HP Business Service Management

for the Windows and Linux operating systems

Software Version: 9.12

TransactionVision Planning Guide Version 9.10

Document Release Date: November 2011

Software Release Date: November 2011

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.

Adobe® and Acrobat® are trademarks of Adobe Systems Incorporated.

AMD and the AMD Arrow symbol are trademarks of Advanced Micro Devices, Inc.

Google™ and Google Maps™ are trademarks of Google Inc.

Intel®, Itanium®, Pentium®, and Intel® Xeon® are trademarks of Intel Corporation in the U.S. and other countries.

iPod is a trademark of Apple Computer, Inc.

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

Microsoft®, Windows®, Windows NT®, Windows® XP, and Windows Vista® 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 Open Group.

Acknowledgements

This product includes software developed by the Apache Software Foundation (http://www.apache.org). This product includes software developed by the JDOM Project (http://www.jdom.org). This product includes software developed by the MX4J project (http://mx4j.sourceforge.net).

Welcome to This Guide

Welcome to the TransactionVision Planning Guide. This guide contains important information for sizing and planning new installations of TransactionVision.

Who Should Read This Guide

This guide is for the following users of TransactionVision:

- Application developers or specialist familiar with the application being monitored.
- System administrators familiar with the application infrastructure components such as application servers and messaging queues.
- Database administrators for the TransactionVision database.

About TransactionVision Documentation

TransactionVision documentation provides information on using the Transaction Management application of HP Business Service Management (BSM) and deploying and administering the TransactionVision components in the BSM deployment environment.

This documentation set includes:

- *TransactionVision Deployment Guide*. Describes the installation and configuration of the TransactionVision components in the BSM deployment environment. This guide is available as a PDF in the BSM Online Documentation Library or in the Processing Server install directory.
- *Using Transaction Management Guide*. Describes how to set up and configure TransactionVision to track transactions and how to view and customize reports and topologies of business transactions. This guide is available as the Transaction Management Portal or as a PDF in the BSM Online Documentation Library.
- *TransactionVision Planning Guide*. Contains important information for sizing and planning new installations of TransactionVision. This guide is available as a download from the HP Software Support Web site.
- *TransactionVision Advanced Customization Guide*. Contains information for how the TransactionVision platform can be extended and customized to achieve further control over its various functions. This guide is available as a PDF in the Transaction Management Portal in the BSM Online Documentation Library. Choose TransactionVision Administration > Advanced Customization Using the APIs topic.

Additional Transaction Management documentation includes:

- **Release Notes**. Provides a list of version limitations and last-minute updates. You can access the most updated release notes file from the product DVD or from the HP Software Support Web site.
- **What's New**. Provides a list of new features and version highlights. In HP Business Service Management, select **Help > What's New**.
- **Online Documentation Library**. The Documentation Library is an online help system that describes how to work with HP Business Service Management and the Transaction Management application. To access the Documentation Library in HP Business Service Management, select **Help > Documentation Library**. Context-sensitive help is available from specific HP Business Service Management pages by selecting **Help > Help on this page** and from specific windows by clicking the **Help** button. For details on using the Documentation Library, see "Working with the HP Business Service Management Documentation Library" in *Platform Administration*.

Support

Visit the HP Software Support web site at:

www.hp.com/go/hpsoftwaresupport

This Web site provides contact information and details about the products, services, and support that HP Software offers.

HP Software online support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support web 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

Troubleshooting & Knowledge Base accesses the Troubleshooting page on the HP Software Support Web site where you can search the Self-solve knowledge base. Choose Help > Troubleshooting & Knowledge Base. The URL for this Web site is *http://h20230.www2.hp.com/troubleshooting.jsp*.

Most of the support areas require that you register as an HP Passport user and sign in. Many also require a support contract. To register for an HP Passport ID, go to:

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

To find more information about access levels, go to:

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

To register for an HP Passport user ID, go to:

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

Contents

1 Introduction to TransactionVision Planning

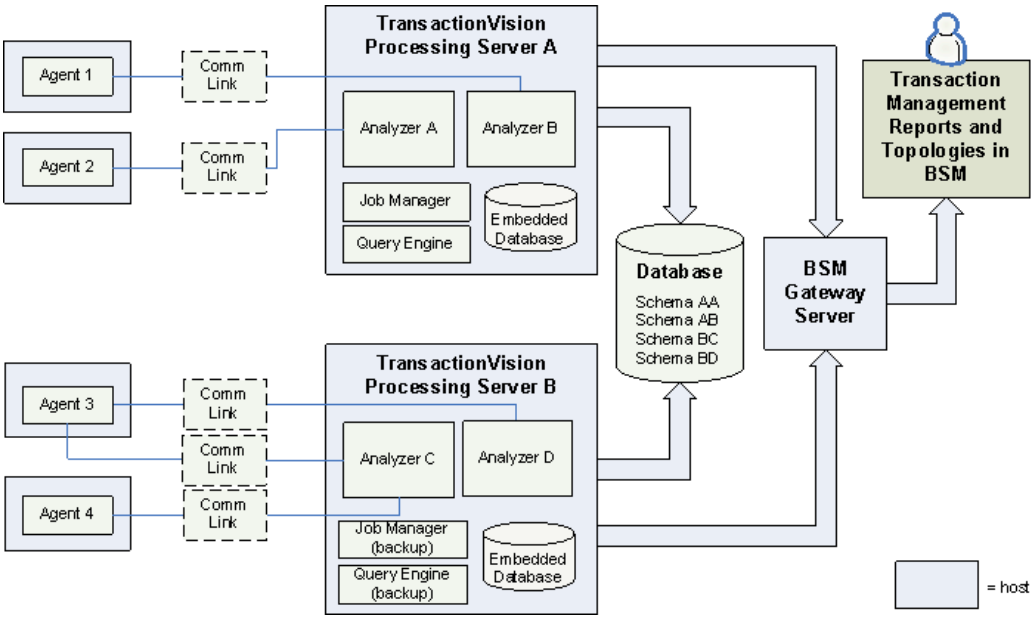
A TransactionVision deployment involves multiple components. It is important to install the components on the appropriate hosts in your environment. This chapter provides guidelines and a roadmap for planning your TransactionVision deployment.

TransactionVision Components

TransactionVision consists of the following major components:

- **Processing Servers.** Container for the TransactionVision components that deliver the core functionality of transaction tracing.
- **Analyzers.** Processes incoming events collected by the agents and constructs corresponding business transactions. Business transactions are represented as Business Transaction CIs.
- **Agents.** Monitors transactional events in applications and collect those that meet certain criteria. Events are API level interactions between an application and the middleware it uses to carry out a transaction.
- **Communication Links**. Enables agents to communicate from the host on which an application is being monitored to the Analyzer. Communication links contains message queues which are managed by a messaging middleware product; either the built-in one can be used or one from a third-party.
- **Database**. Contains a schema for the event information collected by each Analyzer. This database is independent from the BSM databases.

