

Lab 9 TCP versus UDP Response Time

Overview

TCP provides connection-oriented service at the transport layer, and UDP provides connectionless service. As a result, a data exchange using TCP can take longer than the same exchange using UDP. TCP implementations require that the TCP source and destination perform a three-way handshake in order to set up the connection prior to sending data, and a four-way handshake when tearing down the connection. In addition, all data must be acknowledged by the destination when it is received. UDP sources, on the other hand, do not set up connections and so save the overhead in terms of delay and bandwidth usage. UDP is often used when small amounts of data must be sent, for example, when doing credit card verification. If large amounts of data are sent, however, the extra TCP overhead may be negligible in comparison to the entire transaction completion time.

Objective

To examine the response time for large and small data transfers when using TCP and UDP.

Build the Simulation Model

Start up OPNET IT Guru Academic Edition.

Select the **File** tab => **New...**

Choose **Project** and click on **OK**.

Change the **Project Name** to **xx_TCP_vs_UDP** (where **xx** are your initials). Set the **Scenario Name** to **TCP** and click on **OK**.

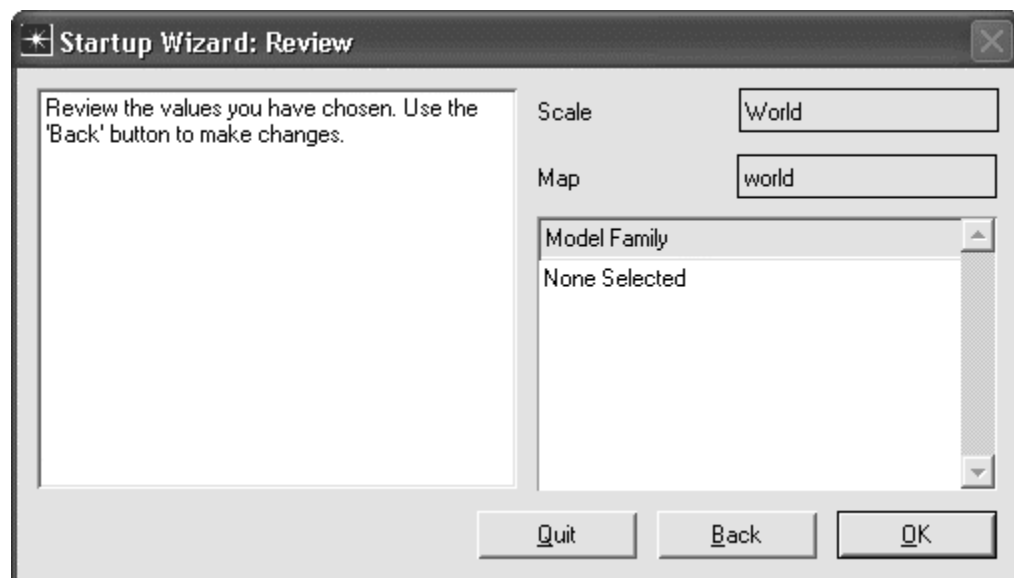
In the **Initial Topology** window, select **Create Empty Scenario** and click on **Next**.

In the **Choose Network Scale** window, select **World** and click on **Next**.

In the **Choose Map** window, choose **world** and click on **Next**.

In the **Select Technologies** window, click on **Next**.

In the **Review** window, click on **OK**.



First, it would be helpful to familiarize yourself with the TCP functionality of OPNET and with the methods for defining new applications.

Select the **Protocols** tab => **TCP** => **Model Usage Guide**. Read through the guide and click on the close window icon to close the PDF viewer when you are done.

Select the **Protocols** tab => **Applications** => **Model Usage Guides** => **Configuring Profiles and Applications**. Read through the guide and click on the close window icon to close the PDF viewer when you are done.

Next, we will create a client-server system running over a wide area network.

Select an **ip32_cloud** object from the Object Palette and place it in the project workspace. Right click on the cloud and choose **View Node Description**. The cloud represents a WAN consisting of IP-capable that supports up to 32 serial links.

Right click on the cloud and select **Set Name**. Set the name to **ip32_cloud**. Click on **OK** to close the window.

In the Object Palette, click on **Configure Palette**. Click on **Node Models**, and choose to include the **ppp_wkstn_adv** model. Click on **OK** twice and save your palette with the default name given when the **Save As** window comes up.

Select a **ppp_wkstn_adv** device from the Object Palette and place it in the project workspace.

Right click on the station and choose **View Node Description**. Note that the station supports the TCP and UDP protocols.

Right click on the station and choose **Edit Attributes**. Modify the **name** attribute of the device to **Application Client**.

Modify the **Client Address** to be **Application Client** as well.

Click on **OK** to close the window.

Select a **ppp_server** device from the Object Palette and place it in the project workspace.

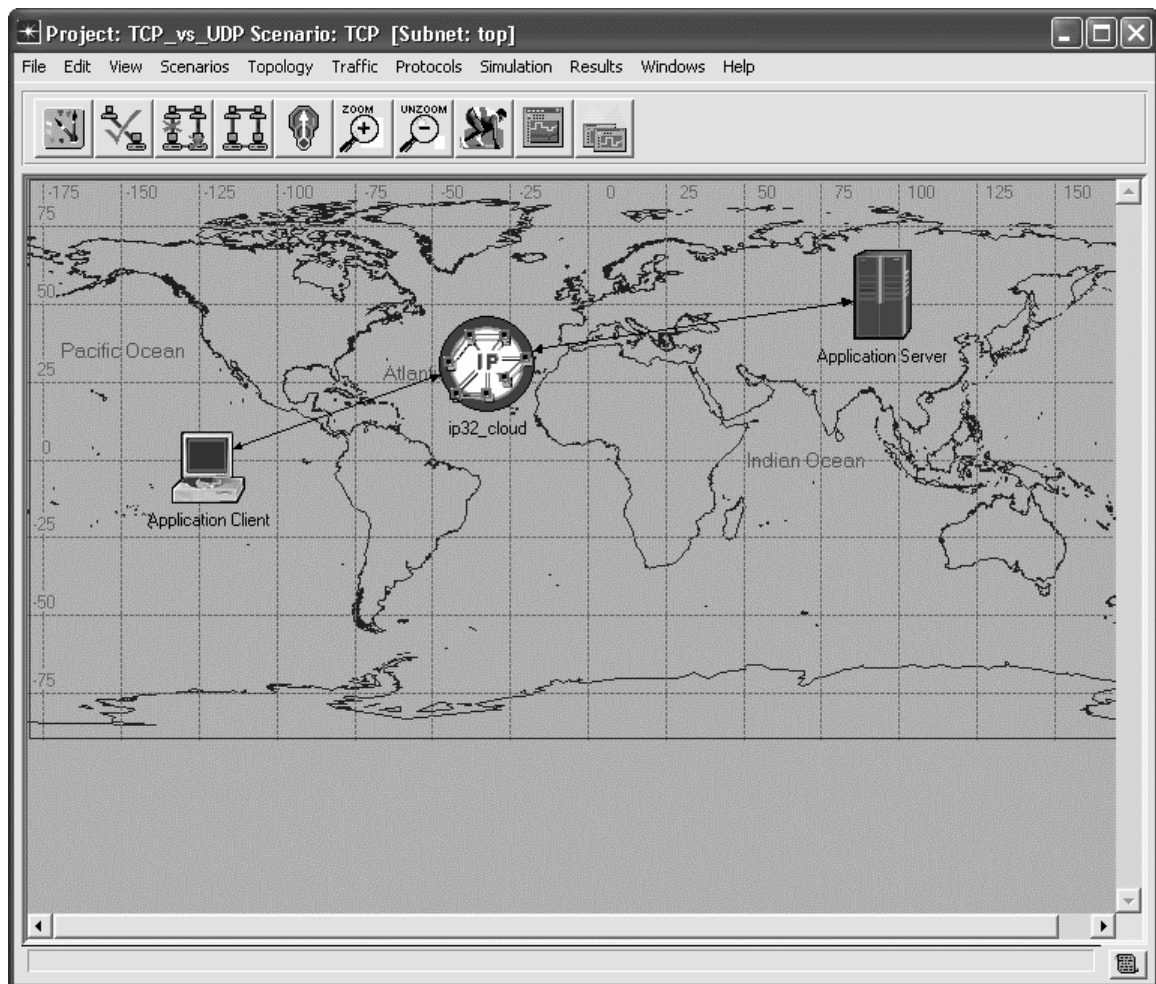
Right click on the server and choose **View Node Description**. Note that the server supports the TCP and UDP protocols.

