28 Modeling Applications over UDP with Transaction Whiteboard

You can run discrete event simulations for UDP-based Transaction Analyzer and Transaction Whiteboard models. Transaction Whiteboard includes a framework for specifying how an application responds if packets are lost over UDP. This framework is called *UDP Drop Response*, and is necessary in order to collect statistics for the response times of applications over UDP.

This section has the following topics:

- Why UDP Drop Response is Needed
- UDP Drop Response Mechanisms in Transaction Whiteboard
- Specifying UDP Drop Response in Transaction Whiteboard Models
- Viewing Drop Response Behavior During a Discrete Event Simulation
- Creating Customized Drop Response Mechanisms (Advanced)

Why UDP Drop Response is Needed

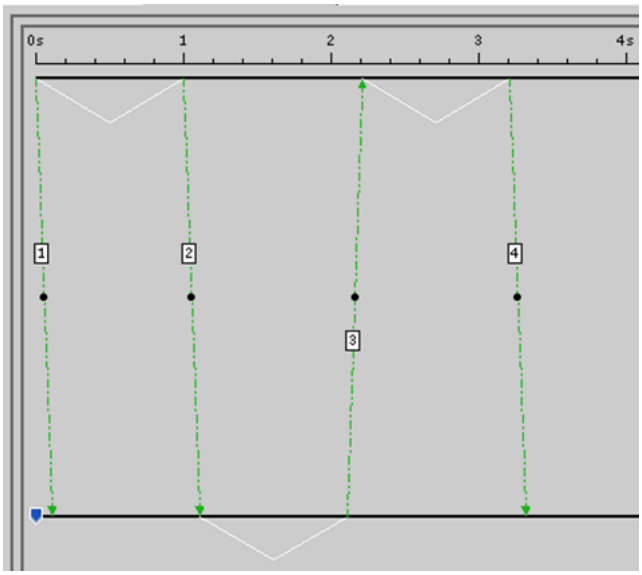
User Datagram Protocol (UDP) is a transport protocol that is used to send messages over the Internet. UDP does not provide the reliability and ordering of TCP: datagrams may arrive out of order, appear duplicated, or go missing without notice. Without the overhead of checking whether every packet actually arrived, UDP is faster and more efficient for many lightweight or time-sensitive purposes.

An application running over UDP must be able to handle missing or duplicate packets. An application can handle a packet drop in many ways—from simply ignoring any drops, to retransmitting the missing packet until it reaches the destination.

Effects of Packet Loss in Discrete Event Simulations

When traffic from a Transaction Analyzer model or a Transaction Whiteboard model is deployed in a discrete event simulation, the kernel reads the file and creates a list of actions to be executed.

Figure 28-1 Example Application



Consider the application in the above figure, which executes the following actions:

1) Tier A sends packet 1 to Tier B
2) A waits 1 second
3) A sends packet 2 to B

   If B receives packet 2, then:

4) B processes packet 2

5) B waits 1 second

6) B sends packet 3 to A

   If A receives packet 3, then:

7) A processes packet 3

8) A waits 1 second

9) A sends packet 4 to B

What happens if the application runs over UDP and packets are lost? If packet 1 is lost, the application can recover; but if packet 2 is lost, the application might be unable to recover. Unless the application has a recovery mechanism, B waits for packet 2 and might not continue (because it does not know how to handle lost packets).

Now consider how a discrete event simulation might handle this. Without a recovery mechanism, the simulation cannot run the entire application if packets 2 or 3 are lost (that is, if either step 3) or step 6) do not complete successfully). If the application does not finish, the simulation cannot collect statistics for AppTransaction Xpert or Custom Application response times.

UDP Drop Response enables you to specify how a simulation handles lost packets. For example, you might want to specify a “time-out” mechanism that ensures that the application continues even if packets are lost:

1) Tier sends packet

2) Simulation starts timer

   If packet is received OR if <timeout_interval> is reached, then:

3) Continue running application

Note—You must specify UDP Drop Response for applications over UDP, even if the application sends only one packet over a lossy link. Otherwise the application might never recover from a packet loss.

