Transaction Whiteboard: Modifying an Application Model

For this tutorial, assume that you are asked to determine the expected response time for a portion of a new “Staff Listing” application that will be rolled out in a few months. You used AppTransaction Xpert to capture a beta version of this application, and you will now use Transaction Whiteboard to create an application model that accurately represents the anticipated final application version.

The objectives of this tutorial are to

1) Open a Transaction Analyzer model in Transaction Whiteboard.

2) Modify the application model so that it matches the final version of the application.

3) Create a parameter that specifies the message size. By making this value a parameter, you enable different applications to use the same Transaction Whiteboard file (with different message sizes) in the Project Editor.

Creating the Transaction Whiteboard Model

The following procedure describes how to create a Transaction Whiteboard model.

Procedure 12-1 Creating a Transaction Whiteboard Model

1 Choose **File > Open Model > Transaction Whiteboard…**

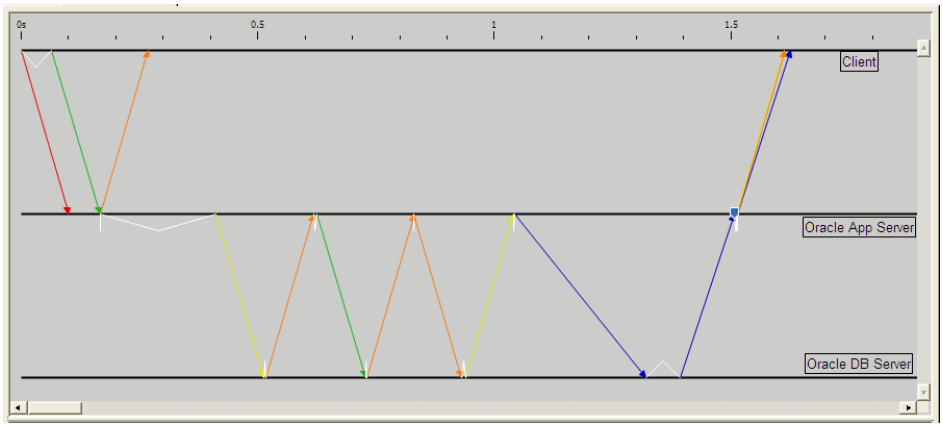
2 In the **Transaction Whiteboard Startup Wizard**, select **Import application from Transaction Analyzer** and click **Next >**.

3 In the **Import Transaction Analyzer model** dialog box, select **tutorial_staff_listing_beta** under **<*reldir*>\sys\examples\AppTransaction Xpert\examples** and click **Import**…

Note—<reldir> is the release directory where AppTransaction Xpert is installed. In the Windows environment, this is often C:\Program Files\OPNET\<*release number*>.

➥ The existing Transaction Analyzer model opens in Transaction Whiteboard.

4 In the **Data Exchange Chart**, notice that the task contains three tiers and some messages, as shown in the following figure.



5 Choose **File > Save** and save the file as **<*initials*>_tutorial_auth**. Keep the file open.

The application contains the messages sent as a result of the client selecting the division of interest from a Web page. The transaction begins with an HTTP message from the Client to the Oracle App Server. The Oracle App Server processes this message and requests the appropriate data from the Oracle DB Server. The Oracle App Server then sends a Web page containing the requested data to the Client.

End of Procedure 12-1

Modifying the Application Model in Transaction Whiteboard

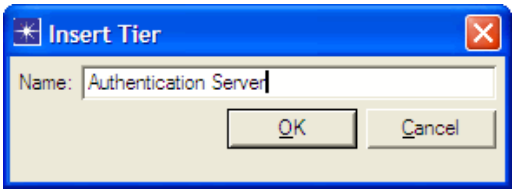
Before sending a message to the Oracle DB Server, the Oracle App Server needs to authenticate with the Authentication Server.

