you plan the TransactionVision implementation in your environment.
- The *TransactionVisionWeb Application Installation and Configuration Guide* provides instructions for installing and configuring the TransactionVision web user interface. This file is also available from the TransactionVision Help menu.
- The *TransactionVision Sensor Installation and Configuration Guide* provides instructions for installing and configuring TransactionVision Sensors. This file is also available from the TransactionVision Help menu.

- The *TransactionVision Administration Guide* provides instructions managing user accounts and communication links, configuring projects and data collection filters, and managing services and schemas. This file is also available from the TransactionVision Help menu.
- The *TransactionVision User Guide* provides instructions using TransactionVision analysis views. This file is also available from the TransactionVision Help menu.
- The *TransactionVision Advanced Customization Guide* provides information for creating custom beans and reports for use with TransactionVision.
- The *TransactionVision Security Guide* provides an overview of the security features and setup procedures of TransactionVision. These features and procedures ensure that data collected by TransactionVision is secure and accessible to the appropriate people.

4 Installing the Analyzer on UNIX Platforms

To install the TransactionVision Analyzer on UNIX platforms, perform the following steps.

1. Change to the directory location of the TransactionVision installation files (either a CD device or download directory). NOTE: On Solaris and HP-UX, you must copy the installation files from the CD device to a temporary directory on your host's hard drive.
2. Log in as superuser:

   ```
   su
   ```

3. Enter the following command to begin the installation procedure:

   ```
   ./install.sh
   ```

   The following menu is displayed:

   ```
   This script will install/uninstall different
   TransactionVision components.

   Unziping/Untaring common package files ...
   Unziping/Untaring Analyzer package files ...
   Unziping/Untaring web package files ...
   Unziping/Untaring Application Server sensor package files ...
   Unziping/Untaring JMS sensor package files ...
   Unziping/Untaring WebSphere MQ sensor package files ...
   ```

The six "Unzipping…" lines above are only for Solaris and HP-UX installations.

```
The following TransactionVision packages are available for
installation:

1. TransactionVision Analyzer
2. TransactionVision Web
3. TransactionVision Application Server Sensor
4. TransactionVision JMS Sensor
```

```
5. TransactionVision WebSphere MQ Sensor

99. All of above
q. Quit install

Please specify your choices (separated by,) by number/letter:
```

Note that actual options and numbers depend on the installation files available on your computer: Also note that the Application Server Sensor package installs both the Servlet and EJB Sensors.

If this is the first time installing TransactionVision components on this computer, continue with Initial Installation. If an earlier version of TransactionVision is installed on this computer, continue with Upgrade Installation on page 17.

Initial Installation

1. To install only a single component, type the number associated with the TransactionVision component package and press **Return**. In this example, type **1** and press **Return** to install only the Analyzer.

   To install multiple, but not all components, type the numbers associated with the components you wish to install, separated by commas, and press **Return**.

   To install all available components, type **99** and press **Return**.

   The installation script installs the specified package(s), then displays the menu again.

2. To quit the installation procedure, type **q** and press **Return**. To install additional components, see the installation instructions for those components.

Upgrade Installation

1 To install only a single component, type the number associated with the TransactionVision component package and press **Return**. In this example, type **2** and press **Return** to install only the Web Application.

To install multiple, but not all components, type the numbers associated with the components you wish to install, separated by commas, and press **Return**.

To install all available components, type **99** and press **Return**. See the installation instructions for the other components you wish to install for information about installing those components.

If the installation script determines that a previous version of TransactionVision is installed, it displays the following message:

```
There is an earlier version of TransactionVision installed on
the system.
The earlier version has to be uninstalled before installing
the current package(s).
Continue with the uninstallation? (Y/N) [N]:
```

2 Type **Y** and press **Return** to uninstall the previous version. If you type **N** and press **Return**, the installation process ends without uninstalling the previous version or installing the new version.

Before uninstalling the previous version, TransactionVision provides the option of migrating configuration to the new installation:

```
Installation has detected a previous installation of
TransactionVision.
Do you want to migrate existing TransactionVision
configuration files? (y/n) [y] :
```

3 Type **Y** and press **Return** to migrate configuration to the new version. The installation script automatically creates a backup copy of existing configuration files for the migration. It then uninstalls the previous version of TransactionVision, installs the new Sensor package (and additional components if specified), and displays the installation menu. Continue to step 4.

If you type **N** and press **Return**, the installation script provides the option of making a backup copy of configuration files for future reference:

```
Although migration will not be performed at this time, you may
optionally back up configuration files from your previous
installation for reference purposes. Note that answering N
will overwrite these files, causing any existing setup
information to be lost. Do you wish to back up configuration
files from the previous installation? (y/n) [y] :
```

Type **Y** and press **Return** to create a backup copy of your configuration files. The installation script prompts you to specify a backup location:

```
Enter the directory to which existing TransactionVision
configuration files should be backed up [/opt/TVision/
migrate_tv750_date_time]:
```

Press **Return** to use the default location, or enter the desired backup directory location. The installation utility performs the following tasks:

- Copies the current configuration files to the specified directory.
- Uninstalls the previous version of TransactionVision.
- Installs the new web package (and additional components, if specified).
- Displays the installation menu again.

4 To quit the installation procedure, type **q** and press **Return**.

5 After performing the upgrade installation it is necessary to migrate the existing database schemas with the `MigrateDB.sh` script. See Appendix A, Utilities Reference for information about this utility. If you need to migrate large amounts of data please also see Appendix C, Database Migration, for more detailed information about the migration process.

Installation Files

The following table shows the installation file names for the TransactionVision web package for each distributed platform.

Platform	Files
AIX	tvision_common_750_aix_power.bff tvision_analyzer_750_aix_power.bff
Linux	tvision_common_750_linux_x86.rpm tvision_analyzer_750_linux_x86.rpm
Solaris	tvision_common_750_solaris_sparc_pkg.gz tvision_analyzer_750_solaris_sparc_pkg.gz

Uninstalling the Analyzer

To uninstall TransactionVision components, perform the following steps:

1. Log in as superuser:

   ```
   su
   ```

2. Enter the following command:

   ```
   install.sh -u
   ```

   The following menu is displayed (note that actual options depend on the TransactionVision packages installed on your computer):

   ```
   The following TransactionVision packages are installed on the
   system:

   1. TransactionVision Web
   2. TransactionVision Analyzer
   3. TransactionVision Application Server Sensor
   4. TransactionVision JMS Sensor
   5. TransactionVision WebSphere MQ Sensor

   99. All of above
   ```

```
q. Quit uninstall

Please specify your choices (separated by,) by number/letter:
```

3 Type the number associated with the TransactionVision package you wish to uninstall (2 for the Analyzer in this example) and press **Return**.

To uninstall all TransactionVision components, type **99** and press **Return**.

The installation script uninstalls the specified package, then displays the menu again.

If you uninstall the servlet Sensor, it will be first turned off in the WebSphere Application Server the Sensor is monitoring. The instrumented jar files under $WAS_HOME/classes will be removed.

4 To quit the uninstall, type **q** and press **Return**. If the common package is the only TransactionVision package still installed, it will be uninstalled automatically.

After uninstalling the TransactionVision web user interface, you must clean up its temporary cache and distribution directory. WebSphere does not do this automatically, and any old files could cause a new installation to work incorrectly.

5 Installing the Analyzer on Windows

The TransactionVision Analyzer and web user interface are installed as a single package on Windows. Note that you must be logged into the target system either as Administrator or as a user with Administrator privileges.

For migration to work properly on Windows when you are upgrading from a previous release, either the JAVA_HOME environment variable must be set or Java must be present in your path before you begin the installation. Migration requires Java; if neither of these is set, the installation may fail to migrate configuration files and you may need to run TVisionSetupInfo manually.

To install this package, perform the following steps:

1. Close all Windows programs currently running on your computer.
2. In the Windows Explorer, double-click **tvision_sensor_750_win.exe**. The InstallShield Welcome screen appears.
3. Click **Next>** and wait until the TransactionVision Setup Welcome screen appears.

If this is the first time installing the TransactionVision Web Application on this computer, continue with Initial Installation on page 22. If an earlier version of the TransactionVision Web Application is installed on this computer, continue with Upgrade Installation on page 22.

If the InstallShield Save Files screen appears, click **Next>** to use the default folder for extracting installation files (for example, C:\TEMP\Hewlett-Packard\TransactionVision), or click **Change** to select the desired folder and click **Next>** to continue.

Initial Installation

For an initial installation, the Setup Welcome screen is displayed.

1. On the Setup Welcome screen, click **Next>** to display the TransactionVision license agreement.
2. Click **Yes** to accept the license agreement. The User Information screen appears.
3. Enter your name and company name, then click **Next>**. The Destination Location screen appears.
4. To use the default installation folder (C:\Program Files\Hewlett-Packard\TransactionVision), click **Next>**. To choose a different installation folder, click **Browse...**, select the desired installation folder, then click **Next>**. The Setup Type screen appears.
5. To install both the Analyzer and web user interface, select Complete and click **Next>**. To install only the Analyzer, select Custom, click **Next>**, deselect the web user interface, and click **Next>**.

   The selected packages are installed in the specified location. The Setup Complete page appears.
6. Click **Finish** to complete the installation.
7. If you are running WebLogic as a Windows service, continue with the steps in Modifying the Installation on page 23.

Upgrade Installation

For an upgrade installation, the installation wizard displays the setup maintenance menu.

1. If you wish to install TransactionVision with different settings from the previous installation, select Remove and click **Next>** to uninstall the previous installation, then begin the installation procedure again. If you are upgrading from a previous release, select Reinstall and click **Next>** to install TransactionVision using the settings from the previous installation. The Configuration File Migration dialog appears:

2 To maintain configuration information from the previous installation, click **Yes**. The installation wizard makes a backup copy of existing configuration files, installs the new version of TransactionVision, and opens an MS-DOS window to migrate existing configuration files to the new version. When complete, the Setup Complete screen appears.

   To overwrite existing configuration files, click **No**. The installation wizard displays a message box asking whether you want to make a backup copy of existing configuration files before continuing the installation.

   Click **Yes** to create a backup copy or No to continue the installation without backing up configuration files. The installation wizard then installs the new version of TransactionVision, overwriting existing configuration files, and displays the Setup Complete screen.

3 Click **Finish** to complete the installation.

4 After performing the upgrade installation it is required to migrate the existing database schemas with the `MigrateDB.bat` script. See Appendix A, Utilities Reference for information about this utility. If you need to migrate large amounts of data see Appendix C, Database Migration for more detailed information about the migration process.

Modifying the Installation

After you install TransactionVision Sensors on a host, you may wish to modify your installation. For example, suppose you initially install the Analyzer on a host, then later decide to install the Servlet and JMS Sensors. To modify your installation perform the following steps:

1 Close all Windows programs currently running on your computer.

2 In the Windows Explorer, double-click **tvision_sensor_750_win.exe**. The InstallShield Welcome screen appears.

3 Click **Next>** and wait until the TransactionVision Setup Welcome screen appears.

4 To install an additional component or remove an installed component, select Modify. To reinstall all TransactionVision components installed by the previous setup, select Repair. To uninstall all components, select Remove.

5 Click **Next>** and select the additional components you wish to install. The Configuration File Migration dialog appears:

6 Click **No**, because migration is not required for installing additional components. The installation wizard displays a message box asking whether you want to make a backup copy of existing configuration files before continuing the installation. Click **No** again to continue the installation without backing up configuration files. The installation wizard then installs the selected TransactionVision components and displays the Setup Complete screen.

7 Click **Finish** to complete the installation. If you wish to display the migration log file created when migrating configuration and project database files to the new version, check View Migration Log File before clicking Finish.

Uninstalling the Analyzer

To uninstall TransactionVision components, perform the following steps:

1 From the Start menu, choose **Settings** > **Control Panel**.

2 Double-click **Add/Remove Programs**.

3 Select the HP TransactionVision package you wish to uninstall and click **Change/Remove**. The maintenance menu screen appears.

4 Select **Remove** and click **Next>** to remove TransactionVision components.

5 Click **OK** to confirm that you wish to uninstall the specified package. The specified package is uninstalled. The following types of files are not deleted:

   — Any files added after the installation

   — Any shared files associated with packages that are still installed

   If shared files do not appear to be associated with any installed packages (for example, if all other TransactionVision packages have been uninstalled), the Shared File Detected screen appears.

   • To leave all shared files installed, check Don't display this message again and click **No**.

- To leave the current file, but display this message for any other shared files, click **No**.
- To delete the shared file, click **Yes**.

If you uninstall the servlet Sensor, it will be first turned off in the WebSphere Application Server the Sensor is monitoring. The instrumented classes under $WAS_HOME/classes will be removed along with its parent directories if they are empty.

6 The Uninstallation Complete screen appears. Click **Finish** to complete the uninstallation procedure.

After uninstalling the TransactionVision web user interface, you must clean up its temporary cache and distribution directory. WebSphere does not do this automatically, and any old files could cause a new installation to work incorrectly.

6 Configuring Databases

Before configuring the TransactionVision Analyzer, it is important to ensure that your DBMS environment is configured properly to work with TransactionVision. This configuration differs for DB2, Oracle, and SQL Server databases.

It is first necessary to create your DB2, Oracle, or SQL Server database before running the TVisionSetup script. The TVisionSetup script does not create a database and relies on a connection to an existing database. Database variables or tuning parameters must then be adjusted to ensure that the database will operate at an acceptable level of performance for your application. Several tools are provided with TransactionVision to aid in the identification and resolution of database performance bottlenecks.

Identifying Database Connection Names

When you run the TVisionSetupInfo utility to configure the TransactionVision Analyzer and Web Application, you will be prompted for the database connection name. This name depends on which database provider you are using and, for Oracle, which database client type.

Oracle OCI Client

If you are using an Oracle oci client with TransactionVision, you will be prompted to enter the database connection name and database name.

The connection name is the Oracle Net Service Name. Use the Oracle Net Manager to find all net service names. Choose the appropriate connection name from the Service Naming folder (for example, TVQA_HEINEKIN).

The database name is the Oracle System Identifier (SID) (for example, tvqa). To identify the SID, click the service name. In the following example, shydb is the database name for the shydb_heineken service name.

Oracle Thin Client

If you are using the Oracle thin driver with TransactionVision, you will be prompted to enter only the database connection name. The connection name is the Oracle SID (for example, tvqa).

IBM DB2

DB2 Universal Driver Type 4

By default, the TransactionVision setup scripts use the DB2 Universal Type 4 Driver. You will be prompted to enter the database name.

DB2 Universal Driver Type 2

If you are using the DB2 Universal Driver Type 2, you will need to enter the database connection name and the database name.

The connection name is the DB2 database alias name. To find the alias name, use the following command:

```
db2 list database directory
```

For local databases, the alias name is typically the same as the database name, unless a remote loopback is defined.

For remote databases, the alias name will typically be different from the database name.