UDP Drop Response Mechanisms in Transaction Whiteboard

This section describes the different UDP Drop Response mechanisms that can be modeled in Transaction Whiteboard. Every UDP connection has an associated drop response mechanism; by default, every message on that connection uses the mechanism defined for that connection.

The following drop response mechanisms are supported:

- Ignore Packet Drops—UDP Drop Response Option #1
- Acknowledgment (ACK) Based—UDP Drop Response Option #2
- Negative Acknowledgement (NACK) Based—UDP Drop Response Option #3

Advanced users can also create their own customized mechanisms, as described in Creating Customized Drop Response Mechanisms (Advanced).

Ignore Packet Drops—UDP Drop Response Option #1

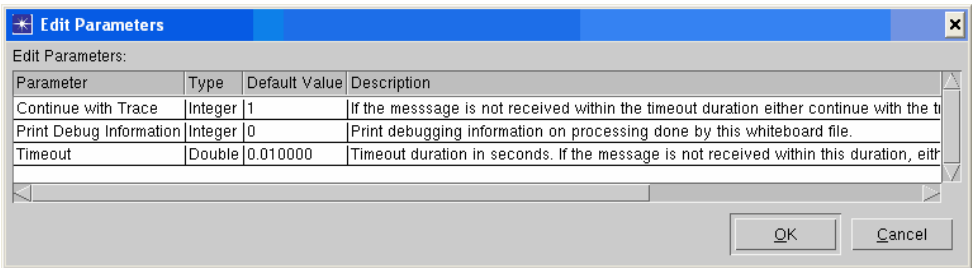
The Ignore mechanism works as follows:

1) The sender tier sends a packet, and the simulation starts a timer.

2) If the receiver does not receive the packet within the timeout window, the simulation either continues running the application or aborts it (depending on how the “Continue with Trace” parameter is set).

In this case, any action that depends on the successful transmission of a previous message is suspended until that message is received or the message timer expires.

Figure 28-2 Parameters for UDP Drop Response (Ignore)



Acknowledgment (ACK) Based—UDP Drop Response Option #2

The ACK-based mechanism works like this:

1) The sender tier sends a packet, and the simulation starts a timer. If the receiver tier receives the packet, it returns an ACK.

2) If the sender does not receive the ACK within the timeout window, it retransmits the packet and the simulation resets the timer. The sender repeats this process until

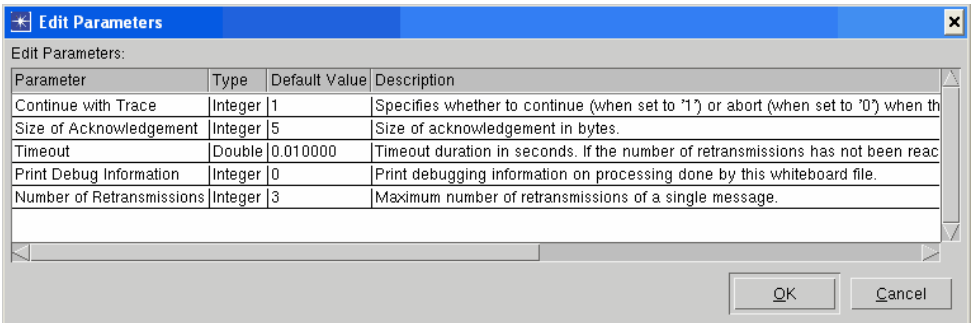
   a) It receives an ACK for the packet, OR

   b) The maximum number of retransmissions is reached.

3) If the sender still has not received an ACK after the maximum number of retransmissions, the simulation continues running the application or aborts.

In this case, if an action depends on the successful transmission of a message, that action is suspended until the message is received or the message timer for the last retransmitted message expires.

Figure 28-3 Parameters for UDP Drop Response (ACK-Based)



Negative Acknowledgement (NACK) Based—UDP Drop Response Option #3

The NACK-based mechanism works as follows:

1) The sender tier sends a packet, and the simulation starts a timer.

2) If the receiver tier does not receive the packet within the timeout window, it returns a NACK.