Right click on the device and choose **Edit Attributes**. Modify the **name** attribute of the server to **Application Server**.

Modify the **Server Address** to be **Application Server** as well.

Click on **OK** to close the window.

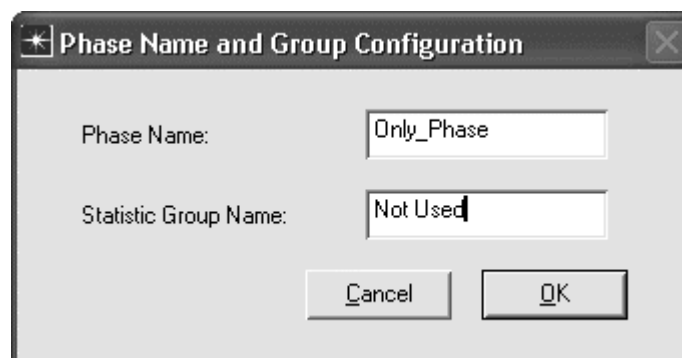
Select two **PPP_DS1** links from the Object Palette and use them to connect the **Application Client** to the **ip32_cloud** and the **Application Server** to the cloud. The PPP protocol is commonly used for long-distance links. Also, remember that DS1 speed is 1.5 Mbps.



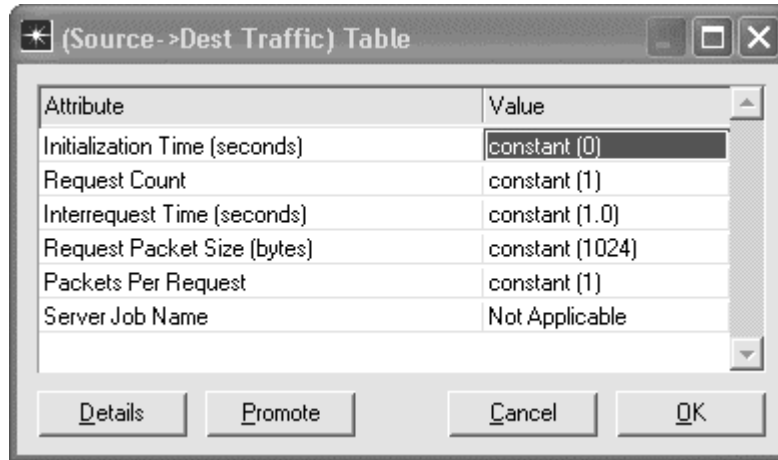
Now we will create a custom application which can run on top of either TCP or UDP. The application will require that the client send a single request to the server and that the server reply with a single response. A task definition must be created that will be used to define an application. That application will then be incorporated into a profile that we can associate with our client and server.

Select a **Task Config** object from the Object Palette and place it in the project workspace. Right click on the object and choose **Edit Attributes**. Set the **name** attribute to **Task**.

Expand the **Task Specification** attribute and set the **rows** attribute to **1**. Expand the **row 0** attribute and set the **Task Name** attribute to **Request_Response_Task**. Now edit the **Manual Configuration** attribute to define the task. Set the number of **rows** to **1**. Edit the **Phase Name** and set it to **Only_Phase**. Click on **OK** to close the window.



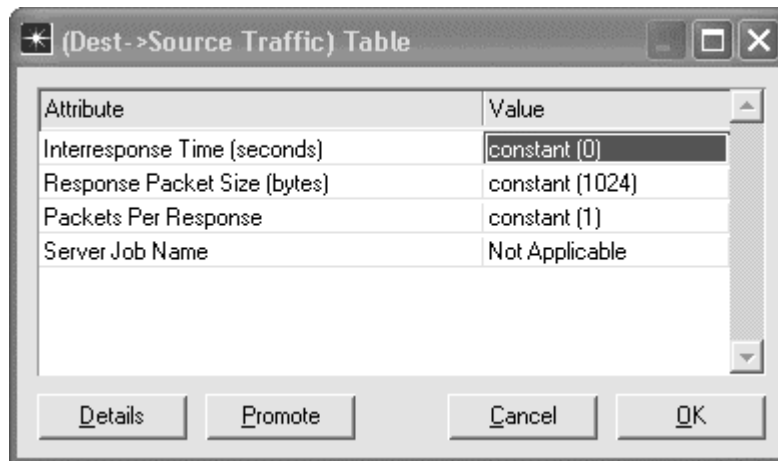
Set the **Source** to be **Application Client**. Set the **Destination** to be **Application Server**. Edit the **Source->Dest Traffic** field, and set the **Initialization Time** to **constant(0)**, the **Request Count** to **constant(1)**, the **Interrequest Time (seconds)** to **constant(1.0)**, and the **Packets per Request** to **constant(1)**. Click on **OK** to close the window. This task will cause the client to send one request packet every second.



Attribute	Value
Initialization Time (seconds)	constant (0)
Request Count	constant (1)
Interrequest Time (seconds)	constant (1.0)
Request Packet Size (bytes)	constant (1024)
Packets Per Request	constant (1)
Server Job Name	Not Applicable