In the following example, the database connection name is TVDB and the database name is TVISION:

```
C:\Program Files\IBM\WebSphere MQ\Java\bin>db2 list database
directory
 System Database Directory

 Number of entries in the directory = 13
```

```
 Database 1 entry:

 Database alias                       = TVDB
 Database name                        = TVISION
 Node name                            = TCP7768C
 Database release level               = a.00
 Comment                              =
 Directory entry type                 = Remote
 Catalog database partition number    = -1

Database 2 entry:

 Database alias                       = BACARDI
 Database name                        = TVISION
 Node name                            = NDE682FE
 Database release level               = a.00
 Comment                              =
 Directory entry type                 = Remote
 Catalog database partition number    = -1
```

SQL Server

If you are using SQL Server, you will be prompted to enter the database name you specified when you created the database

Setting DB2 Variables

Before using TransactionVision, set the values for the following DB2 variables to the values shown in the following table:

Variable	Value	Description
APP CTL HEAP SZ	1024	Maximum application control heap size. This value indicates the number of 4KB blocks.
MAXAPPLS	150	Maximum number of active applications
APPLHEAPSZ	1024	Default application heap size. this value indicates the number of 4KB blocks.
LOCKLIST	Estimate based on system	Amount of storage (in 4KB units) allocated to the lock list.

In addition, you should increase the size of the bufferpool.

APP CTL HEAP SZ, MAXAPPLS, and APPLHEAPSZ

Use the following commands to set these values. Note that the last three commands will drop all active database connections and then stop and start the DB2 server. Be sure to run these steps at an appropriate time when other database users will not be affected.

```
db2 connect to tvision
db2 get db cfg for tvision
db2 update db cfg for tvision using APP_CTL_HEAP_SZ 1024
db2 update db cfg for tvision using MAXAPPLS 150
db2 update db cfg for tvision using APPLHEAPSZ 1024
db2 force application all
db2stop
db2start
```

In the environment in which WebSphere is started on UNIX systems, the DB2INSTANCE environment variable must also be set to the DB2 instance being used by TransactionVision.

LOCKLIST

To determine the number of pages required for your lock list, perform the following steps:

1. Calculate a lower bound for the size of your lock list:

   ```
   (512 * 36 * mapappls) / 4096
   ```

   where 512 is an estimate of the average number of locks per application and 36 is the number of bytes required for each lock against an object that has an existing lock.

2. Calculate an upper bound for the size of your lock list:

   ```
   (512 * 72 * maxappls) / 4096
   ```

   where 72 is the number of bytes required for the first lock against an object.

3. Estimate the amount of concurrency you will have against your data. Based on you expectations, choose an initial value for LOCKLIST that falls between the upper and lower bounds that you have calculated. Use the following command to set the value:

   ```
   db2 update db cfg for database_name using LOCKLIST n
   ```

   where n is the value for LOCKLIST.

4. Using the database system monitor, tune the value of this parameter. For more information, see Chapter 32, “Configuring DB2 Capacity Management,” in the *DB2 V7 Administration Guide*.

DB2_RR_TO_RS

Multiple threads performing concurrent database operations along with a DB2 behavior called Next Row Locking may lead to database deadlocks. If you plan to use multiple event collection threads, set the DB2 variable DB2_RR_TO_RS to ON to avoid deadlocks:

```
db2set DB2_RR_TO_RS=ON
```

This setting is only effective after a DB2 restart. It affects all DB2 applications and cannot be used if other non-TransactionVision applications that require “Repeatable Read” semantics (Transaction Isolation Level RR) are using the same DB2 instance. If other applications using the same

instance require “Repeatable Read” semantics, you must either create a separate DB2 instance for TransactionVision or set the number of event collection threads to 1. For information on setting the number of event collection threads, see the “Managing Communication Links” chapter in the *TransactionVision Administration Guide*.

Bufferpool

The size of the default DB2 bufferpool is only 250 pages. Increase it to 1024 (or more if sufficient memory for DB2 is available) to improve performance. The following command will change the size for the default bufferpool from 250 pages to 1024 pages:

```
db2 ALTER BUFFERPOOL IBMDEFAULTBP SIZE 1024
```

Setting Oracle Variables

If it is expected that TransactionVision will be used in a relatively simple environment where minimal database connections are required, no special configuration is required for the Oracle DBMS. However, environments where a large number of users will be simultaneously accessing the web user interface, several Analyzers will be in use, or where the Analyzer will incorporate higher than the default number of threads, it will be necessary to increase the Open Cursors database parameter. Related error messages may be expected to show up in the TransactionVision UI or Analyzer logs if this limit is exceeded. To increase the number of Open Cursors, execute the following command:

```
alter system set open_cursors = 600
```

This change may be made dynamically while the Oracle server is running. It is not necessary to restart the RDBMS.

Connecting to an Oracle Database

To connect to an Oracle database, perform the following steps:

1. If you are planning to use the Oracle oci driver to make database connections, make sure the target database is defined in your <ORACLE_HOME>/network/admin/tnsnames.ora file. For example:

```
TESTDB_SIMPLEX6.BRISTOL.COM =
  (DESCRIPTION =
    (ADDRESS_LIST =
      (ADDRESS = (PROTOCOL = TCP)(HOST = simplex6)(PORT =
1521))
    )
    (CONNECT_DATA =
      (SID = testdb)
      (SERVER = DEDICATED)
    )
  )
```

   Various Oracle tools (such as Net Manager or Net Configuration Assistant) can create such an entry for you, or you can add it manually.

2. Run the TVisionSetupInfo script (see Chapter 7, Configuring the Analyzer for instructions) and enter Oracle when you are prompted to enter the type of database to use. For thin clients, enter the database listener port and SID for database connection name. For the Oracle oci driver, please follow the steps in Chapter 7, Configuring the Analyzer.

   For a thin client connection, the Database.properties file will contain entries similar to the following example:

```
jdbc_driver=oracle.jdbc.driver.OracleDriver
database_connection_name=testdb
database_name=testdb
database_host=simplex6
database_port=1521
oracle_client_type=thin
```

For more information about these entries, see TransactionVision Database Properties on page 39 and Configuring Database Properties on page 48.

DBMS Performance Tuning

Because TransactionVision uses the DBMS extensively for its data collecting and analyzing process, the performance of the DBMS is vital to the overall performance of TransactionVision. Inserting records and updating records represent the majority of the database operations associated with TransactionVision; therefore the speed of the physical disks/I/O interface has a significant impact on the performance.

Optimizing I/O Throughput

The key to DBMS performance is to overcome the operation bottleneck – I/O throughput limit. Usually this limit is imposed by the physical disk and the I/O interface.

Prior to deployment, it is imperative to make sure the actual DBMS system has a good I/O and disk subsystem attached and that the subsystem has been tuned for writing. This includes checking that the disk is RAID configured for performance, write-cache is enabled for the disks, and the I/O interface is fast (preferably fiber-optic interface).

To achieve high throughput of I/O, some forms of parallel processing should be used:

- Use separate DBMS instances for separate projects - if the data can be partitioned then separate DBMS instances on different hosts can be employed to achieve parallelism. This setup requires setting up multiple instances of TransactionVision.
- Use RAID disks for table space containers. RAID disks provide parallel I/O at hardware level.
- Separate table space containers and log file directories. DB2 log files, Oracle Rollback Segments, and SQL Server Transaction logs hold uncommitted database operations and usually are highly utilized during database insert/update. For this reason they should have their own containers on physically separated disks, and preferably on RAID disks.
- Stripe table spaces. If parallel I/O is not available at the hardware level, DBMS can be set up to span a tablespace across multiple disks, which introduces parallelism at the DBMS space management level.

There are many other database parameters that may impact the performance of TransactionVision. For DB2 in particular, those parameters previously mentioned in Setting DB2 Variables on page 30 must be examined one-by-one to ensure they are optimized.

There is also some benefit when the tablespaces used by TransactionVision are managed by the database directly (DMS or DMS RAW tablespaces).

Testing DBMS and Diagnosing Performance Bottlenecks

HP provides independent tools, DB2Test, OracleTest, and SQLServerTest that can be used to test the performance of DBMS relevant to TransactionVision (especially the record insert rate). The tool is written in Java and should be run where the TransactionVision Analyzer will be installed. For more information about these tools, see Appendix A, Utilities Reference.

The tools simulate the database update operation generated by TransactionVision. Run the test multiple times to get a complete picture of the DBMS performance. Please note that the result of the test does not directly correlate with TransactionVision processing rate; rather, it is an indicator of how well does the DBMS performance for the given configuration.

Make sure each test lasts at least a few minutes to minimize the overhead of any initialization process. The test should be run against the same database and table sets with which the TransactionVision Analyzer will be run.

- Run the insert test with one thread and with record size of 1KB - this will gauge the raw event insert performance.
- Run the insert test with multiple threads and with a record size of 1KB. This will test whether the insert can benefit from multiple threads. Usually the thread count is set to two times the number of CPUs.
- Run the insert test with one thread and with record size of (11KB + average message size) - this will gauge the analyzed event insert performance.
- Run the insert test with multiple threads and with record size of (11KB + average message size). This will test if the insert can benefit from multiple threads. Usually the thread count is set to two to four times the number of CPUs.

- If the Analyzer host and the DBMS host are different, the above tests should be run on the Analyzer host. However at least one test should be run on the DBMS host to see if there is any communication/DB client configuration related issues.

The rate of insert should be on par with the result achieved from similar systems tested by HP. During the test the following parameters of the DBMS system should be monitored:

- Disk I/O usage for all involved physical disks (tablespaces and log files), especially I/O busy percentage.
- CPU usage, including wait time percentage.

If a disk hits 80-90% utilization or the I/O wait time is extraordinary long, that's an indication of a disk I/O bottleneck. If no obvious bottleneck is seen, then a DBMS configuration or O/S configuration issue may exist. Check database, DBMS and kernel parameters with HP for any configuration issues.

Another useful tool for analyzing DBMS performance is the DB2 performance snapshot monitor.

Updating Statistics with RUNSTATS

The database RUNSTATS command updates statistics about the physical characteristics of a table and the associated indexes. These characteristics include number of records, number of pages, and average record length. The optimizer uses these statistics when determining access paths to the data.

This command is called when a table has had many updates, such as when data is continuously collected into DB2 by the TransactionVision Analyzer. It could result in large performance gains in queries made by TransactionVision views and reports, as well as queries made internally by the TransactionVision Analyzer to correlate events.

TransactionVision provides the following scripts for running the RUNSTATS command on TransactionVision tables:

- `DB2RunStats.[sh|bat]` for a DB2 environment
- `OracleRunStats.[sh|bat]` for an Oracle environment

Oracle 10g has a built-in job scheduler that automatically collects statistics. You do not need to run this script for version 10g. The same applies to SQL Server.

These scripts can be set up to run as scheduled batch jobs using either the UNIX **cron** facility or the Windows scheduler.

While these scripts are running, TransactionVision Analyzer processing slows down.

These scripts typically should be run daily, though the frequency of execution could be higher for higher message rates.

Run these scripts from a user account that has privileges to perform database operations. The usage is as follows:

DB2RunStats *database_name schema_name* [-v7]
OracleRunStats *username passwd database_name schema_name*

The database_name option represents the name of the database to connect to, and schema_name is the name of the schema that the project reads and writes data to. For DB2RunStats, use the -v7 flag for DB2 7.x; the default operation is for DB2 8.1.

These scripts need to be customized based on your system to invoke the right environment initialization script like .profile or .bashrc, and to set the correct DB2 installation location and the correct TVISION_HOME location. For OracleRunStats, the file OracleRunStats.sql must be in the same directory where the script is located. Also, because this script uses Oracle SQL Plus, which uses Oracle OCI, there must be an entry in your Oracle's tnsnames.ora file for your database connection (see Connecting to an Oracle Database on page 33 for details).

Example cron Job

For a UNIX environment, the following example is a typical crontab entry for executing the DB2RunStats script every night:

```
5 0 * * * <TVISION_HOME>/bin/DB2RunStats.sh database-name
schema-name > <TVISION_HOME/logs/runstats.log 2>&1
```

In this example, <TVISION_HOME> is the TransactionVision installation directory, database-name is the database name to connect to, and schema-name is the schema on the tables to run RUNSTATS on.

The above entry executes the script every night 5 minutes after midnight. A crontab entry needs to be added for every project schema being used by TransactionVision. Use crontab -e to create a cron job entry. For details, refer to the man pages of cron and crontab.

Example Scheduled Task

For a Windows environment, run these scripts through Windows scheduler at regular intervals. Add a scheduler entry for every project schema being used by TransactionVision. The Windows scheduler can be accessed through the Windows Control Panel. Make sure this script runs in the DB2 command window.

The following is an example command for the Windows scheduler to run the DB2RunStats script:

```
"<DB2-install-dir>\bin\db2cmd.exe" "<TVISION_HOME>\bin\
DB2RunStats.bat" database-name schema-name > "<TVISION_
HOME>\logs\runstats.log"
```

In this example, <TVISION_HOME> is the TransactionVision installation directory, database-name is the database name to connect to, and schema-name is the schema on the tables to run RUNSTATS on.

DBMS Disk Space Requirements

The other factor that needs to be finalized is the amount of disk storage space required for TransactionVision events. This is determined by the average size of the messages, the rate of events collected by TransactionVision, and the duration of which TransactionVision event data needs to be kept in the database.

The formula for calculating disk storage usage (for DB2) is:

(Average message size + 11K Byte) x Event Rate x Event Retention Time

For example, if the average message size is 2K Bytes, the transaction rate is 5 transactions/second (for 8 hours/day and all weekdays, this translates into 720 thousand transactions/week). If there are two collecting points for each transaction. and if TransactionVision data is required to be stored for duration of four weeks before the data is either archived or deleted, then the total required storage is about 75GB ((2 + 11)K Bytes x 720,000 x 2 x 4 = 75G Bytes).

Using the DB2 compact LOB feature and/or the Analyzer compression function may reduce this number by 30% - 70%.

Configuring Databases for Unicode Data

TransactionVision can display Unicode data in views and reports. However, you must create the TransactionVision database with the required code/character set, and set the appropriate database property within TransactionVision.

Database Code/Character Set

When you create the TransactionVision database, you must specify the properties shown in the following table:

Database Provider	**Required Settings**
DB2	The TransactionVision database must be created with Code Set UTF-8.
Oracle	• The TransactionVision database must be created with the character set AL32UTF8 or UTF8. • The NLS_LENGTH_SEMANTICS initialization parameter must be set to BYTE.

TransactionVision Database Properties

If the unicode_db property is set, all character-based XDM columns with the attribute unicode=true set will be generated with double the byte size to allow the specified number of characters to be stored in the database. For character sets requiring more than two bytes per character, set unicode_bytes_per_character=3 in the Database.properties file. For more information on modifying the Database.properties files, see Appendix B, Configuration Files.