3) If the sender receives a NACK, it retransmits the packet and the simulation resets the timer. The sender repeats this process until

   a) it sends a packet without receiving a NACK in return, OR

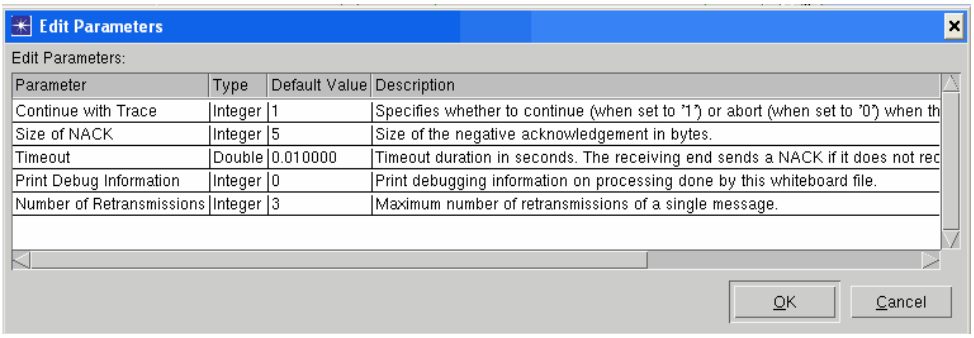
   b) the maximum number of retransmissions is reached.

4) If the sender is still receiving NACKs after the maximum number of retransmissions, the simulation continues running the application or aborts.

In this case, if an action depends on the successful reception of a previous message, that action is suspended until the message timer expires or the message is received. If a NACK is received, the packet is retransmitted.

Note—This mechanism is not immune against NACK losses; if both the packet and the NACK are lost, the packet will not be retransmitted.

Figure 28-4 Parameters for UDP Drop Response (NACK - Based)



Specifying UDP Drop Response in Transaction Whiteboard Models

Note the following restrictions and requirements for specifying UDP Drop Response:

- You can specify UDP Drop Response in Transaction Whiteboard only. If you want an existing Transaction Analyzer model file to support UDP Drop Response, you must import that file into Transaction Whiteboard and set the drop response for all messages that run over UDP.
- The Transaction Whiteboard model cannot have more than two tiers.
- The Transaction Whiteboard model can have only one UDP connection.

The following procedure describes how to specify UDP drop responses for messages.

Procedure 28-1 Specifying UDP Drop Response for Messages in Transaction Whiteboard

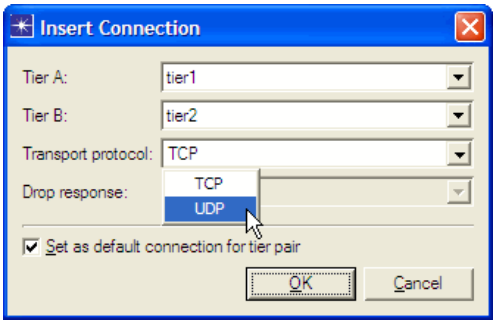
1 Create or open a Transaction Whiteboard model.

To specify drop response in a Transaction Analyzer model, import the model into Transaction Whiteboard.

2 Insert a new connection and specify the UDP parameters:

2.1 Choose Insert > Connection.

➥ The Insert Connection dialog box appears.



2.2 Set the fields as needed. Note the following fields:

- Transport Protocol field = UDP
- Retransmission Mechanism—Specifies the default drop response mechanism for this connection. Every new message assigned to this connection will use this mechanism by default. (You can specify a different mechanism for individual messages in the Message Editor.)

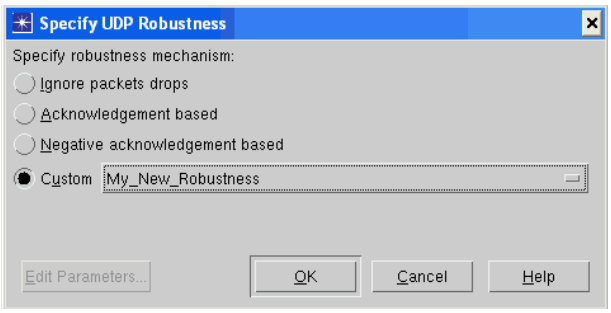
  For more information about the different mechanisms, see UDP Drop Response Mechanisms in Transaction Whiteboard.
- Set as default connection for tier pair—If this option is selected, any new message will be assigned to this connection by default.