The following diagram shows these components in a sample deployment environment. This environment has two Processing Servers that are each running two Analyzers. Each Analyzer is being sent events collected by agents. Agents are installed on the same host as the application they are monitoring:

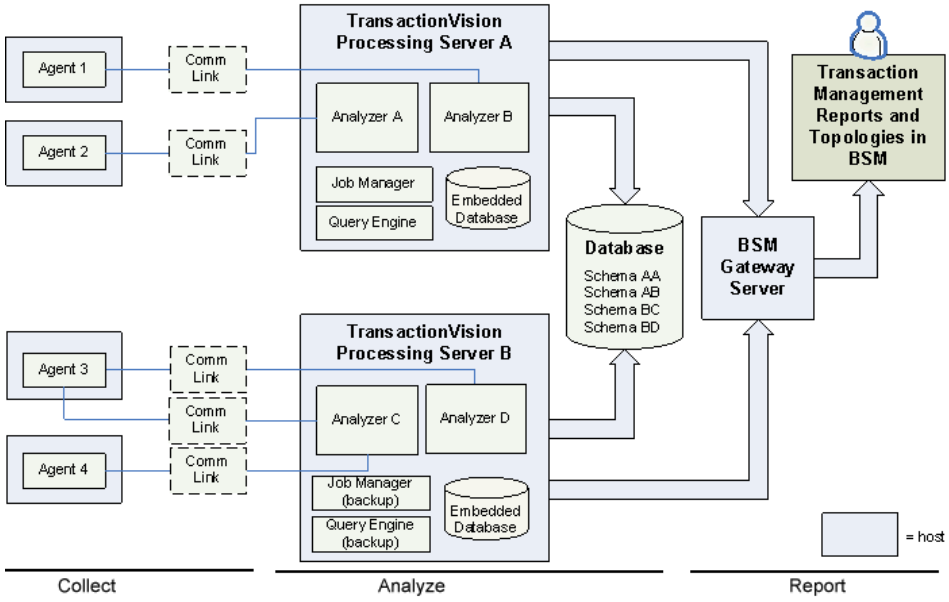


For more information about these components and how events move through them, see Chapter 1, "Introduction," in Using Transaction Management.

Deployment Planning

Proper planning is important for successfully deploying TransactionVision, and the amount of planning necessary depends on the nature of deployment. Large production deployments require more up-front effort in capacity and resource planning, while smaller development and QA deployments with a single monitored application require minimal planning.

Consider the following TransactionVision deployment scenario for initial deployment planning:



TransactionVision agents collect events and send them via communication links to a designated Analyzer. The Analyzer processes the events and builds them into transactions. The Processing Server is a container for the Analyzer and its supporting components as shown above. The Analyzer and Processing Servers write data to the TransactionVision database and to the BSM components.

Collection Strategy Considerations

- What applications and objects need to be monitored by agents? Is the technology used by the application supported by one of the agents?
- Where are the agents going to be deployed? Which agents need to be installed there?
- What kind of messaging middleware needs to be used between the agent and the Analyzer?
- Are there firewalls between the agents and the messaging middleware servers?

Analyze Strategy Considerations

- How many Processing Servers do we need?
- Where will the Processing Servers be hosted?
- How many Analyzers are there on each Processing Server?
- Which agents collect to which Analyzers?
- Are there firewalls between the Analyzers and the messaging middleware servers?

- Where will the Job Manager reside?
- Where will the Query Engine reside?
- Which database type (vendor) will be used?
- Where will the database servers be hosted?
- Are there firewalls between the Processing Servers and database servers?
- How much data needs to be collected by each agent?
- How much data needs to be retained and for how long?
- Will the chosen database support the volume of data?
- How many database instances do we need?
- What access authorizations are to be provided to users administering TransactionVision?

Integration Considerations

- Will TransactionVision be integrated with Diagnostics?
- Will TransactionVision be integrated with BPM (Business Process Monitor)?
- Will TransactionVision be integrated with RUM (Business Process Monitor)?
- Are there firewalls between the Analyzers and the RUM Engine?

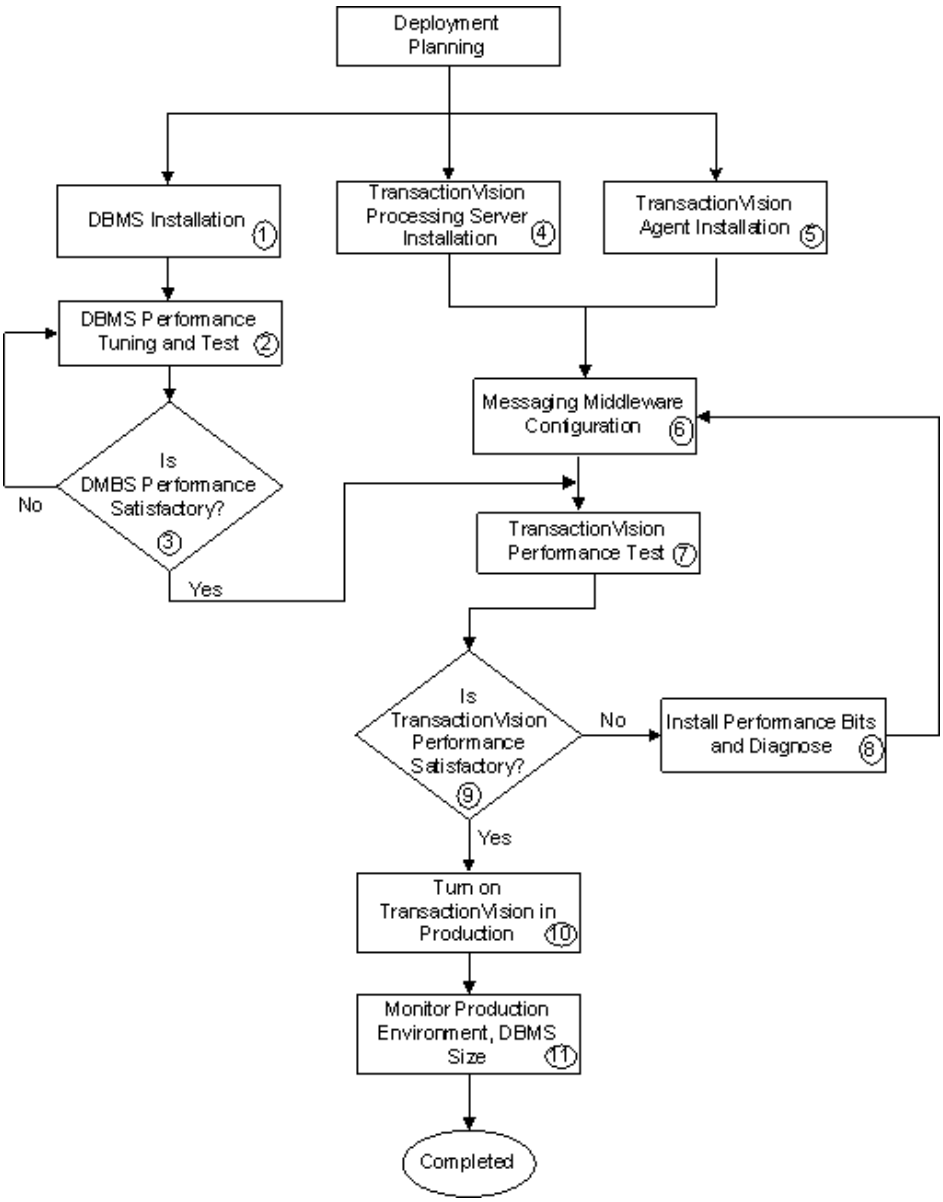
Reporting Strategy Considerations

- What are the reports to be run and how often will they be run?
- What access authorizations are to be provided to users accessing Transaction Management reports and topologies?
- Are there firewalls between the Processing Servers and BSM Gateway servers?
- Are there firewalls between users' web browsers and BSM Gateway servers?

The remainder of this manual guides you in determining the answers to these questions.

Deployment Roadmap

The following diagram shows a high level task flow for a typical TransactionVision deployment scenario. The numbered elements are referenced in the table on the following page.