Details Promote Cancel OK

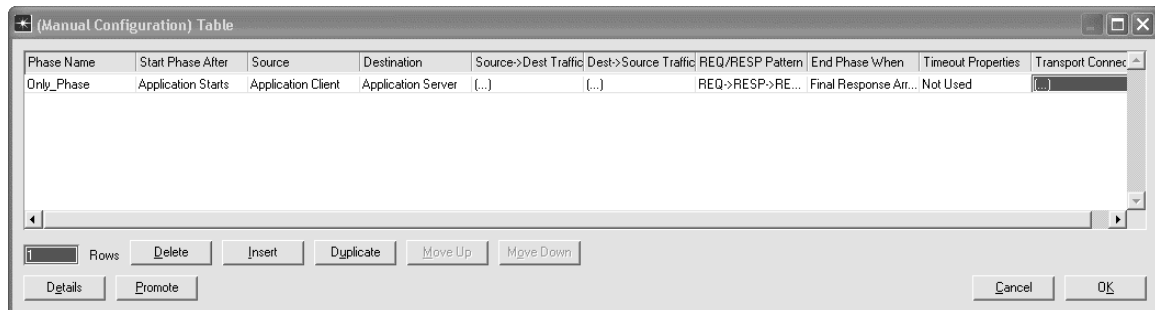
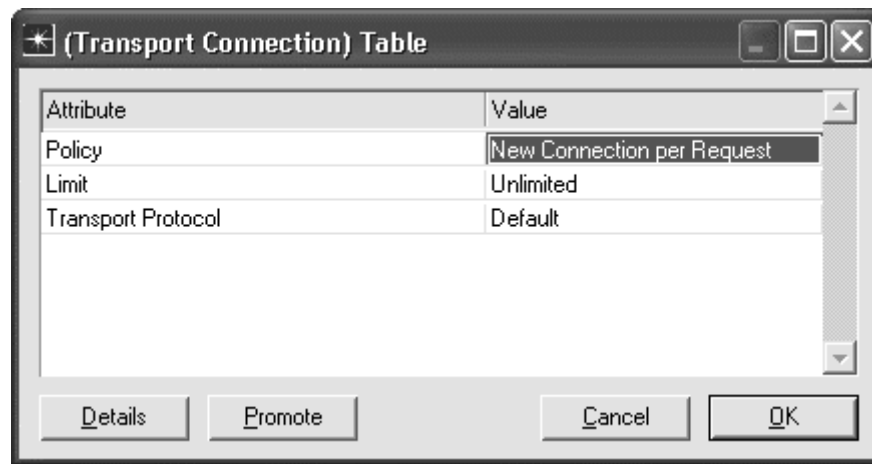
Edit the **Dest->Source Traffic** field, and set the **Interresponse Time** to **constant(0)**. Click on **OK** to close the window. This task will cause the server to respond to requests immediately with a single application-layer packet.

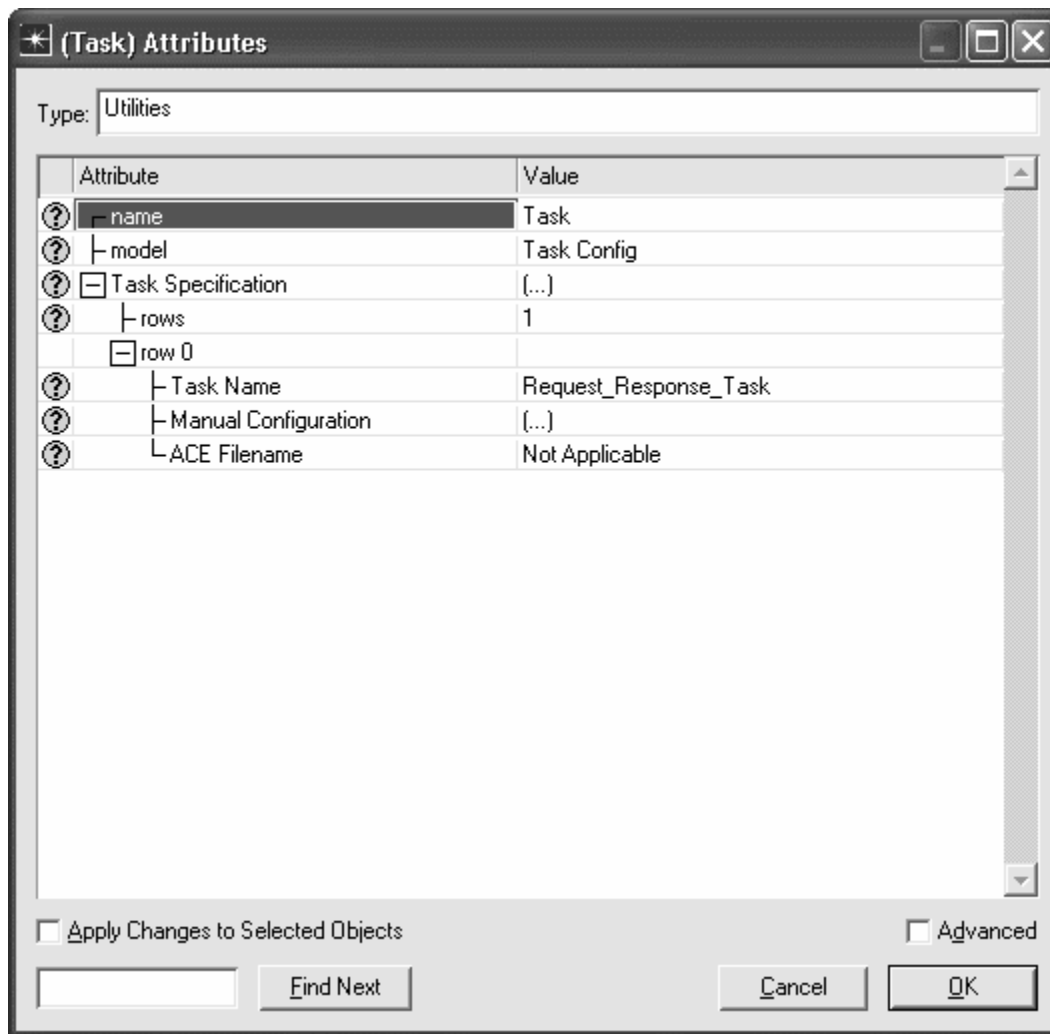


Attribute	Value
Interresponse Time (seconds)	constant (0)
Response Packet Size (bytes)	constant (1024)
Packets Per Response	constant (1)
Server Job Name	Not Applicable

Details Promote Cancel OK

Edit the **Transport Connection** field, and set the **Policy** to **New Connection per Request**. Click on **OK** three times to complete the task configuration.





Right click on the **Application Config** object that appeared in the project workspace when we completed the task definition, and choose **Edit Attributes**. Set the **name** attribute to **Application**.

Expand the **Application Definitions** attribute and set the **rows** attribute to **1**. Expand the **row 0** attribute and set the **Name** attribute to **Request_Response_Application**. Expand the **Description** attribute and edit the value of the **Custom** attribute. Note that the **Transport Protocol** is currently set to **TCP**. Edit the **Task Description** attribute. Set the number of **rows** to **1**. Set the **Task Name** to **Request_Response_Task**. Click on **OK** three times to close the windows and complete the custom application configuration.