Procedure 12-2 Modifying the Application Model in Transaction Whiteboard

1 Insert a new tier for the Authentication Server.

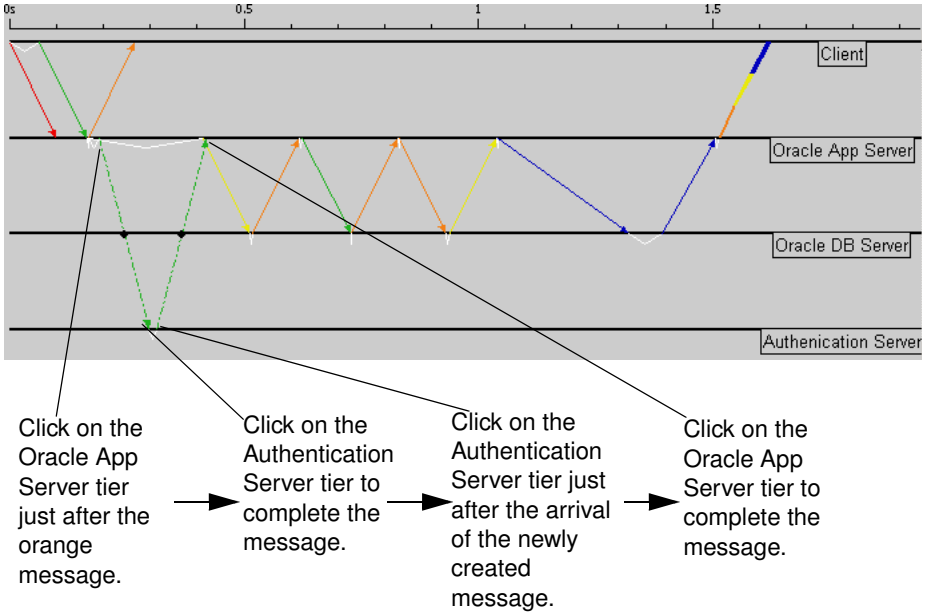
1.1 Choose **Insert > Tier**.

1.2 In the Insert Tier dialog box, enter **Authentication Server** as the tier name.



1.3 Click **OK**.

2 Create a single request/response pattern between Oracle App Server and Authentication Server.



2.1 Choose **Insert > Message**.

2.2 Click on the **Oracle App Server** tier just after the orange message.

2.3 Click on the **Authentication Server** tier to complete the message.

2.4 Click on the **Authentication Server** tier just after the arrival of the newly created message.

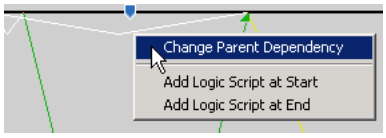
2.5 Click on the **Oracle App Server** tier to complete the message.

3 Right-click in the workspace to exit message creation mode.

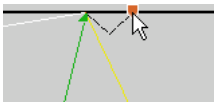
4 Modify the first message between Oracle App Server and Oracle DB Server so that it is dependent on the last message you just created. Notice that this message is currently dependent on the orange message from the Oracle App Server to the Client.

4.1 In the Data Exchange Chart, right-click on the dependency just before the first message from Oracle App Server to Oracle DB Server (the yellow message).

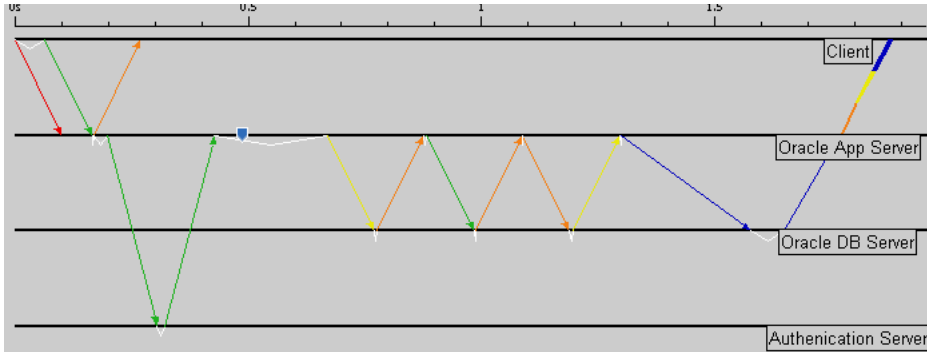
4.2 Select **Change Parent Dependency**.



4.3 Click on the **Oracle App Server** tier near where the green message from Authentication Server to Oracle App Server was received.