The following table describes the numbered elements in the task flow.

Ref No.	Comment
1 - 3	TransactionVision requires a third-party database. This can be IBM DB2, Oracle, or SQL Server. In a test environment where the volume of event collection is in the range of 100,000 to 200,000 events, the embedded database can be used instead of a third-party database. See Chapter 2, *DBMS Sizing and Tuning* for more information.
4	TransactionVision requires at least one Processing Server. On the Processing Server at least one Analyzer must exist. See Chapter 3, *Analyzer Host Sizing* for more information.
5	TransactionVision requires at least one agent to collect events and send them to the Analyzer. To help you plan and list out the agents and the applications they are monitoring, see the Deployment Planning worksheet. Also see the *HP TransactionVision Deployment Guide* PDF for information about agent installation and configuration.
6	TransactionVision uses messaging middleware. You can use either the built-in SonicMQ or a 3rd-party product such as WebSphere MQ. See Chapter 4, *Messaging Middleware Configuration* for more information.
7 - 9	Once everything is up and running, test the initial performance of the system and adjust as necessary. See Chapter 5, *Performance Testing and Tuning* for more information.
10	Move a POC or test deployment to a production environment. See Chapter 6, *TransactionVision Administration* for more information
11	Perform ongoing maintenance of the TransactionVision components. See Chapter 6, *TransactionVision Administration* for more information.

Self-Discovery of Existing Transaction Paths

In many cases, a prototype TransactionVision deployment can be used to self-discover existing transaction paths, which can help to determine the scope and capacity of the installation. The basic procedure for a prototype deployment is as follows:

1. Install the TransactionVision agents on the servers running the major applications of the system to be monitored.
2. Set up communication links from the agents to the Analyzer.
3. Run the applications with the agent enabled, with data collection filters set to collect all data, for an appropriate length of time to collect all necessary data.
4. Use the data collected by TransactionVision to estimate average message size, database size requirements, number of queues and queue managers, application names, and so forth.

See the HP TransactionVision Deployment Guide PDF and Using Transaction Management for information about these tasks.

Assembling the Deployment Team

Deploying TransactionVision involves a team of people with many different roles. The Deployment Planning worksheet can help you list the contacts to assist with your TransactionVision deployment.

Firewall Configuration

In your TransactionVision deployment environment, if there are firewalls between various TransactionVision components, or between TransactionVision servers and other BSM components, you may need to open certain ports on your firewalls. Use the following network traffic tables for reference:

Network Traffic Between Users' Browser and BSM Servers

From	To	Default Ports	Purpose
Browser	BSM Gateway	80	Use the BSM UI
Browser	BSM Gateway	443	Use the BSM UI (SSL)

NetworkTraffic Between BSM Servers and TransactionVision Processing Server Components

From	To	Default Ports	Purpose
BSM Gateway	Process Manager	21100	Get status and administrate TransactionVision Processing Server
BSM Gateway	Process Manager	21101	Get status and administrate TransactionVision Processing Server (SSL)
BSM Gateway	Query Engine	21000	Get status and query transaction instance data
BSM Gateway	Query Engine	21001	Get status and query transaction instance data (SSL)

From	To	Default Ports	Purpose
BSM Gateway	Job Manager	21010	Get status and administrate Job Manager
BSM Gateway	Job Manager	21011	Get status and administrate Job Manager (SSL)
BSM Gateway	Analyzer 1-5	21120, 21130, 21140, 21150, 21160	Get status and administrate Analyzer
BSM Gateway	Analyzer 1-5	21121, 21131, 21141, 21151, 21161	Get status and administrate Analyzer (SSL)
RUM Engine	SonicMQ Broker	21111	Send RUM events to TransactionVision
RUM Engine	SonicMQ Broker	21112	Send RUM events to TransactionVision (SSL)
Job Manager	BSM Gateway	80	Publish CIs and send sample data
Analyzer	BSM Gateway	80	Get time for calculating time skew

Network Traffic Between TransactionVision Processing Server Components and Database Servers

From	To	Default Ports	Purpose
Analyzers, Query Engine, Job Manager	Database server	IBM DB2: 50000	
Microsoft SQL Server: 1433			
Oracle: 1521	Access the database that stores the transaction instance data		

Network Traffic Between TransactionVision Processing Server components and Messaging Providers

From	To	Default Ports	Purpose
Analyzer	SonicMQ Domain Manager	21110	Query event queue depth
Analyzer	SonicMQ Broker	21111	Send configuration messages and get events
Analyzer	WebSphere MQ	1414	Send configuration messages and get events

Notes:

- A TransactionVision Analyzer may use and communicate with a TransactionVision SonicMQ server running on the same or different TransactionVision Processing Server.
- For WebSphere MQ, if an Analyzer and the queue manager are running on the same system, this port is needed only if the Analyzer uses the client connection mode to access the queue manager.

Network Traffic Between TransactionVision Agents and TransactionVision Processing Server Components/Messaging Components

From	To	Default Ports	Purpose
Java Agent	Process Manager (Time Server)	21104	Get time for calculating time skew
Java Agent	SonicMQ Broker	21111	Get configuration messages and send events
Java Agent	SonicMQ Broker	21112	Get configuration messages and send events (SSL)
Java Agent	WebSphere MQ	1414	Get configuration messages and send events
.NET Agent	Process Manager (Time Server)	21104	Get time for calculating time skew
.NET Agent	SonicMQ Broker	21111	Get configuration messages and send events

From	To	Default Ports	Purpose
.NET Agent	SonicMQ Broker	21112	Get configuration messages and send events (SSL)
.NET Agent	WebSphere MQ	1414	Get configuration messages and send events
NonStop TMF Agent	Process Manager (Time Server)	21104	Get time for calculating time skew
NonStop TMF Agent	Analyzer 1-5	21120, 21130, 21140, 21150, 21160	Get configuration messages
NonStop TMF Agent	Analyzer 1-5	21121, 21131, 21141, 21151, 21161	Get configuration messages (SSL)
NonStop TMF Agent	SonicMQ Broker	21113	Send events
NonStop TMF Agent	SonicMQ Broker	21114	Send events (SSL)
Tuxedo Agent	Process Manager (Time Server)	21104	Get time for calculating time skew
Tuxedo Agent	Analyzer 1-5	21120, 21130, 21140, 21150, 21160	Get configuration messages
Tuxedo Agent	Analyzer 1-5	21121, 21131, 21141, 21151, 21161	Get configuration messages (SSL)
Tuxedo Agent	SonicMQ Broker	21113	Send events
Tuxedo Agent	SonicMQ Broker	21114	Send events (SSL)
DataPower Agent	SonicMQ Broker	21113	Send events
DataPower Agent	SonicMQ Broker	21114	Send events (SSL)

Note: Each Java or .NET Agent needs only one message provider.

2 DBMS Sizing and 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.

For information about which DBMSs are certified for use with TransactionVision, see chapter 3 of the HP TransactionVision Deployment Guide PDF.

Disk Storage Requirements

One critical factor to DBMS sizing is the amount of disk storage space required for TransactionVision events. This space can be estimated by the following formula:

```
Storage Size = (Average Message Size + User Data Size) * Event Rate *
Event Retention Time
```

The average message size depends on the database type and the event technology type. The average message sizes are explained in the following sections.