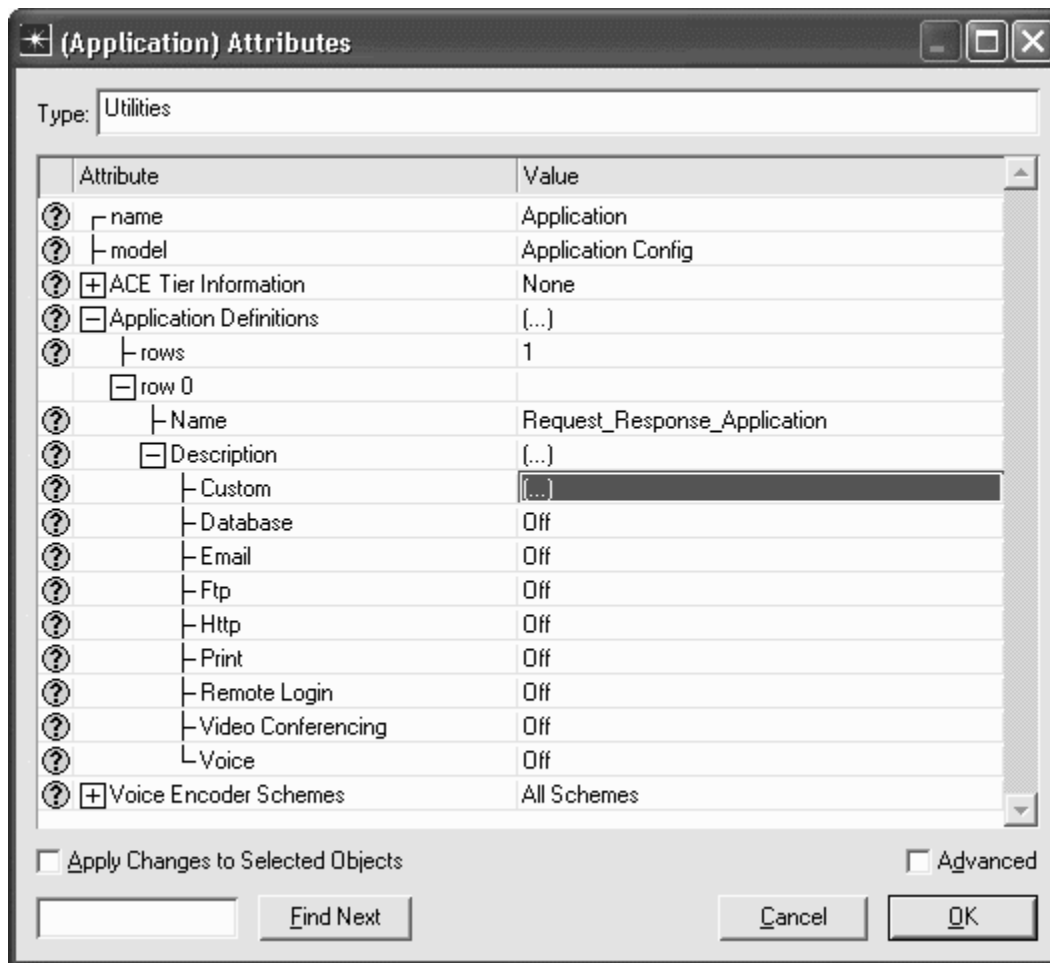
(Task Description) Table

Task Name	Task Weight
Request_Response_Task	10

1 Rows

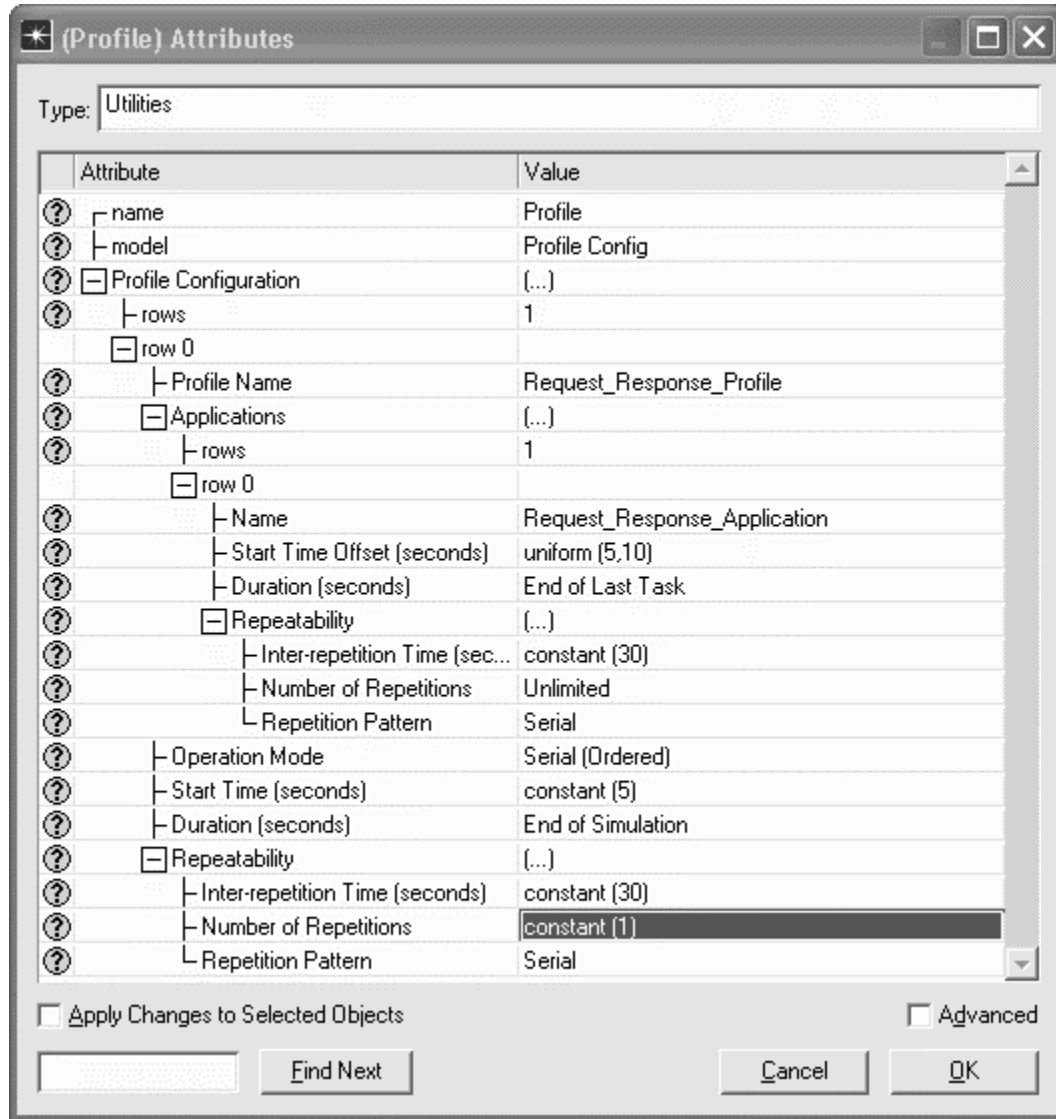
(Custom) Table

Attribute	Value
Task Description	[...]
Task Ordering	Serial (Ordered)
Transport Protocol	TCP
Type of Service	Best Effort (0)
Refresh Connection	After Every Phase
RSVP Parameters	None
Back-End Custom Application	Not Used



Select a **Profile Config** object from the Object Palette and place it in the project workspace. Right click on the object and choose **Edit Attributes**. Set the **name** attribute to **Profile**.

Expand the **Profile Configuration** attribute and set the **rows** attribute to **1**. Expand the **row 0** attribute and set the **Profile Name** to **Request_Response_Profile**. Expand the **Applications** attribute and set the **rows** attribute to **1**. Expand the **row 0** attribute and set the **Name** to **Request_Response_Application**. Set the **Duration (seconds)** to **End of Last Task**. Expand the **Repeatability** attribute and set the **Inter-repetition Time (seconds)** to **constant(30)**. Set the **Start Time (seconds)** to **constant(5)**. Expand the second **Repeatability** item, in the main level hierarchy, and set the **Inter-repetition Time (seconds)** to **constant(30)** and the **Number of Repetitions** to **constant(1)**. Click on **OK** to close the window. This profile will cause the **Request_Response_Application** to run every 30 seconds.



Now we will associate our custom application with our client and server. Right click on the **Application Client** and choose **Edit Attributes**. Edit the **Application: Supported Profiles** attribute. Set **rows** to **1**, and expand the **row 0** attribute. Set the **Profile Name** to **Request_Response_Profile**. Edit the **Application: Destination Preferences** attribute. Set the number of **rows** to **1**, and the **Symbolic Name** to **Application Server**. Edit the **Actual Name** field, set the number of **rows** to **1**, and set the **Name** to **Application Server**. Click on **OK** twice to close the windows. Edit the **Application: Source Preferences** attribute, set the number of **rows** to **1**, and set the **Symbolic Name** to **Application Client**. Click on **OK** to close the window.

