

Transaction traceability of enterprise files, data, and applications: Part 3: Tracking operational transaction data in files, data, and applications using IBM Tivoli Composite Application Manager for Transactions

Devaprasad Nadgir
Abhinav Priyadarshi

September 12, 2012

This article series describes a scenario involving JK Financials, a fictitious bank that needs to gain end-to-end visibility over its integrated enterprise IT infrastructure. Part 3 shows you how to use IBM Tivoli Composite Application Manager for Transactions to track end-to-end movement of files, transactions, applications, and components using IBM Sterling Connect:Direct. It will show you how to use WebSphere Message Broker to emit transaction data to IBM Tivoli Composite Application Manager for Transactions, and quickly identify operational and transactional problems involving a money transfer application.

[View more content in this series](#)

Introduction

This article series describes a scenario involving JK Financials, a fictitious bank that needs to gain end-to-end visibility over its integrated enterprise IT infrastructure. This visibility will help provide transaction traceability and ways to address the IT infrastructure problems that arise during regular operations.

[Part 1](#) described the challenges faced by large enterprises today in integrating file-based systems with enterprise applications spread across multiple networks and supporting different protocols. Part 1 showed you how to integrate IBM® Sterling Connect:Direct (hereafter called Sterling C:D) with IBM WebSphere® MQ in order to generate MQ messages that can be used by an orchestration engine such as WebSphere Message Broker or any application capable of consuming an MQ event.

[Part 2](#) showed you how to build a WebSphere Message Broker orchestration that correlates file arrival events in a WebSphere Message Broker flow, generates a unique ID to track file transfer details using monitoring tools, and computes a routing endpoint and protocol based on the file contents in the Sterling C:D file.

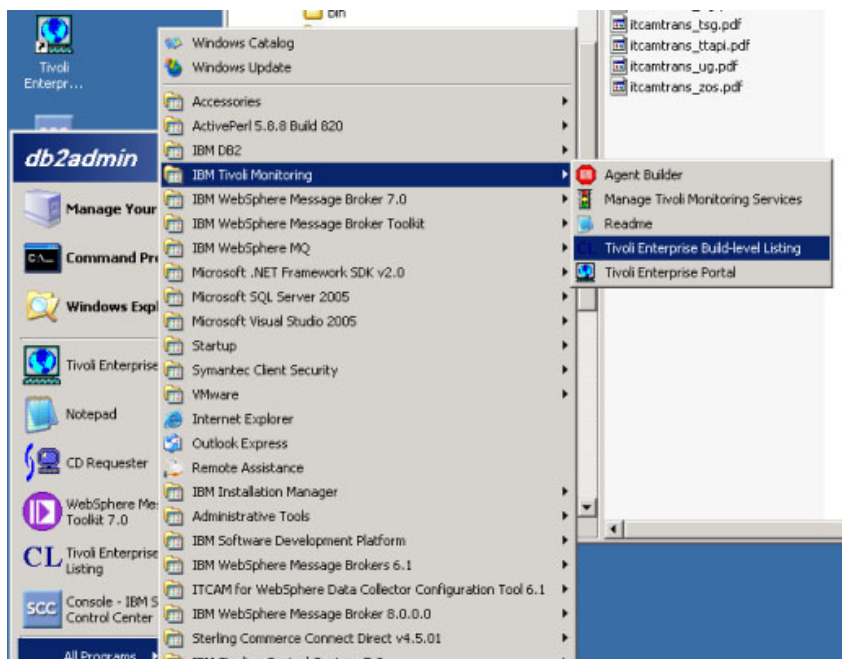
Part 3 will show you how to use IBM Tivoli® Composite Application Manager for Transactions (hereafter called ITCAM for Transactions) to track end-to-end movement of files, transactions, applications, and components using IBM Sterling C:D. It will also show you how to use WebSphere Message Broker to emit transaction data to ITCAM for Transactions, and quickly identify operational and transactional problems involving a money transfer application.

Installing and configuring WebSphere Message Broker and ITCAM for Transactions

Here are the detailed instructions from the information center: [Installing Transaction Tracking components of ITCAM for Transactions V7.3](#). Here is an overview:

1. Download ITCAM for Transactions V7.3 Transaction Tracking for the appropriate platform.
2. After installing ITCAM for Transactions V7.3, make sure that Transaction Collector and Transaction Reporter V7.3 are installed.
3. Verify the installation:

Figure 1. Verify installation of ITCAM for Transactions



4. Search for Transaction Reporter and Transaction Collector in the list of installed Tivoli agents, as shown below:

Figure 2. Search results for Transaction Collector and Transaction Reporter

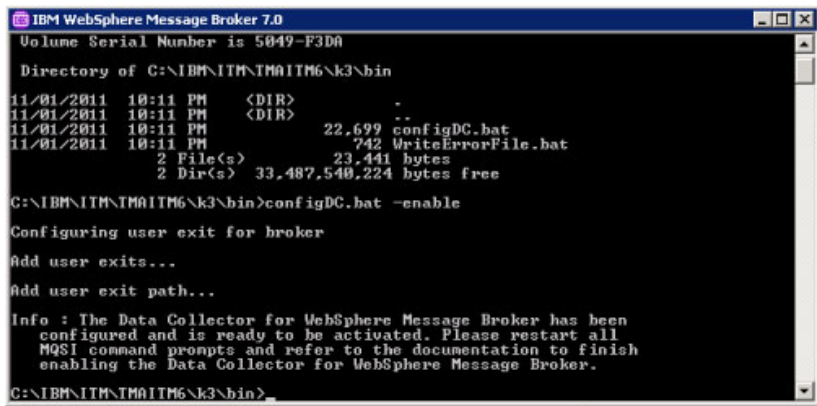
```
Product code TO Title Transaction Reporter
C:\IBMNTM\TOMS - KTOWICMS\I\
Driver TO 07.30.00.00 built 2011/05/18 01:19 platform winnt
...
Product code TU Title Transaction Collector
C:\IBMNTM\TOMS - KTUWICMS\I\
Driver TU 07.30.00.00 built 2011/05/18 01:06 platform winnt
```

Configuring ITCAM for Transactions with WebSphere Message Broker

This section shows you how to install ITCAM for Transactions to monitor transaction details from WebSphere Message Broker.

1. In a command window, change directory to `C:\IBM\ITM\TMAITM6\k3\bin`.
2. Run `configDC.bat -enable`. Make sure the command is successful:

Figure 3. Configuration command execution and results



```

IBM WebSphere Message Broker 7.0
Volume Serial Number is 5049-F3DA

Directory of C:\IBM\ITM\TMAITM6\k3\bin
11/01/2011  10:11 PM  <DIR>      .
11/01/2011  10:11 PM  <DIR>      ..
11/01/2011  10:11 PM             22,699 configDC.bat
11/01/2011  10:11 PM             742 WriteErrorFile.bat
                2 File(s)          23,441 bytes
                2 Dir(s)  33,487,540,224 bytes free

C:\IBM\ITM\TMAITM6\k3\bin>configDC.bat -enable

Configuring user exit for broker
Add user exits...
Add user exit path...

Info : The Data Collector for WebSphere Message Broker has been
       configured and is ready to be activated. Please restart all
       MQSI command prompts and refer to the documentation to finish
       enabling the Data Collector for WebSphere Message Broker.

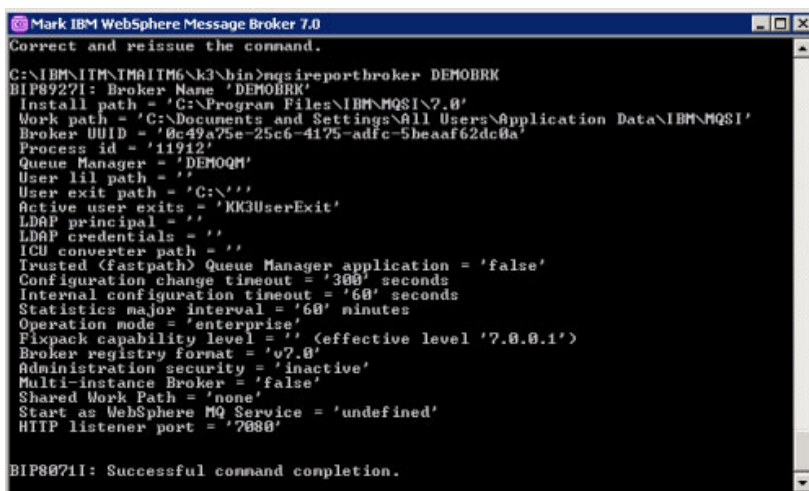
C:\IBM\ITM\TMAITM6\k3\bin>
  
```

3. Using the `mqsichangebroker` command, set the active user exits name as `KK3UserExit`:
`C:\IBM\ITM\TMAITM6\k3\bin>mqsichangebroker <BrokerName> -e KK3UserExit`

Here the user exits are provided by ITCAM, which captures the required transaction data from WebSphere Message Broker.