Average Message Sizes in Oracle

Technology of the Event	**Average Storage Size per Event**	**Size of XML per Event**
EJB	6,200 bytes	3200 bytes
Servlet	10,400 bytes	6600 bytes
JDBC	4,600 bytes	3000 bytes
JMS	7,600 bytes	2800 bytes
WMQ	11,400 bytes	4600 bytes
CICS (average storage estimated)	3,700 bytes	2100 bytes

The average storage size includes all the entities related to the event: its XML, its local and business transaction information, and more. The size of XML for an event is the size of the event message itself.

The XML per event size can help to determine what amount of disk space could be saved if you choose to disable event XML storage.

The table assumes the event has no user data included. See information on user data in *Additional Space Required for User Data*.

Average Message Sizes in DB2

Technology	Average Storage per Event	Size of XML
EJB	6,200 bytes	3200 bytes
Servlet	10,400 bytes	6600 bytes
JDBC	4,600 bytes	2800 bytes
JMS	8,300	2700 bytes
WMQ	12,100	4500 bytes

Average Message Sizes in Microsoft Server

The average message size data for Microsoft SQL Server will be published in a future version. The average size is expected to be similar to the other DBMS vendors above.

Additional Space Required for Users Data

Events can optionally contain user data, also referred to as payload. This portion of the message can provide additional information for correlation or reporting.

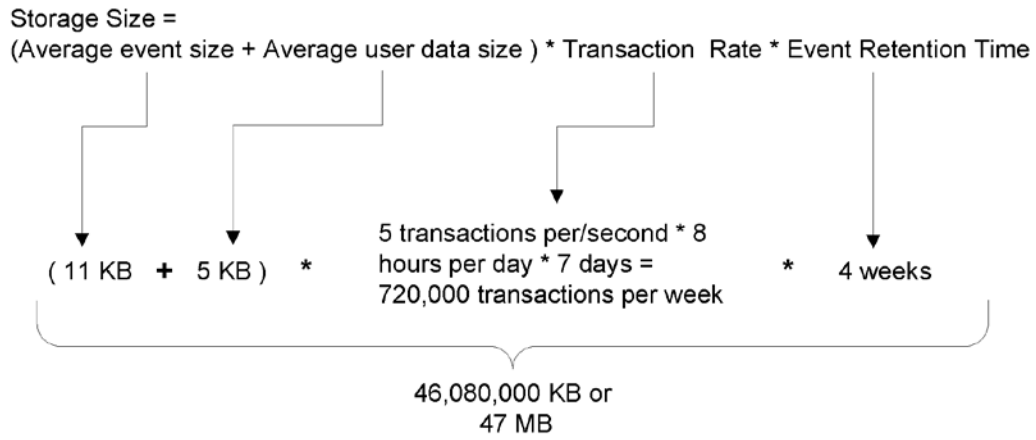
The storage of user data with an event is configurable as follows:

- By default, the user data portion of an event is stored in the database. You can specify that the user data is not stored, even though it is collected. This can be useful in a testing environment as you stabilize other areas and determine if the user data is needed. To specify that user data is not stored, clear the Store User Data check box on the associated communication link configuration pages. For details, see "Communication Links" in *Using Transaction Management*.
- The User Data Range criteria in the data collection filter can specify whether user data is collected, and if so, how much to collect. For example, maybe only a portion of the user data is needed for correlation. For details, see "Data Collection Filters" in *Using Transaction Management*.

Note: By default TransactionVision collects all user data (except for servlets which is only the first 1K). Typically you would look at the user data after collecting some events, and decide what is needed to keep. Then adjust the user data range criteria in order to minimize the data sent/stored. Use the Event Details report to view the user data size.

Example of Calculating Disk Storage Requirements

For example, servlet messages in DB2, with 5K of user data, have the following disk storage requirement:



Tip: Using your DBMS compression features, such as DB2's compact LOB, can reduce this number by 30% - 70%.

Worksheet

The Deployment Planning worksheet can help you estimate the disk space needed for event storage.

DBMS Performance Guidelines

Inserting records and updating records represent the majority of the database operations associated with TransactionVision. Therefore the speed of the physical disks and the I/O interface has a significant impact on the performance.

One of the first things in deployment is 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 schemas - if the data can be partitioned then separate DBMS instances on different hosts can be employed to achieve parallelism. However correlation cannot occur between two different schemas. So all events related to the same transaction must be sent to the same schema.
- Use RAID disks for table space containers. RAID disks provide parallel I/O at hardware level.

- Separate table space containers and log file directories. Log files (DB2 term, Rollback Segment for Oracle) hold uncommitted database operations and usually is highly utilized during database insert/update. For this reason it should have its own container 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.
- Large-scale deployments may require an even higher degree of parallelism at the DBMS level, such as using DB2 EEE (Parallel Edition) database.

There are many other database parameters that may impact the performance of TransactionVision and need to be examined one-by-one to make sure they are optimized to the specific DBMS system.

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

DBMS Performance Testing

HP provides tools to test database performance with respect to TransactionVision: DB2Test, OracleTest, SQLServerTest.

Each tool simulates the database insert operations generated by TransactionVision. The test should be run multiple times to get a complete picture of the DBMS performance.

Note that the result of the test does not directly correlate with TransactionVision processing rate, rather it is an indicator of 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.

To prepare for the test:

1. On the Analyzer host, make sure:
   - Java Runtime Environment 1.5 or above is available
   - DBMS server or client installed
   - JDBC 2.0 driver is set in the CLASSPATH environment variable. For DB2 the driver is db2java.zip. For Oracle the driver is classes12.jar. For SQL Server, the following files must be added to the CLASSPATH:
     - msbase.jar
     - mssqlserver.jar
     - msutil.jar
2. Set up the test structure in the database. Use the SQL statements in the following database-specific sections to create the database structure.

DB2

```
CREATE TABLE SCHEMA_NAME.RAW_EVENT (
 event_id INTEGER NOT NULL,
   event_status INTEGER NOT NULL,
   event_time TIMESTAMP NOT NULL,
    commlink_secs INTEGER NOT NULL,
    commlink_msecs INTEGER NOT NULL,
    client_secs INTEGER NOT NULL,
   client_msecs  INTEGER NOT NULL,
    event_data BLOB(10M) NOT NULL,
    overflow_id INTEGER,
   CONSTRAINT PK_RAW_EVENT PRIMARY KEY (event_id) )
          <IN TABLESPACE>;
```

Replace SCHEMA_NAME with a suitable schema name in the test environment. If not using the default table space USERSPACE, <IN TABLESPACE> should be specified with the actual tablespace name.

Oracle

```
CREATE USER "SCHEMA_NAME"
   PROFILE "DEFAULT" IDENTIFIED BY "SCHEMA_NAME"
    DEFAULT TABLESPACE "USERS" TEMPORARY TABLESPACE "TEMP"
    QUOTA UNLIMITED ON USERS ACCOUNT LOCK;
 GRANT "CONNECT" TO "SCHEMA_NAME";
 CREATE TABLE SCHEMA_NAME.RAW_EVENT (
    event_id INTEGER NOT NULL,
    event_status INTEGER NOT NULL,
    event_time DATE NOT NULL,
    commlink_secs INTEGER NOT NULL,
    commlink_msecs INTEGER NOT NULL,
    client_secs INTEGER NOT NULL,
    client_msecs INTEGER NOT NULL,
   event_data BLOB,
    overflow_id INTEGER,
          CONSTRAINT PK_RAW_EVENT PRIMARY KEY (event_id) );
```

Replace SCHEMA_NAME with a suitable schema name in the test environment. If not using the default table space USERS, USERS should be replaced with the actual tablespace name.