(Actual Name) Table

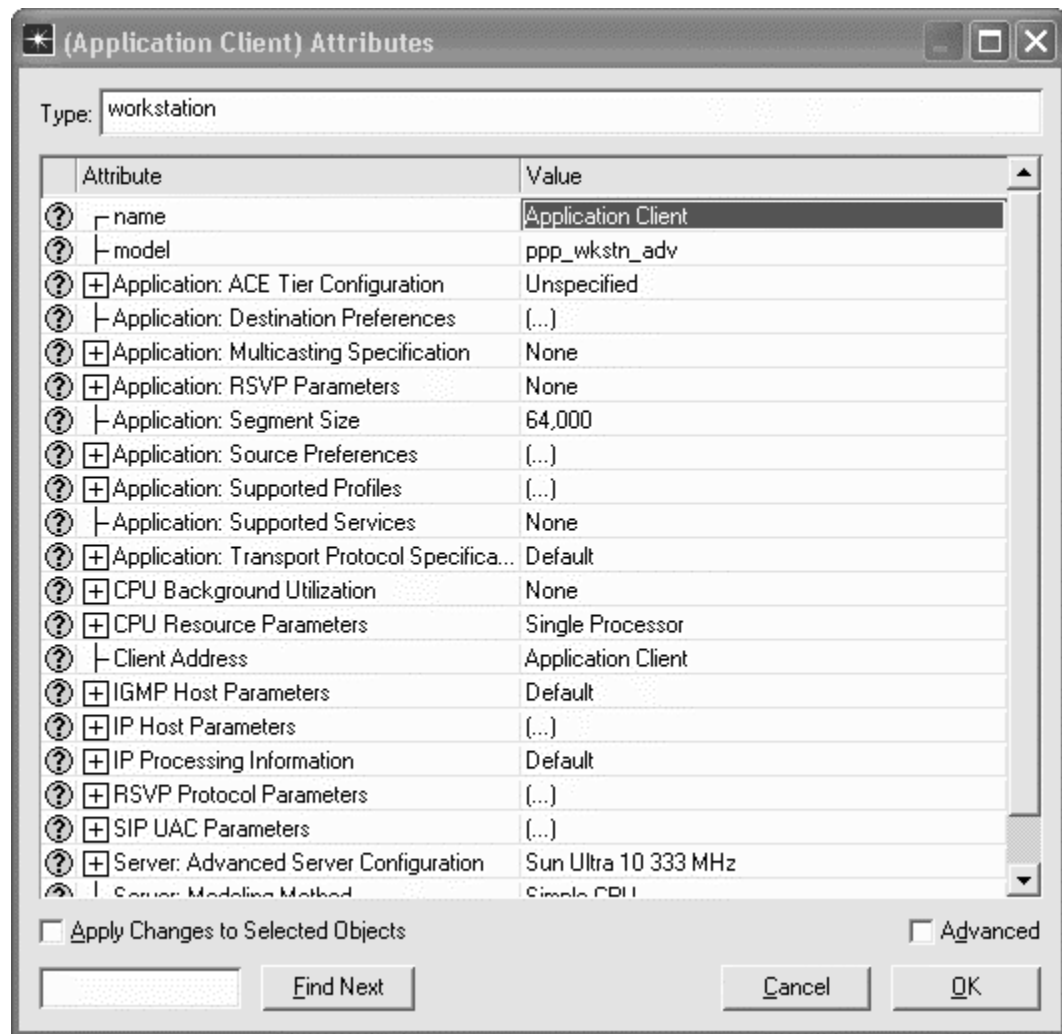
Name	Selection Weight
Application Server	10

1 Rows

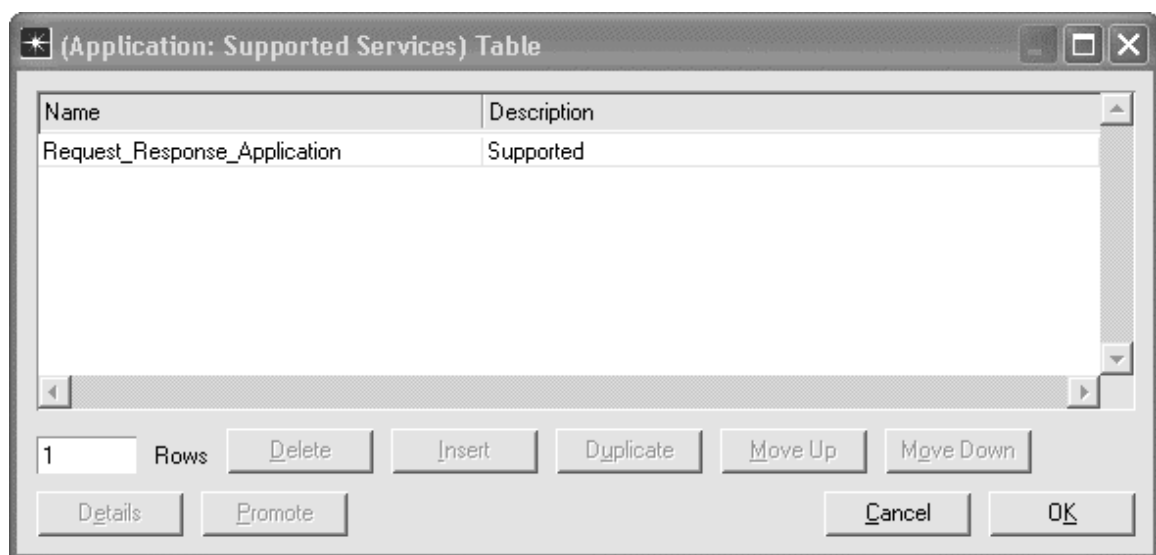
(Application: Destination Preferences) Table

