



IBM Software Group

## WebSphere® Business Monitor V6.1.2

*Integrated model debugger for IBM  
WebSphere Business Monitor 6.1*



@business on demand.

© 2008 IBM Corporation  
Updated August 8, 2008



This presentation should introduce you to the new monitor model debugger, which allows you to debug problems in your monitor models.

## Agenda

- Overview
- Installing
- Launching
- Debug client
- Sending events




This is the agenda for this presentation. You will see an overview of the debugger, along with how to install it. In addition you will review the debug client and how to launch it and how to use it, including how to send events using the integrated test client.



## Overview



- A visual debugger that allows you to troubleshoot the root cause of a problem while running a monitor model.
  - ▶ Values not shown in the dashboard
  - ▶ Incorrect values in the dashboard
  - ▶ Outbound events are not being sent
- You can step through the processing of the model logic when an event arrives. You can set breakpoints, examine the values of event fields and metrics.
- The debugger provides a graphical representation of all elements in a monitor model, such as *metrics*, *triggers*, *stopwatches*, *counters*. And you can also see the sequence of steps that are processed as events are being consumed.



Integrated model debugger 3  
© 2008 IBM Corporation


The debugger tool is integrated with the Monitor development toolkit and is used to debug monitor models as they run on the Monitor test server. It is very helpful when you have a monitor model that is not processing events because the correlation or filter criteria are setup incorrectly. The debugger is also useful when the monitored data appears to be incorrect in your dashboards or if you have missing outbound events.

The debugger provides a graphical user interface so that you can step through the generated code. You can set breakpoints anywhere in the model, and examine values of metrics and event data.



## Restrictions

- No support for JMS and CEI.
- Data resulting from debugger processing is not visible in the dashboard.
- User-defined XPath functions are tested and debugged before their use in a monitor model.




Integrated model debugger © 2008 IBM Corporation 4

This slide lists a few important restrictions that you should be aware of.

You must submit events using the integrated test client and this submits events directly to the debugger without using JMS or CEI.


Persistence is not supported so metrics, measures and KPIs which are created in the debug session are not saved in the database and therefore they are not visible using the dashboards.

If you have any user defined functions in your model, you should validate them outside of the monitor model debugger, since this debugger validates monitor models only and it will not debug Java code.

IBM Software Group 


## Installing

- The debugger is always installed with the Monitor development toolkit. The development and test environments must be present for the debugger to be usable.
- The debugger cannot be installed on a production server.
- The installation involves installing two Java EE applications in the Monitor test server and plug-ins for the monitor model editor.
- Monitor toolkit 6.1 or 6.1.1 cannot be updated to include the debugger, so you need to upgrade to Monitor toolkit 6.1.2.

 5  
Integrated model debugger © 2008 IBM Corporation

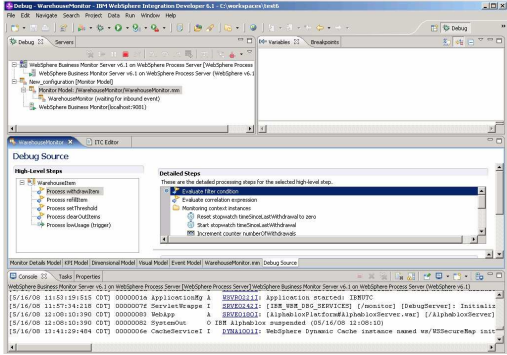
When you install the Monitor toolkit, both the integrated development environment and the integrated test environment must be present for the debugger to be usable. Also, the integrated server must be marked as a development server, otherwise the debugger application is not installed on the server. If it is a development server, then the installer installs two debugger applications on the server.

You cannot install the debugger component itself on an existing Monitor toolkit which is at version 6.1 or 6.1.1. If you have either of these versions, then you must upgrade your Toolkit to version 6.1.2 which will then install the debugger along with the toolkit.

IBM Software Group 

## The debug client

- The monitor model editor includes a debug client capable of debugging monitor models.
- The framework provided by the Eclipse debug platform is used to present a debug experience consistent with debugging any application in the Eclipse platform.
- A new debug page serves as the debug source view.