4. Verify that the user exits are successfully set in WebSphere Message Broker using the command `mqsireportbroker <BrokerName>`, as shown below:

Figure 4. Command execution results for a user exit configuration



```

Mark IBM WebSphere Message Broker 7.0
Correct and reissue the command.

C:\IBM\ITM\TMAITM6\k3\bin>mqsireportbroker DEMOBRK
BIP89271: Broker Name 'DEMOBRK'
Install path = 'C:\Program Files\IBM\MQSI\7.0'
Work path = 'C:\Documents and Settings\All Users\Application Data\IBM\MQSI'
Broker UUID = '0c49a75e-25c6-4175-adfc-5beaf62dc0a'
Process id = '11912'
Queue Manager = 'DEMOQM'
User lib path = ''
User exit path = 'C:\'
Active user exits = 'KK3UserExit'
LDAP principal = ''
LDAP credentials = ''
ICU converter path = ''
Trusted (fastpath) Queue Manager application = 'false'
Configuration change timeout = '300' seconds
Internal configuration timeout = '60' seconds
Statistics major interval = '60' minutes
Operation mode = 'enterprise'
Fixpack capability level = '' (effective level '7.0.0.1')
Broker registry format = 'v2.0'
Administration security = 'inactive'
Multi-instance Broker = 'false'
Shared Work Path = 'none'
Start as WebSphere MQ Service = 'undefined'
HTTP listener port = '7080'

BIP80711: Successful command completion.
  
```

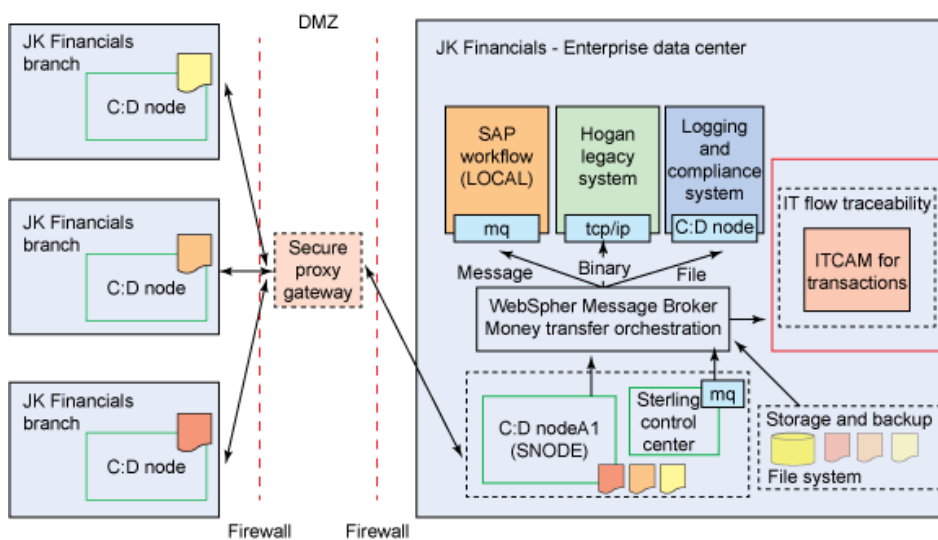
5. Restart WebSphere Message Broker.

Implementing transaction tracking using WebSphere Message Broker and ITCAM for Transactions

Part 1 of this article series explained the JK Financials scenario shown below in Figure 5. This scenario involves transaction tracking using ITCAM for Transactions, and managed file transfer within its enterprise network using IBM Sterling C:D. In this scenario, a branch of JK Financials at a different location from the data center sends money transfer instructions in a file using secure transfer protocol:

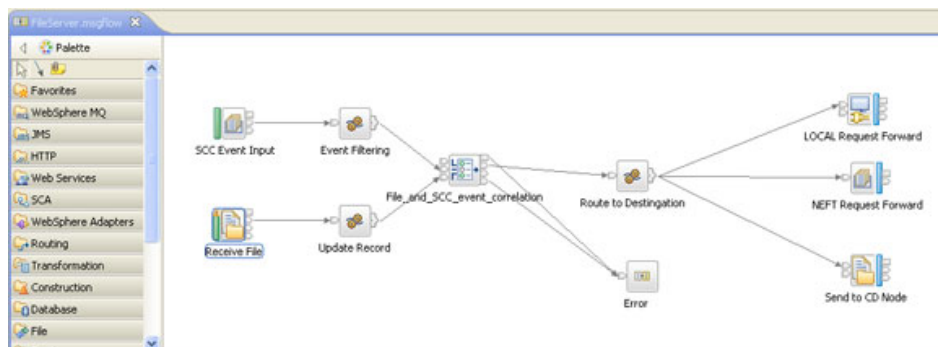
Figure 5. Architectural overview using WebSphere Message Broker

JK Financials - multi-protocol branch integration - money transfer scenario
Component overview and architecture - Using WebSphere Message Broker v6.x



The message flow that implements the banking scenario is shown below. For more details about the flow implementation, see [Part 2 of this article series](#).

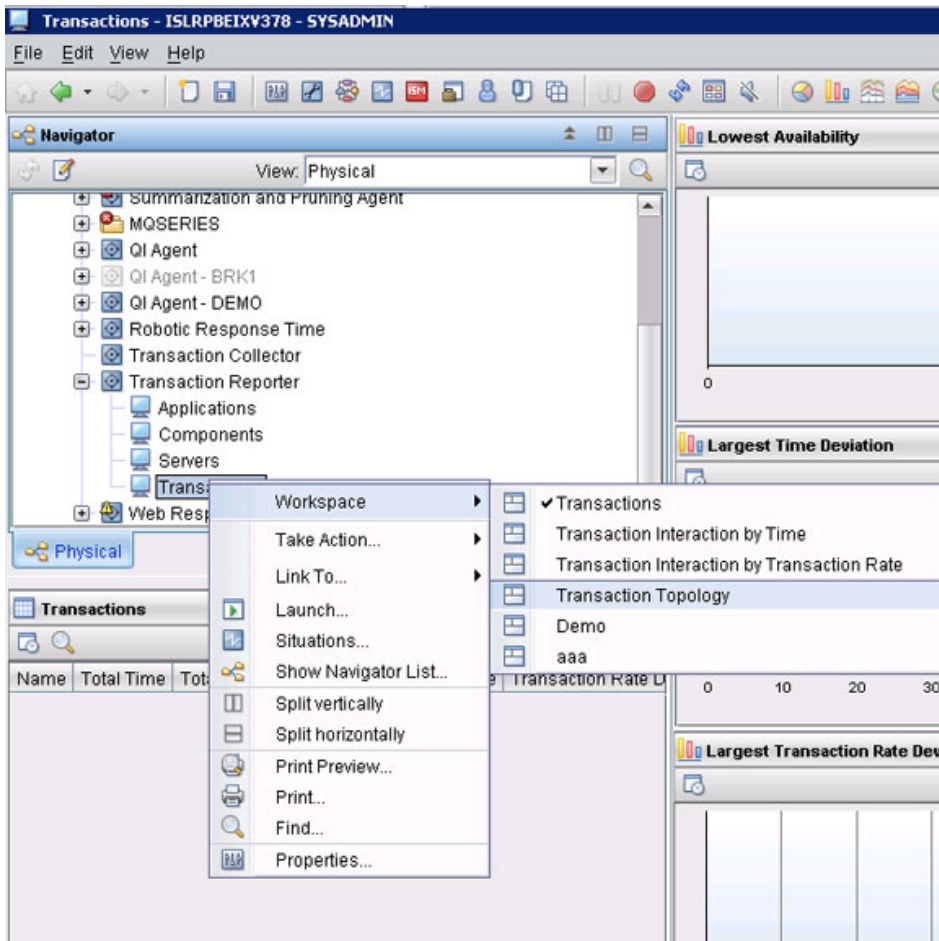
Figure 6. Correlating events of file arrival using WebSphere Message Broker flow



1. Deploy the message flow shown in Figure 5 on the ITCAM configured WebSphere Message Broker execution group.
2. Run the message flow as explained in [Part 2 of this article series](#).