SQL Server

```
exec sp_addrole 'SCHEMA_NAME'
go
 BEGIN TRANSACTION;
 CREATE TABLE SCHEMA_NAME.RAW_EVENT (
    event_id INT NOT NULL,
    event_status INT NOT NULL,
    event_time DATETIME NOT NULL,
    commlink_secs INT NOT NULL,
    commlink_msecs INT NOT NULL,
    client_secs INT NOT NULL,
    client_msecs INT NOT NULL,
   event_data IMAGE,
   overflow_id INT, CONSTRAINT PK_RAW_EVENT
    PRIMARY KEY (event_id) )
 COMMIT
```

Replace SCHEMA_NAMEwith a suitable schema name in the test environment.

To run the test:

1. Access the host on which the TransactionVision Analyzer will be installed.
2. Run the insert test with one thread and with record size of 1KB; this will gauge the raw event insert performance. For Example:

```
$ RunClass.bat com.bristol.tvision.admin.OracleTest bart1 test
1521 system oracle SIZETEST 100000 1024 8 -RAW_EVENT -LOB -
commit 50 -jdbcBatch -thin

0 [main] INFO Trace  - Embedded database is accessible

Inserting into table RAW_EVENT
Using commit count of 50
Using JDBC batching
URL to connect = jdbc:oracle:thin:@flachbart1:1521:test

Thread 1 started, processing 12500 events.
Thread 2 started, processing 12500 events.
Thread 3 started, processing 12500 events.
Thread 4 started, processing 12500 events.
Thread 5 started, processing 12500 events.
Thread 6 started, processing 12500 events.
Thread 7 started, processing 12500 events.
Thread 8 started, processing 12500 events.
Thread 7 finished.
Thread 6 finished.
Thread 8 finished.
Thread 3 finished.
Thread 2 finished.
Thread 4 finished.
Thread 1 finished.
Thread 5 finished.

Stored 100000 Rows in 12.167 seconds
8,218.95 Rows/second
```

3. 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.
4. Run the insert test with one thread and with record size of (7KB); this will gauge the analyzed event insert performance.
5. Run the insert test with multiple threads and with record size of (7KB). 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. For example:

   Example of running the insert test with 7K and multiple threads (4 CPUs):

   DB2

   java com.bristol.tvision.admin.DB2Test localhost tvision 50000 db2inst1 ibmdb2 TESTSCHEMA 1000000 7000 8 -RAW_EVENT -LOB -commit 50 -jdbcBatch

Oracle

java com.bristol.tvision.admin.OracleTest localhost tvision 1521 system oracle TESTSCHEMA 1000000 7000 8 -RAW_EVENT -LOB -commit 50 -oracleBatch -thin

SQL Server

```
java com.bristol.tvision.admin.SQLServerTest localhost 1433
tvison sa sql TESTSCHEMA 1000000 7000 8 -commit 50 -jdbcBatch
```

Note: 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 are any communication/DB client configuration related issues.ust to make sure that there are no network bottlenecks.

Interpreting the Test Results

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 DB, DBMS and kernel parameters with HP for any configuration issues.

Each DBMS has tools to analyze performance, such as DB2's Performance Snapshot.

Next Steps

Configure and tune the database. See chapter 6: Configuring Databases in the *HP TransactionVision Deployment Guide PDF*.

3 Analyzer Host Sizing

Reviewing the Benchmark Data

Reviewing the existing benchmark data can help you determine your basic Analyzer host requirements.

Benchmark Number 1

The deployment scenario for benchmark number 1 is a single Analyzer on Linux with a mixture of agent types.

The benchmark details and result are as follows:

Benchmark Number 1	
Processing Server Host:	Software environment: Linux RH Enterprise 4, DB2 EE 9.2 Hardware environment: 4-way Quad Core HP DL580 G5 ProLiant Server (1.6 GHz CPUs)
Database:	IBM DB2 EE 9.2
Number of Analyzers:	1
Test data used:	**Trade Simulation data:** 50 events in one business transaction on avg 2 local transaction per business transaction **Technologies:** 20% WMQ events 20% CICS events 60% J2EE (Servlet,EJB,JDBC,JMS) events **Messaging Events/User Data:** 30% of all events are messaging events (WMQ,JMS) with user data < 1K
Throughput of 1,000,000 messages	1,600 messages per second 96,000 messages per minute 5,760,000 messages per hour 138,240,000 messages per day

Note: Performance is affected by factors such as the type of messages collected, the payload size, and any custom correlation. Each environment is different and may exhibit different results than shown above.

Benchmark Number 2

The deployment scenario for benchmark number 2 is a single Analyzer on Linux with a mixture of agent types.

The benchmark details and result are as follows:

Benchmark Number 2	
Processing Server Host:	Software environment: Linux RH Enterprise 4, DB2 EE 9.2 Hardware environment: 4-way Quad Core HP DL580 G5 ProLiant Server (1.6 GHz CPUs)
Database:	Oracle 11g
Number of Analyzers:	1
Test data used:	**Trade Simulation data:** 50 events in one business transaction on avg 2 local transaction per business transaction **Technologies:** 20% WMQ events 20% CICS events 60% J2EE (Servlet,EJB,JDBC,JMS) events **Messaging Events/User Data:** 30% of all events are messaging events (WMQ,JMS) with user data < 1K
Throughput of 1,000,000 messages	900 messages per second 54,000 per minute 3,240,000 per hour 77,760,000 per day

Understanding the Target Transaction Rate

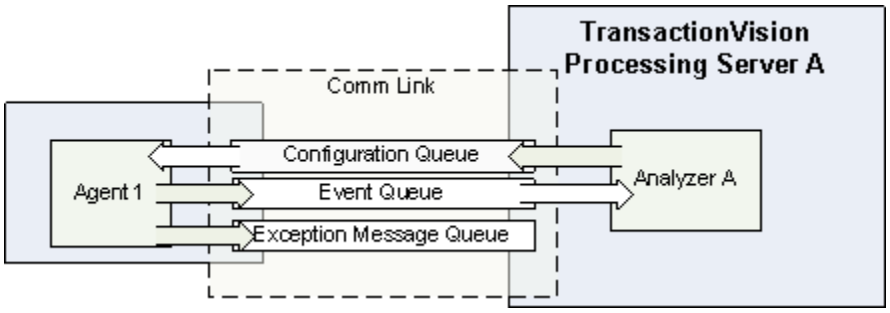
Before deploying TransactionVision it is important to know the volume of events/transactions coming in that the Analyzer must be able to process.

You can obtain this metric from the environment using other tools.

Choosing the Analyzer Host

The TransactionVision Analyzer communicates with TransactionVision agents via messaging middleware.

It generates and delivers configuration messages to agents by placing them on a designated configuration queue. The configuration messages tells the agents what event messages to send back. The Analyzer retrieves events placed on an event queue and processes them for analysis:



Therefore the host you choose for the Analyzer must be one that supports the message middleware product you want to use to communicate between the agent and the Analyzer. Each type of agent has its own messaging middleware restrictions. See the Deployment Planning spreadsheet.

By default, TransactionVision uses SonicMQ as the messaging middleware provider and this product is bundled in. WebSphere MQ is also supported.

The Analyzer can run on Windows or Linux. See the HP TransactionVision Deployment Guide PDF for more information about the supported platforms for the Analyzer.

The host on which the Analyzer runs must also have access to the BSM Gateway Server. For details about the BSM deployment environment, see the HP Business Service Management Deployment Guide PDF.

Scaling Up

Your deployment can include multiple Analyzers and multiple Processing Servers.

If you reach a point during testing where the database access is slowed down, you could move some of the work to another Analyzer on another Processing Server.

