24 Transaction Whiteboard: Overview

Transaction Whiteboard enables you to design and edit an application's behavior in a graphic environment similar to Transaction Analyzer. You can import and edit existing Transaction Analyzer models, or create Transaction Whiteboard models entirely from scratch. You can also write and insert logic scripts into a Transaction Whiteboard model file to model complex application-layer logic and behavior. After you model an application in Transaction Whiteboard, you can predict the application's performance using discrete event simulations or QuickPredict.

This section includes the following topics:

- Transaction Whiteboard Features
- Components of a Transaction Whiteboard Model
- Transaction Analyzer vs. Transaction Whiteboard

Transaction Whiteboard Features

Transaction Whiteboard provides the following features:

- **Transaction Analyzer-like graphical interface—**Transaction Whiteboard has Tree View, Data Exchange Chart, and Tier Pair Circle views that are similar to their Transaction Analyzer equivalents. You can quickly create and remove messages in the Data Exchange Chart using mouse clicks and keyboard shortcuts.
- **Importing Transaction Analyzer models—**Import Transaction Analyzer models and thereby edit your existing applications directly.
- **Message Editor—**Use the Message Editor table to modify message attributes such as sizes, source and destination tiers, delay times, and message dependencies. The Message Editor includes a “multi-message editing mode” that can be use to edit multiple messages at once.
- **Logic scripts—**Insert logic scripts into a Transaction Whiteboard model file to model advanced application-layer logic and events, such as conditional message transmissions and delays. Transaction Whiteboard includes a library of Python functions for writing logic scripts.
- **Modeling variability using application parameters—**Create parameters for a Transaction Whiteboard model (for example, “message size”) and write logic scripts to specify behavior based on these parameters. Then users can set these parameters before running QuickPredict or discrete event simulations.
- **Application task chaining—**Write logic scripts so that a tier in one task launches another task. This means that you can execute complex transactions that consist of individual tasks “chained” together using logic scripts.

Related Topics

- *Transaction Whiteboard: Overview*

Components of a Transaction Whiteboard Model

Like a Transaction Analyzer model, a Transaction Whiteboard model should describe one application task in its entirety. Examples of an application task include downloading a web page, querying and retrieving a record from a database, or transferring a file via FTP.

A Transaction Whiteboard model has the following components:

- **Tiers and Tier Pairs**
  Every Transaction Whiteboard model contains at least two tiers. As in Transaction Analyzer, a tier is a computer that sends and/or receives traffic. You can insert any number of tiers into a Transaction Whiteboard model.

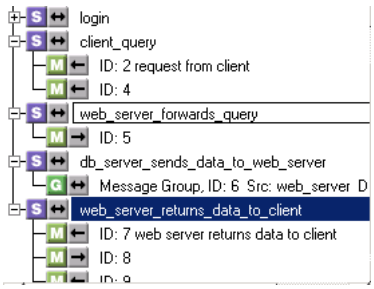
- **Connections**
  Every tier pair has at least one connection. You can create additional connections if you want to model an application where two tiers exchange data using multiple connections.

  Every tier pair has a *default connection*. Whenever you create a new message, the “connection” attribute for that message is set to the default connection.You can change the default connection for a tier pair and edit messages to use different connections.

- **Subtasks**
  A subtask is a logical grouping of messages, message groups, and other subtasks. Subtasks are useful for organizing a Transaction Analyzer model to make the application task easier to understand and navigate. You can insert new subtasks into a file and move elements into different subtasks.

  You can view and edit subtasks in the Tree View page. Subtasks are similar to transactions in Transaction Analyzer: when you import a Transaction Analyzer model, the Transaction Whiteboard organizes messages into subtasks based on the original transactions. When you create an application from scratch, you might want to organize the messages into subtasks that reflect separate transactions or phases, as shown in the following figure.

Figure 24-1 Organizing Messages Using Custom Subtasks: Example



For more information, see Subtask Operations in Transaction Whiteboard.

- **Messages**
  In most cases, a Transaction Whiteboard model contains at least one message transferred from one tier to another. You can create individual messages or message groups. Every message or group has a set of attributes to specify message size, subtask, connection, and dependency information.You can view and edit these attributes using the Message Editor table at the bottom of the Transaction Whiteboard window.

  For more information, see Message Operations.