7 Configuring the Analyzer

Configuration Overview

Once TransactionVision is installed, you must create an IBM DB2, Oracle, or SQL Server database for TransactionVision to save event data. Refer to Chapter 6, Configuring Databases for the required database parameters. The TransactionVision setup utilities do not create the database.

After the database is created, the following setup tasks must be completed before you can use TransactionVision Analyzer:

- Create the TransactionVision database schema.
- Start the Analyzer service.

TransactionVision provides two utilities to guide you through the setup process:

- The TVisionSetupInfo utility enables you to enter information it needs for the setup process and sets required environment variables.
- The TVisionSetup utility uses information you provide to TVisionSetupInfo to perform setup tasks.

This chapter provides instructions for configuring the Analyzer only. If you are installing both the Analyzer and Web Application on a single system, refer to the *TransactionVision Web Application Installation and Configuration Guide* for instructions on installing the Web Application and then configuring both components.

Using TVisionSetupInfo

If only the Analyzer is installed, the TVisionSetupInfo utility modifies the following files:

- TVISION_HOME/config/datamgr/Database.properties
- TVISION_HOME/config/setup/Setup.properties

In addition, TVisionSetupInfo does the following:

- Saves the installation path for software tools in TVISION_HOME/config/setup/DefaultInstallPath.xml
- Generates TVISION_HOME/bin/SetupEnv.[sh | bat], which is run by TVisionSetup to set the JAVA_HOME, CLASSPATH, and system library path environment variables required by TransactionVision

Required Information

The TVisionSetupInfo utility prompts you to enter the following information necessary to complete the configuration. Please review these information requirements before running TVisionSetupInfo so you will be prepared to enter appropriate responses.

The following tables show parameters and associated technology areas (database and message system provider) that will be required. See the sections that follow for detailed descriptions of the parameters.

	Technology Database		
Parameter	**DB2**	**Oracle**	**SQL Server**
Log file location			
Database instance name	✓		✓
Database connection name	✓	✓	
Database name	✓	✓	✓
Database host	✓	✓	✓
Database user name	✓	✓	✓

Parameter	Technology Database		
	DB2	Oracle	SQL Server
Database user password	✓	✓	✓
Database listener port	✓	✓	✓
Database installation path	✓	✓	
JDBC driver installation path			✓

Parameter	Technology Message System Provider			
	WebSphere MQ	TIBCO EMS	Sonic MQ	WebLogic JMS
WebSphere MQ installation path	✓			
TIBCO EMS installation path		✓		
SonicMQ installation path			✓	
WebLogic JMS installation path				✓

Log File Location

You must supply the pathname of the directory where TransactionVision should store log files. The default value is TVISION_HOME/logs.

Database Properties

You must specify the type of database you plan to use with TransactionVision (IBM DB2, Oracle, or SQL Server).

The TVisionSetupInfo utility automatically sets up the DB2 Universal JDBC driver (type 4) or the Oracle thin client driver. For instructions on using other drivers, such as the DB2 Universal Driver (type 2) or the Oracle oci driver, see Running the Analyzer with a DB2 Type 2 or Oracle oci JDBC Driver on page 64.

For a DB2 database, you must supply the following information:

- The value of the DB2INSTANCE environment variable. The default value is DB2. Before starting your WebSphere 5 server, your DB2INSTANCE environment variable must be set. Failure to set this variable will cause a failure during TransactionVision startup and cause the WebSphere server to fail to start and initialize.
- The name of the database connection to be used by TransactionVision. See Chapter 6, Configuring Databases for more information on identifying the connection name and database name.
- The name of the database that TransactionVision will connect to. This name may be different from the "database_connection_name" if a client database connection is used. See Chapter 6, Configuring Databases for more information on identifying the connection name and database name.
- The name of the host that the database runs on. The default value is local_host.
- The user name for connecting to the database. If this field is empty, the currently logged in user is used to make the database connection. Make sure that the user specified or currently logged in has privileges for database access.
- The password for the database user name. If this field is empty, the currently logged in user's password is used to make the database connection. If TVisionSetupInfo is able to detect an installed JCE provider that supports the DES encryption algorithm, it sets the jce_provider value in the Database.properties file to the class name of the JCE provider and encrypts the password. Otherwise, it disables the password encryption feature and stores the password as unencrypted text.

- The database port number. This is the TCP/IP port the database server is listening on. A typical value is 50000. To identify the correct value, select the following menu items from the DB2 Control Center while the DB2 administration server is running:

  `All Systems > system_name > Instances > instance_name`

  Right-click and select **Setup Communications** from the context menu. Click **Properties** to get the port number your database instance is using.

For an Oracle database, you must supply the following information:

- The name of the database connection to be used by TransactionVision. The default value is TVISION. See Chapter 6, Configuring Databases for more information on identifying the connection name and database name.
- The name of the database that TransactionVision will connect to. The default value is TVISION. This name may be different from the "database_connection_name" if a client database connection is used. See Chapter 6, Configuring Databases for more information on identifying the connection name and database name.
- The name of the host that the database runs on. The default value is local_host.
- The user name for connecting to the database. If this field is empty, the currently logged in user is used to make the database connection. Make sure that the user specified or currently logged in has privileges for database access.
- The password for the database user name. If this field is empty, the currently logged in user's password is used to make the database connection. If TVisionSetupInfo is able to detect an installed JCE provider that supports the DES encryption algorithm, it sets the jce_provider value in the Database.properties file to the class name of the JCE provider and encrypts the password. Otherwise, it disables the password encryption feature and stores the password as unencrypted text.
- The database port number. This is the TCP/IP port the database server is listening on. A typical value is 1521. To identify the correct value, run the lsnrctl command on the server side. Use the status command to find out the port number.

For a SQL Server database, you must supply the following information:

- The name of the host that the database runs on. The default value is local_host.

- The name of the SQL Server instance. This should typically be empty unless you are running multiple server instances.
- The port number to connect to on the SQL Server
- The name of the SQL Server database that TransactionVision will connect to. The default value is TVISION.
- The user name for connecting to the database. If this field is empty, the currently logged in user is used to make the database connection. Make sure that the user specified or currently logged in has privileges for database access.
- The password for the database user name. If this field is empty, the currently logged in user's password is used to make the database connection. If TVisionSetupInfo is able to detect an installed JCE provider that supports the DES encryption algorithm, it sets the jce_provider value in the Database.properties file to the class name of the JCE provider and encrypts the password. Otherwise, it disables the password encryption feature and stores the password as unencrypted text.

Messaging System Provider

You must specify the messaging system provider used to transfer TransactionVision configuration messages and events between Sensors and the Analyzer. Supported providers include WebSphere MQ and TIBCO EMS. You may specify one or both providers.

Installation Paths

You will be prompted to enter installation paths for the following required software:

- Database
- WebSphere MQ or TIBCO EMS

Analyzer Settings

You must specify the following Analyzer information:

- The types of Sensors you are using in your environment

- The location of any JAR files that need to be added to the Analyzer CLASSPATH. These JAR files contain plugin beans that customize the Analyzer.

TVisionSetupInfo Syntax

1. Set the TVISION_HOME environment variable to the absolute path of the installation directory. For example, on Solaris, TVISION_HOME would be /opt/HP/TransactionVision; on AIX it would be /usr/lpp/HP/TransactionVision.
2. Run the TVisionSetupInfo.[sh|bat] script. This utility modifies TransactionVision configuration files, so you must have file modification privileges to run it (typically root for UNIX or Administrator for Windows).

Operating System	Script Command
AIX, Linux, Solaris	<TVISION_HOME>/bin/TVisionSetupInfo.sh
Windows	<TVISION_HOME>\bin\TVisionSetupInfo.bat

TVisionSetupInfo prompts you to enter information required to configure the following:

- Log files
- Database properties
- Messaging system provider
- Installation paths for dependent software
- Analyzer settings
- Analyzer runtime environment

When responding to user input prompts from TVisionSetupInfo, press **Enter** to accept the default value shown in brackets; otherwise, type the correct value and press **Enter**. To specify an empty value, press the spacebar, and then press **Enter**. In the following sections, sample input is shown in italics.

As TVisionSetupInfo completes each configuration section, it displays messages indicating which files have been updated.

Regarding integrations:

By default all of the integration columns are created in the database when the schema is created. These should be dropped only by an advanced user, probably for performance reasons, if they are sure that none of the integrations will ever be used. You can disable this by running TVisionSetupInfo with the `-integration` option.

Configuring Log Files

- `Please specify name of directory where you want to store your log files [C:\Program Files\Hewlett-Packard\TVision\logs]: C:\Program Files\Hewlett-Packard\TransactionVision\logs`

  Enter the pathname of the directory where you want TransactionVision to store Analyzer log files. The default value is TVISION_HOME/logs.

Configuring Database Properties

- `Type of Database (DB2/Oracle/SQLServer)? [DB2]: DB2`

  Enter the type of database you are using to store TransactionVision event data. The remaining database property prompts depend on the type of database you specify.

DB2 :

- `Value of your environment variable DB2INSTANCE [DB2]: DB2`

  Enter the name of the DB2 instance.

Before starting your WebSphere 5 server, your DB2INSTANCE environment variable must be set. Failure to set this variable will cause a failure during TransactionVision startup and cause the WebSphere server to fail to start and initialize.

- `Database connection name to be used by TransactionVision [TVISION]: TVDB`

Enter the name of the database connection to be used. See Chapter 6, Configuring Databases for more information on identifying the connection name and database name.

- `Name of database TransactionVision connects to [TVISION]: TVDB`

  Enter the name of the database on the server to connect to. See Chapter 6, Configuring Databases for more information on identifying the connection name and database name.

- `Name of the host the database is running on [local_host]: devpc`

  Enter the name of the host the database server is running on.

- `Database user name [db2admin]:`

  Enter the user name to be used while making the database connection. Leave blank to use the currently logged in user to make the database connection.

- `User password [ibmdb2]:`

  Enter the password to be used while making the database connection. Leave blank to use the currently logged in user's password to make the database connection.

- `Database port number [50000]:`

  Enter the TCP/IP port number the database server is listening on. See Required Information on page 42 for instructions on identifying the correct port number.

  The TVisionSetupInfo utility automatically searches for an installed JCE provider that supports the DES encryption algorithm. If such a provider is found, TVisionSetupInfo sets the jce_provider value in the Database.properties file to the class name of the JCE provider. If no JCE provider is found, TVisionSetupInfo displays the following message:

  ```
  Java Cryptography Extension (JCE) is not present in your
  current JDK. Password encryption feature will be disabled
  and stored in plain text.
  ```

Oracle:

- `Database connection name to be used by TransactionVision [TVISION]: TVDB`

  Enter the Oracle Local Net Service Name for an oci client connection or the SID (system identifier) for a thin client connection. See Chapter 6, Configuring Databases for more information on identifying the connection name and database name.

- `Name of database TransactionVision connects to [TVISION]: TVDB`

  Enter the Oracle SID. This prompt does not appear for an Oracle thin client connection. See Chapter 6, Configuring Databases for more information on identifying the connection name and database name.

- `Name of the host the database is running on [local_host]: devpc`

  Enter the name of the host the database server is running on.

- `Database user name []:`

  Enter the user name to be used while making the database connection. Leave blank to use the currently logged in user to make the database connection.

- `User password []:`

  Enter the password to be used while making the database connection. Leave blank to use the currently logged in user's password to make the database connection.

- `Database port number [1521]:`

  Enter the TCP/IP port number the database server is listening on. See Required Information on page 42 for instructions on identifying the correct port number.

  The TVisionSetupInfo utility automatically searches for an installed JCE provider that supports the DES encryption algorithm. If such a provider is found, TVisionSetupInfo sets the jce_provider value in the Database.properties file to the class name of the JCE provider. If no JCE provider is found, TVisionSetupInfo displays the following message:

```
Java Cryptography Extension (JCE) is not present in your
current JDK. Password encryption feature will be disabled
and stored in plain text.
```

SQLServer:

- `Name of the host the database is running on [local_host]:`

  Enter the name of the host the database server is running on.`

- `Enter the SQL Server instance name (typically it is empty unless you have multiple instances of SQL Server running in the same host) []:`

  Enter the name of the SQL Server instance you wish to use, or leave the field blank if only a single instance is installed.

- `Enter the port number that the SQL Server listener is on [1433]:`

  Enter the port for the SQL Server listener process.

- `Name of database TransactionVision connects to [TVISION]: TVDB`

  Enter the name of the database on the server to connect to.

- `Name of the host the database is running on [local_host]: devpc`

  Enter the name of the host the database server is running on.

- `Database user name []:`

  Enter the user name to be used while making the database connection. Leave blank to use the currently logged in user to make the database connection.

- `User password []:`

  Enter the password to be used while making the database connection. Leave blank to use the currently logged in user's password to make the database connection.

The TVisionSetupInfo utility automatically searches for an installed JCE provider that supports the DES encryption algorithm. If such a provider is found, TVisionSetupInfo sets the jce_provider value in the Database.properties file to the class name of the JCE provider. If no JCE provider is found, TVisionSetupInfo displays the following message:

```
Java Cryptography Extension (JCE) is not present in your
current JDK. Password encryption feature will be disabled
and stored in plain text.
```

Configuring Messaging System Provider

- `Please select your Queuing Product Type by number.`

```
1.  WebSphere MQ
2.  TIBCO EMS
3.  SonicMQ
4.  WebLogic JMS
5.  All of the above
```

`Your Queuing Product [1]: 5`

Enter the number corresponding to the messaging system provider you wish to use to transfer configuration and event messages between the Analyzer and Sensors. TVisionSetupInfo uses this information to generate the proper shell script to set up the Analyzer runtime environment correctly. If you want the Analyzer to collect events from any message system provider, select option **5**.

Configuring Installation Paths for Dependent Software

The following prompt only appears if you selected DB2 as the database.

- `DB2 installation location [C:\Program Files\sqllib]:`

  Enter the full pathname of the installation location of your DBMS software.

The following prompt only appears if you selected Oracle as the database.

```
Oracle installation location [C:\oracle\product\10.2.0\db_1]:
```

The following prompt only appears if you selected SQLServer as the database. This will prompt you for the JDBC driver installation location instead:

- `SQLServer JDBC Driver installation location []:`

  Enter the full pathname of the SQL Server JDBC driver installation directory.

The following prompt only appears if you selected WebSphere MQ as a message system provider.

- `WebSphere MQ installation location [C:\Program Files\IBM\WebSphere MQ]:`

  Enter the full pathname of the WebSphere MQ installation directory.

The following prompt only appears if you selected TIBCO EMS as a message system provider.

- `TIBCO EMS installation location [C:\tibco\ems]:`

  Enter the full pathname of the TIBCO EMS installation directory.

The following prompt only appears if you selected SonicMQ as a message system provider.

- `SonicMQ installation location [C:\Sonic\MQ7.5]:`

  Enter the full pathname of the SonicMQ installation directory.

The following prompt only appears if you selected WebLogic JMS as a message system provider.

- `WebLogic installation location [C:\bea\weblogic81]:`

  Enter the full pathname of the WebLogic JMS installation directory.