3 Optionally, change the UDP settings for individual messages:

3.1 Select the messages.

3.2 In the Message Editor table, set the Connection and Drop Response fields for the selected messages as desired. To change or configure the mechanism, click in the Drop Response field and choose Edit.

Figure 28-5 Specifying a UDP Drop Response Mechanism



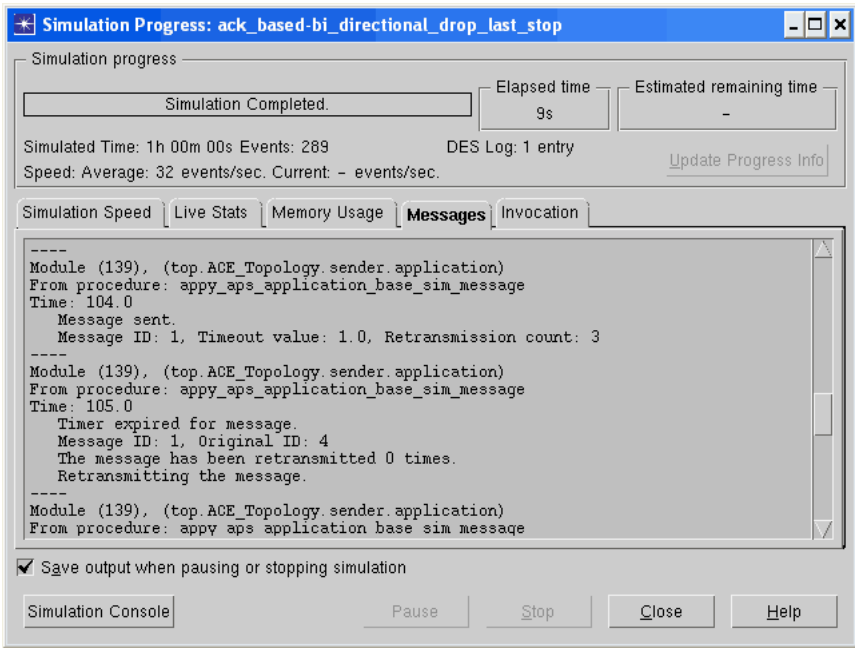
End of Procedure 28-1

Viewing Drop Response Behavior During a Discrete Event Simulation

When running a simulation that invokes UDP Drop Response behavior, you can view log messages about the actions taken. These messages appear in the Messages tabbed page of the Simulation Progress dialog box (or in the OPNET Debugger console). The following figure shows information generated when messages are sent and retransmitted: time, message ID, and some configuration details.

Note—These messages are generated only if the UDP Drop Response attribute “Print Debug Information” is set to 1 on the connection used by the packet.

Figure 28-6 Viewing Trace Output for a Custom UDP Drop Response Algorithm



Creating Customized Drop Response Mechanisms (*Advanced*)

You can define a custom drop response mechanism in a Transaction Whiteboard model, as described in Procedure 28-2. This type of Transaction Whiteboard model defines

- Configurable parameters for that action
- The actions to be run when a message is received or when its timer expires

When the simulation detects a UDP Drop Response mechanism in association with a packet, it does not process that packet. Instead, it runs the Transaction Whiteboard model that defines the drop response mechanism. The Transaction Whiteboard model should contain a script that does the following:

- Ensures that the packet size and connection information for the original message are retained.
  (This is the message that launched the UDP Drop Response mechanism, as shown in Figure 28-7.)
- Registers a timer callback for the message.
  This callback specifies the action(s) to be executed when the timer expires. The script should set a timer callback on any message that is sent (including ACKs).
- Registers a receipt callback for the message.
  This callback specifies the action(s) to be executed—such as cancel the timer or send an ACK—when a message is received.

The timer and receipt callbacks are defined in the Function Block (Scripting > Function Block) of the Transaction Whiteboard model file.