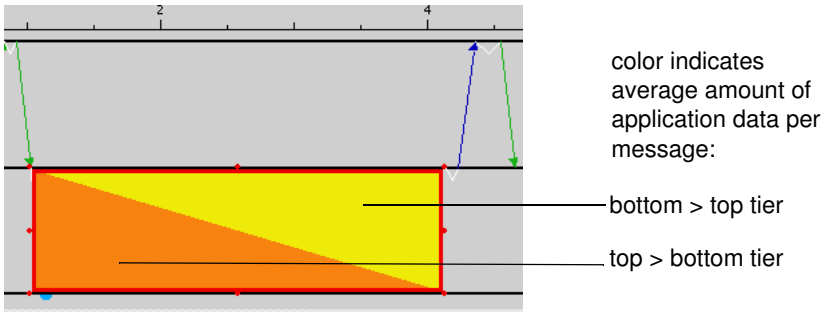
- **Message Groups**
  A message group is a sequence of messages between two tiers in which the general characteristics of the data transferred are modeled. Message groups are useful when you need to model a well-defined series of requests and responses (for example, a series of chatty database requests).

  You can create a message group in one operation (Insert > Message Group). You can specify the number of application turns, the total number of bytes transferred, and the total processing time at each tier.

  For more information, see Message Group Operations.

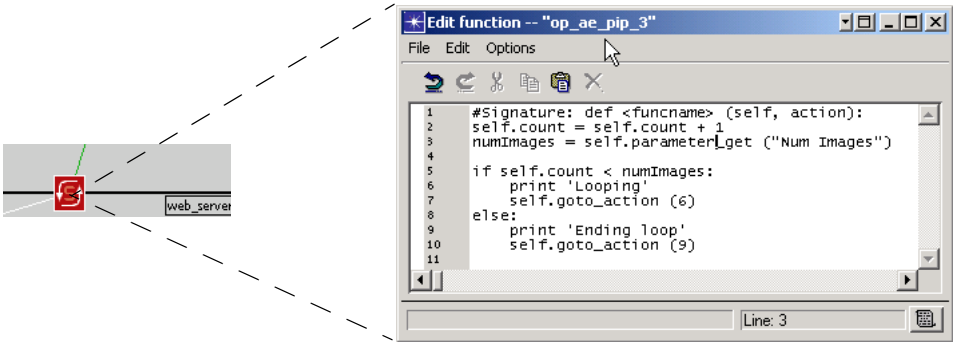
  A message group appears as a solid bar in the Data Exchange Chart; each group is split diagonally to show the average application payload per message in each direction (as specified in the legend at the bottom of the Data Exchange Chart page).

Figure 24-2 Message Group in the Data Exchange Chart: Example



- **Logic Scripts**
  A *logic script* specifies one or more actions and the internal logic that defines when those actions are run. Every logic script is associated with a specific tier; the associated tier "runs" the script at a specific time in the application. You can use logic scripts to create "smart" application models that respond to network and application events.

Figure 24-3 Logic Script in the Data Exchange Chart: Example



For more information, see Logic Scripts.

- **Parameters**
  You can specify one or more parameters for a Transaction Whiteboard model. For example, you might want to create parameters such as "Graphics files in page" or "Size of downloaded file." Then you can write logic scripts that define application behavior based on the task parameters. Parameters make a task configurable, because they can be set by a user before the task is run. A task can also set parameters when it runs a child task in a task-chaining scenario.

  For more information, see Parameters.

Related Topics

- *Transaction Whiteboard: Overview*

Transaction Analyzer vs. Transaction Whiteboard

Although Transaction Analyzer and Transaction Whiteboard have a similar "look and feel," there are important differences between the two environments. The differences are especially important to remember when importing a Transaction Analyzer model into Transaction Whiteboard. This section describe environment differences, including:

- Supported Operations
- Network Delays and Application Time Scales
- Tier Delays and Application Message Dependencies
- Message Bars vs. Message Groups
- Transactions vs. Subtasks

Supported Operations

Transaction Whiteboard includes operations for editing applications, but does not include many of the analytical and diagnostic features found in Transaction Analyzer. The following table lists features of Transaction Analyzer and Transaction Whiteboard for comparison.