TransactionVision Analyzer Settings

- `Please select your Sensor types by number. This enables the Analyzer to receive events from the selected Sensors. If you are selecting multiple Sensor types, please separate them with ",".`

  1. `CICS`
  2. `EJB`
  3. `Servlet`
  4. `TIBCO EMS, SonicMQ JMS or WebLogic JMS`

```
5.  WebSphere MQ
6.  WebSphere MQ IMS Bridge
7.  WebSphere MQ JMS
8.  JDBC
9.  All of the above

Your Sensor type(s) [9] :
```

- Enter the number corresponding to the types of TransactionVision Sensors you plan to use. For multiple Sensor types, separate the numbers with a comma. For example, if you plan to use the EJB, WebSphere MQ, and WebSphere MQ JMS Sensors, enter the following:

  ```
  Your Sensor type(s) [8] :2,5,7
  ```

  If you plan to use all Sensor types, enter **9**.

Customize the Analyzer Runtime Environment

- The Analyzer can optionally be customized by plugin beans in JAR files.

  The location of these JAR files needs to be added to the Analyzer CLASSPATH.

  Please specify a semicolon delimited list of any additional JAR files you wish to be added to the CLASSPATH. (for example, 'C:\TVision\myext.jar;C:\TVision\myutil.jar') []:

  Specify the location of any JAR files that need to be added to the Analyzer CLASSPATH. These JAR files contain plug-in beans that customize the Analyzer.

Completion Information

Upon completion, TVisionSetupInfo displays the following messages:

```
TransactionVision Info(TVisionSetupInfoSuccess)
: TVisionSetupInfo has completed successfully.
All program output and user input are logged to "C:/Program
Files/Hewlett-Packard/TransactionVision/logs/setup.log".
Please run TVisionSetup to complete TransactionVision setup.
```

```
Note:
- TVisionSetup needs to be run in a user environment which has
the correct database environment set.
- For IBM DB2 this includes checking the DB2INSTANCE
environment variable setting and that the user has database
access rights.
- An IBM DB2 database, required to store TransactionVision
data, needs to be created before running TVisionSetup.
- Refer to your IBM DB2 manual for further information on
database setup.
```

Use the TVisionSetup utility, described in the following section, to continue initialization.

Using TVisionSetup

Run TVisionSetup once after installation. When only the Analyzer is installed, you can use this utility to do any or all of the following:

- Initialize the database for TransactionVision, or verify that the database is initialized
- Verify that WebSphere MQ and/or TIBCO EMS is set up for TransactionVision
- Initialize the TransactionVision Analyzer, or verify that the Analyzer is initialized

Before You Begin

Before you run TVisionSetup, you must set the correct database environment in your user environment:

- For IBM DB2, the DB2INSTANCE environment variable must be set and you must have database access rights.
- For Oracle, the ORACLE_HOME environment variable must be set and you must have database access rights.
- A database to store TransactionVision data must be created before running TVisionSetup.

See Chapter 6, Configuring Databases for database configuration information such as required settings and disk space. Refer to your IBM DB2, Oracle, or SQL Server documentation for further information on database setup.

TVisionSetup Syntax

Use the following commands to run TVisionSetup:

Operating System	Script Command
AIX, Linux, Solaris	<TVISION_HOME>/bin/TVisionSetup.sh
Windows	<TVISION_HOME>\bin\TVisionSetup.bat

The syntax for TVisionSetup is as follows:

```
<TVISION_HOME>/bin/TVisionSetup [-rmiregp PORTNUMBER]
[-debug][-h]
```

- The default port number where the Analyzer listens for RMI calls is 1099; use the -rmiregp option to specify a different port number.
- Use the -debug option to start rmid and rmiregistry in debug mode.
- Use the -h option to print the usage message for this utility.

TVisionSetup must be run in a user environment which has the correct database environment set. For IBM DB2 this includes checking the DB2INSTANCE environment variable setting and that the user has database access rights. Refer to your IBM DB2 manual for further information.

This command provides different menus, depending on whether the host is a WebSphere or WebLogic AppServer with the TransactionVision web user interface installed, and whether LDAP demo mode is used.

Enter the number corresponding to the desired menu action, or enter 99 to perform all actions. The results of each menu action are displayed as they are performed; press the **Enter** key to continue with the next menu item. Some menu actions require user input.

The following menu appears if the TransactionVision Analyzer is installed. For information about using TVisionSetup if the TransactionVision web user interface is installed, see the *TransactionVision Web Application Installation and Configuration Guide*.

```
1. Initialize database for TransactionVision
2. Verify database initialization for TransactionVision
3. Verify WebSphere MQ setup for TransactionVision
4. Verify TIBCO setup for TransactionVision
5. Verify SonicMQ setup for TransactionVision
6. Initialize TransactionVision Analyzer
7. Verify TransactionVision Analyzer initialization
99. Select all
V. Display version info for software
Type "quit" or "q" to exit
```

The following sections describe the required input for each menu item.

Initialize Database for TransactionVision

Select this item to initialize the database that you have created for TransactionVision events. This selection does not require any input. For DB2 and Oracle, this menu item always creates the TVision schema in the default tablespace. To use a different tablespace, run the CreateSqlScript script the -ts option.

Verify Database initialization for TransactionVision

Select this item to verify that the database is initialized for TransactionVision. This selection does not require any input.

Verify WebSphere MQ setup for TransactionVision

Select this item to verify that the Analyzer is able to communicate with the specified queue manager. You may test a server or client connection.

This menu item first prompts you to enter the name of the queue manager to test:

```
Verifying WebSphere MQ Java setup for TransactionVision ...
Please specify WebSphere MQ queue manager name:
```

Enter the name of the queue manager where TransactionVision Sensors will check for configuration messages and place event messsages.

To test using a server connection, enter **Y** to the following prompt:

```
Do you want to test WebSphere MQ server connection? (Y/N) [N]:

TransactionVision Info(MQInitialized): WebSphere MQ is loaded
properly for WebSphere MQ server connection
```

To test using a client connection, enter **Y** to the following prompt:

```
Do you want to test WebSphere MQ client connection? (Y/N) [N]:
```

When testing a client connection, you must enter the following additional information:

```
Host name for WebSphere MQ queue manager:
Channel name for WebSphere MQ queue manager:
Port number for WebSphere MQ queue manager:
CCSID for WebSphere MQ queue manager (leave blank if unknown):
Hit <Enter> to continue ...:
```

Verify TIBCO setup for TransactionVision

Select this item to verify that the Analyzer is able to communicate with the specified host. You must specify the host name, port number, and the user name and password to use for the connection.

```
Verifying TIBCO EMS setup for TransactionVision ...
  TIBCO EMS Server Host Name []:
  TIBCO EMS Server Port Number []:
  User Name []:
  Password []:
Testing TIBCO EMS server connection ...
2005-04-12 09:21:35,631 [main] - TransactionVision
Info(TIBCOInitialized): TIBCO  is loaded properly for TIBCO
connection
Hit <Enter> to continue...:
```

Initialize TransactionVision Analyzer

Select this menu item to start the Analyzer the first time after installing and configuring it. This menu item initializes the database connection, checks the Analyzer license, and connects to the RMI registry. On Windows, the TransactionVision Analyzer Windows Services is started; you may control this service via the Windows Control Panel Services Console.

The Analyzer uses an embedded RMI registry so that Analyzers may be controlled by the TransactionVision web user interface running on remote hosts.

```
Initializing TransactionVision Analyzer ...
Do you want to start TransactionVision Analyzer? (Y/N) [Y]:
Hit <Enter> to continue ...:
```

For instructions on starting, stopping, and exiting the Analyzer after you run TVisionSetup, see Starting and Stopping the Analyzer on page 59.

Verify TransactionVision Analyzer initialization

Select this menu item to verify that the TransactionVision Analyzer is initialized correctly.

Starting and Stopping the Analyzer

When you run TVisionSetup, you may choose to start the Analyzer from the Verify TransactionVision Analyzer initialization menu item. Once you've run TVisionSetup, however, you will use the ServicesManager utility to manage the Analyzer. This utility has the following syntax:

```
ServicesManager
 {(-start [-project (-proj) PROJECTNAME]) |
  (-stop [-quiesce][-project (-proj) PROJECTNAME]
[-keepcollect]) |
  -exit [-quiesce][-keepcollect]) |
  (-status [-project (-proj) PROJECTNAME]) |
  (-killserver) |
  (-reconfig {classification | logging | analyzer}) |
  (-versioninfo)
  ([-host HOST][-rmiregp PORTNUMBER][-debug])
```

Option	Description
-start	Starts the Analyzer process if it is not already running.
-stop	The Analyzer stops collecting event data.
-exit	The Analyzer stops collecting event data, then the process exits.
-quiesce	The default behavior of the Analyzer on a stop or exit is to immediately close down collection. If this flag is set, the Analyzer clears out any pending events in the event queue before stopping or exiting. Note that if there is a large event backlog, the Analyzer may take some time to stop or exit.
-keepcollect	Do not send stop message to Sensor. Used in combination with the –stop or –exit option so that the Sensor keeps collecting.
-status	Reports the current Analyzer status.
-reconfig	Reloads Analyzer configuration settings (classification/ logging/all settings) without stopping event collection. • **classification** – Re-initializes the classification rule engine and reloads the Analyzer classification files reading in any changes that may have been made to that file. • **logging** – Reloads the log4j XML configuration files in the directory <TVISION_HOME>/config/logging. • **analyzer** – Re-initializes the Analyzer by reloading most Analyzer settings in the Analyzer.properties file, all beans described in the Beans.xml file, re-initializing logging and all XML based rule files. Settings in the “Collection Properties” section of the Analyzer.properties file, except the write_to_buffer_table property, are not reloaded by the Analyzer with the -reconfig option.
-versioninfo	Returns Analyzer version information.

Option	Description
-killserver	Shuts down the Analyzer immediately. This flag is not recommended; -exit is the preferred method for shutting down the Analyzer cleanly.
-project (-proj) PROJECTNAME	Name of the project for the specified command. If the project name contains a space, enclose the project name in double quotation marks (for example, -proj “Project Name”). The project must have a communication link in order for the Analyzer to process events. Only the -start, -stop, and -status commands can be performed on a single project. This option cannot be used in combination with the -host or -rmiregp options. When the -project option is used, the Analyzer host and port is looked up from the database.
-host HOST	Name of the host where the Analyzer runs. Can be either the name or IP address. Local host is used if not specified.
-rmiregp PORTNUMBER	Port number on which the Analyzer listens for RMI connections. The default value is the value specified by the analyzer_port property in the Analyzer.properties file. This option only takes effect when communicating with an Analyzer that is already running; the Analyzer always uses the value specified in Analyzer.properties when starting up.
-debug	Start the Analyzer process in debug mode

Starting the Analyzer

To start the Analyzer on the local host, use the following command:

```
ServicesManager –start
```

To start the Analyzer on a remote host, use the following command:

```
ServicesManager –start –host HOSTNAME
```

To start the Analyzer, if it is not already running, and start event collection for a specific project, use the following command:

```
ServicesManager -start -proj PROJECTNAME
```

Stopping Event Collection

To stop event collection by the Analyzer on the local host, enter the following command. Note that the Analyzer process remains active.

```
ServicesManager -stop
```

You may also stop collection for a specific project with the following command:

```
ServicesManager -stop -proj PROJECNAME
```

Stopping the Analyzer

To stop event collection and exit the Analyzer process on the local host, use the following command:

```
ServicesManager -exit
```

To clear out any pending events in the event queue before exiting, enter the following command. Note that if there is a large event backlog, the Analyzer may take some time to exit.

```
ServicesManager -exit -quiesce
```

To exit the Analyzer without sending a stop collection message to the Sensor (so that the Sensor continues to collect events), use the following command:

```
ServicesManager -exit -keepcollect
```

Note that the Sensor only continues to collect events until the last configuration message has expired, since there will not be any new configuration messages sent by the analyzer. You can change the configuration message expiry on the analyzer settings page in the TransactionVision UI.

Additional Analyzer Configuration

Check the following parameters for the Analyzer:

- Make sure the DBMS schema has been tuned up and tested. For more information, see Chapter 6, Configuring Databases.
- Configuration Message Expiry should be set to a value small enough so that the amount of events generated by an “orphaned” configuration message can fit into the event queues without causing major production issues. For more information, see the *TransactionVision Administration Guide*.
- Analyzer thread count should be set to match the test results from the DBMS insert test. For more information, see the *TransactionVision Administration Guide*.

Windows Service Properties

On Windows, the Analyzer is a Windows service named TransactionVision Analyzer Service, which can be managed through the Services administrative tool.

Service Properties

By default, the Analyzer service is configured to use the local system account and to startup manually when the system starts. To change the account name or to start the Analyzer automatically, use the Windows Services administrative tool, accessible through the Control Panel > Administrative Tools > Services.

Note that the Analyzer must have access to WebSphere MQ and a database. For it to function correctly, the assigned Log On account must have permission to access these resources.

Error Logging

Once the Analyzer is running, it logs its errors to TVISION_HOME/logs/analyzer.log. If an error occurs while initializing the service portion of the Analyzer, however, they can be found in the analyzer_startup.log file.

Running the Analyzer on a 64-Bit JVM

The TransactionVision Analyzer can be run on a 64-bit JVM on the AIX and Solaris platforms. This allows usage of larger JVM memory sizes for large caches. It also allows usage for 64-bit JDBC and MQ drivers.

On Solaris, no change is required to run the Analyzer in a 64-bit JVM. On Solaris, the 64-bit JVM binary (in the <JAVA_HOME>/bin/sparcv9 directory) is used by default by the ServicesManager.sh script.

On AIX, set the JAVA_HOME environment variable explicitly to the 64-bit JDK in the environment the Analyzer is started in. For example, in the tcsh environment, if the 64-bit JDK is installed at /usr/java14_64:

```
setenv JAVA_HOME /usr/java14_64
$TVISION_HOME/bin/ServicesManager.sh
```

If the JDBC type 2 driver is used on DB2 or Oracle, or if a WebSphere MQ connection is made using server bindings, the library path environment variable is automatically set by the <TVISION_HOME>/bin/SetupEnv.sh script.

Running the Analyzer with a DB2 Type 2 or Oracle oci JDBC Driver

The TVisionSetupInfo utility sets up the DB2 Universal JDBC driver(type 4) or the Oracle thin client driver. To use other drivers such as the DB2 Universal Driver(type 2) or the Oracle oci driver, perform the following steps:

1. Run TVisionSetupInfo at least once to input required parameters such as the database host name, database name, etc.
2. To use the DB2 Universal Driver(type 2), add the property jdbc_url in the property file <TVISION_HOME>/config/datamgr/ Database.properties in the following format:

   ```
   jdbc:db2:<database-name>,
   ```

   where <database-connection-name> is the database alias name used by the Analyzer. Refer to Identifying Database Connection Names on page 27 to determine the database connection name.
3. To use the Oracle oci driver, add the property jdbc_url in the property file <TVISION_HOME>/config/datamgr/ Database.properties in the following format:

```
jdbc:oracle:oci:<user>/<password>@<database-name>
```

Where <database-connection-name> is the Oracle Net service name used by the Analyzer, and <user>, <password> are the user name and passwords required for the Oracle connection. Refer to Identifying Database Connection Names on page 27 to determine the database connection name.