Deploying multiple Analyzers requires partitioning data. That is, you need to determine which Analyzer handles what kind of event data. This is because all event data related to the same transaction must be processed by the same Analyzer.

There are two alternatives for partitioning data.

- TransactionVision's data collection filters can be used to partition data based on technology or other factors. For example, the MQ header often contains information that can be used to identify specific business applications.
- Dedicated agents can be used to partition data. For example, appserver A has one agent and all events are sent to analyzer 1, and appserver B has one agent which sends all events to analyzer 2. No filtering is necessary, since all events that get transported on the defined commlinks go to the same Analyzer.

Next Steps

Configure and tune the Analyzers as necessary. See chapter 3: Analyzers in Using Transaction Management.

4 Messaging Middleware Configuration

WebSphere MQ

The following sections describe the queue requirements for WebSphere MQ.

Message Length

The message length requirements for configuration queues are small. A default configuration message uses approximately 500 bytes. However, specifying data collection filter conditions increases the size of configuration messages. A minimum message length of 10,000 bytes is required.

The length of event messages can vary greatly, depending on the parameters passed to the API, the user data passed through the calls, and data collection filter conditions. For event queues, it is recommended to use the default maximum message length of 4194304 bytes; a minimum message length of 10,000 bytes is required. To restrict the message length based on your applications, consider the length of messages sent by your application and allow an additional 4000 bytes for event information.

Queue Depth

For configuration queues, the default queue depth should be sufficient. Analyzers send configuration messages to start data collection, to change data collection filter conditions, to end data collections, and to determine time skews across hosts. The number of configuration messages sent to a configuration queue depends on the number of Analyzers sending messages to the queue.

Event queue depth should be set to a value that matches the queue manager storage space based on the average event package message size. It should be adequate to handle the peak volume. It is recommended to use the largest queue depth possible for the event queue and, in the case of remote communication link configurations, all queues in between. In some cases, agents put event messages on the event queues faster than Analyzers retrieve them. If event delay retry is enabled on the communication link, insufficient queue depth may result in event queues filling up and agents slowing down waiting for the Analyzer to catch up. Increasing the event queue depth helps prevent this situation.

Event Queue Manager

It is best to have the actual event queue hosted on a queue manager other than the production queue manager, so that issues related to event queues (such as running out of disk storage due to event backlog) does not affect normal operations.

Note: The channel on which the TransactionVision event queue is located should have the channel property CONVERT set to NO. If this property is set to YES, event messages are discarded to the exception message queue. If the channel is being shared by queues that have messages that require the property to be set to YES, please create a separate channel specifically for the event queue.

Event Queue Storage

Event queue storage should match the event queue depth based on the event message size. It should not exceed the storage capability of the queue manager host system. This is especially important when the event queue and exception message queue are hosted by the production queue manager because running out of storage space may stop the queue manager completely.

Set the Configuration Message Expiry in the Analyzer to a small value. This option 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.

Event Queue Message Persistency

The event queue message persistency property should match the event collecting policy.

Note that non-persistent messages typically means better Analyzer performance and persistent messages means no loss of data.

Queue Parameters

Queue parameters require the following:

- Configuration and event queues must:
  - have the Default Share Option (DEFSOPT) set to SHARED.
  - allow Shared Access (SHARED).
  - allow Get (GET(ENABLED)) and Put (PUT(ENABLED)) operations.
- TransactionVision configuration queues must allow messages of at least 10,000 bytes in length (MAXMSGL).
- TransactionVision event queues must allow messages of at least 10,000 bytes in length (MAXMSGL).
- All channels (including client channels) must have MAXMSGL of at least 10,000 bytes.

Agent Security Permissions

Userids of programs using the agent require the following:

- GET and BROWSE authority to the configuration queue.
- PUT authority to the event queue.

Analyzer Security Permissions

Users of the Analyzer require the following:

- PUT, GET, BROWSE, and INQ authority to all configuration and event queues.
- Passid permission to test communication links.

Note that Analyzer users can be given authority to only subsets of the configuration/event queues if you want to limit access to certain agents.

If a user does not have the required access permission on the configuration queue (TVISION.CONFIGURATION.QUEUE by default), the following message is found in the agent logs:

```
TransactionVision agent: cannot open event queue
TVISION.CONFIGURATION.QUEUE on queue manager merce.es1.manager:
Not authorized for access.
```

To set access permission of TVISION.CONFIGURATION.QUEUE for a user (tester in this example), run the following command:

```
setmqaut -m queuemanager -n TVISION.CONFIGURATION.QUEUE -t queue
-p tester. +get +browse +put +inq
```

It returns a message that the command completed successfully.

To display the access permission of TVISION.CONFIGURATION.QUEUE for a user (tester in this example), run the following command:

```
dspmqaut -m queuemanager -n TVISION.CONFIGURATION.QUEUE -t queue
-p tester
```

It returns output similar to the following example:

```
Entity tester has the following authorizations for object
TVISION.CONFIGURATION.QUEUE:
   get
   browse
   put
   inq
```

WebSphere MQ Capacity Planning

This section provides a methodology for determining how TransactionVision would impact the number of messages handled by a given system. Then, these adjusted numbers can be used with the information in the MQSeries Planning Guide (CSQZAB03) to calculate storage needs, and so on.

Use the following formula to determine the impact of TransactionVision monitoring on message volume:

`impact factor = 1 + (application efficiency factor / event packaging number)`

where:

Variable	Description
impact factor	The impact of TransactionVision monitoring on message volume. This number can be multiplied times the existing message volume to show how much message volume would be increased with TransactionVision monitoring. Note that when TransactionVision is not actively being used to monitor a system, there is no impact on message volume, so this factor becomes one.
application efficiency factor	A number from 1 to 5 approximating how efficiently the applications use the WebSphere MQ APIs. It is the ratio of the number of WebSphere MQ APIs called to the number of messages handled. In the worst case scenario, each MQPUT or MQGET will be accompanied by an MQCONN, MQOPEN, MQCLOSE and MQDISC. In this case, the efficiency of the application would be 5, since approximately five APIs are called for every message processed. At the other end of the spectrum, if an application were to call MQCONN, MQOPEN and then MQPUT 1,000 messages and MQCLOSE, MQDISC, the total number of APIs is 1,004. Dividing this by the number of messages processed (1,000) gives us a ratio of 1.004. Estimating this ratio is necessary to determine how many events TransactionVision will generate, and hence how many event messages TransactionVision will send.
event packaging number	The parameter set in the data collection filter telling the agent to pack a certain number of events into one event message. Note that if event packaging is not enabled, then this factor has a value of 1.

This impact factor may have to be calculated independently for different applications which are being monitored, since these applications may have different efficiency factors.

Consider an example. Suppose TransactionVision is being used to monitor an application running in batch mode overnight. This application runs on z/OS and reads data from some internal datasets and pushes that data out using messages to a variety of other servers. On average the application sends out 2,500 messages spread evenly over 25 different queues. The application is written efficiently - there is only one MQCONN and MQDISC needed and then an MQOPEN/MQCLOSE pair for each queue. Each batch of 1,000 messages is also committed or rolled back using MQCMIT or MQBACK.

An efficiency rating for this application can be calculated as follows:

```
2,500 (msgs) + 2 (MQCONN/MQDISC) + 50 (MQOPEN/MQCLOSE) + 25
(MQCMIT) = 2577 apis
```

```
2,577 apis / 2,500 msgs = 1.03 application efficiency factor
```

Now, apply the application efficiency factor to the number of messages sent and calculate different impact factors based how the event packaging is set.

