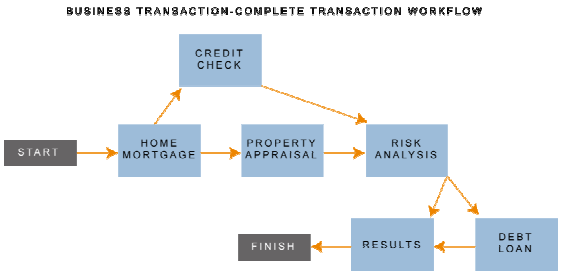
The TransactionVision Solution

Bristol's TransactionVision is transaction tracking and analysis software that provides a real-time view of business transactions flowing through a distributed enterprise environment via WebSphere MQ, CICS, JMS, J2EE, JDBC, TIBCO, and Web Services. TransactionVision automatically and non-intrusively tracks transactions across each touch point, self-discovering transaction flows and content while providing real-time monitoring to pinpoint failures and ensure service levels.

What is a Transaction?

TransactionVision is revolutionary in its ability to connect the technical and business perspectives of transactional activity. To fully understand the value, it is important to understand how Bristol defines a "transaction."



From a technologist's point of view, a transaction is a series of actions within a single application. These actions form an atomic unit of work in which the actions are all made permanent (committed) or undone (backed out) as one, rather than independently.

But to a business person a transaction is a unique business interaction, such as a stock trade or an online purchase. Business transactions are complex events made up of a series of interrelated activities (simple events) that, when executed, form the transaction. In enterprise environments, business transactions frequently are distributed, requiring numerous related units of work, which are tied together with transactional middleware such as WebSphere MQ.

TransactionVision transactions refer to the definition of a business transaction (e.g., stock trade or online purchase). These transactions (or complex events) are comprised of multiple units of work (or simple events).

TransactionVision uses patent-pending algorithms to construct the business transactions from the multiple units of work as they are executed, recording them in an open repository and providing business and technology oriented views. It automatically constructs and displays graphical views of the flow, timing, and business content of each transaction.

TransactionVision accomplishes this by tracking the interaction between applications at the middleware API level. This enables TransactionVision to observe the details necessary to construct a business transaction's execution as it flows across multiple applications on heterogeneous, geographically dispersed systems.

TransactionVision Benefits

TransactionVision's transaction tracking capabilities enable our clients to:

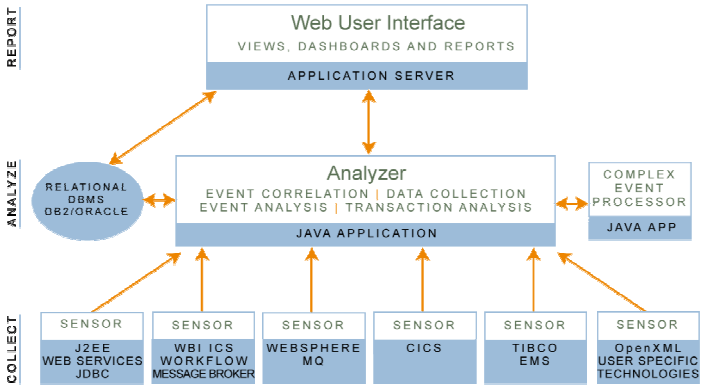
- Accelerate problem resolution by quickly pinpointing transaction issues across distributed heterogeneous business processes.
- Improve service level management by providing a transaction-based service level view.
- Mitigate risks by quantifying business consequences against IT performance.
- Monitor business activities in real-time by capturing the business content of each transaction to measure key performance indicators.
- Monitor both transactional patterns and transaction content patterns to identify issues and trigger automated processes or notifications.
- Optimize transaction performance and improve capacity planning.

These benefits are readily demonstrable in TransactionVision's views.

By depicting critical information such as data volumes (indicated by the width of the lines between components) or average communication latencies, essential information for capacity planning and performance optimization is rapidly visualized.