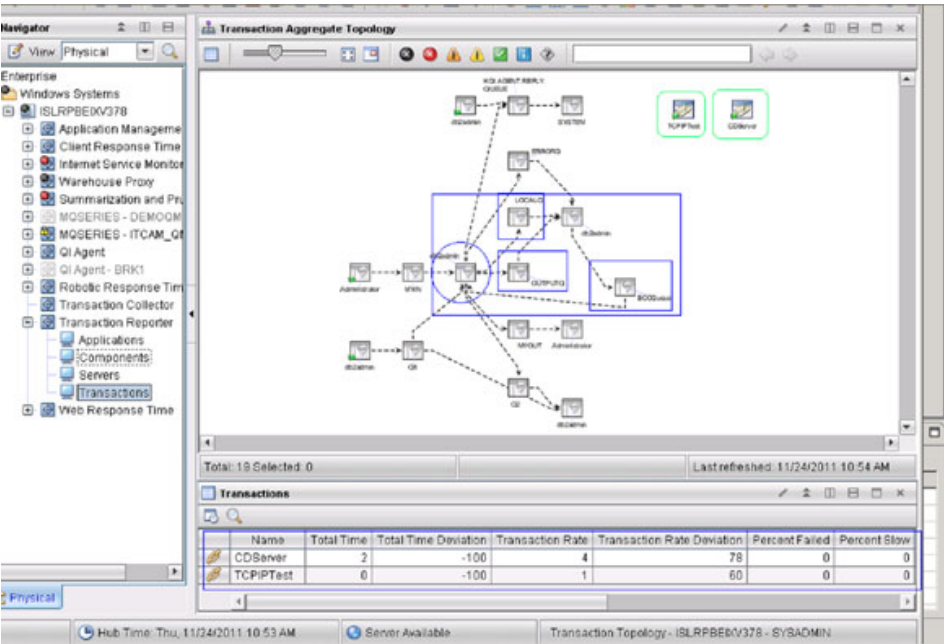
3. Open Tivoli Enterprise portal and then open the Transaction Topology workspace.
4. Launch Tivoli Enterprise Portal and then open the Transaction Reporter workspace: Select **Transaction Reporter => Transactions => Transaction Topology**:

Figure 7. Transaction Viewer in Tivoli Enterprise Portal



The Transaction Aggregate Topology captures the transaction processing path through various application components, and it shows you how different applications in the system interact to process the transaction. Figure 8 shows the Transaction Aggregate Topology view:

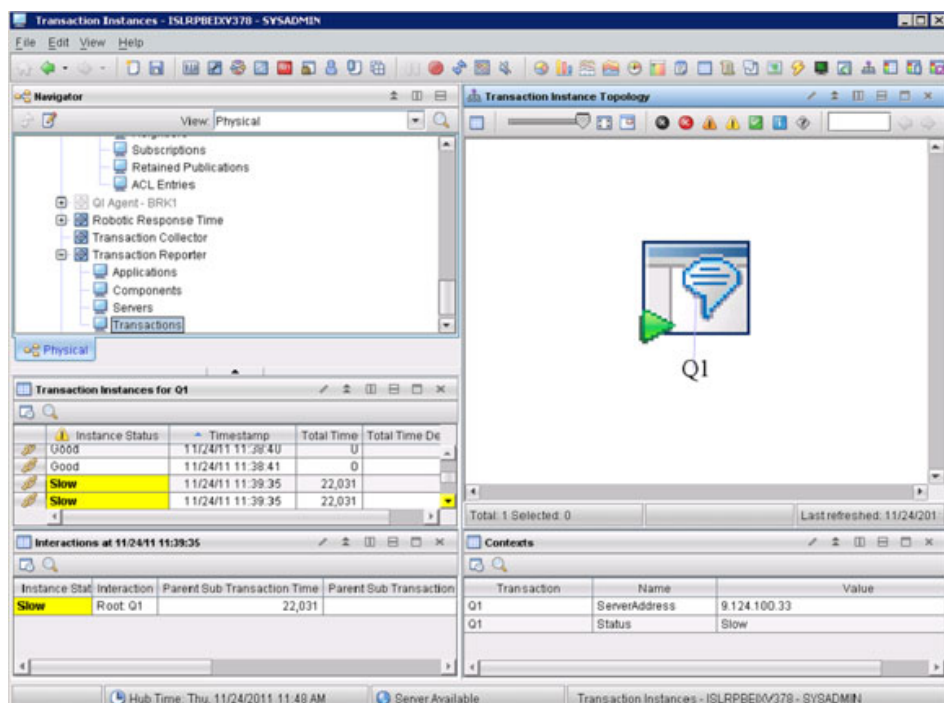
Figure 8. Transaction Aggregate Topology view in Transaction Reporter



In the Transaction topology workspace, you can see how transactions flow through various application end points connected to a WebSphere Message Broker orchestration. Here the event message comes through SCCQueue, goes to a message flow called db2admin, and then goes to either OUTPUTQ or ERRORQ.

In this topology, you can see individual transaction instances captured by ITCAM, check transaction performance, and view details about individual transactions and context views, as shown below in Figure 9. As you can see, there are some slow transactions, which you can act on in order to meet service-level agreements (SLAs).

Figure 9. Transaction instances view in ITCAM for Transactions

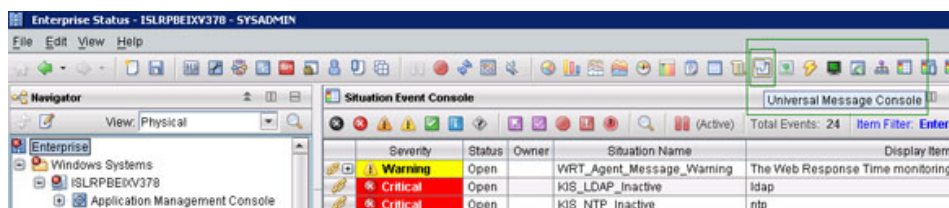


Similarly, you can view the Application Topology and Component Topology workspaces to view the applications and components involved in a transaction in a topology diagram. If any application is down, then ITCAM will detect it and show an alert.

You can use the Situation Editor and set some criteria to show an alert in ITCAM using the Universal Message Console. For example, you could specify that if the currentDepth of a queue on the Message Broker queue ERRORQ is greater than 0, then show an alert. In the Transaction Topology view, you can select ERRORQ as an application endpoint where a failure message arrives, and send a Short Message Service (SMS) text. An SMS gateway needs to be installed, and then you run a command to send an SMS text when a message arrives in the ERRORQ. You can configure alerts like these, as shown below.

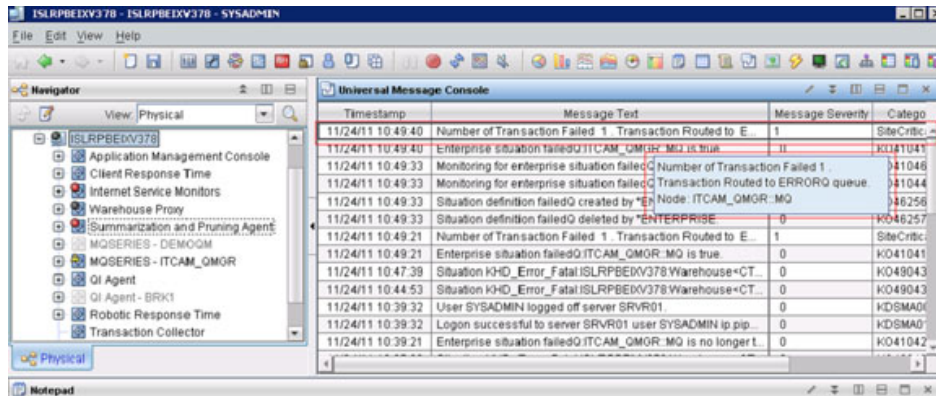
The Situation Event Console below shows various system alerts classified as "Warning" and "Critical," which helps the administrator of ITCAM for Transactions react appropriately.

Figure 10. Situation Event Console for alerts



The Universal Message Console shows customized situation events. Here a situation event is created to show an alert when any transaction message is routed to ERRORQ. The current depth property of ERRORQ is used to capture such event. If the current depth of ERRORQ exceeds 0, the Situation Handler sends an alert to the Universal Message Console:

Figure 11. Viewing failed transactions in Universal Message Console

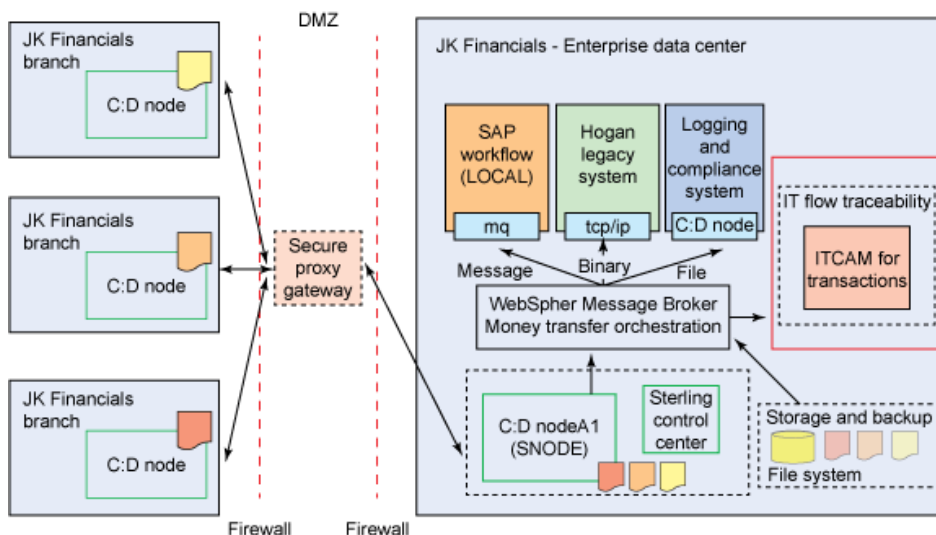


Viewing transactions using ITCAM for Transactions

Part 1 of this article series explained the architecture of the JK Financials multi-protocol branch integration using WebSphere Message Broker. Figure 12 shows the overall architecture:

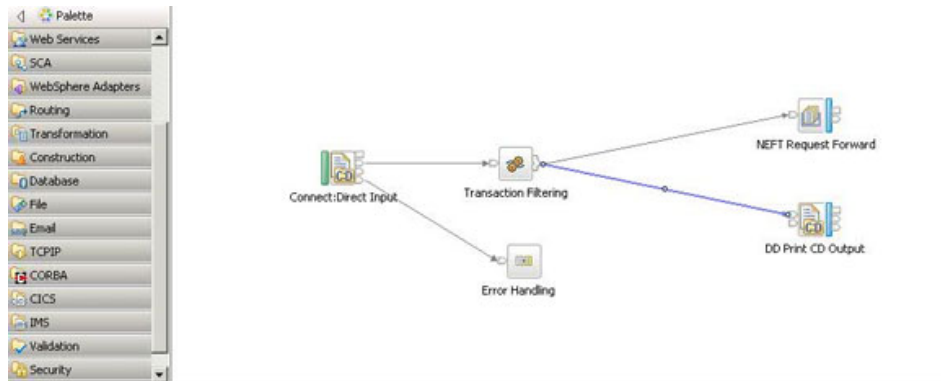
Figure 12. Component overview using WebSphere Message Broker

JK Financials - multi-protocol branch integration - money transfer scenario
Component overview and architecture - Using WebSphere Message Broker v7.x



Part 2 of this article series explained the message flow for the scenario, as shown below:

Figure 13. Simplified WebSphere Message Broker orchestration



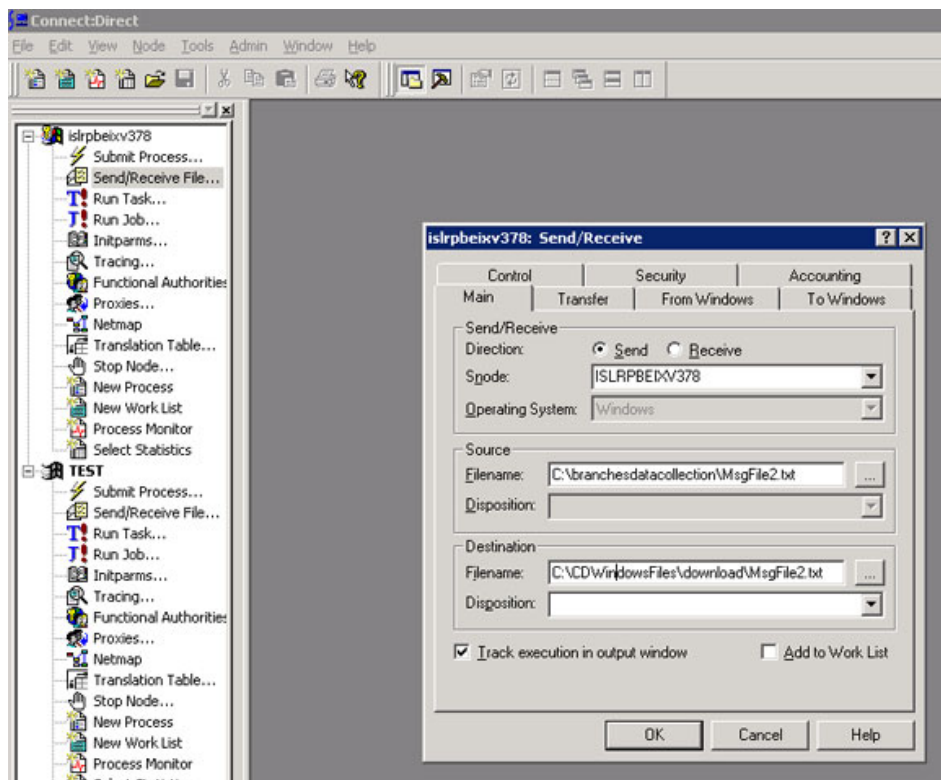
This flow receives the file transferred by different bank branches to the enterprise data center at CDNode A1. The input messages file details are provided in Part 1. It processes all transactions available in each file. The CDInput node sends each transaction in the file to the next compute node, called BaseOnRequestTypeRouteIndividualTransaction. This compute node routes the messages to either the NEFT queue or DD Print endpoints, based on the RequestType field value in the input message.

The message flow is deployed and transaction data files are sent to the enterprise data center using the Sterling C:D Server available at different bank branches. The Sterling C:D secondary node at the enterprise data center receives the data files from all branches. The CDInput node fetches the data file received at the enterprise data center after it is reprocessed through the local Sterling C:D node at the enterprise data center.

Running the message flow

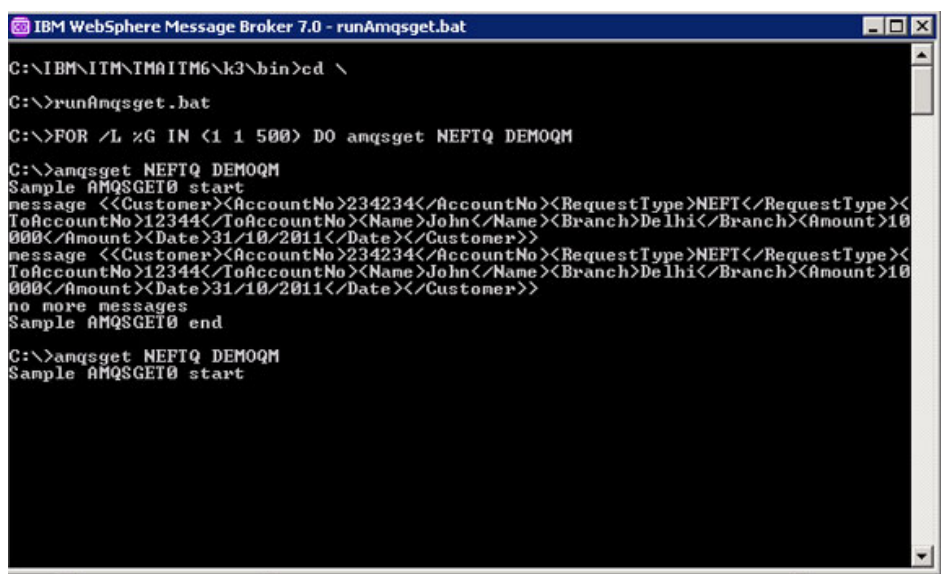
After deploying the message flow in the execution group, open the Sterling C:D Requester Tool and process `MsgFile2.txt` from the `branchesdatacollection` folder messages into the `CDWindows\download\` folder. Send all files from different branches using the Send/Receive File Utility of the Sterling C:D node:

Figure 14. Testing end-to-end flow using Sterling C:D Console



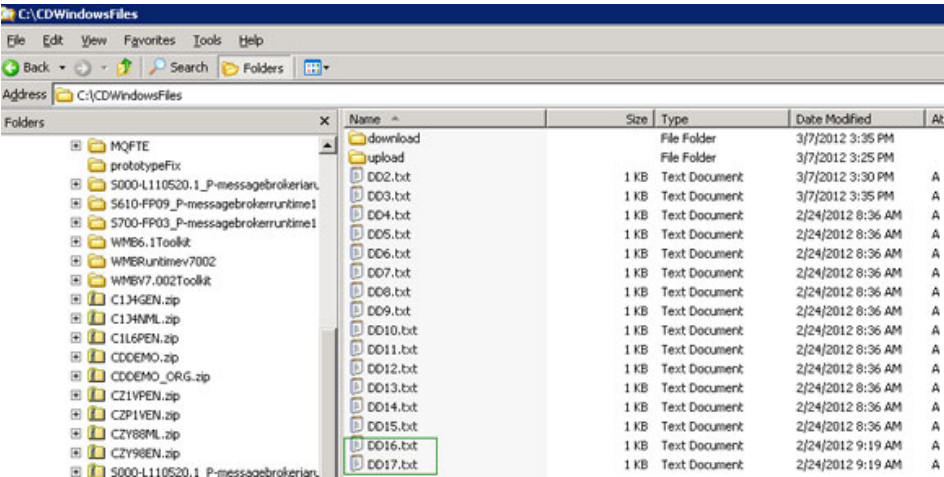
Use the amqsget utility to receive any messages routed to the NEFF queue as shown in Figure 15 below. Two DDn.txt messages will be stored in the file system.