Event Packaging Number	**Impact Factor**
1	1+(1.03/1) = 2.03
10	1+(1.03/10) = 1.103
25	1+(1.03/25) = 1.0412

Since this is a very efficient application, the TransactionVision impact can be minimized quite well with a reasonable packaging number. Simply multiply the impact factor times the number of messages processed, 2,500 in this case, and the result indicates that the total message traffic has increased to approximately 2,603 using the a packaging number of 25.

The impact factor can be used in the standard capacity planning formulae in the MQSeries Capacity Planning Guide which depend on either total messages or messages/second figures. Simply multiply the impact factor times the current messages or messages/second value to get a new result which takes into account the increased traffic due to TransactionVision. Note that the use of this factor only accounts for increased message volume, but does not account for the message size.

Note: Modify the inetd configuration file correctly. Incorrect configuration of inetd is the most common cause for not being able to establish new MQ client connections. Making sure the configuration is correct (including spelling!) avoids this issue.

SonicMQ

TransactionVision is bundled with a version of SonicMQ that is configured to work effectively out-of-the-box for smaller demo and POC environments. This installation of SonicMQ can be configured using SonicMQ tools and documentation. See the Progress SonicMQ documentation for more information.

TransactionVision also works with a stand-alone installation of SonicMQ. This is typically preferred for production environments because you have more control over where the queues are placed.

Next Steps

Configure and tune the communication links. See chapter 4: Communication Links in Using Transaction Management.

5 Performance Testing and Tuning

This chapter describes strategies to keep TransactionVision running at the best possible service level, especially in environments were event traffic is very high.

Setting Objectives to Optimize Performance

Most organizations establish service level standards for a complete, end-to-end-business process. This business process can include application logic, information retrieval from one or more data sources, database inserts, deletes and updates, data transformation and middleware messaging. The contribution of any one of these steps toward the end-to-end business process performance depends on the overall business process design. For example, a process that is solely responsible for routing messages relies significantly more on middleware performance than a business process that needs to access a database and performs application logic.

Business processes typically do more than just flow through middleware—they also usually involve accessing data, performing application logic and often data transformation. Therefore, performance overhead should be evaluated based on the entire end-to-end business process. Once you have identified the end-to-end business process that you need to optimize, the performance overhead of TransactionVision can be evaluated and optimized.

Key Performance Guidelines

The following sections give you guidelines for managing the performance of your TransactionVision deployment.

Collect Only the Event Messages You Need

Use the data collection filters assigned to the communication link to get only the required messages. This will result in less data stored in the queues and written to the database.

Data collection filters can be set based on several parameters including: Host Name, User Name, Program Name, Time, CICS Region, CICS Transaction, Job Name, API, API Return Code, QueueManager, MQSI Broker, MQSI Message Flow, IMS Region Type, IMS Region Identifier, IMS Identifier, IMS Transaction, and IMS PBS.

The default filter is designed to be good starting point. This filter can be customized but has the following initial settings:

For WebSphere MQ:

- Only MQGET, MQPUT, and MQPUT1 API names are included.
- Do not send browsing MQGET is set.

For JMS: send, receive, receiveNoWait, publish, and onMessage methods are included.

For Servlet: only the first 1024 bytes of HTTP request and response data are included.

The JDBC technology is off.

Collect Only the Part of the Message You Need

For WebSphere MQ, CICS, and JDBC messages, TransactionVision can be configured to collect User data, sometimes referred to as payload data. This data can be quite large.

Parameters can be set to collect only critical sections of the user buffer. That is, only the data needed for custom correlation or business context. For example, the parameters can:

- Set the Data Collection Filters to collect only APIs with failure or warning return codes.
- Set Data Collection Filters to collect only the first 100 bytes of each message user buffer.
- Set Data Collection Filters to collect only specific data of the message user buffer.

Use Event Packaging to Buffer Messages

Without event packaging in effect, TransactionVision calls MQPUT each time a message matches a data collection filter. The event packaging option buffers multiple messages at the agent and packages them into a single MQPUT call.

For example, event packaging can enable TransactionVision to send one TransactionVision message for every ten application messages.

The default data collection filter has event packaging enabled and the number of events to package is set to 10 events.

Set Analyzer Thread Count Accurately

Analyzer thread count should be set to match the test results from the DBMS insert rate.

Run TransactionVision Performance Tests

DBMS performance maintenance such as RUNSTATS (for DB2), or something similar for other databases, needs to be run on a regular basis (recommended daily) to ensure optimal performance.

Purge Events When as Soon as They are No Longer Needed

By default, no collected event data is purged. To avoid the database getting full unnecessarily, you should purge event data as soon as possible. You schedule purging to be performed automatically, based on event age or other factors. For details, see "Configuration Tab, Data Purge Tab" in "Analyzers" in Using Transaction Management.

Perform Regular Database Maintenance

DBMS performance maintenance such as RUNSTATS (for DB2), or something similar for other databases, needs to be run on a regular basis (recommended daily) to ensure optimal performance.

TransactionVision in a z/OS Environment

Runtime support for language environment must be included in affected CICS regions. The language environment must be enabled in the CICS region for C or the MQ listener will be disabled or possibly MQ connections will be disabled.

TransactionVision Performance Test

You can use the tvblast test application to verify that the installed and configured TransactionVision components can handle the volume generated by the production applications.

This test should be run in the same environment (or a mirrored environment) where the production application will run. The test result should be used to check whether the system has been tuned up to the same level of similarly configured systems.

The test should be run to generate events with the same size expected in production, and at a similar event rate. The whole event collecting and analyzing path will be tested this way. During the test, system resources across all involved platforms should be monitored for any potential issues.

Prerequisites

- WMQ server installed
- Agent installed on the test platform (supported platforms are Windows NT 4.0/2000, Solaris, AIX and HP-UX)
- TransactionVision Analyzer installed and configured for collecting events from the test platform
- One available queue defined at the queue manager to be tested
- Create a communication link between the Analyzer and agent.

PUT Format

Syntax:

```
tvblast -c count -b message_size  -W wait_interval -t trace_level
-pprint_iteration -u unit_of_work -s status_report queue_manager
queue
```

Where:

Option	Description
count	Number of total messages to write.
message_size	Size of message user buffer in bytes (default is 1000).
wait_interval	Amount of time in milliseconds to wait before proceeding to the next message.
trace_level	Controls how detailed the trace information is. Usually set to 2.
print_iteration	Controls how often (in messages) the trace message is given.
unit_of_work	Controls how many messages are included for each unit of work.
status_report	Controls how often (in messages) a statistics report is generated.
queue_manager	The name of the queue manager the program connects to.
queue	The name of the queue that is to be used for the test.

GET format

Syntax

```
tvblast -c count -b buffer -r -C -W wait_interval -t trace_level
-p print_iteration -u unit_of_work -s status_report queue_manager
queue
```

Where:

Option	Description
count	Number of total messages to read.
buffer	Size of the receiving buffer.
wait_interval	Amount of time in milliseconds to wait before proceeding to the next message.
trace_level	Controls how detailed the trace information is. Usually set to 2.
print_iteration	Controls how often (in messages) the trace message is given.
unit_of_work	Controls how many messages are included for each unit of work.
status_report	Controls how often (in messages) a statistics report is generated.
queue_manager	The name of the queue manager the program connects to.
queue	The name of the queue that is to be used for the test.

Note: To see the usage information for tvblast, enter: `* tvblast -x`.