5 Notice that this causes the yellow message to shift to the right, as seen in the following figure.



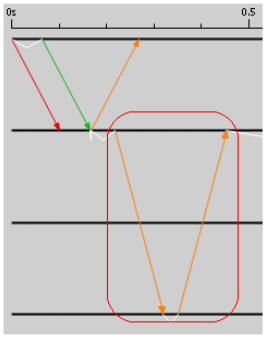
6 Edit the two messages you created in step 2.

6.1 Select the two messages in the **Data Exchange Chart**. You can select two messages by clicking in the background, holding the mouse button down, and drawing a rectangle around the two messages. Alternatively, you can shift-click to select the messages individually.

6.2 In the message editor, change the **Bytes** of the first green message (ID: 15) to **50**.

6.3 Change the **Bytes** of the second green message (ID: 16) to **100**.

6.4 Change the **Processing Time** value of the first message (ID:15) to **0.05**.



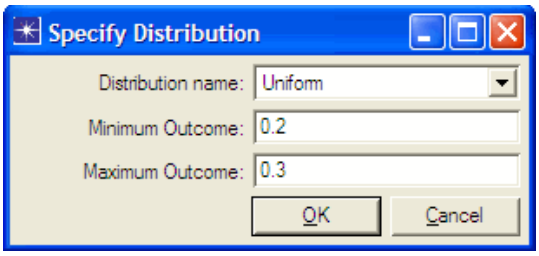
6.5 In the **Message Editor**, change the **Processing Time** value for the second message (ID: 16) to a **Uniform** distribution from **0.2** to **0.3**:

Click into the **Processing Time** field for the response message (ID: 16).

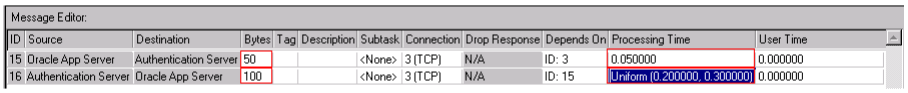
Select **Edit As Distribution…**, and then change the distribution to **Uniform** in the Specify Distribution dialog box.

Change the **Minimum Outcome** to **0.2**.

Change the **Maximum Outcome** to **0.3**, as shown in the following figure.

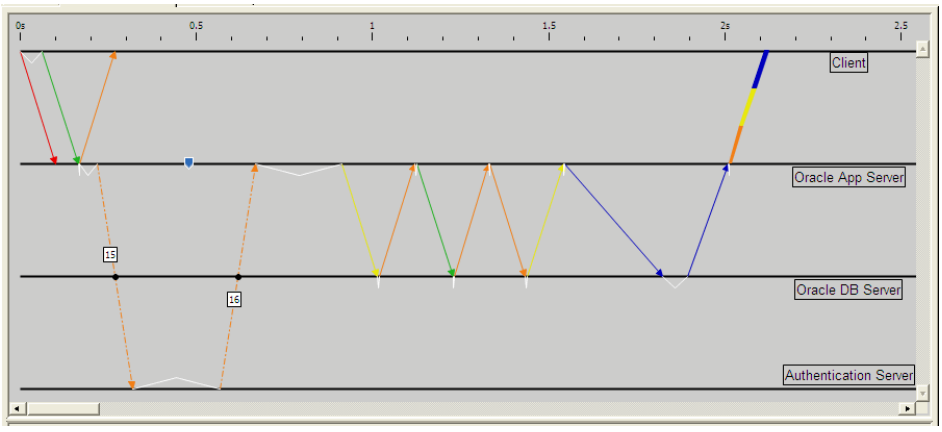


Click **OK**. The Message Editor should now look like the following figure.



7 Change the zoom level to Full Zoom by right-clicking on a blank portion of the **Data Exchange Chart,** and then choosing **Full Zoom**.

8 The **Data Exchange Chart** should look similar to the following figure.



9 Choose **File > Save** to save the file. Keep the file open.

End of Procedure 12-2

At this point, you have opened a Transaction Analyzer model in Transaction Whiteboard, created a new tier, added two messages, used a distribution to specify a processing time dependency, and reassigned a message's parent dependency.