Figure 15. Running amqsget utility to send sample data file



The two other transactions on DD print will be stored in the file system using the CDOOutput node. For the two transaction requests of type DD, the message is routed to the CDOOutput node, where it generates DD16.txt and DD17.txt files in the output file system folder, as shown below:

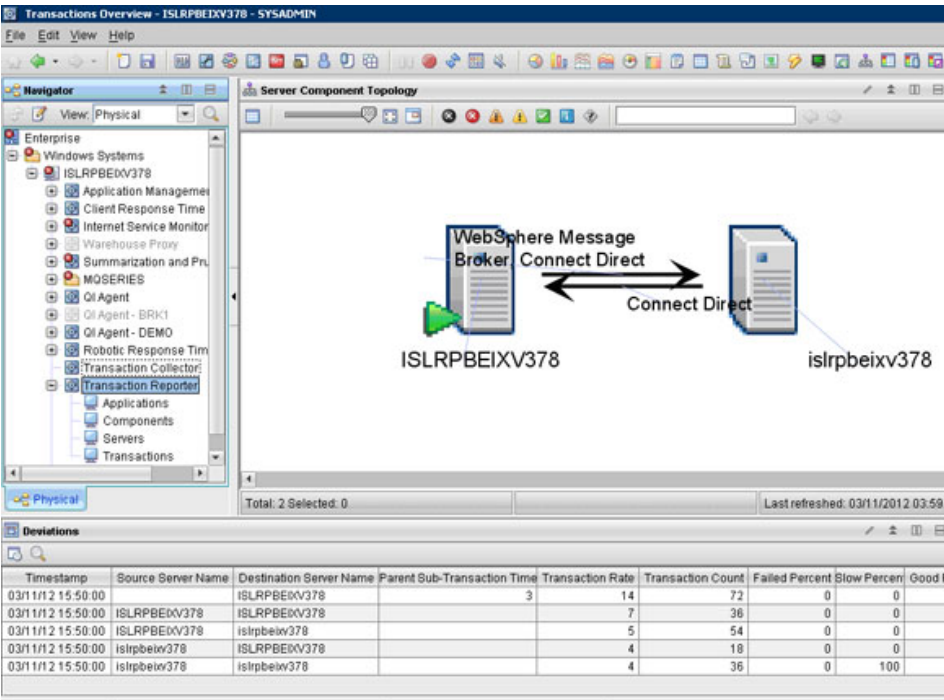
Figure 16. Testing arrival of a file in Sterling C:D configured node folder



After processing the messages through the Sterling C:D Server and the WebSphere Message Broker flow, open the Tivoli Enterprise Portal, using userid sysadmin and password sysadmin.

Go to Transaction Reporter. The Sterling C:D Node connected to the broker system is shown. Here it can detect a remote C:D node also and show the C:D node details. Figure 17 shows that the broker system ISLRPBEIXV378 is connected to C:D node islrpbeixv378. The C:D server node is a local node. It will also show the remote C:D server node connection if any file transfer happens through a remote C:D server node to the broker system ISLRPBEIXV378.

Figure 17. Server Component Topology view



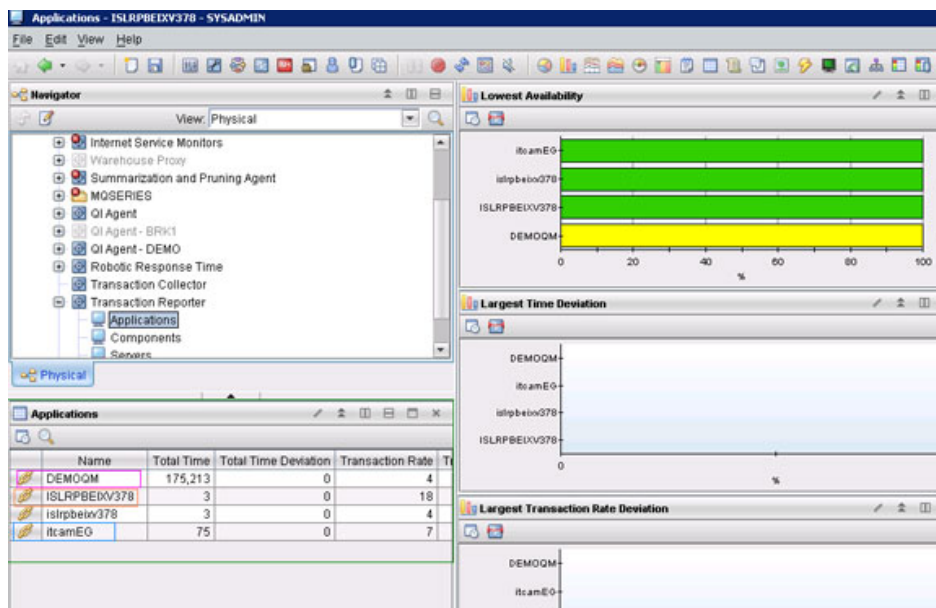
Viewing the application in ITCAM

Three applications have been detected:

- DEMOQM -- WebSphere MQ application
- itcamEG -- WebSphere Message Broker application execution group process
- ISLRPBEIXV378 -- Sterling C:D node server application

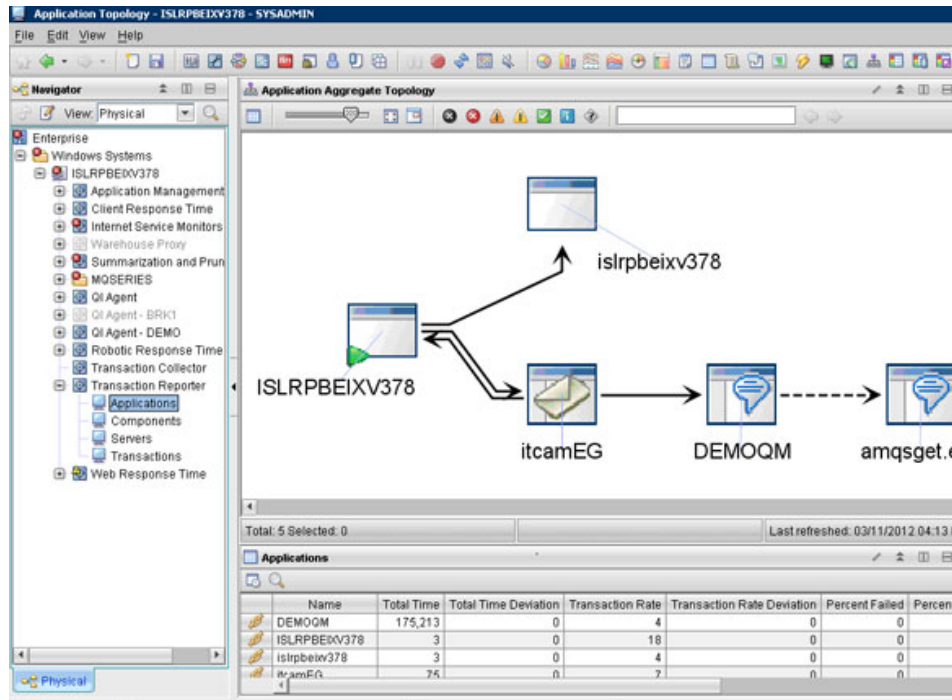
In Figure 18, the Applications panel on the left shows four applications: DEMOQM, ISLRPBEIXV378, islrpbeixv378, and itcamEG. It shows the availability of these applications in the Lowest Availability panel on the right. The three green bars show that three applications are running well, while the yellow bar indicates that DEMOQM has a performance issue and message processing through it is slow:

Figure 18. Applications detected by ITCAM for Transactions



1. Open the Application Aggregate Topology workspace to view how these composite applications are connected.
2. In the Tivoli Enterprise Portal, select **Transaction Reporter => Applications => Workspace => Application Topology**.
3. The Application Topology view is shown below. The Application Aggregate Topology shows you how various applications are wired together in the overall solution:

Figure 19. Application Aggregate Topology view



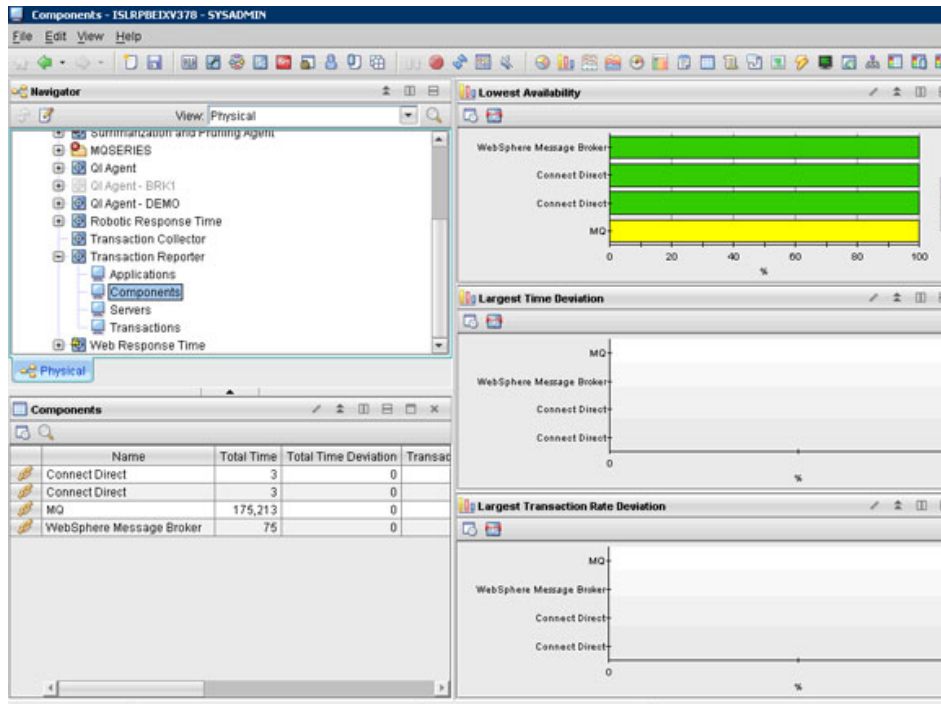
4. In the Application Topology workspace view, you see that a Sterling C:D Server ISLRPBEIXV378 is connected to the WebSphere Message Broker execution group itcamEG, which is connected to the WebSphere MQ application DEMOQM. DEMOQM is lightly coupled with amqsget.exe, which consumes the NEFT messages from the queue. The execution group itcamEG application is connected to the Sterling CD server islrpbeixv378 to create an output file for any DD type request.