Table 24-1 Supported Features: Transaction Analyzer vs. Transaction Whiteboard

Category	Transaction Analyzer	Transaction Whiteboard
Editing and customizing applications	• QuickRecode • Rename tiers	• Create tiers, messages, message groups, connections and subtasks • Edit messages and message groups • Logic scripting • Create parameters • Chain applications • Rename tiers
Predicting Application Performance	• QuickPredict • QuickRecode • Traffic flows • Discrete event simulations	• QuickPredict bar charts • Discrete event simulations[1]
Analyzing applications and networks	• AppDoctor • Protocol Decodes • PathProbe	—
Reporting	• Web, RTF, and spreadsheet reports	—

1. For information about running discrete event simulations, including license requirements, see Workflow: Running a Discrete Event Simulation.

Network Delays and Application Time Scales

The time scales of an application might differ in Transaction Analyzer and Transaction Whiteboard because the two interfaces model the underlying network differently.

In Transaction Analyzer, network delays are based on the observed transmission and arrival times in the underlying packet traces. In other words, network delays reflect the delays in the network when the application was captured.

In Transaction Whiteboard, network delays are based on the "Simple Network Estimation" setting in the Data Exchange Chart:

- If network estimation is turned off, network delay time is 0—that is, the transmission and arrival times for each message are the same.
- If network estimation is turned on, the Bandwidth and Latency fields (under "Simple Network Estimation") determine the network delay time between the transmission of a message from the source tier and its arrival at the destination tier.

Note—Simple Network Estimation is used for visualization purposes only and has no effect on results from QuickPredict or discrete event simulations, which use their own attributes and settings for modeling network delay.

Tier Delays and Application Message Dependencies

Transaction Analyzer and the Transaction Whiteboard model application message dependencies differently. An application message dependency defines the relationship between two subsequent events at the same tier: the transmission/arrival of one message (event 1) and the subsequent transmission/arrival of the next message (event 2). In Transaction Analyzer, a dependency has three attributes:

1) The dependency relationship (that is, event 2 depends on the completion of event 1)

2) The total *application delay* between event 1 and event 2

3) The total amount of *network delay* between event 1 and event 2

Because Transaction Whiteboard ignores network-specific information in a Transaction Analyzer model, a dependency does not have a specified network delay time. Instead, a dependency has the following components in Transaction Whiteboard:

1) The dependency relationship

2) The total amount of *processing delay* between event 1 and event 2

3) The total amount of *user think time delay* (rather than network delay) between event 1 and event 2

When you import a Transaction Analyzer model into Transaction Whiteboard, the network delays are removed from all dependencies; all network delays are based on the Simple Network Estimation settings in the Data Exchange Chart. As a result, each dependency consists of processing delay only (application delay from the original Transaction Analyzer model plus user delay, which is 0 by default).

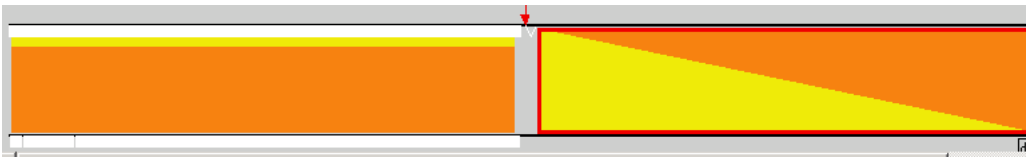
Message Bars vs. Message Groups

As in Transaction Analyzer, the Data Exchange Chart shows groups of messages as solid bars when individual messages are too close together to be seen at the current zoom level.

A message bar is different from a message group, which is a distinct object that represents a series of messages between two tiers. QuickPredict and simulations convert message groups into individual messages, but the component messages cannot be seen in the Transaction Whiteboard. For more information, see How a Message Group Models Individual Messages.

You can distinguish between message bars (which are split horizontally) and message groups (which are split diagonally) in the Data Exchange Chart.

Figure 24-4 Message Bars vs. Message Group: Example



message bar (divided horizontally) shows individual messages when zoomed in

message group (divided diagonally) models overall data transfer, but individual message cannot be seen

Transactions vs. Subtasks

When you open a Transaction Analyzer model, Transaction Whiteboard groups the application task into subtasks based on the application transactions found in the original Transaction Analyzer model. However, subtasks and application transactions are related but different concepts.

When a traffic packet trace is opened in Transaction Analyzer, application messages are grouped into transactions. In Transaction Whiteboard, you can group elements into subtasks according to your preferences. Thus the Transaction Whiteboard user—not the software—ultimately determines the organization of an application task.

Related Topics

- *Transaction Whiteboard: Overview*