TransactionVision Architecture

TransactionVision uses a three-tier design to capture, analyze and present transactional information. The diagram below shows its

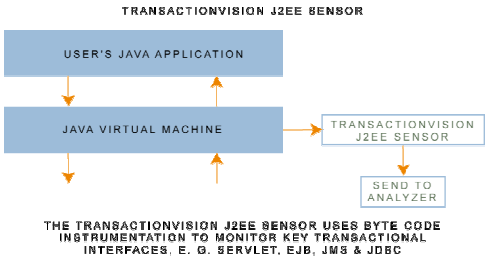


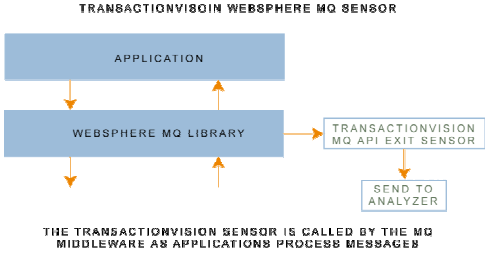
major components: Sensors, the Analyzer, and the Web User Interface. As shown, the Analyzer and Web User Interface utilize a standard relational database as a repository for collected transaction data.

Sensors

The Sensors collect transactional events from the various applications and report them to the TransactionVision Analyzer. Events are collected by monitoring the interactions between the applications and the middleware, including WebSphere MQ, CICS, JMS, J2EE, JDBC, TIBCO EMS and Web Services.

Sensors are lightweight and high-performance, and they are non-intrusive to the applications being monitored, requiring no changes to the application logic or source code. Sensors have negligible, often immeasurable, impact on end-to-end transaction response time. The information they report allows the reconstruction of the flow, timing, and content of the monitored transactions.





Sensors are specific to the platform and type of middleware, and are available for a wide-range of combinations. Platform support includes UNIX, OS/400, Linux, OS/390, z/OS, and Windows.

The mechanism used to monitor the interaction between an application and the middleware varies by type of middleware and by platform. As an example, on distributed platforms the WebSphere MQ Sensor consists of a small shared library that is included in the application's load library path. The application communicates with the middleware, and the sensor quickly scans the events as they pass from the application to the middleware. The CICS Sensor uses the CICS Global User Exit (GLUEs) and Task Related User Exits (TRUEs) to monitor key APIs inside CICS transactional flows.

The installation mechanism varies by type of sensor and platform, but in each case the installation and configuration of the Sensor is designed to be quick and straightforward; even complex systems can be installed in days.

Analyzer

The Analyzer is a Java application that performs a variety of functional tasks including Event Collection, Event Analysis, and Transaction Analysis.

Event Collection

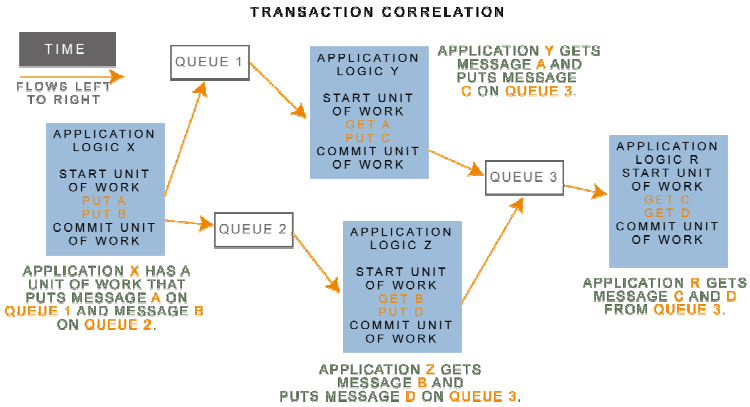
The Analyzer interfaces with all of the Sensors, providing them with configuration information to control the data collection process. This configuration information includes data collection filtering settings, which specify to the Sensors what information and data to report – for example, which programs, hosts, time periods, or infrastructure components are of interest. Communication between the Sensors and the Analyzer is via asynchronous messaging to ensure scalability, reliability, and security.

The Analyzer receives the reported events from the Sensors and performs Event Analy-

sis, as described below, upon them. The Analyzer components responsible for Data Collection and Event Analysis are multithreaded to provide scalability to support high transaction volumes, even across multiple processors when necessary.

Event Analysis

The Analyzer's Event Analysis processing consists of unmarshalling, correlation, analysis and data management functions. Most notably, the Transaction Constructor component of the Analyzer, as described below, constructs groups of related individual reported events (API calls, servlet invocations, etc.) into coherent end-to-end business transactions.