4 Once the jdbc_url property has been changed, both the Analyzer and the application server will use the property value to connect to the database. Application servers such as WebSphere or WebLogic use 32-bit JVMs and do not require any changes to the library path environment variable.

5 If you are sharing the same TVISION_HOME for the Analyzer and the web application (that is, you are running the Analyzer and the web application on the same host), and the web application runs in a 64-bit JVM (on AIX or Solaris), you will need to modify the native library path for the application server to include the 64-bit library directory.

Multithreaded Servlet/JMS Events

By default, if a servlet spins off a thread to make some JMS calls, the servlet passes tracking information to the child thread. The result is that both the servlet and JMS events belong to the same business transaction. However, there may be some cases in which you wish to separate these events into different transactions. For example, a servlet may spin off a long-running thread that you do not want to be part of the same transaction as the servlet.

To change the default behavior so that the child thread is not considered by TransactionVision to be part of the same business transaction as the servlet that spins it, perform the following steps:

1 Open the <TVISION_HOME>/config/services/Analyzer.proper-ties file.

2 Add the name of the program that spins off threads to the separate_child_thread_txns property, as in the following example:

```
separate_child_thread_txns=program1, program2
```

Separate multiple program names with a comma.

3 Close and save Analyzer.properties.

4 Restart the Analyzer.

Disabling unneeded analyzer beans

If you configured the analyzer for JMS event collection during TVisionSetupInfo, the JMSPubSubRelationBean will get enabled by default. This bean is needed to correlate JMS Publish and Subscribe events during event analysis.

If you:

- do not collect any PubSub JMS events from your sensors, AND -
- require maximum performance of the analyzer, then you can disable this bean to avoid the overhead related to handling the Publish/ Subscribe mechanism.

To disable the JMSPubSubRelationBean

Edit `<TVISION_HOME>/config/Beans.xml` in the following way:

```
<!--  this bean is used to correlate Publish/Subscribe events -->

<Module type="Bean"
class="com.bristol.tvision.services.analysis.eventanalysis.JMSPu
bSubRelationshipBean"/>
```

To:

```
<!--  this bean is used to correlate Publish/Subscribe events -->

<!--Module type="Bean"
class="com.bristol.tvision.services.analysis.eventanalysis.JMSPu
bSubRelationshipBean"/-->
```

Local Transaction Matching

By default, the Analyzer uses strict local transaction matching to group WebSphere MQ events into local transactions. However, there may be times when you want to disable strict local transaction matching and instead use the default MQ local transaction bean, which groups events into local transactions according to unit of work.

To disable strict local transaction matching for certain WebSphere MQ events, you must define criteria in the TransactionVision MQStrictLocalTxnExclude.xml file. If an event matches any of the criteria specified in this file, the Analyzer will not put it into a separate local transaction, but will use the default MQ local transaction bean instead.

Criteria consist of a program name, queue manager, and object name combination. The following example disables strict local transaction matching for events that meet either of the following criteria:

- Program name amqcrsta, queue manager QM1, and object TEST.Q1
- Program name amqcrsta, queue manager ALT_QM, and object TEST.Q

```
<Criterias>
  <Match id="0">
    <value xpath="/Event/StdHeader/ProgramName">amqcrsta</
value
    <value xpath="/Event/Technology/MQSeries/MQObject/
@queueManager"> QM1</value>
    <value xpath="/Event/Technology/MQSeries/MQObject/
@objectName"> TEST.Q1</value>
  </Match>

  <Match id="1">
    <value xpath="/Event/StdHeader/ProgramName">amqcrsta</
value
    <value xpath="/Event/Technology/MQSeries/MQObject/
@queueManager"> ALT_QM</value>
    <value xpath="/Event/Technology/MQSeries/MQObject/
@objectName"> TEST.Q1/value>
  </Match>
</Criterias>
```

Analyzer Failure Mode

The Analyzer can operate in two modes: standard mode and failure mode. In failure mode, the Analyzer will only store event data of failed business transactions (or transactions violating SLA). For successful business transactions, only the corresponding business transaction rows will be saved. Using failure mode reduces the data storage requirement for a project. For more information about enabling failure mode, see the *TransactionVision Administration Guide*.

JDBC Sensor Database Resolution

All JDBC events report the current database connection URL.

For DB2, the URL syntax can be one of the following:

- jdbc:db2:<db2 database alias> for type 2 driver.
- jdbc:db2://<host>:<port>/<db2 database alias> for type 4 driver.

For Oracle, the URL syntax may take the following forms:

- jdbc:oracle:thin:@//<host>:<port>/<service name> for type 4 driver.
- jdbc:oracle:thin:<host>:<port>:<SID> for type 4 driver.
- jdbc:oracle:oci:@<TNSName> for type 2 driver.
- jdbc:oracle:oci:@ for local default connection.

The JDBC sensor uses the connection URL to identify which database the corresponding events are associated with. In many cases the URL contains all the information necessary to identify the database. This is typically the case with a type 4 style URL that contains the host, port and database name. In such a case making changes to this file is not needed. It is also possible that the URL does not have this information, then, in order to identify the database name the TransactionVision analyzer follows a set of rules defined in the `JDBCSystemModelDefinition.xml`, Knowing whether you need to configure this file depends on how your application connects to the database, and whether it is desired that any database aliases that may be used are resolved to the actual name of the underlying database.

TVISION_HOME/config/services/JDBCSystemModelDefinition.xml

This file contains data to associate JDBC connection URLs to database level objects. It contains these four sections:

- Default database instance name for different vendors (optional).
- A list of database instance objects in the monitored environment that can be referred to in the local default database and mapping section.
- A list of default local database connections. This applies to Oracle only.
- Mapping between JDBC URL attributes to the corresponding database.

Default database instance

This optional section defines the default database instance string to be used for a DB vendor and platform:

- <defdb2instwin>db2</defdb2instwin> This defines the default database instance name for DB2 on a Microsoft Windows platform.
- <defdb2inst>db2inst1</defdb2inst> This defines the default name for DB2 on all other platforms.
- <deforacleinst>1521</deforacleinst> This defines the default port number for the oracle instance.

If these entries do not exist, TransactionVision assumes the default values shown above.

Database instance objects

The second section defines the different database instances in the environment:

```
<dbinstance vendor="oracle" host=" myhost " name="1544">
    <protocol name="tcp">
       <param name="host" value=" myhost "/>
       <param name="port" value="1544"/>
    </protocol>
 </dbinstance>
```

Each database instance element should have the following attributes: vendor (db2 or oracle), host, and instance name. The instance can have one or more protocol sections.

For DB2, the instance name is set to the DB2 instance. For Oracle, it is set to the port number of the listener process.

Default local database connections

The third section defines the default database for the given host and database instance. Note that there should be a database instance definition (see above section) for the instance referred to in the local default database definition.

```
<localdefdb vendor="oracle" host="myhost" inst="1544"
          name="mydb"/>
```

An example use case of the default local database setting is if a JDBC program made a connection using a URL like "jdbc:oracle:oci:@'. In this case TV has no way of identifying the database name being used, so it will look in the `JDBCSystemModelDefinition.xml` file and see what is defined as the default database for the host the event is occurring under. Assuming this was running on 'myhost' from the example above, it would associate this event with the Oracle database named 'mydb'.

JDBC URL Mapping

The fourth section defines the mapping between the database alias in the JDBC URL and the actual database name.

The mapping element looks like the following:

```
<mapping vendor="vendor">
     <src host="host name" dbalias="db alias"/>
     <dst host="host name" inst="instance name" dbname="db
 name"/>
</mapping>
```

- The vendor attribute's value can be set to "db2" or "oracle".
- The dbalias attribute in the <src> element, is set to the TNS name for Oracle, and database alias in DB2.
- The host of the <src> element is the host that the monitored application is running on.
- The instance attribute only is needed in DB2 mapping entries.
- The <dst> element contains the definition that a match will map to, and its host and instance attributes must match a <dbinstance> entry defined earlier.

For example, given the following mapping entry:

```
 <mapping vendor="oracle">
    <src host="apphost" dbalias="DBNAME"/>
    <dst host="myhost" inst="1544" dbname="mydb"/>
</mapping>
```

In this situation, if a sensored application running on 'apphost' used a JDBC URL connection such as 'jdbc:oracle:oci:@DBNAME', this would enable the analyzer to resolve this database to the oracle 'mydb' database running on 'myhost'.

How a database name is resolved

After defining rules in the `JDBCSystemModelDefinition.xml`, when the events are processed by the analyzer it will follow a set of rules in determining the database name to report. The order of this resolution follows these steps:

Database Resolution for DB2 Type 2 Driver

1. Search the database mapping list and find if there is any entry with the source attributes matching the event host and database alias name.
2. If a matching entry can be found, create a database object using the information from the mapping destination section.
3. If no such entry can be found, create a database object using the event host, local database instance name (or default value if no such data exists in the event), and database alias name.

Database Resolution for DB2 Type 4 Driver

1. Follow Step 1 and 2 in the resolution instructions for DB2 type 2 driver.
2. If (1) fails, follow the below steps.
3. Parse the URL to retrieve the server name and listening port number.
4. Find an entry in the database instance list with matching host and port number.
5. If a matching entry can be found in (2), the event is to be associated with the found database instance. The database name is set to the one reported by the event URL.
6. If no such entry can be found in (2), and the event host is the same as the URL host, use the local database instance data from the event as database instance, or the default value if no such data exists in the event. Create a database object with the host, database instance, and URL database name as determined above.
7. If no such entry can be found in (2), and the event host is different from the URL host, create a database object using the URL host, default database instance name, and URL database name

Database Resolution for Oracle Type 2 Driver

1. If the URL contains a database name (TNS name), check if there is an entry in the database mapping list with the source host and database alias name matching the event host and database name.
2. If a matching entry can be found, use the mapping host section data (host, instance, database name) to create a database object for the event.
3. If no matching entry can be found, create a database object using (a) the JDBC event host, (b) default Oracle database instance (port), and (c) event database name.
4. If the URL contains no database name, check if there is an entry in the local default database section with matching host name. If more than one is found, use the first returned match. Create a database object using the host, database instance, and database name data from the matching entry.
5. If no matching entry can be found, create a database object using (a) the JDBC event host, (b) the default value for Oracle database instance, and (c) the string "default" as database name

Database Resolution for Oracle Type 4 Driver

1. No resolution is necessary in this case. The URL should provide the database host, listener port number, and SID/service name.
2. Check if the configuration data provides details on the Oracle database instance identified by the host and port number. Use the information if such record exists.
3. Create a database object in the system model table with the database instance and SID/service name information.

Reducing Event Database Size

TransactionVision provides an XML event compression bean. Use this bean to reduce the database size for each event.

If the XML Event Compression bean is enabled, it is not possible to query on user buffer data.

After running TVisionSetupInfo, open the file <TVISION_HOME>/config/services/Beans.xml, change the following segment:

Original Segment:

```
<Module name="DBWriteEventCtx" type="Context">

  <!-- This context contains beans that write the XML event
(or part of it) to the database. -->
  <!-- Each registered bean in the chain is called. -->

  <Module
class="com.bristol.tvision.services.analysis.dbwrite.DBWriteE
ventDefaultBean" type="Bean"/>


  <!-- Replace the default bean with this one if you want ZIP
compression for the XML event -->
  <!--Module type="Bean"
class="com.bristol.tvision.services.analysis.dbwrite.DBWriteE
ventCompressedBean" /-->
</Module>
```

Modified Segment:

```
<Module name="DBWriteEventCtx" type="Context">

  <!-- This context contains beans that write the XML event
(or part of it) to the database. -->
  <!-- Each registered bean in the chain is called. -->

  <!--<Module
class="com.bristol.tvision.services.analysis.dbwrite.DBWriteE
ventDefaultBean" type="Bean"/-->
```

```
  <!-- Replace the default bean with this one if you want ZIP
compression for the XML event -->
  Module type="Bean"
class="com.bristol.tvision.services.analysis.dbwrite.DBWriteE
ventCompressedBean" />
</Module>
```

Make sure only one bean is enabled at one time; otherwise, when both DBWriteEventDefaultBean and DBWriteEventCompressedBean are enabled, an exception will be thrown.

When using the compression bean, your DatabaseDef.xml must be updated to store the event data in BLOB format. The EVENT and EVENT_OVERFLOW table definitions must be updated, and the event_data column type changed from CLOB to BLOB. After this change has been made you will need to restart your Analyzer and Web application, and create a new project for the changes to take effect. The following example shows DatabaseDef.xml before the change to store event data in BLOB format:

```
<Table name="EVENT" volatile="true">
[…]
  <Column name="event_data" type="CLOB" size="1M"/>
[…]
  </Table>
  <Table name="EVENT_OVERFLOW" volatile="true">
  […]
  <Column name="event_data" type="CLOB" size="1M"
notNull="true"/>
  […]
</Table>
```

8 Configuring Analyzer Logging

Log Files

By default, all TransactionVision components log error and warning messages to the appropriate log files. The location of log files is specified when you run TVisionSetupInfo or SensorSetup and stored in the Setup.properties file.

The Analyzer logs error messages to the analyzer.log file. On Windows, the Analyzer uses three additional log files:

- **analyzer_startup.log** contains information about the running of the Windows service portion of the Analyzer. It typically contains information about what options the Analyzer started under. If errors are encountered during the initializing of the service portion of the Analyzer, they can be found in this file.
- **analyzer_stderr.log** and **analyzer_stdout.log** represent the standard output and error of the Analyzer process. If you have custom analysis beans that print to the console or to standard error, you can find their output in these files. These files should also be referred to for further information if you see problems in starting or running the Analyzer and the standard analyzer.log file does not contain anything indicating an error.

Circular Logging

By default, the Analyzer employs a form of circular logging. When the log file reaches the configured maximum size, it is renamed as a backup file and a new, empty log file is created. By default, the maximum log size is 10 MB and there is one backup log file.

Using the defaults, when a log file (for example, the Analyzer log file analyzer.log, reaches 10 MB in size, it is renamed analyzer.log.1 and a new analyzer.log file is created. If you change the configuration so that there are two backup files, the following events take place when analyzer.log reaches 10 MB:

- analyzer.log.2 is removed if it exists.
- analyzer.log.1 is renamed analyzer.log.2.
- analyzer.log is renamed analyzer.log.1.
- A new analyzer.log is created.

If you do *not* wish to use circular logging, you may change the configuration to use linear logging, in which a single log file is generated.