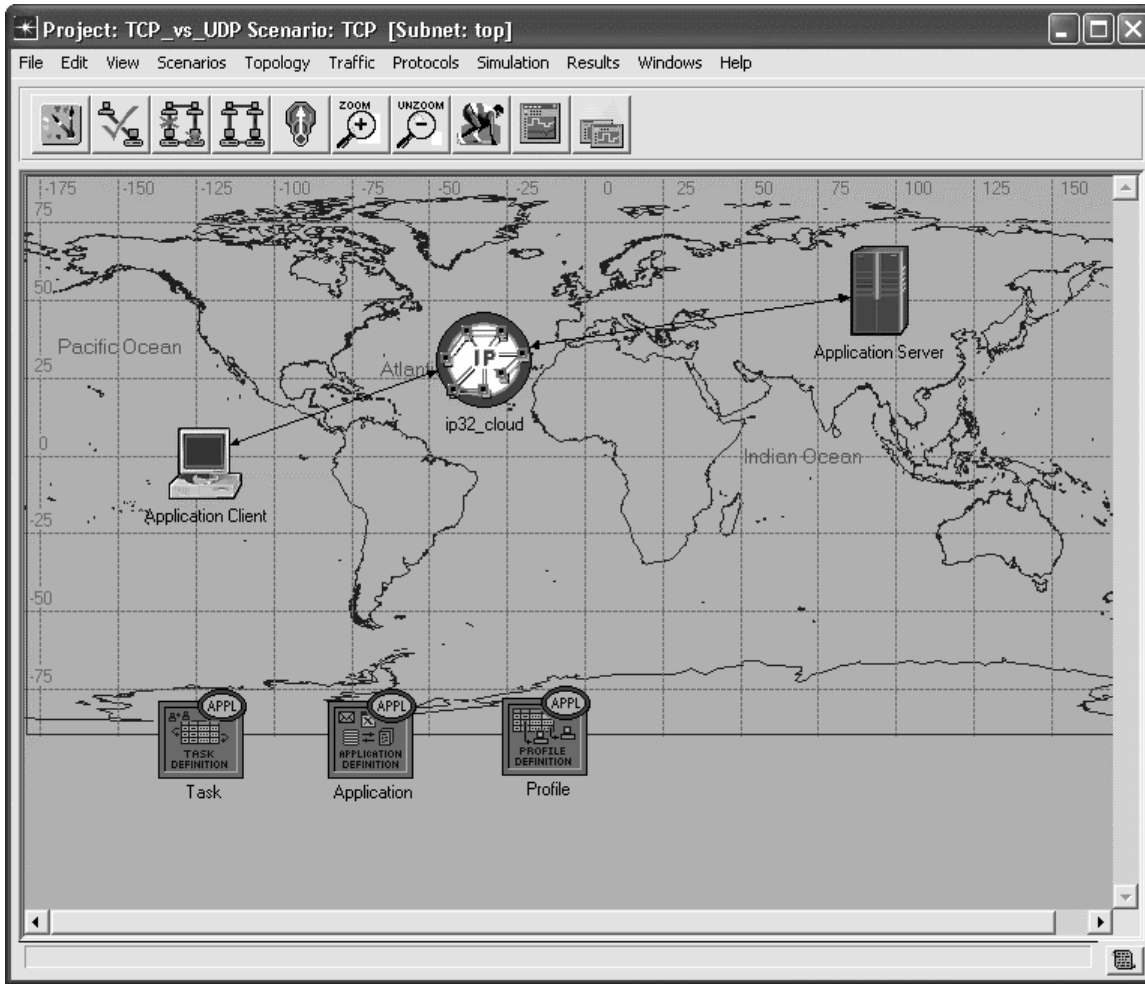
Symbolic Name	Actual Name
Application Server	[...]

1 Rows



Right click on the Application Server and choose **Edit Attributes**. Edit the **Application: Supported Services** attribute. Set **rows** to **1**, and set the **Name** to **Request_Response_Application**. Click on **OK** twice to close the windows.





Configure the Simulation

Select the **Simulation** tab => **Choose Individual Statistics...**

Expand the **Global Statistics** item and the **Custom Application** item, and select the **Task Response Time (seconds)** statistic.

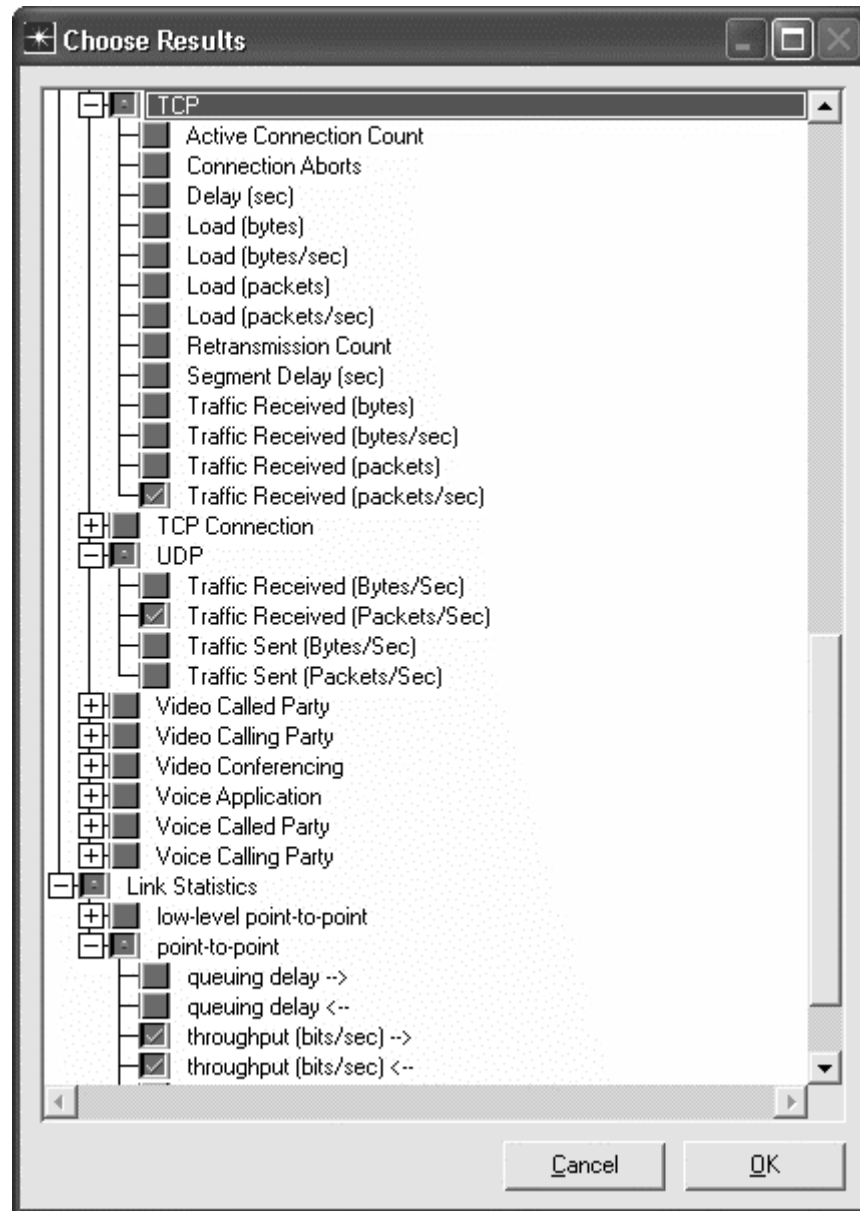
Expand the **Node Statistics** item and the **IP** item, and select the **Traffic Received (packets/sec)** and **Traffic Sent (packets/sec)** statistics.

Expand the **TCP** item and select the **Traffic Received (packets/sec)**.

Expand the **UDP** item and select the **Traffic Received (packets/sec)**.

Expand the **Link Statistics**, and the **point-to-point** item, and select **throughput (bits/sec) -->** and **throughput (bits/sec) <--**.

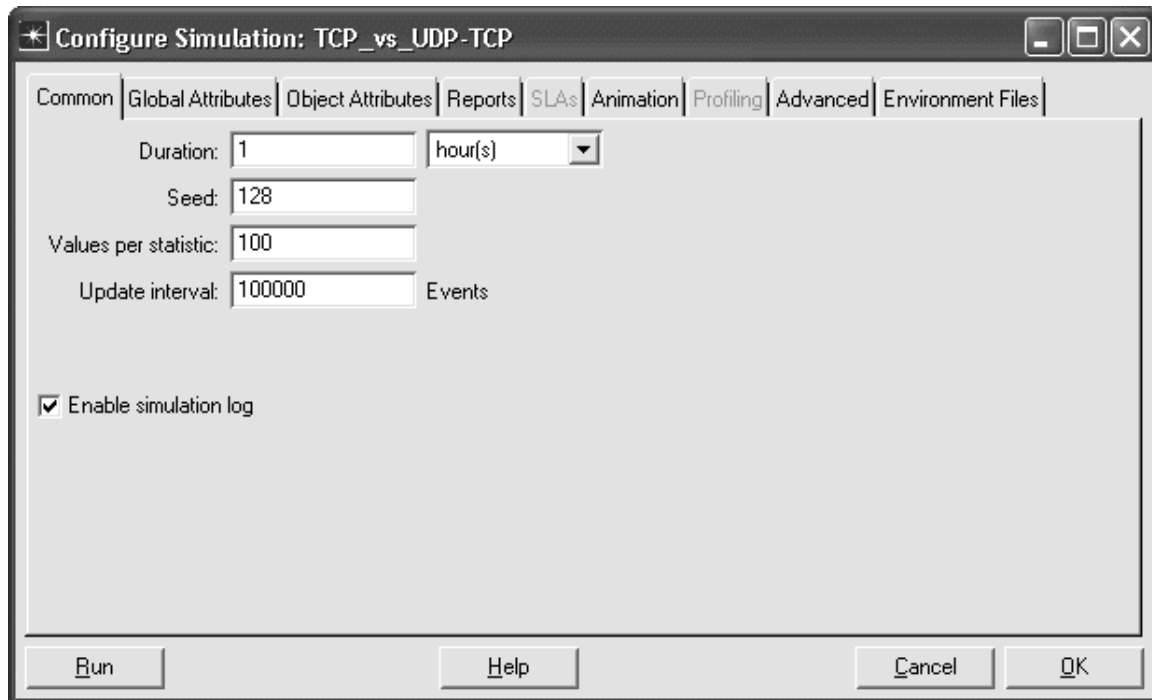
Click on **OK** to close window.