Adding a Parameter to the Transaction Whiteboard Model

Assume that based on your knowledge of the application, you know that the size of the last response message from the Oracle DB Server to the Oracle App Server depends on the number of employees in the division (specified by the user in the initial request). Since you want to be able to change this value in the Project Editor before running a simulation, you should parameterize the response size.

Procedure 12-3 Adding a Parameter to the Transaction Whiteboard Model

1 Add a parameter to the task:

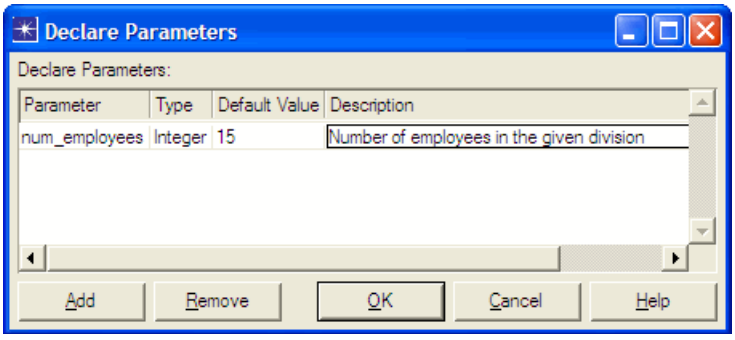
1.1 Choose **Scripting > Declare Parameters…**

➥ The Declare Parameters dialog box displays.

1.2 Click **Add** to add a new row to the table and specify the following:

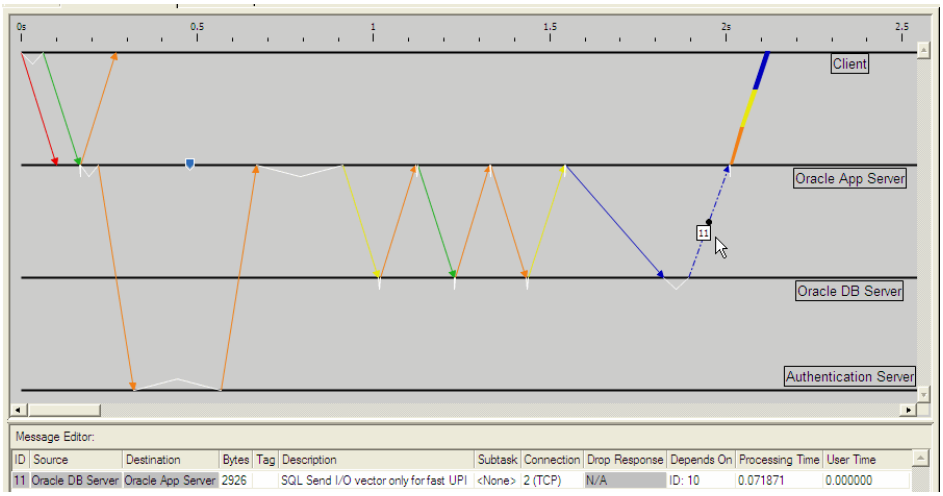
- **Parameter:** num_employees
- **Type:** Integer
- **Default Value:** 15
- **Description:** Number of employees in the given division

The Declare Parameters dialog box should look as follows.



1.3 Click **OK**.

2 Identify the last message from Oracle DB Server to Oracle App Server by giving it a unique tag. This enables you to access this message from within a logic script.

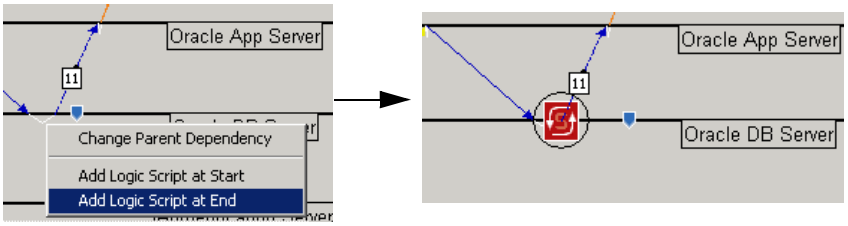


2.1 In the **Data Exchange Chart**, select the last message from Oracle DB Server to Oracle App Server.

2.2 In the **Message Editor**, click the **Tag** field for message 11, and then change the value to **db_response**.

3 Logic scripts can be used to dynamically modify the behavior of an application model. You will write a logic script that changes the size of the **db_response** message based on the value of the parameter **num_employees**.