Viewing the Components in ITCAM

Select the components in the Transaction Reporter, as shown below in the Tivoli Enterprise Portal.

In the Components view, you see that it has detected four components: two Sterling C:D, one WebSphere Message Broker, and one WebSphere MQ.

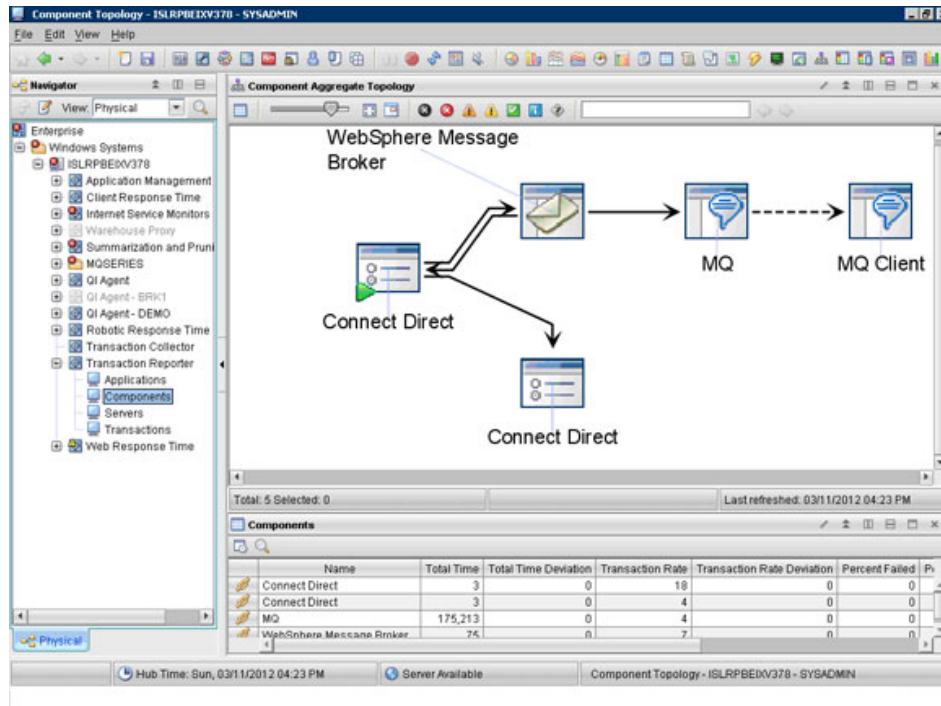
In Figure 20, the Components panel on the left shows the four components detected by the Transaction Collector and Transaction Reporter. The Lowest Availability panel on the right shows the status of these components:

Figure 20. Components view in Transaction Reporter

To see how these components are aggregated, perform the following steps:

1. Select **Transaction Reporter => Components => Workspace => Component Topology**.
2. The Component Aggregate Topology shows you how various application components are wired together and running in WebSphere Message Broker, as shown in Figure 21. The Sterling C:D Server is the starting point that sends the file to WebSphere Message Broker, which picks up the transactions in the transferred file and routes them to either WebSphere MQ or C:D components. The WebSphere MQ component puts the message into a queue, while the C:D component creates an output file. Finally, the WebSphere MQ client component consumes the messages in the message queue. The dotted line shows that the WebSphere MQ client is loosely coupled, while the other components are tightly coupled:.

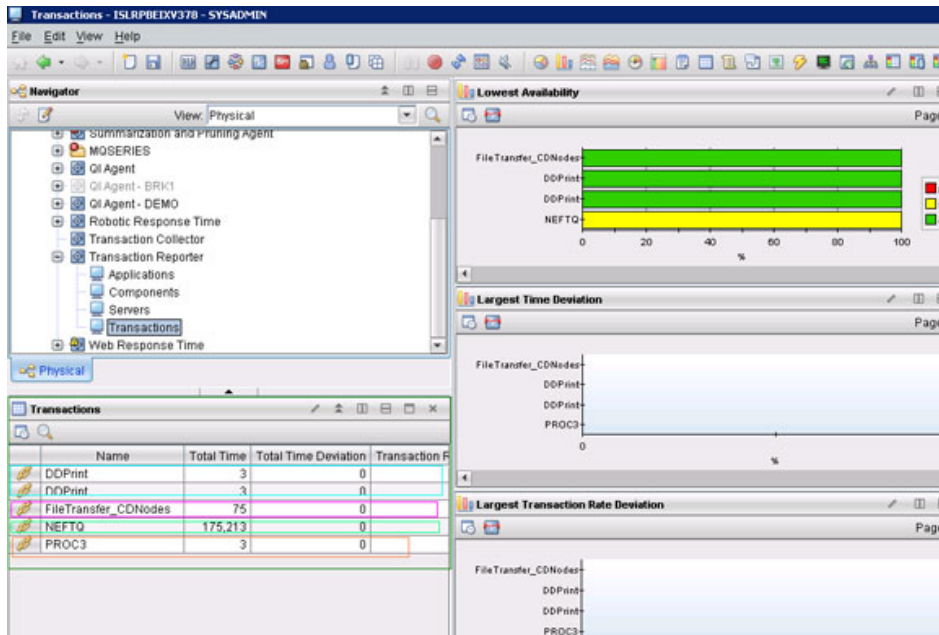
Figure 21. Component Aggregate Topology view



Viewing the transaction in ITCAM

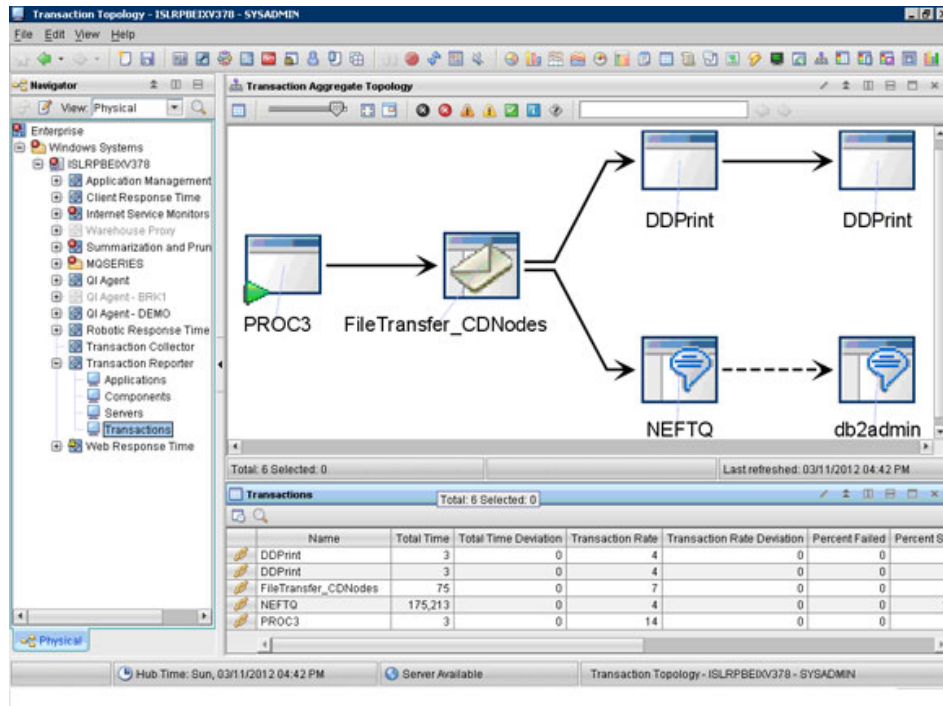
The Transaction view lets you view transaction performance at five minute intervals, and you can adjust this polling interval.