Integrated model debugger © 2008 IBM Corporation 6

The debug client provides the user interface for you to interact with the Eclipse debug platform for debugging your monitor models. This platform provides a consistent debugging experience which is much like running the Java debugger or any other Eclipse based debugger. A new tabbed debug page is included in the monitor model editor so you will see it listed alongside the tabs for the other sub-models in the editor.

IBM Software Group

IBM

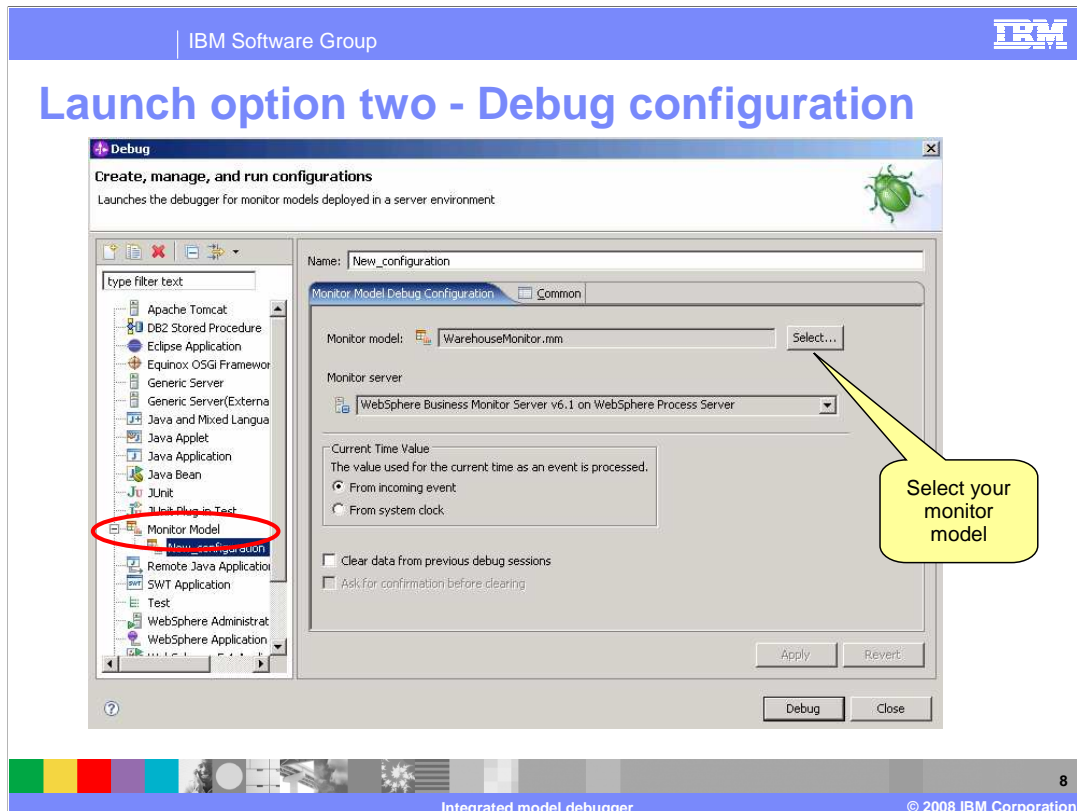
## Launching from the monitoring perspective

- Option one - automatically launches debug configuration and the integrated test client
  - ▶ Right click on the model in the Project Explorer > Debug As > Monitor Model
- Option two
  - ▶ Right click on the model in the Project Explorer > Debug As > Debug...
  - ▶ Right click on Monitor Model > New
  - ▶ Select the model > Apply > Debug
  - ▶ In the monitor perspective, right click on the model in the Project Explorer > Launch Integrated Test Client (ITC)
  - ▶ In the test client editor click the server tab and select the correct debug configuration, then click the events tab
- If the debug perspective does not open automatically, select Window > Preferences... > Run/Debug > Perspectives > In the box 'Open the associated perspective when launching' select 'Always' > Apply > OK

Integrated model debugger © 2008 IBM Corporation 7

You can launch the debugger in two different ways. Option one is the simplest way to do it, because it automatically creates a debug configuration and opens the integrated test client. Option two provides some debug session options on the debug launcher that you can tailor to your needs. However you can also access these same options in the Monitor debugger preferences for the workspace. With the second option, you must create the debug configuration yourself, and you must also manually start the integrated test client and configure it for your debug session.