The Transaction Constructor

Individual transactions within enterprise applications are grouped into logical units of work. Units of work are denoted by middleware APIs visible to TransactionVision that allow developers to commit or rollback middleware transactions. This grouping of events allows TransactionVision to know what events are related to a single local transaction. Events leaving one application are matched with events arriving at another application by information inherent to the middleware infrastructure. The transaction correlation feature works whether or not the customer elects to collect business content data as part of transaction monitoring. TransactionVision monitors units of work as the transaction events flow from one process to another and when these cascading events end, a picture of the complete business process is presented.

The typical TransactionVision customer will never need to customize its correlation logic; however, each of the Event Analysis functions is performed by chained JavaBeans, which the customer may override or sub-class to tailor the analysis to their specific requirements. Examples include:

Custom beans in the unmarshalling process to convert binary message data into XML formats specific to particular business applications.

Custom beans in the correlation and analysis process to provide correlation with unsupported middleware technologies or proprietary inter-process communication solutions. This capability allows tailored correlation extensions that can tie two parts of a transaction that flows through a proprietary messaging system together.

Custom beans in the analysis step recognize application specific or IT infrastructure exceptions and send SNMP or other alerts to system monitoring products (e.g. Tivoli Enterprise Console, HP OpenView, and Micromuse NetCool.)

Custom beans in the data management process pull application specific fields from the transaction events and insert this data into database tables for reporting and querying. In addition, select information can be filtered out of the data collection process to protect information privacy policies or requirements.

This flexibility benefits systems integrators or users with complex requirements or unique integration needs.

For less technical users, TransactionVision's complex event processor allows business analysts to develop custom solutions without programming in Java.

The resulting description of the reconstructed transaction is stored into a relational database such as DB2 or Oracle. Data collected by TransactionVision is stored in XML, providing an open format that is easily queried or transformed into other formats. TransactionVision provides open access to information via standard database interfaces

Transaction Analysis

The Analyzer's Transaction Analysis processing evaluates the reconstructed transactions and invokes actions on these events as defined by the user. This analysis includes classi-

fication of transactions into user defined transaction classes (e.g., equity trades, payments), and maintaining transaction level response time metrics for each transaction. From this, service level performance requirements can be automatically evaluated. Separate service level targets for different transaction classes can be readily set and evaluated.

The Transaction Analysis process is customizable through JavaBeans, in a manner similar to the Event Analysis process or through an XML-based rules classification file. The action to take in response to an identified problem within a transaction also is configured by the user. For example, if a response time SLA target is violated an alert is automatically generated.

Complex event processor

The complex event processor extends TransactionVision's ability to react to failures, predict problems based on event patterns and create alerts triggered by event patterns. This complex event processing engine allows businesses to monitor both transactional patterns and transaction content patterns to identify issues sooner to notify people and/or trigger automated processes. The following table lists the basic building blocks used to construct the patterns in the rules engine for situational analysis:

Pattern Class	Examples
Transaction attributes	state, results, location, transaction content
Transaction content	stock price, order destination, customer id
Temporal settings	between 9 am and 11 am, 3 times in one minute, last week
External data	database field, system information, reconciliation table

The complex event processor feature of TransactionVision monitors event and event timing patterns enabling real-time reaction to events or series of events. Consider the following scenarios:

- Service Level Agreement Alerts – TransactionVision sends an alert message if an order was sent for processing and no response was received within the time specified by the SLA.
- Capital Markets Compliance or Regulatory Checks – TransactionVision sends an alert message related to a bulk buy transaction that is performance in less than the regulatory allowed time after a bulk sell
- Credit Card Fraud Detection – TransactionVision sends an alert message when multiple credit card purchases are executed within a specified brief time frame at a distance greater than 100 miles.

Installation Architecture

The Analyzer is a Java-based application that is installed on a server that has high bandwidth connections to the RDBMS server and the Web User Interface application server. As depicted below, the TransactionVision architecture allows the Analyzer, the Web User Interface and the associated RDBMS to be installed on the same server or on separate servers, depending on the customer's need for scalability.

Web User Interface

TransactionVision's user access is via a J2EE Web interface. This becomes especially important when TransactionVision is used for operational reporting by users throughout the global enterprise. The interface services build on either IBM WebSphere Application Server or BEA WebLogic Application Server for a robust, scalable, and secure solution that operates in a variety of platform environments. All of TransactionVision's user interfaces provide role-based security controls, integrating with LDAP solutions to manage which users have access to which information and functionality.