The <TVISION_HOME>/config/logging/*.Logging.xml files specify the type of logging used, the maximum log file size, and the number of backup log files for each component. For example, Sensor.Logging.xml specifies the configuration for the servlet and JMS Sensors. This file contains entries similar to the following:

```
<appender
class="tvision.org.apache.log4j.RollingFileAppender"
name="SENSOR_LOGFILE">
  <param name="File" value="c:/Program Files/Hewlett-Packard/
TransactionVision/logs/sensor.log"/>
  <param name="Append" value="true"/>
  <param name="MaxBackupIndex" value="2"/>
  <param name="MaxFileSize" value="10MB"/>
  <layout class="tvision.org.apache.log4j. PatternLayout">
  <param name="ConversionPattern" value="%d [%t] %-5p %c %x -
%m%n"/>
  </layout>
</appender>
```

Maximum Log File Size

To change the maximum size of the log file, change the value of the MaxFileSize parameter to the desired size. Values provided should end in "MB" or "KB" to distinguish between megabytes and kilobytes.

Maximum Number of Backup Log Files

To change the number of backup files, change the value of the MaxBackupIndex parameter to the desired number of backup files.

Changing from Circular to Linear Logging

To use linear logging rather than circular logging, do the following:

1. In the appender class value, change RollingFileAppender to FileAppender. For example, in the previous example, change the first line to the following:

   ```
   <appender class="tvision.org.apache.log4j.FileAppender"
   name="SENSOR_LOGFILE">
   ```

2. Remove the entries for the MaxBackupIndex and MaxFileSize parameters.

Trace Logging

Trace logging provides verbose information of what a TransactionVision Analyzer is doing internally. It is used mainly to troubleshoot problems and should not be turned on in production environments.

To enable trace logging for the TransactionVision Analyzer, set the value of the trace property in the <TVISION_HOME>/config/services/Analyzer.properties file to on. After modifying this configuration file, you must restart the Analyzer for the change to take effect.

Using Windows and UNIX System Logs

On UNIX and Windows platforms, you can configure TransactionVision to log output to the system event logging facilities—the event log for Windows or syslog for UNIX. Examples of the logging configuration files needed to do this can be found in TVISION_HOME/config/logging/system/*/Sensor.Logging.xml.

For both Windows and UNIX, you must define a specialized event appender.

Windows Event Appender

The following example shows how to configure the Windows event appender to use the event log:

```
<appender name="NT_EVENT_LOG"
class="tvision.org.apache.log4j.nt .NTEventLogAppender">
  <layout class="tvision.org.apache.log4j.PatternLayout">
    <param name="ConversionPattern" value="%d [%t] - %m%n"/>
  </layout>
</appender>
```

NT_EVENT_LOG can then be referenced in a category definition of your choice. For example:

```
<category additivity="false" class="com.bristol.tvision.util.
log.XCategory" name="sensorLog">
  <priority class="com.bristol.tvision.util.log.XPriority"
value="info"/>
  <appender-ref ref="NT_EVENT_LOG"/>
</category>
```

On Windows, you must also add a special DLL to your path. This DLL, NTEventLogAppender.dll, can be found in the config\logging\system\bin directory. For example:

```
set path=%TVISION_HOME%\config\logging\system\bin;%PATH%
```

UNIX Event Appender

The following example shows a UNIX event appender to use syslog:

```
<appender name="SYSLOG" class="tvision.org.apache.log4j.net.
SyslogAppender">
  <param name="SyslogHost" value="localhost"/>
  <param name="Facility" value="local0"/>
  <layout class="tvision.org.apache.log4j.PatternLayout">
     <param name="ConversionPattern" value="[%t] %-5p %c %x
- %m%n"/>
  </layout>
</appender>
```

Specify the SyslogHost and Facility parameters as appropriate for your environment.

Enabling SMTP Logging

The SMTPAppender sends an email to the SMTP server when an error log event at the specified threshold reaches the appender. To enable the SMTPAppender, add the following to your Analyzer.logging.xml file:

```
<appender name="EMAIL"
class="tvision.org.apache.log4j.net.SMTPAuthenticateAppender"
>
    <param name="SMTPHost" value="smtp.myserver.net"/>
    <param name="To" value="analyzer_log4j@myserver.net"/>
    <param name="From" value="administrator@myserver.net"/>
    <param name="UserName" value="smtp_user"/>
    <param name="Password" value=""/>
    <param name="Authenticate" value="true"/>
    <param name="BufferSize" value="1"/>
    <param name="Threshold" value="info"/>
    <layout class="tvision.org.apache.log4j.PatternLayout">
       <param name="ConversionPattern" value="%d [%t] %-5p %c
%x - %m%n"/>
    </layout>
  </appender>
```

The threshold parameter specifies the logging level that is allowed to append into the SMTPAppender.

To define a customized triggering event evaluator, add the EvaluatorClass parameter:

```
<param name="EvaluatorClass" class="tvision.org.apache.
log4j.spi.TriggeringEventEvaluator"/>
```

This interface provides the following function for determining when an email should be sent:

```
public boolean isTriggeringEvent(LoggingEvent event) {
    long l = 0;
    synchronized(lock) {
      l = (msgCount ++);
```

```
    }
    return (((l + 1)%msgPkgSize) == 0); // fire email on every
msgPkgSize events.
  }
```

Enabling SNMP Logging

The JoeSNMPTrapSender appender sends email when a specified error level occurs. To enable JoeSNMPTrapSender, add the following definition to the Analyzer.logging.xml file:

```
<!-- SNMP TRAP appender !-->
<appender name="TRAP_LOG"
class="tvision.org.apache.log4j.ext.SNMPTrapAppender">
  <param name="ImplementationClassName"
value="tvision.org.apache.log4j.ext.JoeSNMPTrapSender"/>
  <param name="ManagementHost" value="127.0.0.1"/>
  <param name="ManagementHostTrapListenPort" value="162"/>
  <param name="EnterpriseOID" value="1.3.6.1.4.1.24.0"/>
  <param name="LocalIPAddress" value="127.0.0.1"/>
  <param name="LocalTrapSendPort" value="161"/>
  <param name="GenericTrapType" value="6"/>
  <param name="SpecificTrapType" value="12345678"/>
  <param name="CommunityString" value="public"/>
  <param name="ForwardStackTraceWithTrap" value="true"/>
  <param name="Threshold" value="DEBUG"/>
  <param name="ApplicationTrapOID"
value="1.3.6.1.4.1.24.12.10.22.64"/>
  <layout class="tvision.org.apache.log4j.PatternLayout">
  <param name="ConversionPattern"
value="%d,%p,[%t],[%c],%m%n"/>
  </layout>
</appender>
```

You must add joesnmp.jar to your CLASSPATH because it is required by JoeSNMPTrapSender. This JAR file can be downloaded from the JoeSNMP project at http://sourceforge.net/projects/joesnmp.

Enabling JMS logging

TransactionVision provides a mechanism to send log messages via a JMS messaging provider. This is done through the `com.bristol.tvision.appender.JMSAppender` appender configured in the `Analyzer.logging.xml`.

The JMS appender is also used by the TransactionVision integration with Business Process Insight (BPI). This can be configured in the analyzer logging configuration, but the recommended way of configuring BPI integration is through the TransactionVision integrations facility, through **Administration > Integrations.**

The JMS appender can be configured one of two ways. Typically you use JNDI settings to configure the JMS connectivity, but direct WMQ JMS configuration is also allowed.

Choose whether you are configuring using:

- JNDI - or
- Direct WMQ JMS.

If both methods are specified, the WMQ JMS settings will take precedence and the JNDI settings are ignored.

In order to manually configure the BPI JMS connectivity, or to configure a separate logging facility to publish logs through JMS queues, use the following JMSAppender options:

- *ConnectionRetryDelay* is the time before a retry is made if connection fails.
- *ConnectionRetryTimeout* is the time it will wait before an unresponsive connection times out.
- *QueueName* is the name of the JNDI object (if JNDI is used), or the actual name of the queue (in the case of WMQ JMS).
- *Username* and *Password* are optional settings if authentication to the JMS provider is required.

For JNDI settings

- *InitialContextFactoryName* is the classname of your JNDI context factory. This value will depend on which JMS vendor you use (see their documentation for details). Some examples are com.sun.jndi.fscontext.RefFSContextFactory, or com.tibco.tibjms.naming.TibjmsInitialContextFactory.
- *ProviderURL* is the url to connect to the JNDI repository, and depends on which JMS vendor you use. A RefFSContextFactory has a URL similar to file:/C:/jndi. For TIBCO, you might use something like tibjmsnaming://host:7222.
- *QueueConnectionFactoryName* is the name of the Queue Connection Factory JNDI object.

WMQ JMS settings

The WMQ JMS specific settings correspond to the queue manager name, host, port and channel that you are using to connect to WMQ JMS. TargetMQClient enables/disables whether MQ uses RFH2 headers in its JMS message.

```
<appender class="com.bristol.tvision.appender.JMSAppender"
name="JMS_APPENDER">

  <!--connection retry interval in millseconds -->

 <param name="ConnectionRetryDelay" value="0"/>

 <param name="ConnectionRetryTimeout" value="0"/>

 <param name="QueueName" value=""/>

 <param name="UserName" value=""/>

 <param name="Password" value=""/>

<!-- enable the following to provide JNDI context parameters for

    JMS connection -->

<!--<param name="InitialContextFactoryName" value="" />

<param name="ProviderURL" value="" />

        <param name="QueueConnectionFactoryName" value=""/>

-->
```

```
<!-- enable the following to provide WebSphere MQ parameters for
     JMS connection -->
<!--
<param name="MQQueueManagerName" value="" />
<param name="MQClientConnectionHost" value="" />
<param name="MQClientConnectionPort" value="" />
<param name="MQClientConnectionChannel" value="" />
<param name="TargetMQClient" value="false"/>
->


 </appender>
```

A Utilities Reference

CreateSqlScript

Location:

```
TVISION_HOME/bin/CreateSqlScript.[sh|bat]
```

Purpose:

Allow the user to create and optionally execute a SQL script to create, drop, import or export TransactionVision system tables or project tables.

The TransactionVision Analyzer and web application need to be stopped before performing any database imports or exports. This is to avoid causing the database import/exports to fail because of database locks held by the Analyzer or web application. If a schema is dropped, make sure that there are no active projects or users logged in that are using that schema.

Syntax:

```
CreateSqlScript
  {-create(-c) | -drop(-d) | -import(-i) | -export(-ex) |
{-system(-sys) | -schema(-s)SCHEMA | -table(-t) TABLE SCHEMA}
  [[-noscript(-n)] -execute(-e)]
  [-noprompt(-np)] [-noinsert(-ni)] (
  [-tablespace(-ts) TABLESPACE] [-dbMove(-m)]
  [-fileType(-f) IXF|DEL] [-lobPath(-lp) PATH]
  [-noLob(-nl)] [-dbproperties(-db) FILE]
```

Options:

Option	Description
-drop (-d)	Drop tables
-create (-c)	Create tables
-execute (-e)	Execute script
-noscript (-n)	No script generation
-system (-sys)	Create/drop system tables (schema TVISION)
-schema (-s) SCHEMA	Create/drop project tables in schema SCHEMA
-table (-t) TABLE SCHEMA	Create/drop table TABLE in schema SCHEMA
-tablespace (-ts) TBSPC	Use tablespace TBSPC
-dbproperties (-db) FILE	Use Database.properties file FILE
-import (-i)	Generates a database import script. To run database scripts, use the command db2 -n -t -f <sql script filename>. For a DB2 database, you can combine this option with the -fileType and -lobPath options to customize the data format and the location of LOB. You may NOT combine this option with the -noscript or -execute options.
-export (-ex)	Generates a database export script. To run database scripts, use the command db2 -n -t -f <sql script filename>. For a DB2 database, you can combine this option with the -fileType and -lobPath options to customize the data format and the location of LOB. You may NOT combine this option with the -noscript or -execute options.
-resetseq (-r)	Resets the sequence start number to match to match the imported data.
-dbMove (-m)	Display the corresponding db2move command for importing/exporting a schema instead of generating an IMPORT/EXPORT script (only in conjunction with -import/-export).

Option	Description
-noprompt (-np)	Do not prompt for confirmation when dropping database tables.
-noinsert (-ni)	Do not insert any initial table rows defined in the XDM file when creating database tables.
-fileType (-f) IXF | DEL	Specify the DB2 data output file format used for importing/exporting data. The default type is IXF. For the IXF file type, the import/export script uses the LOBFILE option for rows that contain greater than 32K data. For the DEL type, the import/export script exports LOBFILEs into a single file (requires FixPack 8).
-lobPath (-lp) PATH	Specifies the directory for DB2 LOBFILEs. The default value is the current directory.
-noLob (-nl)	Do not generate DB2 export/import SQL with LOBINFILE option. This option will truncate LOB data to the first 32K bytes.

Examples:

1 Create system tables with schema as TVISION and execute the procedure without generating SQL script:

```
CreateSqlScript –e –n –c –sys
```

2 Generate SQL script for creating project tables with schema as PROJECT without executing the procedure:

```
CreateSqlScript –c –s PROJECT
```

3 Drop table EVENT in schema PROJECT and execute the procedure without generating SQL script:

```
CreateSqlScript –e –n –d –t EVENT PROJECT
```

DB2RunStats

Location:

```
TVISION_HOME/bin/DB2RunStats.[sh|bat]
```

Description:

Updates statistics about the physical characteristics of a table and the associated indexes. These characteristics include number of records, number of pages, and average record length. The optimizer uses these statistics when determining access paths to the data.

This command is called when a table has had many updates, such as when data is continuously collected into DB2 by the TransactionVision Analyzer. It could result in large performance gains in queries made by TransactionVision views and reports, as well as queries made internally by the TransactionVision Analyzer to correlate events.

This script can be set up to run as a scheduled batch job using either the UNIX **`cron`** facility or the Windows scheduler.

While this script is running, TransactionVision Analyzer processing slows down.

This script typically should be run daily, though the frequency of execution could be higher for higher message rates.

This script needs to be customized based on your system to set the correct DB2 installation location and the correct TVISION_HOME location.

If the user has additional tables defined for the project schema, new RUNSTATS statements must be added to cover the additional tables.

Syntax:

```
DB2RunStats user_name passwd database_name schema_name [-v7]
```

Options:

Option	Description
user_name	The user account that has privilege to execute RUNSTATS and make database connections
passwd	The password associated with user_name
database_name	The name of the database to connect to
schema_name	The name of the schema that the project reads and writes data to
-v7	Use in a DB2 7.x environment. The default operation is for DB2 8.1.

DB2Test

Location:

```
com.bristol.tvision.admin.DB2Test
```

Description:

Measures DB2 database INSERT performance.