In either case, if the debug perspective is not opened automatically, then you can go to the workspace preferences and adjust the setting to control this.




If you are using the second launch option then this is the page for creating a new debug configuration. Underneath the option for monitor model, you click to create a new configuration, and select the monitor model to debug. You can also specify some other settings on this page.

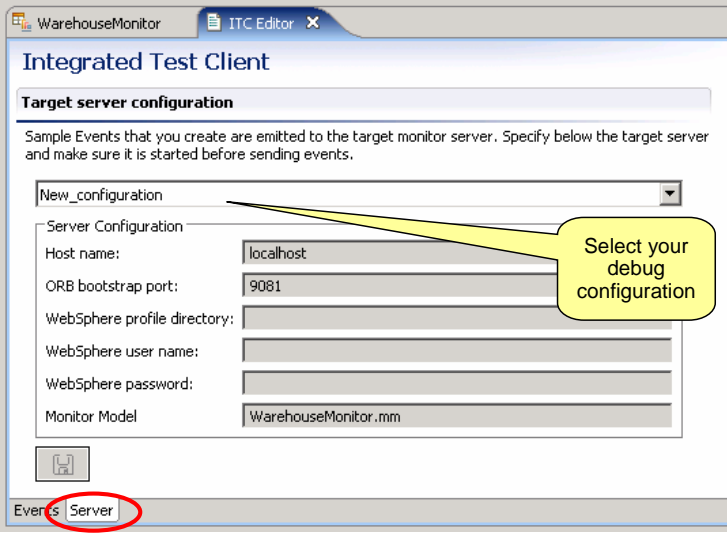
To empty the event queue and remove any remaining instances from the previous debugging sessions before launching the debugger, select 'Clear data from previous debug sessions'. Don't select this if you want to submit events and have the existing instances process the new events.

In a monitor model, a stopwatch can have a source of type trigger or inbound event. In either case, the creationTime field on the triggering inbound event is used to set the start time and stop time on the stopwatch. So the server time is not used. However, when using the debugger, the processing of the stopwatch depends on what you select for the debug setting 'Current time value'. If you select 'From incoming event' then it uses the creationTime field on the event to set the start time and stop time of the stopwatch. If you select 'From system clock' then the start times and stop times are based on the server time.



IBM Software Group 

## Launch option two - ITC editor



WarehouseMonitor ITC Editor

### Integrated Test Client

**Target server configuration**

Sample Events that you create are emitted to the target monitor server. Specify below the target server and make sure it is started before sending events.

New\_configuration

Server Configuration

Host name: localhost

ORB bootstrap port: 9081

WebSphere profile directory:

WebSphere user name:

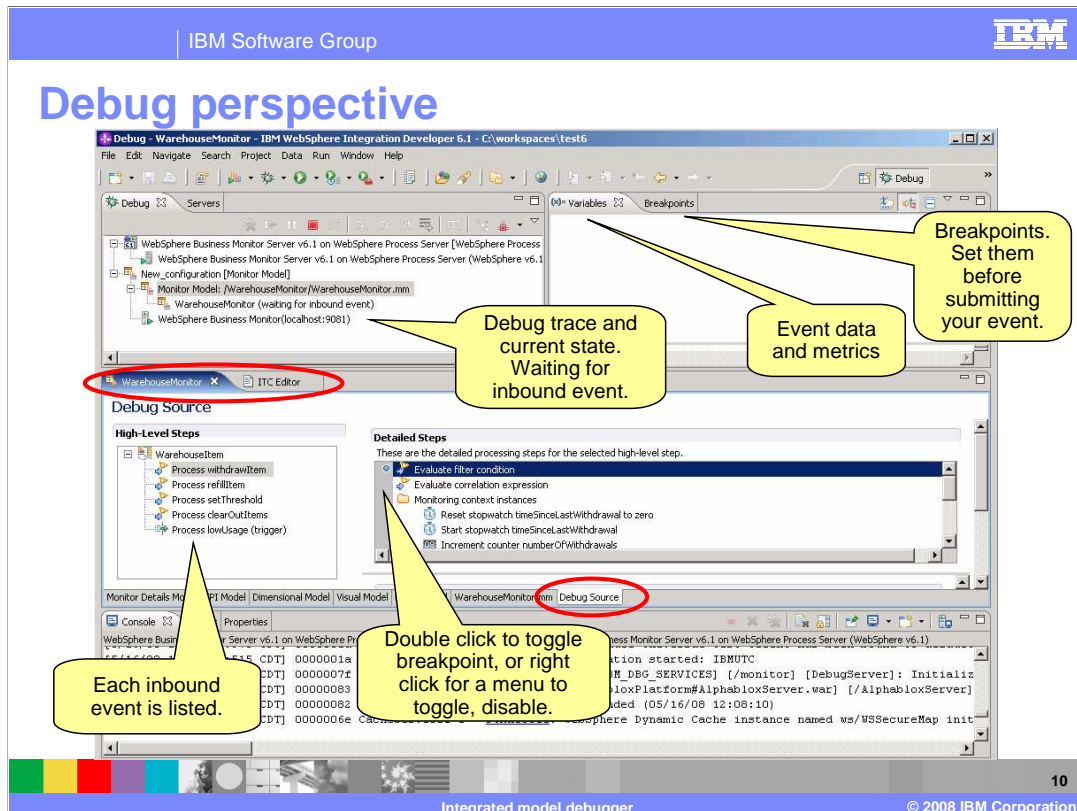
WebSphere password:

Monitor Model: WarehouseMonitor.mm

Events Server

Integrated model debugger © 2008 IBM Corporation 9

If you are using the second launch option, then you must manually start the integrated test client. Then you need to click the server tab and select the debug configuration that you created previously, and verify that the server configuration information is correct.



This is a screen capture of the debug perspective immediately after a debug session has been started for a monitor model. The debug tab shows you the name of the monitor model and shows that the debugger is waiting for an inbound event. The variables tab will eventually show you the values of fields in your inbound events and the metric values in the monitoring context instances. The breakpoints tab shows a list of all of your breakpoints in the model.

You will notice that the monitor model editor is opened in the middle of this screen capture, and a new tab has been added for the debug source view. This is where you can see the detailed steps that the debugger takes through your model. Also notice that the integrated test client tab is available and placed next to the tab for the editor. In the debug source view, there is a list of the high level steps in the model which shows you the inbound events and triggers for each monitoring context. When you select one of them, then the associated detailed steps are shown to the right. In the detailed steps, there is a gray bar where you can double click to toggle breakpoints. You need to set at least one break point before sending in your first event.

IBM Software Group IBM

## Submitting test events using the ITC

**Select your context and event**

**Enter event values**

**Add to the test script**

**Submit the event script**

Name	ID	Type	Path
transactionInfo	transactionInfo	trns:deposit	cbe:CommonBaseEve...

Name	Type	Value
@trns:sku	string	abc
@trns:quantity	positiveInteger	123

Script filename: untitled\*

Emit refillItem

Server Profile: New\_configuration

Submit events to debug server: Directly

Integrated model debugger 11 © 2008 IBM Corporation

You must use the integrated test client to submit events to the debugger. Here you can select the monitoring context and event definition, then enter values for fields in the event. You can create a script containing one or more events to submit. When you have finished building your event script, then press the green arrow to submit the script.

IBM Software Group IBM

## Event received - filtering

**Variables**

Name	Value
CommonBaseEvent	
creationTime	2008-05-16T18:19:35.234Z
localInstanceId	ITC17737403822090371234687951749131363
version	1.0.1
xmlns	http://www.ibm.com/ACJ/commonbaseevent1_0
xmlns:xsi	http://www.w3.org/2001/XMLSchema-instance

**High-Level Steps**

- WarehouseItem
- Process withdrawItem
- Process refillItem
- Process setThreshold
- Process clearOutItems
- Process lowUsage (trigger)

**Detailed Steps**

- Evaluate filter condition
- Evaluate correlation expression
- Monitoring context instances
- Start stopwatch timeSinceLastWithdrawal
- Set counter numberOfWithdrawals to zero
- Update metric sku
- Update metric quantityInStock

**Filter Condition**

This condition is based on attributes in the event definition. The inbound event is checked against the filter condition. If the event passes, the correlation expression is evaluated. If the event fails, the event is rejected and processing continues with the filter condition of the next inbound event definition.

```
fn:exists(refillItem/transactionInfo/@tns:sku)
```