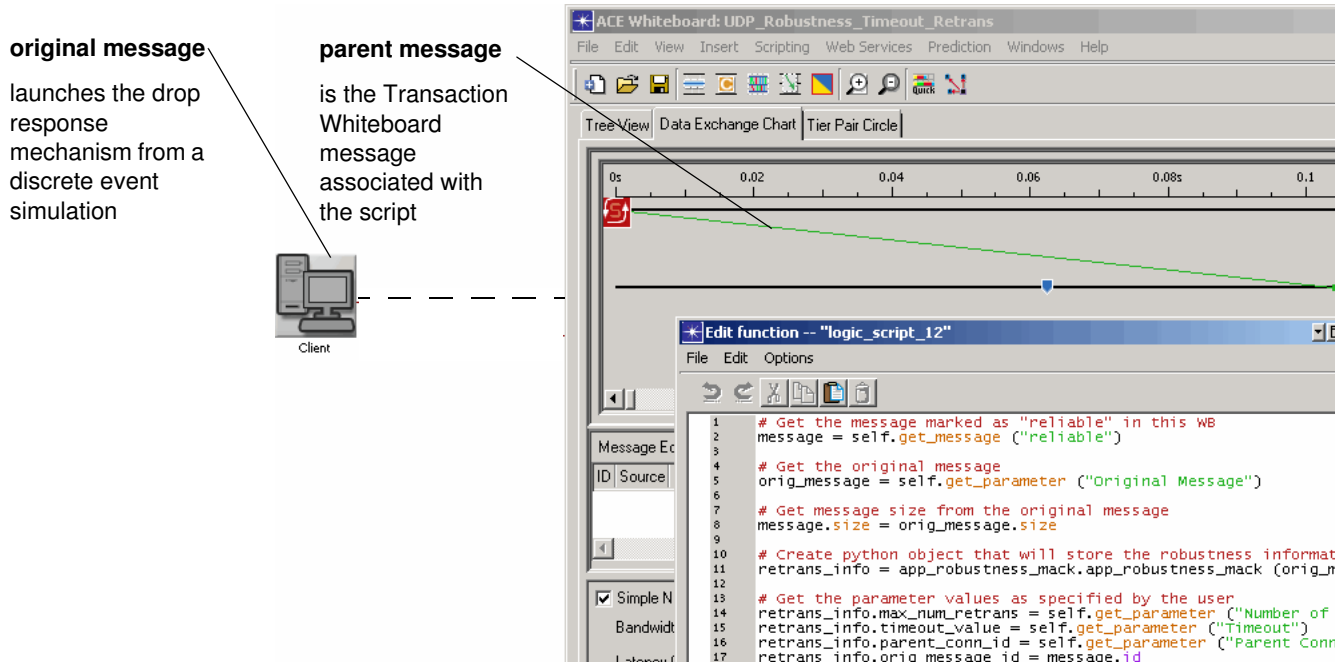
Procedure 28-2 Creating a Custom UDP Drop Response Mechanism

1. Create a new Transaction Whiteboard model file.
2. Define parameters for the new model (choose Scripting > Declare Parameters in Transaction Whiteboard).
3. Create a message between two tiers.
4. Add a script to the beginning of the message you just created.

   Note—In this procedure, you need to work with two different messages. This procedure uses the following terms:

   - Original message—The message in the simulation that launched this drop response mechanism Transaction Whiteboard model file
   - Parent message—The message in the current Transaction Whiteboard model file that is the parent of the current script

Figure 28-7 Original Message and Parent Message



5 Specify a tag for the parent message in the Message Editor table.

6 In the Function Block (Scripting > Function Block), write the receipt and timeout callback functions for the original message. These functions define the actions to take when a message is received and when its timer expires.

7 Edit the script you created in step 4. This script needs to do the following:

7.1 Get a handle to the original message. You can access this message using get_parameter(); use “Original Message” for the `name` argument.

7.2 Set the parent message size to the size of the original message.

7.3 Read and store the parameters defined in the Declare Parameters dialog box.

7.4 Set the connection index of the parent message to the index of the original message.

7.5 Register the receipt and timeout callback functions that define the actions to take when a message is received and when its timer expired. There should be a timer associated with any message (including ACKS) sent in Transaction Whiteboard.

8 After defining drop response logic in this Transaction Whiteboard model, you can invoke the model in discrete event simulations.

For more information, see:

- Specifying UDP Drop Response in Transaction Whiteboard Models
- Viewing Drop Response Behavior During a Discrete Event Simulation

End of Procedure 28-2