The utility inserts sample event data into the RAW_EVENT table of the specified schema. Before running the test, create a new project schema with CreateSqlScript (which can be deleted after running the test). See the *TransactionVision Planning Guide* for details on how to set up the required test environment

Syntax:

```
java com.bristol.tvision.admin.DB2Test databaseName user
passwd schema eventCount eventSize threadCount {-commit
n}{-jdbcBatch}
```

Options:

Option	Description
databaseName	Name of the DB2 Database that contains the schema to be used for the test
user	DB2 user name
passwd	DB2 password
schema	TransactionVision schema in which sample event data will be saved
eventCount	Number of events to generate

Option	Description
threadCount	Number of threads to use to generate events
-commit n	Executes every n insert statements as part of a single batch operation. Default is to commit each insert individually.
-jdbcBatch	Use JDBC standard batching

MigrateConfig

Location:

```
TVISION_HOME/bin/MigrateConfig.[sh|bat]
```

Description:

This is an internal script called during TransactionVision installation to migrate configuration files from an older version of TransactionVision to the current version.

Do NOT call this script directly; it may only be run during installation.

MigrateDB

Location:

```
TVISION_HOME/bin/MigrateDB.[sh|bat]
```

Description:

Migrates project database files from an older version of TransactionVision to the current version. This script has to be run after an upgrade installation process. It must be run in a configured TransactionVision environment; the `Database.properties` must be set correctly for communication with the database.

Syntax:

```
MigrateDB
```

OracleRunStats

Location:

```
TVISION_HOME/bin/OracleRunStats.[sh|bat]
```

Description:

Oracle 10g has a built-in job scheduler that automatically collects statistics. You do not need to run this script for version 10g.

Updates statistics about the physical characteristics of a table and the associated indexes. These characteristics include number of records, number of pages, and average record length. The optimizer uses these statistics when determining access paths to the data.

This command is called when a table has had many updates, such as when data is continuously collected into Oracle by the TransactionVision Analyzer. It could result in large performance gains in queries made by TransactionVision views and reports, as well as queries made internally by the TransactionVision Analyzer to correlate events.

This script can be set up to run as a scheduled batch job using either the UNIX **`cron`** facility or the Windows scheduler.

While this script is running, TransactionVision Analyzer processing slows down.

This script typically should be run daily, though the frequency of execution could be higher for higher message rates.

Run this script from a user account that has privileges to perform database operations. This script must be run under an Oracle user account.

This script needs to be customized based on your system to invoke the correct environment initialization script like .profile or .bashrc, and to set the correct Oracle installation location and the correct TVISION_HOME location. Additionally, the file OracleRunStats.sql must be in the current directory when running this script.

Syntax:

```
OracleRunStats user_name passwd database_name schema_name
```

Options:

Option	Description
user_name	The Oracle user account to run the script under
passwd	The password associated with user_name
database_name	The name of the database to connect to
schema_name	The name of the schema that the project reads and writers data to

OracleTest

Location:

```
com.bristol.tvision.admin.OracleTest
```

Description:

Measures Oracle database INSERT performance.

The utility inserts sample event data into the RAW_EVENT table of the specified schema. Before running the test, create a new project schema with CreateSqlScript (which can be deleted after running the test). See the *TransactionVision Planning Guide* for details on how to set up the required test environment

Syntax:

```
java com.bristol.tvision.admin.OracleTest databaseName host
port user passwd schema eventCount eventSize threadCount
[-VARCHAR | -BLOB |
-LONGRAW] {-commit n} {-jdbcBatch} {-OracleBatch} {-thin}
{-parallel} {-url URL}
```

Options:

Option	Description
databaseName	Name of the Oracle database that contains the schema to be used for the test.
host	Name of host system on which the Oracle server exists
user	Oracle user name
passwd	Oracle password
schema	TransactionVision schema in which sample event data will be saved
eventCount	Number of events to generate
eventSize	Size of event user data buffer (default is 1024 bytes)

Option	Description
threadCount	Number of threads to use to generate events
-VARCHAR	Use this option if the RAW_EVENT table has been created with a VARCHAR column definition.
-BLOB	Use this option if the RAW_EVENT table has been created with a BLOB column definition.
-LONGRAW	Use this option if the RAW_EVENT table has been created with a LONGRAW column definition.
commit n	Executes every *n* insert statements as part of a single batch operation. Default is to commit each insert individually.
-jdbcBatch	Use JDBC standard batching.
-oracleBatch	Use Oracle update batching
-thin	User thin client driver. Default is to use oci client driver.
-parallel	Use the Oracle INSERT PARALLEL option instead of the standard INSERT INTO.
-url URL	By default, OracleTest will use an appropriate JDBC URL for thin or oci client drivers. However the default may be overwritten by specifying the JDBC URL here.

ServicesManager

Location:

```
TVISION_HOME/bin/ServicesManager.[sh|bat]
```

Purpose:

Manage the TransactionVision Analyzer service. The Analyzer uses an embedded RMI registry so that Analyzers may be controlled by the TransactionVision web user interface running on remote hosts.

Syntax:

```
ServicesManager
 {(-start [-project (-proj) PROJECTNAME]) |
  (-stop [-quiesce][-project (-proj) PROJECTNAME]
[-keepcollect]) |
  (-exit [-quiesce][-keepcollect]) |
  (-status [-project (-proj) PROJECTNAME]) |
  (-killserver) |
  (-reconfig {classification | logging | analyzer}) |
  (-versioninfo)
  ([-host HOST][-rmiregp PORTNUMBER][-debug])
```

Options:

Option	Description
-start	Starts the Analyzer process if it is not already running
-stop	The Analyzer stops collecting event data
-exit	The Analyzer stops collecting event data, then the process exits
-quiesce	The default behavior of the Analyzer on a stop or exit is to immediately close down collection. If this flag is set, the Analyzer clears out any pending events in the event queue before stopping or exiting. Note that if there is a large event backlog, the Analyzer may take some time to stop or exit.

Option	Description
-keepcollect	Do not send stop message to Sensor. Used in combination with the –stop or –exit option so that the Sensor keeps collecting.
-status	Reports the current Analyzer status
-reconfig	Reloads Analyzer configuration settings (classification/ logging/all settings) without stopping event collection. • **classification** – Re-initializes the classification rule engine and reloads the Analyzer classification files reading in any changes that may have been made to that file. • **logging** – Reloads the log4j XML configuration files in the directory <TVISION_HOME>/config/logging. • **analyzer** – Re-initializes the Analyzer by reloading most Analyzer settings in the Analyzer.properties file, all beans described in the Beans.xml file, re-initializing logging and all XML based rule files. Settings in the"Collection Properties" section of the Analyzer.properties file, except the write_to_buffer_table property, are not reloaded by the Analyzer with the -reconfig option.
-versioninfo	Returns Analyzer version information
-killserver	Shuts down the Analyzer immediately. This flag is not recommended; -exit is the preferred method for shutting down the Analyzer cleanly.
-project (-proj) PROJECTNAME	Name of the project for the specified command. If the project name contains a space, enclose the project name in double quotation marks (for example, -proj “Project Name”). The project must have a communication link in order for the Analyzer to process events. Only the -start, -stop, and -status commands be performed on a single project. This option cannot be used in combination with the -host or -rmiregp options. When the -project option is used, the Analyzer host and port is looked up from the database.

Option	Description
-host HOST	Name of the host where the Analyzer runs. Can be either name or IP address. Local host is used if not specified.
-rmiregp PORTNUMBER	Port number on which the Analyzer listens for RMI connections. The default value is the value specified by the analyzer_port property in the Analyzer.properties file. This option only takes effect when communicating with an Analyzer that is already running; the Analyzer always uses the value specified in Analyzer.properties when starting up.
-debug	Start the Analyzer process in debug mode.

Examples:

1 Start project PROJECT:

```
ServicesManager –start –proj PROJECT
```

2 Stop Analyzer on host HOST:

```
ServicesManager –stop –host HOST
```

SQLServerTest

Location:

```
com.bristol.tvision.admin.SQLServerTest
```

Description:

Measures SQL Server database INSERT performance.

The utility inserts sample event data into the RAW_EVENT table of the specified schema. Before running the test, create a new project schema with CreateSqlScript (which can be deleted after running the test).

Syntax:

```
java com.bristol.tvision.admin.SQLServerTest databaseName
user passwd schema eventCount eventSize threadCount {-commit
n}{-jdbcBatch}
```

Options:

Option	Description
databaseName	Name of the SQL Server Database that contains the schema to be used for the test
user	SQL Server user name
passwd	SQL Server password
schema	TransactionVision schema in which sample event data will be saved
eventCount	Number of events to generate
threadCount	Number of threads to use to generate events
-commit n	Executes every n insert statements as part of a single batch operation. Default is to commit each insert individually.
-jdbcBatch	Use JDBC standard batching

B Configuration Files

The TransactionVision setup utilities save Analyzer configuration information necessary for TransactionVision to operate in the following configuration files. You may also modify these files directly if you need to make any changes to your configuration.

If you modify property files, you must restart the associated application for you changes to take effect. To restart the Analyzer on Windows, use the Windows Services management control panel.

Analyzer.properties

The <TVISION_HOME>/config/services/Analyzer.properties file provides general, collection, and analysis configuration information for the TransactionVision Analyzer.

General Properties

- rmi_client_timeout

  This entry specifies the timeout for the RMI client in minutes. The default value is 2.

- logging_xml

  Specifies the name of the logging configuration file used by the Analyzer. The default value is Analyzer.Logging.xml.

- trace

  This property specifies whether Analyzer trace logging is on or off. The default value is ON.

- debug
- This property specifies whether Analyzer debug logging is on or off. The default value is OFF. Enable this setting only if advised to do so by support.
- service_jvm_flags

  Specifies any additional JFM flags you want the Analyzer to run with.
- service_additional_classpath

  Specifies the classpaths for any custom beans to be used with your Analyzer.
- jvm_dll
  service_classpath
  service_libpath

  On Windows, these properties are automatically generated by TVisionSetupInfo. It should not be necessary to change them. If you do change them, note that they will be reset if TVisionSetupInfo is run again. These properties have no effect on the TransactionVision environment other than the Analyzer running as a Windows service.
- analyzer_port

  This entry specifies the default port number that the Analyzer runs on. The default value is 1099. The Analyzer always uses this port number when it starts. The ServicesManager.[bat|sh] -rmiregp command can be used to set the port number for an Analyzer that is already running.
- rmi_server_port

  This entry specifies the default port number that the RMI server object is listening to.

  The default value is 0, which means that the java runtime will use an arbitrary, random port whenever the analyzer starts. If you are running the analyzer behind a firewall and require a fixed port then you can set this property to a specific port value.
- time_server

  Set to on to run the time server within the Analyzer. The time server is required for TIBCO EMS and generic JMS communication link types to calculate the time skew information. To run the time server on a different host, use the TimeServer.[sh|bat] utility.

- time_server_port

  Specifies the listening port for the time server to use, if time_server is set to on. The default port is 9037.

Collection Properties

The following properties are used for the event collection operations of the Analyzer:

- batch_commit_count

  Specifies the number of events to batch before a commit is issued. The default value is 50.

- commit_time_threshold

  Specifies the amount of time, in seconds, after which a commit is forced. The default value is 1.

- commit_byte_threshold

  Specifies the number of bytes after which a commit is forced. The default value is 1,000,000.

- write_to_buffer_table

  Specifies whether to write a copy of raw events from the queue into the RAW_EVENT table, in addition to normal processing. The default value is False. The setting is global to all projects, opposed to the corresponding setting on the Communication Link page in the UI which allows to enable raw event writing for a specific Communication Link. This property should never be set to True if read_from_buffer_table is also set to True. Use this property to test custom beans, or if advised by support to troubleshoot issues. It should always be set to False in a production environment.

- read_from_buffer_table

  Specifies whether to pull any events from the RAW_EVENT table in addition to reading from the queue. The default value is False. This property should never be set to True if write_to_buffer_table is also set to True. Use this property to test custom beans, or if advised by support to troubleshoot issues. It should always be set to False in a production environment.

- keep_events

  Specifies whether the Analyzer should delete raw events after they have been successfully processed. Set it to true to keep the raw events in the RAW_EVENT table. Their event_status column will be set to PROCESSED so that they are not processed again. If you want to process the events again (for example, with a custom Java bean for a different type of analysis), you must first set the event_status to NEW. Set it to false (the default value) for the Analyzer to delete raw events from the RAW_EVENT table after it processes them.

- write_to_jar

  Similar to write_to_buffer_table, but the analyzer stores the raw events in a JAR file instead of a table in the database. The location of the JAR file is $TVISION_HOME/logs, and the file name will be `SCHEMA_raw_events.jar`. For each project schema a separate JAR file is created. You must stop the project in order to close the JAR file correctly.

- read_from_jar

  Similar to read_from_buffer_table, but the analyzer reads the raw events from a JAR file instead of a table in the database. For a project with database schema SCHEMA the analyzer will look for a file `SCHEMA_raw_events.jar` in the $TVISION_HOME/logs directory. Note that you can copy the jar file while the analyzer is running, it will periodically scan the directory for the corresponding files. Once the analyzer starts processing the events in the JAR file it is renamed to `SCHEMA_raw_events.jar.processed.TIMESTAMP`. Note that you should never set this setting to True if write_to_jar is also set to True.

- fail_safe_collection_shutdown

  If true, causes the Analyzer to stop sending configuration messages if a serious failure occurs in event collection and processing. The default value is true.

- jdbc_batching

  The default jdbc_batching value (on) causes TransactionVision to execute database statements in batch mode.

- jdbc_batch_count

  Specifies The number of SQL operations to batch in JDBC batching mode. Note that this number should be equal to or a multiple of the batch_commit_count property. The default value is 50.

Analysis Properties

The following properties are used for the event analysis operations of the Analyzer:

- save_event_document

  Specifies whether to save event documents in the database. The default is on. If this property is set to off, event detail is not available. Furthermore, the Analyzer will not be able to recover asynchronously flushed transaction and static topology data. This means that in the event of a crash or other abnormal program termination, transaction data and statistics data for the static topology view may no longer be accurate.

- eventmatching_interval

  Specifies the time in seconds to wait between every invocation of event matching. The default is value is 600.

- partial_event_lifetime

  The maximum time in minutes a partial event entry may exist in the partial_event table. When this time limit is reached, the partial event will be flushed. The default value is 10 minutes.

- latency_resolution

  The resolution used to calculate latency between events. Possible values are:

Value	Description
1	Seconds
10	1/10 seconds
100	1/100 seconds (default)
1000	Milliseconds

- generate_api_only_txns

  By default the analyzer will not perform any transaction analysis on events that have been collected with a DataCollection filter "API only". Set this property to true to change this behavior.

- correlation_limit

This setting limits the number of event relationships created by the analyzer for a specific correlation key and can be used as a safeguard against flawed correlation rule design in custom correlation, leading to performance issues. The default value is -1 (no limit).