**Callouts:**

- Suspended at breakpoint
- Event values
- Stopped on this inbound event
- A breakpoint is reached. The rest of the steps to process for this event are listed.
- The first step is the filter

Integrated model debugger © 2008 IBM Corporation

When an event has been received then the debugger processes the event as it steps through the model logic. When it reaches a breakpoint, then you see the results as shown in this screen capture. In the debug tab, you can use the icons to step through the logic, resume processing or terminate the session. Here you see that the status of the model is suspended which means that the debugger has found a breakpoint and has stopped processing. In the variables tab you see the values for each field in the inbound event. In the high level steps you see that the corresponding inbound event is highlighted and that the breakpoint is highlighted in the detailed steps. In this case, there is a breakpoint on the filter condition, and below the detailed steps you see the filter condition that is processed. If the filter condition resolves to true then the next step to process is the correlation expression.

IBM Software Group IBM

## Second step - correlation

WarehouseMonitor x ITC Editor

**Debug Source**

**High-Level Steps**

- WarehouseItem
  - Process withdrawItem
  - Process refillItem
  - Process setThreshold
  - Process clearOutItems
  - Process lowUsage (trigger)

**Detailed Steps**

These are the detailed processing steps for the selected high-level step.

- Evaluate filter condition
- Evaluate correlation expression**
  - Monitoring context instances
    - Start stopwatch timeSinceLastWithdrawal
    - Set counter numberOfWithdrawals to zero
    - Update metric sku
    - Update metric quantityInStock
    - Update metric quantityWithdrawn
    - Update metric dateOfLastReplenishment

**Correlation Expression**

After an event has passed the filter, this expression determines where to deliver the event. It is evaluated once for each instance of the monitoring context that exists. It might result in 0, 1, or many matching instances, or might create a new instance. The settings of the inbound event definition determine the action to take.

refillItem/transactionInfo@tns:sku = sku

If no instances are found

If one instance is found

If multiple instances are found

13

Integrated model debugger © 2008 IBM Corporation

In this screen capture the debugger is stopped on the correlation step. Below the detailed steps you can view the correlation expression. If the correlation expression resolves to true then the next step in the sequence is the monitoring context instances, so a new monitoring context is created or an attempt is made to match to existing instances.

IBM Software Group IBM

## Processing a monitoring context

You can step through the event processing logic for each instance, or continue to the next breakpoint.

The screenshot shows the following components:

- Servers:** A tree view showing the hierarchy of the application, including 'WarehouseMonitor' and 'Processing WarehouseItem'.
- Variables:** A table listing variables and their values for the current instance.
 

Name	Value
threshold	10
lowInventory_LastEvaluation	false
lowUsage_LastEvaluation	false
lowUsage_nextEvaluation	2008-05-16T23:06:59.515Z
itemDiscontinued_LastEvaluation	false
sku	abc
...	...
- Debug Source:** Shows the 'High-Level Steps' and 'Detailed Steps' for the selected step. The 'Detailed Steps' pane shows the current step: 'Start stopwatch timeSinceLastWithdrawal'.

The metrics for this instance are listed including the context key.







An instance is being processed.

14

Integrated model debugger © 2008 IBM Corporation

In this screen capture you can see in the detailed steps that the debugger is stopped at the first step in the monitoring context instances. So, either a new instance has been created or an existing instance has been matched. You can view all the metrics for this instance in the variables tab. Also, you can view the key for the instance, so it is easy to determine which instance is being processed by observing the value of the key metric.

## Debug toolbar

-  Step into - F5
  - ▶ Advance to the next step in the sequence
-  Step over - F6
  - ▶ Advance to the next step at the same level. Any sub-steps such as for a trigger are processed without stopping.
-  Step return - F7
  - ▶ Advance to the next step at the parent level. Stop at the next context instance, but if none then stop at the next inbound event definition.
-  Resume – F8
  - ▶ Run to the next breakpoint.
-  Terminate
  - ▶ End the debugging session but leave it listed in the debug view.
  - ▶ You can update your model on the fly, but you must terminate your debug session and re-launch it to see the changes
-  Remove all terminated
  - ▶ Remove all terminated debug sessions from the debug view.