3.1 Add a logic script before the message you just tagged as **db_response**. Right-click on the dependency before the message (not on the message itself) and choose **Add Logic Script at End**.



➥ The **Edit function** window displays.

3.2 Add the following lines of code to the logic script:

```
num_employees = self.get_parameter ('num_employees')

response_msg = self.get_message ('db_response')

response_msg.size = num_employees * 2000
```

3.3 Choose **File > Commit** to save the logic script.

End of Procedure 12-3

Using QuickPredict to Determine the Expected Response Time

Use QuickPredict to determine the expected response time.

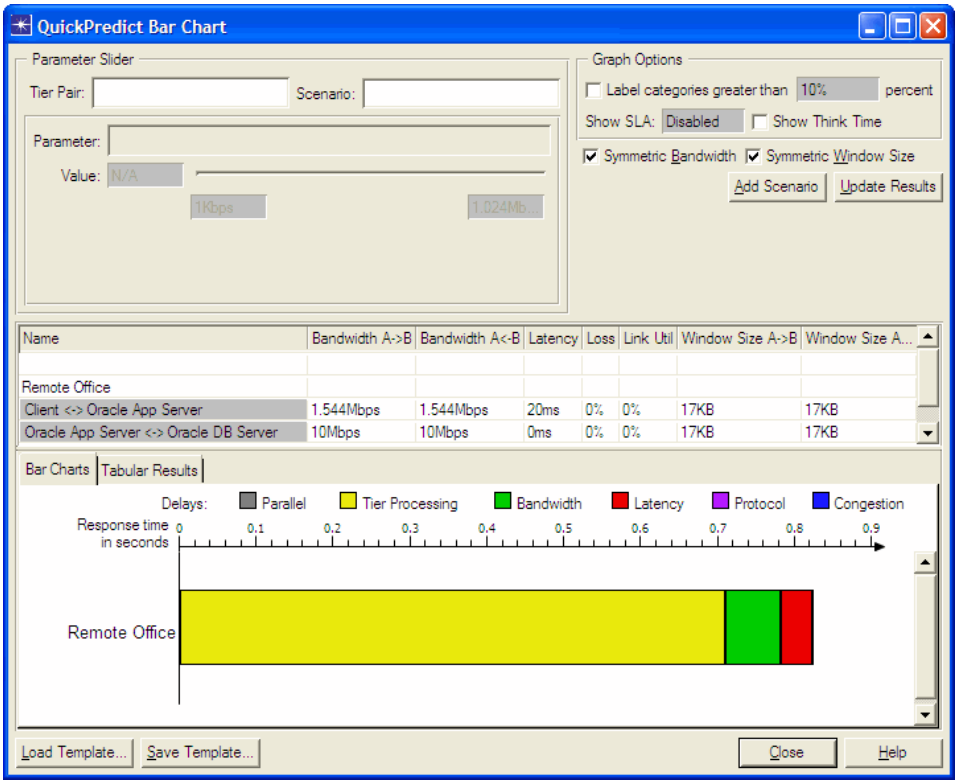
Procedure 12-4 Determining the Expected Response Time (with QuickPredict)

1. In the Transaction Whiteboard window, choose **Prediction > QuickPredict** to open the QuickPredict dialog box.

2. Click **Load Template…**

3. Select the **tutorial_staff_listing_modified.ace.qpb** in **<*reldir*>\sys\examples\AppTransaction Xpert\examples** and click **Open**.