Select **Simulation => Configure Discrete Event Simulation...**

Under the **Common** tab, set the **Duration** to **1**, and the unit to **hour(s)**.

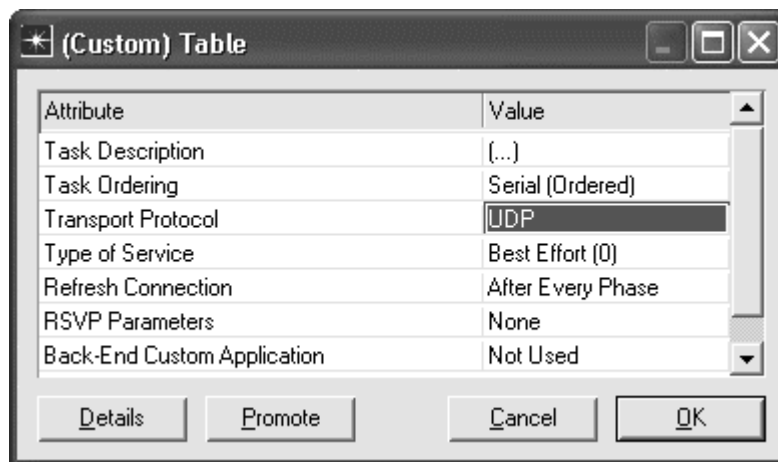
Click on **OK** to close the window.



Duplicate the Scenario

We are now going to duplicate the scenario and configure the application to use UDP instead. This will allow us to compare the performance of the application across the two transport layer protocols. Choose **Scenarios** => **Duplicate Scenario**, and name the new scenario **UDP**.

Right click on the **Application Config** object, expand the **Application Definitions** item, the **row 0** item, and the **Description** item. Edit the **Custom** application definition, and modify the **Transport Protocol** to **UDP**. Click on **OK** twice to close the windows.

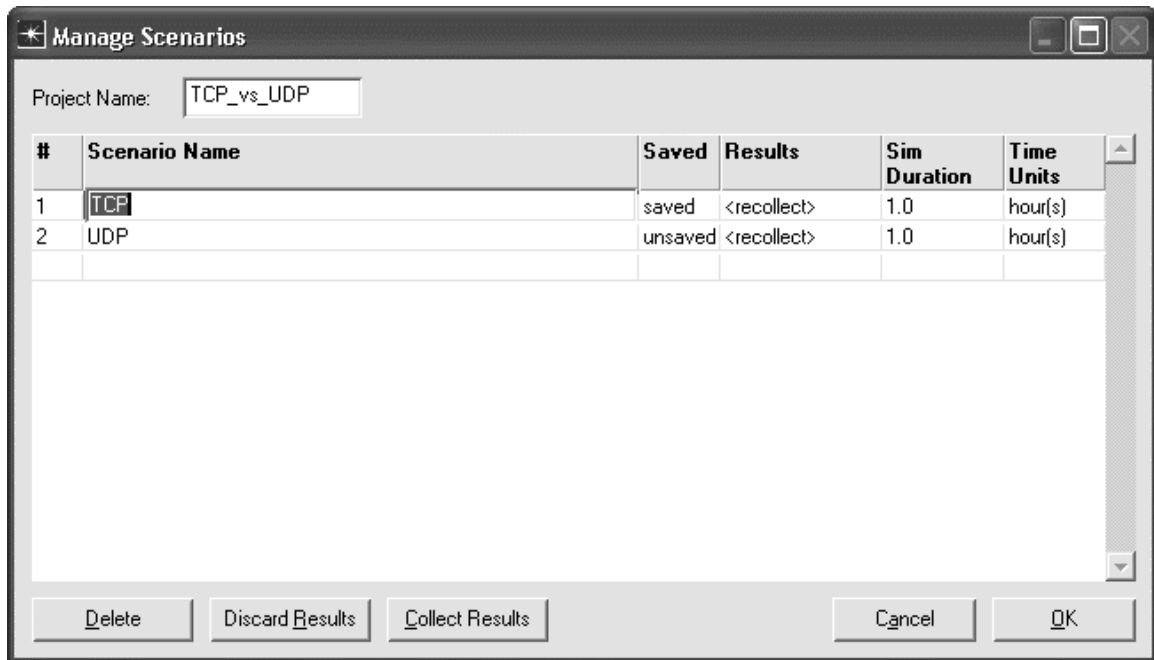


Run the Simulation

Select the **Scenarios** tab => **Manage Scenarios...**

Edit **Results** field in both rows and set the value to **<collect>** or **<recollect>**.

Click on **OK** to run both scenarios (one after the other).

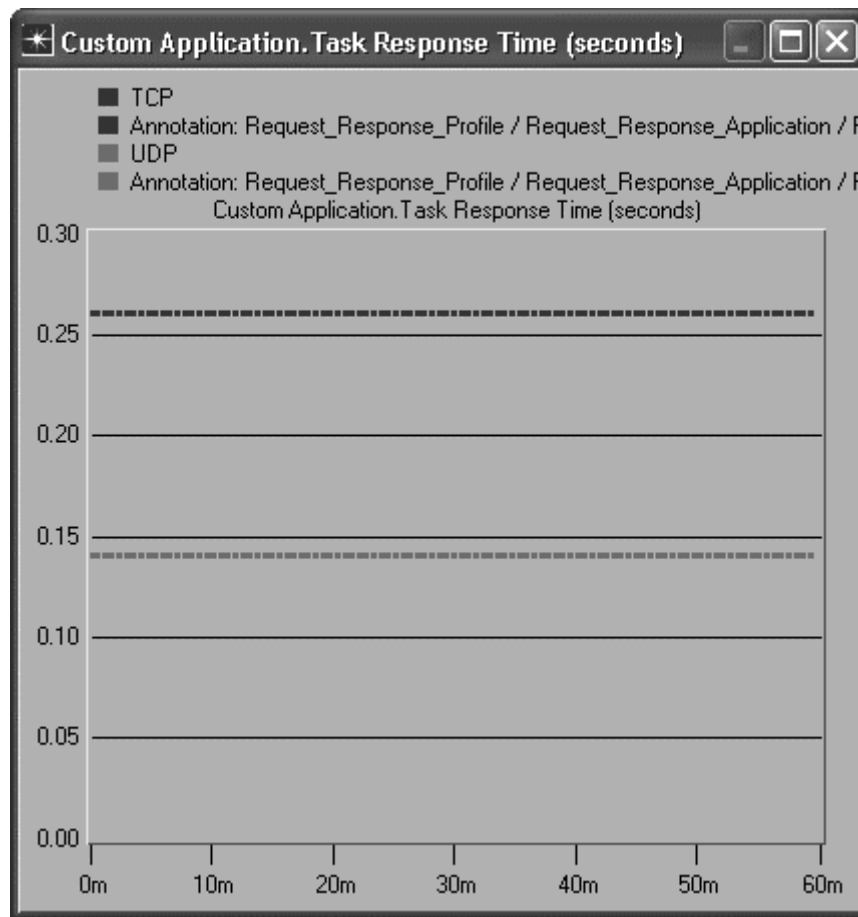


When the simulation has completed, click on **Close** to close the window.