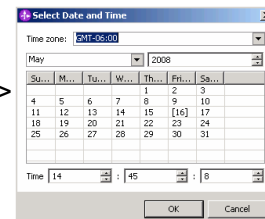
On the debug toolbar you can select different options for controlling the behavior of the debugger. 'Step into' is used to continue to the next step in the sequence. 'Step over' advances to the next step but only if it is at the same level as the current level. So, for example, if the next step was a trigger which has sub-steps then it is processed without stopping in the sub-steps. 'Step return' advances to the next step at the parent level, so the debugger stops at the next monitoring context instance or the next inbound event definition.

You also have icons to resume, terminate and 'remove all terminated'. Note that if you update your monitor model in the middle of a debug session, then you need to terminate and then restart your debug session in order to see the updates.

Note that for 'Step over' and 'Step return', breakpoints are respected.

## Time based triggers

- Any time-based trigger, for example a trigger that is evaluated every minute, is shown in the high-level steps along with inbound events.
- To test these triggers during debugging, you must send a TIME-CHECK event to the debug server.
  - ▶ Generic - target all time-based triggers in the model
  - ▶ Specific - target a specific time-based trigger
- Submitting the events
  - ▶ Create an XML file and submit using the ITC
  - ▶ Right click on the trigger in the debug source view > Emit Time Check Event...



For triggers which are based on a recurring time interval, you can specify a special time check event to fire the trigger so that you can debug the steps based on the trigger. You can create an XML file and identify a generic or specific time check event. The generic type of event applies to all time based triggers in the model, but the specific type of event applies to a single named trigger in the model. You submit the XML file using the integrated test client. Optionally, you can submit the time check event graphically by using the pop-up menu on the trigger. When you use the menu to emit the time check event, it is created as a specific type event that is for the selected trigger.



IBM Software Group

## KPIs and measures

Debug Source

**High-Level Steps**

- ClipsAndTacks MC
  - Process Activity Event
  - Process New Order Event
- My KPI Context
  - Process KPIs
  - Process Declined Order Tri
  - Process Order Fulfillment T
- ClipsAndTacks MC Cube
  - Process measures

**Detailed Steps**

These are the detailed processing steps for the selected high-level step.

- Evaluate filter condition
- Evaluate correlation expression
- Update KPI Average Order Fulfillment KPI August 2006**
- Update KPI Order Count KPI
- Update KPI Ship Count KPI
- Update KPI Percent of Orders Approved KPI
- Update KPI Average Order Price KPI (Dollars)
- Update KPI Declined Order KPI

You can step through the processing for each KPI. Data filters are evaluated and applied.

Debug Source

**High-Level Steps**

- ClipsAndTacks MC
  - Process Activity Event
  - Process New Order Event
- My KPI Context
  - Process KPIs
  - Process Declined Order Tri
  - Process Order Fulfillment T
- ClipsAndTacks MC Cube
  - Process measures

**Detailed Steps**

These are the detailed processing steps for the selected high-level step.

- Evaluate filter condition
- Evaluate correlation expression
- Update measure Average Order Price**
- Update measure Sum Order Price
- Update measure Order Count

You can step through the processing for each measure

Integrated model debugger 17 © 2008 IBM Corporation

You can debug your KPIs and measures in the same way that you debug monitoring contexts. Each KPI context is listed in the high level steps and underneath each context are triggers and an entry for the KPIs called 'Process KPIs'. If you select 'Process KPIs', then you see all of the KPIs listed in the detailed steps, where you can step through their processing.

Each cube is listed in the high level steps and underneath each cube is an entry for the measures called 'Process measures'. If you select 'Process measures', then you see all of the measures listed in the detailed steps, where you can step through their processing.

IBM Software Group

## Nested monitoring contexts

Debug Source

**High-Level Steps**

- Country
  - Process newKindOfItem
    - Warehouse
      - Process newKindOfItem
        - Item
          - Process withdrawItem
          - Process refillItem
          - Process newKindOfItem
          - Process clearOutItems
          - Process lowUsage (trigger)

You can step through the processing for each context

**Detailed Steps**

These are the detailed processing steps for the selected high-level step.