1. Open the Tivoli Enterprise Portal and select **Transaction Reporter => Transaction**.
2. There are four transaction endpoints as shown below in Figure 22 in different colored boxes. PROC3 is the process that initiates the transaction and starts the file transfer. FileTransfer_CDNodes is the flow that consumes the file transfer started from PROC3 and routes messages inside the file to NEFTQ and two messages to DDPrint, since the Sterling C:D server created two different processes to create a DDPrint output file. The Transactions panel on the left contains columns for different performance evaluation parameters. Total Time shows the time elapsed in each of these application components:

Figure 22. Transaction view in Transaction Reporter

- To view details about transaction performance in these application components, open the Transaction Aggregate Topology view: Select **Transaction Reporter => Workspace => Transaction Topology**.
- In this scenario, the Sterling C:D server process PROC3 sends a file having multiple transaction records to the message flow FileTransfer_CDNodes, which routes the transaction records to either NEFTQ (a WebSphere MQ queue) or DDPrint process (a Sterling C:D process to create the output file). The Transaction Aggregate Topology view shows how these individual applications are wired as part of the overall transaction processing system. The topology is detected automatically by the Transaction Collector and Transaction Reporter of ITCAM for Transactions.
- In Figure 23 below, the ITCAM Transaction Reporter detects the file transfer initiated by the Sterling C:D server process PROC3. It reaches the WebSphere Message Broker flow FileTransfer_CDNodes, and then the message is routed either to the CDOOutput node or to the MQOutput node in the message flow, based on the message contents. Either the CDOOutput node's Sterling C:D Server process DDPrint writes the output file to the file system, or else the MQOutput node puts the message to NEFTQ. In Figure 23, the FileTransfer_CDNodes message flow contains two routing paths: DDPrint and NEFTQ. The amqsget application shown as db2admin consumes the messages that arrive in the NEFTQ output queue:

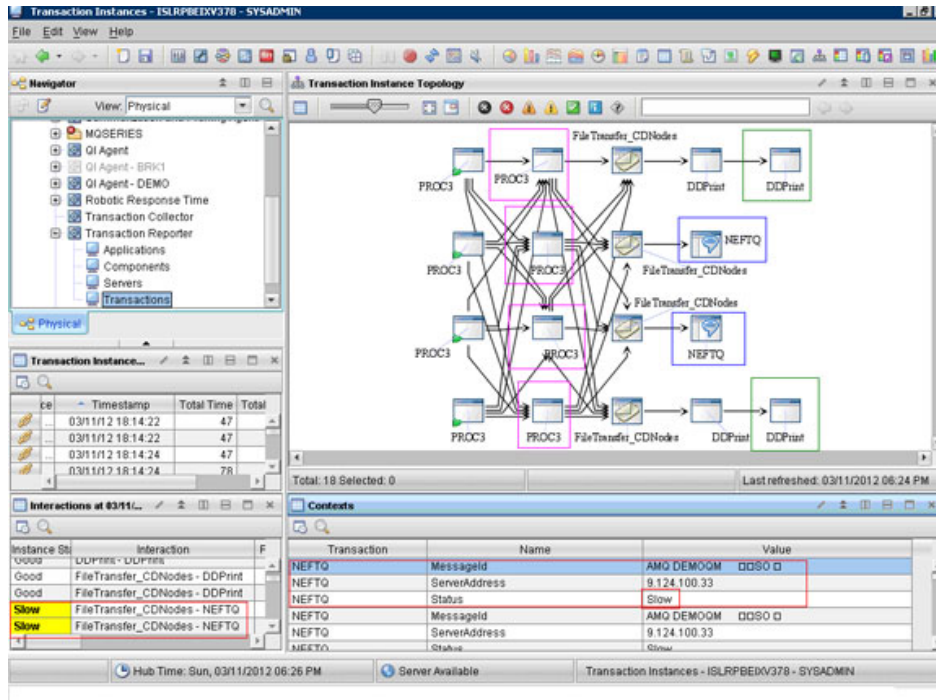
Figure 23. Transaction Aggregate Topology view



Viewing the transaction instance

An individual transaction instance can help you identify details about a specific transaction, so that you can query on what happened to an individual transaction using a `localtransactionId` or sub-transaction, in case a transaction contains multiple sub-transactions. These kinds of queries can help you maintain security and SLAs. For example, if a message is routed to an error queue or dead letter queue instead of to the desired destination, you can diagnose and correct the problem and resend the request.

In this scenario, a file contains four transactions, which get processed and routed individually to various destinations by the WebSphere Message Broker flow. In the individual Transaction Instance view, you can see that the transaction file transferred by the bank contains four transaction records, shown below in pink. Two of them (shown in green) are routed to DDPrint, and two (shown in blue) are routed to NEFTQ. The view also shows that the NEFTQ transaction is slow, and therefore you should take some corrective action to maintain the SLA.

Figure 24. Transaction Instance Topology view

The CDInput node of the message flow FileTransfer_CDNodes reads each of the four messages individually, and the message flow Compute node routes two of them to NEFTQ and two of them to DDPrint to a file to print the demand draft. The slow transaction on the WebSphere MQ NEFT queue is shown as an alert in yellow, indicating that WebSphere MQ has a performance issue. If the file contains more transaction records, then the destination of each transaction record is shown in the Transaction Instance Topology view.

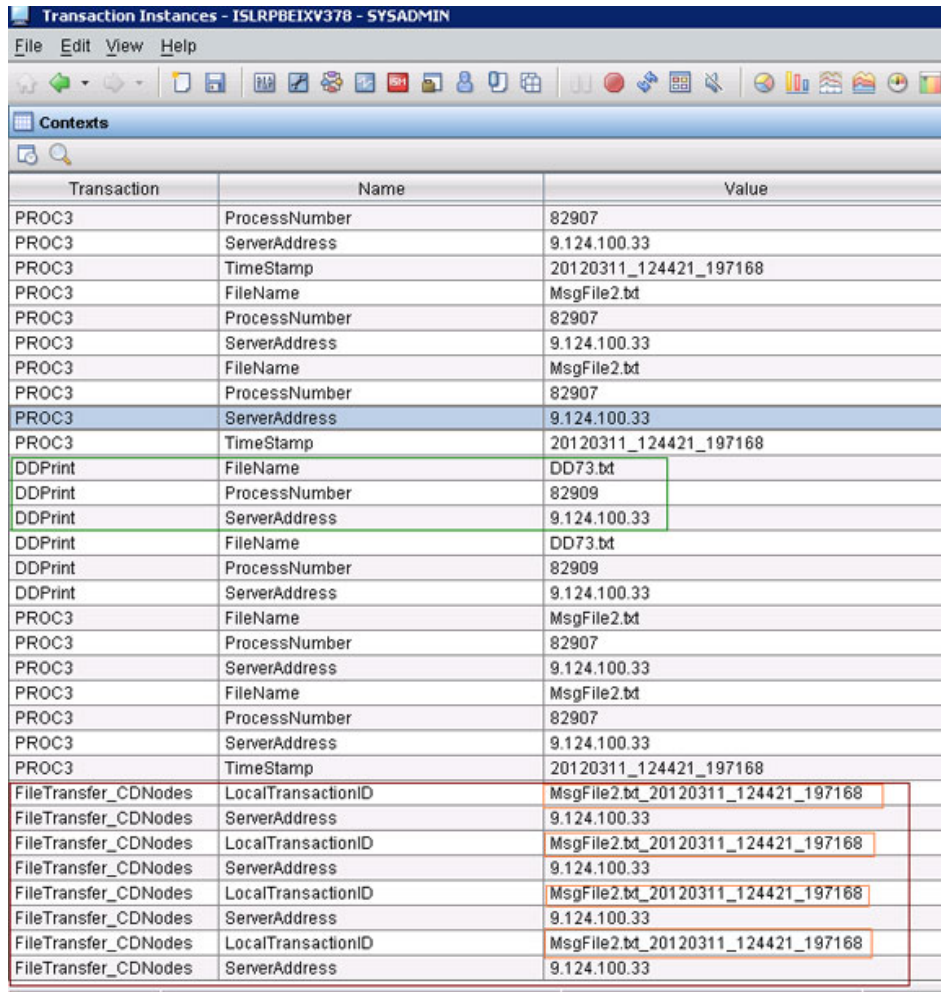
Transaction Contexts view

The Transaction Contexts view is part of the Transaction Instance Topology view, and it shows details about an individual input or output transaction instance in a table.

In the Contexts view shown in Figures 25 and 26, you see that the transaction instance contains a file named `MsgFile2.txt` transferred by the Sterling C:D Server PROC3 (in pink) and it comes to the WebSphere Message Broker flow as a transaction. `MsgFile2.txt` contains four sub-transaction messages identified by LocalTransactionID: `MsgFile2.txt_20120311_124421_197168`. Due to a limitation in ITCAM, the individual sub-transaction ID is not getting overridden or populated, and therefore four sub-transactions are shown with the same ID. However, the LocalTransactionID shows where all four transactions have gone -- two of them to NEFTQ with respective WebSphere MQ MessageId's (shown in blue), and two to the DDPrint application (shown in green), which is a Sterling C:D application and creates output files named `DD72.txt` and `DD73.txt`.