Job Scheduling

The Web User Interface also enables the user to schedule recurring tasks configured for each user environment. For example, it can execute custom analysis logic to aggregate business metrics to enable real-time dashboards, run regular reports, or trim transactional data. Job Scheduling provides extremely fast reporting of key performance indicators, even if their computation involves a large quantity of transaction data.

Built-in Reports

TransactionVision Web User Interface also offers an array of standard reports, including:

- Transaction Tracking
- Service Level Agreement Analysis
- Charge-Back
- Performance Dashboard
- Capacity Planning

Custom Web Reporting Framework

TransactionVision provides a framework to further refine the standard reports, or create new customized reports. In addition, since all data is stored in open relational databases, and the data itself is structured as XML, third party reporting tools can access the repository.

Architectural Advantages

TransactionVision is built for enterprise deployment. This means it meets the scalability, security and reliability requirements for today's demanding enterprises. It also means TransactionVision fits into the existing environment – easily integrating with business applications, supporting corporate standards for databases and middleware, and tying to standard enterprise management systems.

Some of the architectural advantages of TransactionVision are:

Scalability

TransactionVision is a multi-tier infrastructure. The Sensors are distributed – fitting into each distributed system in a low overhead, easily installed manner. The Analyzer is multithreaded to take advantage of a wide range of server capacities. In addition, there can be multiple Analyzers deployed in an enterprise. TransactionVision uses industry standard relational database management system (RDBMS) to store data, and it runs on a comprehensive range of systems including UNIX, Linux and Windows.

Performance

The modular design of TransactionVision allows its work to be split across multiple processes and multiple servers to best fit the requirements of each installation. This modular design provides transaction analysis even in extraordinarily high volume transaction environments.

Security

TransactionVision builds upon existing enterprise security frameworks and provides fine-grain control over user access to sensitive transactional data. It integrates with LDAP enabled directory services and industry leading single sign-on solutions.

Data Management

By building upon standard RDBMS products, such as IBM's DB2 and Oracle, TransactionVision leverages the storage management, back-up and recovery, and scalability that these products provide. Fine grain tracking information can be purged from the database after a set period of time, while aggregate metrics derived from these details can be retained longer. In addition, all data is stored in XML format for ease of use and complete openness for integration with other tools.

Thin Client/Java Application Server Architecture

TransactionVision's thin-client architecture allows global concurrent access without having to install client applications on every user's PC.

Extensible Platform/Data Accessibility

TransactionVision opens up its rich data repository by making all transactional information accessible through direct database connections, and XML data formats.

Customizable Reporting

Custom view of TransactionVision data can be created using J2EE technologies such as EJBs, JSPs, and Servlets, as well as with the integrated report generation tool. Reporting is also used to identify and alert users when rule-based business exception conditions occur.

Relationship to Other Enterprise Technologies

TransactionVision is a unique product, so it is worth noting how it relates to other relevant enterprise technologies:

- Middleware System Management
- Transaction Monitoring & Tracking
- EAI, Business Integration, or Enterprise Service Bus Solutions
- Web Services Management

Bristol is focused on providing solutions to customers who need to track business transactions across a complex, distributed environment. To understand this focus, it is first useful to break the types of enterprise applications into two categories:

1. Distributed Messaging

The purpose of this architecture is to enable multiple application components residing on separate systems to communicate with each other. WebSphere MQ is the dominant form of this infrastructure today and is in use both internally as well as in inter-company networks.

2. Web Application Servers

This architecture became popular with the Web – providing a consolidating platform to host applications that received requests from Web Browsers, retrieved information from databases or backend systems and then responded back to the browser.

Numerous system management and monitoring products address the need of the enterprise. It is useful to look at each of the above topics within the context of enterprise applications and understand where Bristol's TransactionVision fits in the landscape.

Middleware (WebSphere MQ or J2EE Application Server) Management

TransactionVision is not a middleware system management product. However, it is complimentary to these products.

The traditional approach to managing a WebSphere MQ environment is to monitor the status of each of the queue managers, queues, and channels, and report status through a central console. While important, this type of monitoring does not give a transaction-oriented view of the system – only a point solution view of a set of resources.

The same sort of comparative issues apply with regard to Application Server management solutions. Customers often use the traditional system management products for infrastructure monitoring and use TransactionVision to provide transaction tracking and analysis. Moreover, TransactionVision complements system management solutions by leveraging existing enterprise alert management capabilities as needed with business transaction issues. TransactionVision is a certified Ready for IBM Tivoli product.