- local_txn_limit
- This setting limits the number of local transactions assigned by the analyzer to one business transaction, and can be used as a safeguard against flawed local transaction rule design in custom local transaction analysis, leading to performance issues. The default value is -1 (no limit).
- separate_child_thread_txns

  By default, if a servlet spins off a thread to make some JMS calls, the servlet passes tracking information to the child thread. The result is that both the servlet and JMS events belong to the same business transaction. However, there may be some cases in which you wish to separate these events into different transactions. For example, a servlet may spin off a long-running thread that you do not want to be part of the same transaction as the servlet. If you do not want threads spun off by a servlet to be included in the same business transaction as the servlet, list the servlet program name as the value for this property. Separate multiple program names with a comma, as in the following example:

  ```
  separate_child_thread_txns=program1, program2
  ```

- enable_dbcaching

  Enables standard mode in addition to or instead of failure mode. If all of your projects only use Failure Mode, the Standard Mode can be disabled to save some system resources.

- enable_failure_mode

  Enables failure mode so that the Analyzer will only store event data of failed business transactions (or transactions violating SLA). For successful business transactions, only the corresponding business transaction rows will be saved. Since this mode relies heavily on a meaningful value of the ‘result’ attribute of the business transaction, it requires a transaction classification with suitable rules for the ‘result’ column to be in place. The failure mode can be enabled on a per-project basis, allowing it to run certain projects in failure mode while running others in the conventional Standard Mode. The default value is off. For more information about failure mode, see the *TransactionVision Administration Guide*.

- dbcache_thread_count

  Specifies the number of threads to use for writing the cached analysis data to the database. The best value is dependent on the hardware the Analyzer is running on and can only be determined by performance measuring with different values, but a good starting point would be to set this number to half the number of collection threads used for a particular schema. The default value is 2.

- failure_mode_thread_count

  Defines the number of flushing threads to use for Failure Mode. This number should be set high enough so that the cache data can be flushed fast enough, but low enough to not waste too many system resources. The default value is 2.

- failure_mode_process_delay

  Defines the amount of time, in milli-seconds, which gets added to the SLA value for determining the age out timeout and is a crucial value for the Failure Mode. If this value is set too low, a lot of successful transaction data will unnecessarily age out and be written to the database, which will result in poorer performance, and wasted database disk space. If this value is set too high, too much data will be held in the memory caches. If the cache sizes are not large enough for the system load, cache overflows will occur frequently and lead to very poor performance The default value is 5 seconds.

- failure_mode_discard_overflow

  In case of a cache overflow all the data in the caches will normally be written to the database. If the overflow was caused, for whatever reason, by a longer lasting system slowdown, the constant flushing of the cache could further worsen the situation and slow down the Analyzer even more. If the main concern is to avoid any chance of filling up the event queue in such situations, you can set the above parameter to true and force the discarding of all cache data in case of an overflow. The default value is false.

CacheSize.properties

The <TVISION_HOME>/config/services/CacheSize.properties file specifies the size of the caches used in the event analysis. A bigger cache size minimizes database access and increases performance, but also increases the amount of memory used by TransactionVision. It sets the following properties:

- system_model_objects

  Specifies the number of system model object other than PII cached.

- pii_objects

  Specifies the number of PII system model objects cached.

- event_based

  Specifies the number of events cached.

- transaction_based

  Specifies the number of transactions cached.

Database.properties

The `<TVISION_HOME>/config/datamgr/Database.properties` file specifies the database the Analyzer and the TransactionVision web application read from and write to.

Required Entries

The following mandatory entries are required:

- jdbc_driver=COM.ibm.db2.jdbc.app.DB2Driver

  This entry is the class name of the JDBC driver used. The default for the IBM DB2 JDBC driver is as above. For Oracle, the default is oracle.jdbc.driver.OracleDriver. For SQL Server, the default is com.microsoft.jdbc.sqlserver.SQLServerDriver

- database_connection_name

  For DB2, this entry is the name of the database connection to be used (typically the database alias). For Oracle, this entry is the database name for an oci client connection or the SID (system identifier) for a thin client connection. For SQL Server, this entry is not used.

- database_name

  For DB2, this entry is the name of the database on the server to connect to. This name may be different from the "database_connection_name" if a client database connection is used. For Oracle, this entry is the SID. For SQL Server, this is the name of the database.

- database_host

  This entry is the host the database server is running on.

- database_port

  For Oracle, this entry specifies the port number for the Oracle listener on the target host. The default value is 1521. This attribute is only used if you are using the Oracle thin client. For more information on using Oracle with TransactionVision, see Chapter 6, Configuring Databases. For SQL Server, the entry specifies the port to connect to on the server, the default value is 1433.21."

- oracle_client_type

  Specifies whether to use the Oracle thin or client JDBC driver. Set it to thin (recommended) or oci for Oracle 9 or oci8 for Oracle 8.1.7. The default is thin. This attribute is only used if you are using an Oracle database. For more information on using Oracle with TransactionVision, see Chapter 6, Configuring Databases.

- oracle_user_password

  This entry specifies the user password that will be set for the Oracle user which gets created for a new project schema. The default value is ABC99DEF. You can change this value if you need to conform to specific password policies.

- db2_instance_env

  This entry specifies the value of the DB2 environment variable DB2INSTANCE.

Optional Entries that Override Automatic Detection

The following optional entries may be setup in this file. These entries override automatic detection.

- connection_type

  This entry defines how to obtain JDBC connections. Use one of the following values:

- JDBC: Uses connections obtained from the JDBC driver, with no driver connection pooling (default).
- DB2DataSource: Uses DB2 connection pooling.
- OracleDataSource: Uses Oracle connection pooling.
- JNDI: Uses the data source registered in JNDI. To use a JNDI connection, you must also set the jndi_url property.
- database_url

  This entry defines a custom URL to use with the JDBC driver manager.

- jndi_url

  This entry defines the JNDI name for the database (defined in WebSphere) when using JNDI connections.

- unicode_db

  If this property is set, all character-based XDM columns with the attribute unicode=true will be generated with double the byte size to allow the specified number of characters to be stored in the database.

- unicode_bytes_per_character

  If this property is set, all character-based XDM columns with the attribute unicode_true will be generated with a size using this value as a multiplier of the base value. The default value of 2 matches the behavior of setting unicode_db=true and not setting this value. Values larger than 3 may cause database creation problems. This property will not take effect unless unicode_db=true is also set.

- jdbc_url

  This entry specifies a database driver to use other than the default DB2 Universal JDBC driver (type 4) or the Oracle thin client driver.

  For the DB2 Universal Driver (type 2), this entry has the following format:

```
jdbc:db2:<database-name>
```

where <database-name> is the database to which the Analyzer connects to.

For the Oracle oci driver, this entry has the following format:

```
jdbc:oracle:oci:<user>/<password>@<database-name>
```

Where <database-name> is the database to which the Analyzer connects to, and <user>, <password> are the user name and passwords required for the Oracle connection.

Optional Entries with Default Values

The following optional entries may be setup in this file. If they are not specified, default values are used.

- user

  This optional entry is the user name to be used while making the database connection. If this field is empty, the currently logged in user is used to make the database connection. Make sure that the user specified or currently logged has privileges for database access.

- passwd

  This optional entry is the password to be used while making the database connection. If this field is empty, the currently logged in user's password is used to make the database connection.

- jce_provider

  This optional entry specifies the Java Cryptographic Extension (JCE) package name to encrypt the database password. The following table shows the valid package names. If this entry is blank, TransactionVision stores the password as plain text.

Provider	Package Name
Sun	com.sun.crypto.provider.SunJCE
IBM	com.ibm.crypto.provider.IBMJCE

The TVision_SetupInfo utility automatically searches for a an installed JCE provider that supports the DES encryption algorithm. If such a provider is found, TVision_SetupInfo sets the jce_provider value to the class name of the JCE provider. If no JCE provider is found, TVision_SetupInfo displays the following message:

```
Java Cryptography Extension (JCE) is not present in your
current JDK. Password encryption feature will be disabled
and stored in plain text.
```

- reconnect_interval

  This entry specifies the frequency in seconds that TransactionVision should try to reconnect with the database. The default value is 10.

- reconnect_timeout

  This entry specifies the amount of time in seconds that TransactionVision should try to reconnect to the database (at the reconnect_interval). The default value is 600.

- query_timeout

  This field specifies the timeout in seconds, for queries run in the UI. By default this setting turned off.

License.properties

The `<TVISION_HOME>/config/license/License.properties` file specifies the TransactionVision license code supplied by HP.

Performance.properties

The `<TVISION_HOME>/config/services/Performance.properties` file contains the following settings for performance logging:

- performance

  Specifies whether to performance logging is on or off. The default is off.

- count_interval

Specifies the number of events after which performance data is logged. The default is 500.

- detail

  Specifies whether to generate detailed performance and statistics logs. The default is off.

- start_at

  Defines how many events have to get processed before the performance logging will start. The default is 500.

- get_queuedepth

  Specifies whether to retrieve the current event queue depth for each log interval. The default is off.

Setup.properties

The `<TVISION_HOME>/config/setup/Setup.properties` file specifies the following properties:

- default_tool_install_path

  The directory location of the DefaultInstallPath.xml file, which lists the locations of software components required by TransactionVision

- logs_dir

  The name of the directory in which to store log files

- logging_xml

  The name of the logging configuration file

- minimum_java_version

  The minimum Java version required by TransactionVision

- minimum_java_version_sun

  The minimum Java version required by TransactionVision on the Solaris platform

- maximum_java_version

  The highest Java version supported by TransactionVision

C Database Migration

Time and space requirements

TransactionVision version 7.50 contains major changes to the database schema that have a large impact on the time and resources required by the migration process. Most of the table data of a pre-7.50 project will need to be copied during the migration. Due to the fail-safe implementation of the migration process this requires an additional amount of free space equal to the amount of space used for the largest existing project in the database. For example, if your largest TransactionVision project is using 2 GB of space, the migration will need an additional 2 GB to finish successfully. Also, depending on the amount of data in your database the migration process can take a considerable amount of time to complete.

Disabling unused integration columns

The new database schema contains some new table columns to be used for the integration of TransactionVision with other products in the BAC suite. If you are operating TransactionVision in a performance critical environment and do not intend to use any of the integrations, you can avoid creating those new columns in the migration process. Run the 'MigrateDB' script (see Appendix A, Utilities Reference) with the following command line option:

```
MigrateDB.sh|.bat  -disableMigration,
```

Migration of customized database schemas

If you have added custom XDM definition files in the previous TransactionVision installation, these files will be preserved in the new configuration. But you must change the 'proginst_id' key field from INTEGER to BIGINT for any custom event tables, and the "business_trans_id" key field from INTEGER to BIGINT for any custom transaction tables. If you added or modified columns in the standard XDM files, those changes will not get preserved in the new installation. Choose to save the old configuration files at setup time and make the necessary updates after installation.

Database migration - technical details

The following list contains all changes to the database schema since TransactionVision version 5.00:

TVISION system schema

- Table SCHEMA_VERSION: renamed column schema -> schema_name.
- Table ID_TABLE: renamed column key -> key_name.
- Renamed table SCHEMA -> SCHEMA_TABLE.
- Added table CLASSIFICATION.
- Added table CLASSIFICATION_REL.
- Added table TXN_CLASS_ATTRIBUTE.
- Added table TXN_CLASS_ATTR_VALUE.
- Added table PROPERTIES.

Project schema

- Table ID_TABLE: renamed column key -> key_name.
- Table SCRATCH: renamed column key -> key_name.

- Changed type of column 'proginst_id' from INTEGER to BIGINT for the following tables: PARTIAL_EVENT, EVENT, EVENT_OVERFLOW, USER_DATA, USER_DATA_OVERFLOW, RELATION_LOOKUP.
- Changed type of column 'proginst_id' from INTEGER to BIGINT in all tables defined via xdm files in $TVISION_HOME/config/xdm with documentType="/Event".
- Changed type of column 'local_trans_id' and 'business_trans_id' from INTEGER to BIGINT in table LOCAL_TRANSACTION.
- Changed type of column 'local_trans_id' from INTEGER to BIGINT in table TRACKING_OVERFLOW.
- Changed type of column 'business_trans_id' from INTEGER to BIGINT in table BUSINESS_TRANSACTION.
- Changed type of column 'proginst_id' and 'proginst_id2' from INTEGER to BIGINT in table EVENT_RELATION.
- Changed type of column 'object_id' from INTEGER to BIGINT in table SYS_MDL_OBJECT.
- Changed type of column 'object_id' and 'object_id2' from INTEGER to BIGINT in table SYS_MDL_OBJECT_RELATION.
- Table LOCAL_TRANSACTION: renamed column key -> key_name.
- Table BUSINESS_TRANSACTION: renamed column sequential_id -> update_id and changed column type from INTEGER to BIGINT.
- Table EVENT_LOOKUP: renamed column sequential_id -> seq_id and changed column type from INTEGER to BIGINT.
- Table TRANSACTION_STATS: renamed column begin-> begin_time and end -> end_time.
- Table JMS_LOOKUP: added column exception_class (INTEGER).
- Table BUSINESS_TRANSACTION: added column exception (INTEGER).
- Added the following columns for integration purposes to SERVLET_LOOKUP: probe_id, probe_group_id, uri, bpm_profile_id, bpm_location_id, bpm_txn_id, client_ip, session_id, url (data types can be found in $TVISION_HOME/config/xdm/Servlet.xdm).
- Added the following columns for integration purposes to BUSINESS_TRANSACTION: trans_id, is_bpievent (data types can be found in $TVISION_HOME/config/xdm/Transaction.xdm).

- Changed unique index (tracking_id, tracking_seq) on table TRACKING_OVERFLOW to non-unique.
- Added table JDBC_LOOKUP.
- Added table JDBC_STATS.
- Added table SYS_MDL_OBJECT_ATTR.

Optimizing TV in non-integration environments

TransactionVision 7.50 introduces several new project table columns for integrating with other products in the BAC product family. If you are not using any of the integrations, NULL values are written into these columns during analyzer processing. If your database product is updating index files even for NULL values it might be possible that this has a small impact on the analyzer performance. So if you are running the TransactionVision analyzer in a high-performance environment and do not intend to integrate with other products, we suggest to disable the integration columns as follows:

- Run `TVisionSetupInfo.bat|.sh` from $TVISION_HOME/bin with the command line switch `-integrations`.
- Answer "**n**" to the following questions:

```
------------------------------------------------------------

 Integration Settings

------------------------------------------------------------

Retrieving current integration settings...

Notice:

  If you disable integrations, projects created afterwards may
not work properly

  as the database columns won't exist. It's safer to enable them
now and just

  disable them in the UI later.

IntegrationEnableBAC (y/n) [n]:

IntegrationEnableDiag (y/n) [n]:
```

This disables all integration columns in the XDM files (currently `Servlet.xdm` and `Transaction.xdm`). For new project schemas, the tables will be created without those columns. For existing project schemas, the columns will remain in the tables, but the analyzer will no longer access them. You can change this setting any time by running `TVisionSetupInfo` with the corresponding command line switch again.

Index

I

J

L

M

N

O

P

R

S

T

U

W


invent


Java™
COMPATIBLE