Figure 25. Contexts table view of individual transaction instance

Transaction	Name	Value
NEFTQ	MessageId	AMQ DEMOQM □□S□ □
NEFTQ	ServerAddress	9.124.100.33
NEFTQ	Status	Slow
NEFTQ	MessageId	AMQ DEMOQM □□S□ □
NEFTQ	ServerAddress	9.124.100.33
NEFTQ	Status	Slow
PROC3	FileName	MsgFile2.bt
PROC3	ProcessNumber	82907
PROC3	ServerAddress	9.124.100.33
PROC3	FileName	MsgFile2.bt
PROC3	ProcessNumber	82907
PROC3	ServerAddress	9.124.100.33
PROC3	TimeStamp	20120311_124421_197168
DDPrint	FileName	DD72.bt
DDPrint	ProcessNumber	82908
DDPrint	ServerAddress	9.124.100.33
DDPrint	FileName	DD72.bt
DDPrint	ProcessNumber	82908
DDPrint	ServerAddress	9.124.100.33
PROC3	FileName	MsgFile2.bt
PROC3	ProcessNumber	82907
PROC3	ServerAddress	9.124.100.33
PROC3	FileName	MsgFile2.bt
PROC3	ProcessNumber	82907
PROC3	ServerAddress	9.124.100.33
PROC3	TimeStamp	20120311_124421_197168
PROC3	FileName	MsgFile2.bt
PROC3	ProcessNumber	82907
PROC3	ServerAddress	9.124.100.33
PROC3	FileName	MsgFile2.bt
PROC3	ProcessNumber	82907

Figure 26. Contexts table view of Individual transaction instance


Transaction	Name	Value
PROC3	ProcessNumber	82907
PROC3	ServerAddress	9.124.100.33
PROC3	TimeStamp	20120311_124421_197168
PROC3	FileName	MsgFile2.bt
PROC3	ProcessNumber	82907
PROC3	ServerAddress	9.124.100.33
PROC3	FileName	MsgFile2.bt
PROC3	ProcessNumber	82907
PROC3	ServerAddress	9.124.100.33
PROC3	TimeStamp	20120311_124421_197168
DDPrint	FileName	DD73.bt
DDPrint	ProcessNumber	82909
DDPrint	ServerAddress	9.124.100.33
DDPrint	FileName	DD73.bt
DDPrint	ProcessNumber	82909
DDPrint	ServerAddress	9.124.100.33
PROC3	FileName	MsgFile2.bt
PROC3	ProcessNumber	82907
PROC3	ServerAddress	9.124.100.33
PROC3	FileName	MsgFile2.bt
PROC3	ProcessNumber	82907
PROC3	ServerAddress	9.124.100.33
PROC3	TimeStamp	20120311_124421_197168
FileTransfer_CDNodes	LocalTransactionID	MsgFile2.bt_20120311_124421_197168
FileTransfer_CDNodes	ServerAddress	9.124.100.33
FileTransfer_CDNodes	LocalTransactionID	MsgFile2.bt_20120311_124421_197168
FileTransfer_CDNodes	ServerAddress	9.124.100.33
FileTransfer_CDNodes	LocalTransactionID	MsgFile2.bt_20120311_124421_197168
FileTransfer_CDNodes	ServerAddress	9.124.100.33
FileTransfer_CDNodes	LocalTransactionID	MsgFile2.bt_20120311_124421_197168
FileTransfer_CDNodes	ServerAddress	9.124.100.33

To achieve the transaction tracking in WebSphere Message Broker, ITCAM for Transactions provides a user exit that instruments its code into the WebSphere Message Broker runtime, and fetches the necessary transaction information, and displays it in various Transaction Reporter workspace views. The Transaction Collector uses the WebSphere Message Broker user exit to collect various transaction messages processed through the deployed message flow, and Transaction Reporter then shows the activities in its various workspaces.

Conclusion

This article showed you how to use ITCAM for Transactions to trace a transaction that is processed through IBM Sterling Connect:Direct, WebSphere Message Broker, and WebSphere MQ. ITCAM for Transactions can help you meet SLAs by showing alerts for slow transactions and enabling you to act before an SLA is violated.

This article series showed you how to integrate IBM Sterling Connect:Direct and WebSphere Message Broker to provide end-to-end visibility of transactions involving files and application-specific data across different protocols and networks.

Related topics

- **Sterling, Tivoli, and WebSphere Message Broker resources**

- [Documentation for IBM Sterling Connect:Direct](#)
- [Installing IBM Tivoli Composite Application Manager for Transactions using the Installer](#)
- [IBM Tivoli Composite Application Manager for Transactions information center](#)
- [Configuring Data Collector for WebSphere Message Broker using IBM Tivoli Composite Application Manager for Transactions V7.3](#)
- [WebSphere Message Broker V8 information center](#)

A single Web portal to all WebSphere Message Broker V8 documentation, with conceptual, task, and reference information on installing, configuring, and using your WebSphere Message Broker environment.
- [WebSphere Message Broker developer resources page](#)

Technical resources to help you use WebSphere Message Broker for connectivity, universal data transformation, and enterprise-level integration of disparate services, applications, and platforms to power your SOA.
- [WebSphere Message Broker product page](#)

Product descriptions, product news, training information, support information, and more.
- [Download free trial version of WebSphere Message Broker](#)

WebSphere Message Broker is an ESB built for universal connectivity and transformation in heterogeneous IT environments. It distributes information and data generated by business events in real time to people, applications, and devices throughout your extended enterprise and beyond.
- [WebSphere Message Broker documentation library](#)

WebSphere Message Broker specifications and manuals.
- [WebSphere Message Broker forum](#)

Get answers to your technical questions and share your expertise with other WebSphere Message Broker users.
- [WebSphere Message Broker support page](#)

A searchable database of support problems and their solutions, plus downloads, fixes, and problem tracking.
- [Youtube tutorial: Integrating Microsoft .NET code in a WebSphere Message Broker V8 message flow](#)

This five-minute youtube tutorial shows you how simple it is to use WebSphere Message Broker V8 to build a message flow that includes Microsoft .NET code. Microsoft Visual Studio is used to build .NET code in C#, which is then integrated into a message flow using Message Broker and an HTTP RESTful interface.
- [WebSphere Message Broker V8 Development Training](#)

In this IBM Training course, you will learn about the components of the WebSphere Message Broker development and runtime environments. The course also explores message flow problem determination, and shows you how to construct message flows that use ESQL, Java, and PHP to transform messages.

- **WebSphere resources**

- [developerWorks WebSphere developer resources](#)

Technical information and resources for developers who use WebSphere products. developerWorks WebSphere provides product downloads, how-to information, support resources, and a free technical library of more than 2000 technical articles, tutorials, best practices, IBM Redbooks, and online product manuals.

- [developerWorks WebSphere application integration developer resources](#)
How-to articles, downloads, tutorials, education, product info, and other resources to help you build WebSphere application integration and business integration solutions.
- [developerWorks WebSphere business process management developer resources](#)
WebSphere BPM how-to articles, downloads, tutorials, education, product info, and other resources to help you model, assemble, deploy, and manage business processes.
- [Most popular WebSphere trial downloads](#)
No-charge trial downloads for key WebSphere products.
- [WebSphere forums](#)
Product-specific forums where you can get answers to your technical questions and share your expertise with other WebSphere users.
- [WebSphere on-demand demos](#)
Download and watch these self-running demos, and learn how WebSphere products and technologies can help your company respond to the rapidly changing and increasingly complex business environment.
- [developerWorks WebSphere weekly newsletter](#)
The developerWorks newsletter gives you the latest articles and information only on those topics that interest you. In addition to WebSphere, you can select from Java, Linux, Open source, Rational, SOA, Web services, and other topics. Subscribe now and design your custom mailing.
- [WebSphere-related books from IBM Press](#)
Convenient online ordering through Barnes & Noble.
- [WebSphere-related events](#)
Conferences, trade shows, Webcasts, and other events around the world of interest to WebSphere developers.
- **developerWorks resources**
 - [Trial downloads for IBM software products](#)
No-charge trial downloads for selected IBM® DB2®, Lotus®, Rational®, Tivoli®, and WebSphere® products.
 - [developerWorks blogs](#)
Join a conversation with developerWorks users and authors, and IBM editors and developers.
 - [developerWorks cloud computing resources](#)
Access the IBM or Amazon EC2 cloud, test an IBM cloud computing product in a sandbox, see demos of cloud computing products and services, read cloud articles, and access other cloud resources.
 - [developerWorks tech briefings](#)
Free technical sessions by IBM experts to accelerate your learning curve and help you succeed in your most challenging software projects. Sessions range from one-hour virtual briefings to half-day and full-day live sessions in cities worldwide.
 - [developerWorks podcasts](#)

Listen to interesting and offbeat interviews and discussions with software innovators.

- [developerWorks on Twitter](#)

Check out recent Twitter messages and URLs.

- [IBM Education Assistant](#)

A collection of multimedia educational modules that will help you better understand IBM software products and use them more effectively to meet your business requirements.

© Copyright IBM Corporation 2012

(www.ibm.com/legal/copytrade.shtml)

[Trademarks](#)

(www.ibm.com/developerworks/ibm/trademarks/)