Examples

If the production environment has a transaction rate of 5 transactions/second with two data collecting points (MQPUT and MQGET) and average message buffer size is 2000 bytes. The following test can simulate the production scenario in a 10-minute test using two tvblast instances for writing and reading:

- Message rate: 5 message/sec for each tvblast instance. Set the message wait interval to 200ms (-W 200).
- Message size: 2000 bytes. Set the buffer size to 2000 (-b 2000).
- Test duration: 10 minutes. Set the total message count to 3000 messages (-c 3000).

- Unit of work: assuming each message is in its own unit of work (-u 1).

```
tvblast -c 3000 -b2000 -t 2 -p 50 -u 1 -W 200 -s 300 test_manager
test_queue
```

```
tvblast -c 3000 -b2000 -C -r -t 2 -p 50 -u 1 -s 300 test_manager
test_queue
```

Note: The two tests should be run simultaneously. The data collection filter should be set to collecting MQPUT and MQGET in general.

TransactionVision Performance Tuning

If the performance test results show that the system does not perform as expected, further investigation is required to identify the bottleneck.

The first place to look is the performance data from the test, particularly the system resource utilization data. There may be resource bottlenecks (CPU utilization on the Analyzer hosts, disk I/O utilization on the DBMS hosts, etc.) that can be immediately recognized from this data.

If there is no obvious source for the under-performance, then a diagnostic test needs to be performed to break down the event processing time. A TransactionVision build with the performance diagnosis function enabled needs to be installed and the performance test needs to be repeated. TransactionVision will generate detailed performance information that is helpful in pinpointing the cause of the performance issue.

6 TransactionVision Administration

Moving to Production from Testing Environment

Once you have tested and configured TransactionVision components in a test environment, you are ready to move to a production environment. Moving to a production environment is not a simple move. It can require that you redo all the configuration changes that you applied in the test environment.

If you use a completely new BSM, you basically need to reconfigure everything from scratch and apply all TransactionVision changes and definitions again (such as creating the Processing Servers, Analyzers, configuring those objects, creating communication links, classification, and so on).

If you use the same BSM and only move from a test Processing Server to a production Processing Server on a different host, some of the configuration (which is not Processing Server or Analyzer specific) can be reused, such as queries or classification rules (as long as their definitions still apply to the production environment).

However, the bulk of configuration (for Processing Server/Analyzer definitions and customizations, and most likely communication links) still must be set up from scratch.

Ongoing Administration

The following tasks are required for ongoing TransactionVision administration:

- DBMS storage space management. You must monitor and control the storage space used by TransactionVision data. You can set purging behavior on each Analyzer and TransactionVision will manage the data cleanup automatically. By default, no data is purged.

  Or you can maintain the storage space by using your DBMS tools.

- DBMS performance maintenance. Database statistics need to be generated on a regular basis to ensure optimal performance.
- Adding and changing users. See Chapter 23 "Configuring Security" in the HP TransactionVision Deployment Guide PDF.
- Installing and configuring new agents as needed. Removing agents when no longer needed. See the HP TransactionVision Deployment Guide PDF
- Queue depth monitoring. A fail-safe method (such as running a crontab job) should be devised to make sure that if any of the messaging queues overflow appropriate action is taken.

7 Customizing TransactionVision

Moving to Production from Testing Environment

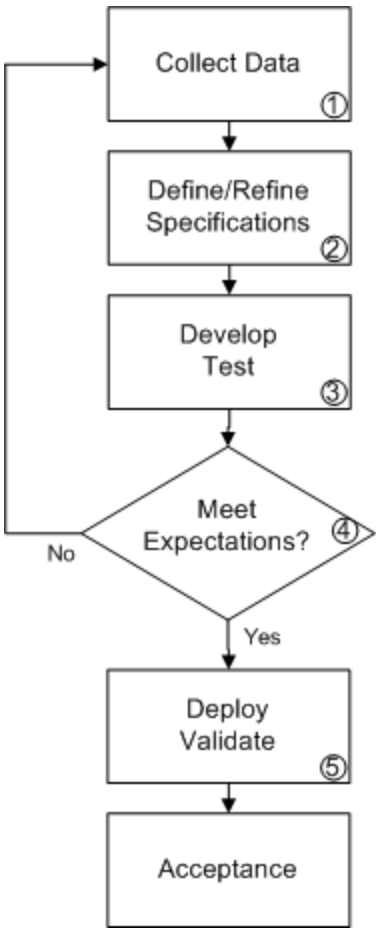
Depending on the data collected, TransactionVision can be customized in several ways. The primary customizations include:

- Writing Analyzer beans and XDM files to extract message data fields, writing custom event correlation beans or writing transaction classification rules. Often, message data contains useful information from which custom reports can be built. The first step for using this information is to convert binary message data into XML by writing an unmarshaller bean and then mapping XML fields into database table columns using TransactionVision XDM files. Event correlation beans may need to be written if parts of your system are not being monitored by TransactionVision. Transaction classification rules can be added to classify your system transactions into different categories and add attributes to those categories.
- Writing job beans to perform batch data analysis or administration tasks. Job beans may be written if you need to perform any kind of a batch job such as data backup, deletion, custom analysis and so forth.

For details on implementing the above customizations, see the TransactionVision Advanced Customization Guide.

- Writing custom reports. Custom reports perform analysis on data collected by TransactionVision that is specific to your system being monitored.

The task flow below describes the process to create a custom report. The numbered elements are referenced in the table on the following page which provides additional details about the steps and references to more information.



The following table describes the numbered elements in the task flow

Ref. No.	Comment
1	Collect TransactionVision data from customer environment. This step usually involves installing, configuring and running TransactionVision in some customer environment. The customer environment can be development environment, QA environment or production environment, as long as the actual (or mirrored) transaction flow can be captured by TransactionVision. The captured data can be used as the base for demonstrating TransactionVision capability, developing requirement for custom reports and later testing the reports. This step may be done during the Proof-of-Concept stage.
2	Define/Refine custom report requirement specification. Once customer has better understanding of the data collected by TransactionVision and how the data can be used to solve technical and business issues, a specification shall be developed by the customer to formalize the requirement for the custom reports. The specification should cover the logic for generating the reports, detailed user interface description (possibly with screen mock-ups or samples) if applicable, the manner the reports will be invoked (scheduled, interactive, and so on) and response time and any relevant topics. HP will provide an estimate for the amount of work that is involved to implement the reports.

Ref. No.	Comment
3	Develop and test custom reports. Once the specification is finalized, work can start to implement the logic to create the reports. The work will be conducted at HP. The work may involve both front end (user interface) and back end (data analysis) programming. HP will conduct the development in house and test the reports against the data collected from customer environment.
4	Review reports. Once the reports have been tested, the customer will have an opportunity to review the reports. If the reports does not meet all customer expectations, or based on the reports the customer has developed additional requirement, go back to step 2 to refine the requirement specification.
5	Deploy and validate report. Once the customer has approved the reports, they can be deployed to the customer environment for final validation. Sometimes re-configuration of the reports is necessary if the final deployment environment is different from the environment where the initial test data were collected. A user acceptance test usually is conducted after the deployment to make sure that both the content of the reports and the response time of the reports are acceptable to the customer.

8 Database Performance Testing Utilities

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 schema with CreateSqlScript (which can be deleted after running the test). See “DBMS Performance Testing” in this 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
-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

Oracle Test

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 "DBMS Performance Testing" in this 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)
-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

Option	Description
	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.

SQLServer Test

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). See “DBMS Performance Testing” in this guide for details on how to set up the required test environment.

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.

Example:

```
AnalyzerManager –stop –host HOST
```