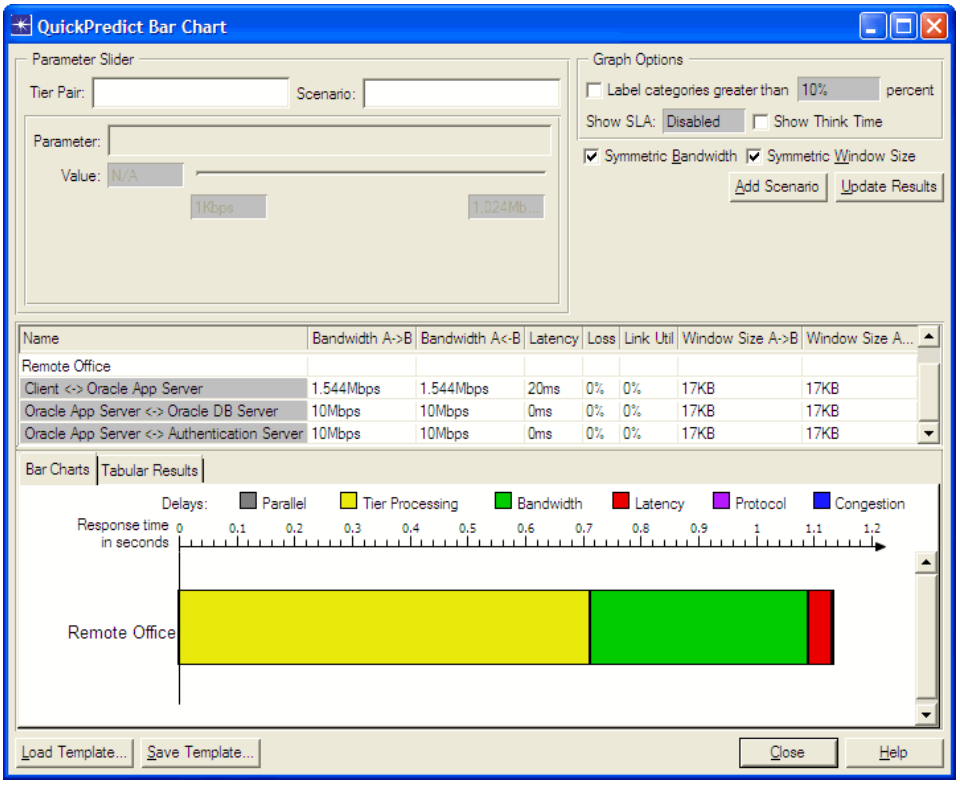
4. If the **Choose Tier Pair Mapping** dialog displays, make sure the tiers map correctly, then click **Done**. (You may have to click the **From Trace** field for the new tier, Authentication Server, and select **Authentication Tier**.) The specified template is applied to the QuickPredict results and the following bar chart displays.



   Notice that the revised application takes less than 1 second to complete when there are 15 employees. (**Note**—We gave the parameter num_employees a default value of 15 earlier in this tutorial.)

5. In the Transaction Whiteboard window, select **Scripting > Declare Parameters…** Change the default value of **num_employees** to **200**, and then click **OK**.

6 In QuickPredict, click the **Update Results** button.



Notice that increasing the default value for the num_employees parameter caused an increase in the response time of the application.

7 Click the **Close** button to close QuickPredict.

8 Choose **File > Save** to save the Transaction Whiteboard file.

9 Choose **File > Close** to close Transaction Whiteboard.

End of Procedure 12-4

Conclusion

Starting with a Transaction Analyzer model of a beta version of your application, you used Transaction Whiteboard to modify the behavior of the application. These modifications included adding a request/response pattern to a new Authentication Server tier, changing the parent dependency of a message, adding variability to the application using a distribution, and specifying the size of a message based on a parameter. You also learned how to use QuickPredict to determine the response time of the application for two different values of the parameter.

Additional Details

- In this tutorial, you wrote a logic script to dynamically change the size of a message based on a parameter. If you had deployed this application on a network in the Project Editor to run a Discrete Event Simulation, you would be able to specify the message size in the AppTransaction Xpert **Parameters** field of the task specification.
- Instead of manually adding a request/response pattern to an authentication server, you could have created a separate Transaction Whiteboard file to handle the authentication. Then, you could have invoked your authentication Transaction Whiteboard model. You will see an example of this in the next tutorial.
- When making multiple changes to a Transaction Whiteboard model, it is often useful to turn on Multi-Message Editing Mode. (Choose **Edit** > **Enable Multi-Message Editing**.)