- Evaluate filter condition
  - Evaluate correlation expression
    - Monitoring context instances
      - Start stopwatch timeSinceLastWithdrawal
      - Update metric sku
      - Update metric threshold

**Correlation Expression**

After an event has passed the filter, this expression determines where to deliver the event. It is evaluated once for each instance of the monitoring context that exists. It might result in 0, 1, or many matching instances, or might create a new instance. The settings of the inbound event definition determine the action to take.

```
newKindOfItem/thresholdInfo/tns:address/tns:country = ../countryName and newKindOfItem/thresholdInfo/tns:address/tns:zip = ../zipCode and newKindOfItem/thresholdInfo/@tns:sku = sku
```

If no instances are found

If one instance is found

If multiple instances are found

18

Integrated model debugger © 2008 IBM Corporation

If your monitor model contains multiple monitoring contexts or nested monitoring contexts then you can debug all of them. All of the contexts are listed in the high level steps in the order that they are processed and you can select each context so that you can set breakpoints and step through processing for each monitoring context.

## Notes

- Open the debug session before opening the ITC, otherwise there won't be a debug configuration to set on the server tab of the ITC
- Set a breakpoint before sending the event, otherwise the debugger won't stop
- For this release, there isn't a navigator to see the list of existing monitoring context instances
- You can't see the return value of the filter or correlation expression, but you know the value by the path it takes



If you start the integrated test client before you create a debug configuration, then the debug configuration is not available for selecting on the server tab of the integrated test client. To fix this problem, just close the client and re-open it.

Also remember to set a breakpoint before sending your first event, otherwise the debugger won't stop.

There isn't a monitoring context instance navigator in the debugger, but you can determine which instance is being processed by checking the value of the key metric in the variables list.

Also, you don't get a confirmation of the returned value from the filter and correlation on an inbound event, but you can determine the returned value by the path that the debugger takes through the model. If the filter returns true, then it proceeds to correlation, but if it returns false then it moves to the next inbound event. If the correlation returns true, then it proceeds to the monitoring context, but if it returns false then it moves to the next inbound event.

## Summary

- Covered the new monitor model debugger



In summary, you have seen an overview of the new monitor model debugger component which is very helpful in resolving problems with your monitor models.

## Feedback

### Your feedback is valuable

You can help improve the quality of IBM Education Assistant content to better meet your needs by providing feedback.

- Did you find this module useful?
- Did it help you solve a problem or answer a question?
- Do you have suggestions for improvements?

Click to send e-mail feedback:

[mailto:iea@us.ibm.com?subject=Feedback\\_about\\_WBMonitorV612\\_Debugger.ppt](mailto:iea@us.ibm.com?subject=Feedback_about_WBMonitorV612_Debugger.ppt)

This module is also available in PDF format at: [..WBMonitorV612\\_Debugger.pdf](..WBMonitorV612_Debugger.pdf)



You can help improve the quality of IBM Education Assistant content by providing feedback.

## Trademarks, copyrights, and disclaimers

The following terms are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both:

IBM            WebSphere

A current list of other IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>

Product data has been reviewed for accuracy as of the date of initial publication. Product data is subject to change without notice. This document could include technical inaccuracies or typographical errors. IBM may make improvements or changes in the products or programs described herein at any time without notice. Any statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only. References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business. Any reference to an IBM Program Product in this document is not intended to state or imply that only that program product may be used. Any functionally equivalent program, that does not infringe IBM's intellectual property rights, may be used instead.

Information is provided "AS IS" without warranty of any kind. THE INFORMATION PROVIDED IN THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IBM EXPRESSLY DISCLAIMS ANY WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT. IBM shall have no responsibility to update this information. IBM products are warranted, if at all, according to the terms and conditions of the agreements (for example, IBM Customer Agreement, Statement of Limited Warranty, International Program License Agreement, etc.) under which they are provided. Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products.

IBM makes no representations or warranties, express or implied, regarding non-IBM products and services.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents or copyrights. Inquiries regarding patent or copyright licenses should be made, in writing, to:

IBM Director of Licensing  
IBM Corporation  
North Castle Drive  
Armonk, NY 10504-1785  
U.S.A.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. The actual throughput or performance that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput or performance improvements equivalent to the ratios stated here.

© Copyright International Business Machines Corporation 2008. All rights reserved.

Note to U.S. Government Users - Documentation related to restricted rights-Use, duplication or disclosure is subject to restrictions set forth in GSA ADP Schedule Contract and IBM Corp.