In brief, management products look at anticipated exceptions only. TransactionVision looks at transactions, not just exception transactions, and can identify patterns that expose unanticipated exceptions as well as anticipated exceptions. For example, TransactionVision can identify a denial of service attack on web sites by observing the volume of legitimate activity. In this case, the exception is not associated with any one transaction, however, the rate of incoming transactions and the fact that they originate from relatively few connections could be indicative of a potential attack.

Transaction Tracking & Monitoring

TransactionVision is the only product that tracks business transactions across both distributed messaging environments and Web environments. There are tools that provide a purely Web-to-Application Server view of transactions, but TransactionVision offers a much broader and unique end-to-end transaction view across messaging and Web environments.

Enterprise Application Integration, Business Process Management, or Enterprise Service Bus Solutions

Most of the EAI, BPM and ESB vendors have some level of process monitoring. This is typically built around their workflow engines, and provides business process monitoring for the modeled part of their transactions. The problem most customers have is that they may not use the workflow tools, or they may have many applications or segments of their transactions that are implemented outside of the workflow products, greatly limiting visibility.

TransactionVision's auto-correlation capability discovers the workflow of legacy applications and can track transactions end-to-end across diverse integrated components.

Web Services Management

There are many early efforts to define web services management. The early entrants into this market broadly define it as including management of security and failover and load balancing. However, Web Services Management does not focus on the correlation of the messages as they flow through a distributed network, since most web service applications are simple point-to-point implementations. TransactionVision focuses on the complex needs of a distributed heterogeneous environment

Conclusion

The TransactionVision technical and architectural differentiators provide customers with significant advantages:

- High levels of scalability, performance, and security
- A non-intrusive solution requiring no changes to business applications or middleware
- A thin-client application server-based user interface that maximizes enterprise accessibility

- An open, extensible platform for data management and access.

Bristol's TransactionVision is a breakthrough solution for enterprises that rely on electronic transactions. Global customers across many industries rely on TransactionVision and its unique capability to track transactions across each touch point, self-discovering transaction flows and content while providing real-time monitoring to pinpoint failures and ensure service levels.

EXECUTIVE SUMMARY

Unique Solution

Bristol's TransactionVision is transaction tracking and analysis software that provides a real-time view of business transactions flowing through a distributed enterprise environment. TransactionVision automatically and non-intrusively tracks transactions across each touch point, self-discovering transaction flows and content while providing real-time monitoring to pinpoint failures and ensure service levels.

No other solution duplicates the key TransactionVision feature set: non-intrusive end-to-end tracking and automatically reconstructing each step of business transactions across WebSphere MQ, CICS, JMS, J2EE, and Web Services, on a range of platforms including UNIX, Windows, and IBM mainframes.

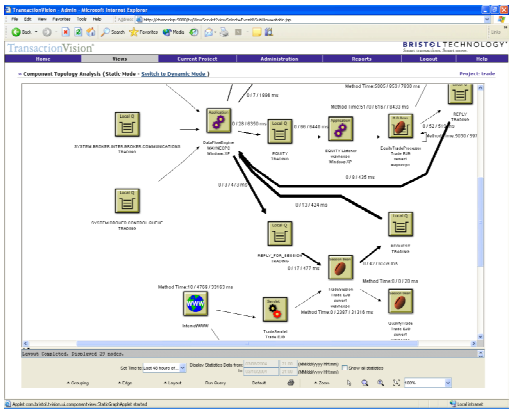
TransactionVision architecture features:

- High levels of scalability, performance, and security
- A non-intrusive solution requiring no changes to business applications or middleware
- A thin-client application server-based user interface that maximizes enterprise accessibility
- An open, extensible platform for data management and access.

Usage Benefits

TransactionVision provides the extensive transaction tracking capabilities necessary to:

- Accelerate problem resolution by quickly pinpointing transaction issues across distributed heterogeneous business processes.
- Improve service level management by providing a transaction-based service level view.
- Mitigate risks by quantifying business consequences against IT performance.
- Monitor business activities in real-time based on key performance indicators.
- Monitor both transactional patterns and transaction content patterns to identify issues and trigger automated processes or notifications.
- Optimize transaction performance and improve capacity planning.



Please contact us for more Information:

Scientific Computers GmbH
Friedlandstrasse 18
52064 Aachen / Germany
☎ +49 241 40008-0
+49 241 40008-13
www.scientific.de
info@scientific.de