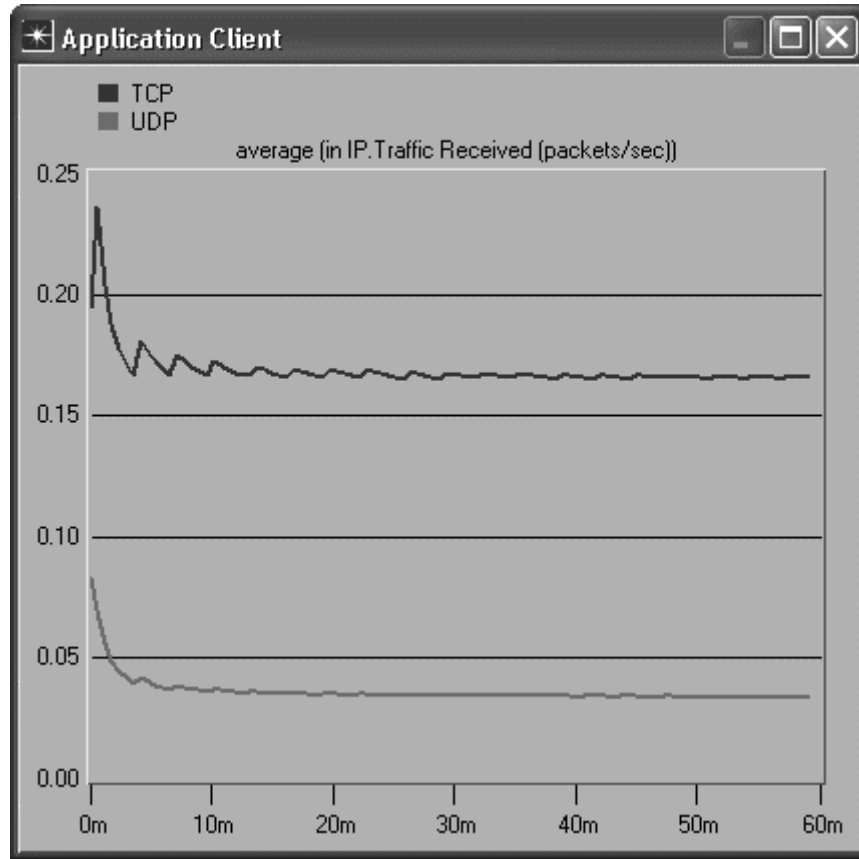
Inspect and Analyze Results

Select the **Results** tab => **Compare Results...**

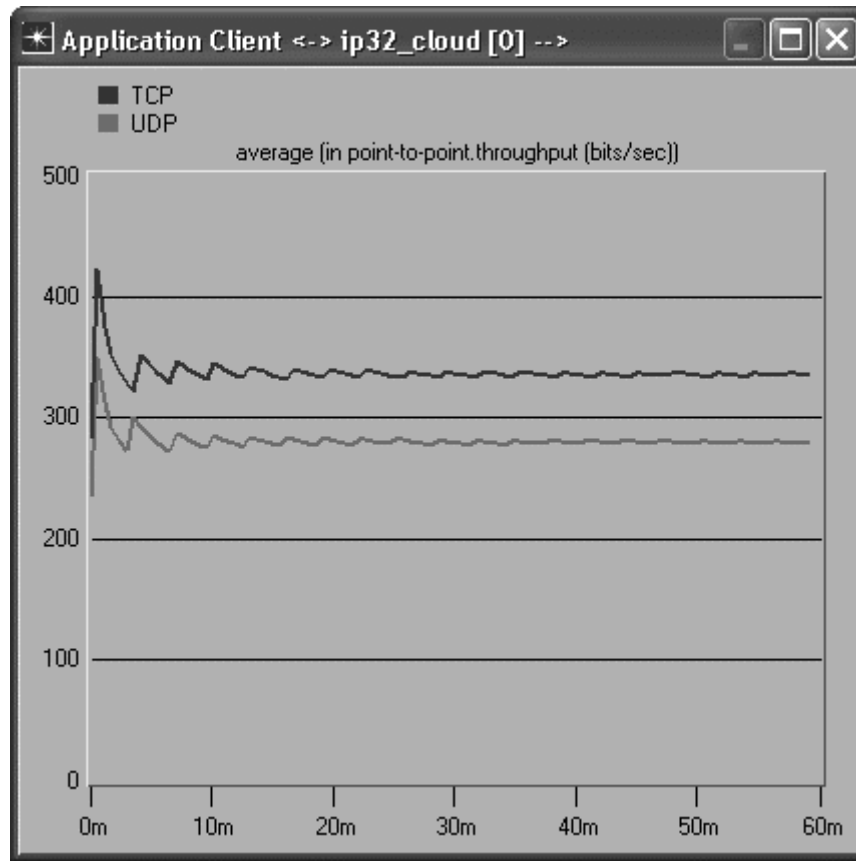
Select and expand the **Global Statistics** item and the **Custom Application** item, and select the **Task Response Time (seconds)** statistic. This statistic shows how long each task took to complete. Use **As Is** mode to view the statistic. Click on **Show** to see a more detailed graph. There will be one point for each time the application was run and one set of points for TCP and another for UDP. You can see that the application took nearly twice as long to complete each task when using TCP as it did when using UDP. This is likely due to the lack of connection setup and teardown delay in UDP. Click on the close window icon and **Delete** the panel. Click on the statistic again to disable the preview.



Select and expand the **Object Statistics** item, the **Application Client** item, and the **IP** item. Select the **Traffic Received (packets/sec)** statistic and use **average** mode to view the statistic. Click on **Show**. You can see the client received a larger number of packets per second in the TCP case than the UDP case, nearly five times as many. This is due to the need to set up connections and tear them down again in the TCP case. Click on the close window icon and **Delete** the panel. Click on the statistic again to disable the preview.



Expand the **Application Client** <-> **ip32_cloud[0]** item, the **point-to-point** item, and select the **throughput (bits/sec)** --> statistic. Use **average** mode to view the statistic. Click on **Show**. This statistic shows the amount of traffic that was seen on the link between the client and the WAN. Even though the same number of requests and responses were executed in both scenarios, the bandwidth usage is less in the UDP case. This is again due to the fact that UDP does not need to send connection setup and teardown packets. Click on the close window icon and **Delete** the panel. Click on **Close** to close the **View Results** window.



Save your model and close all windows.

Questions

1. Based on the configuration parameters, what would the throughput on the links be if there were no control packets (such as connection setup and teardown packets)? You can use the values you set in the Task Configuration and Profile Configuration objects to calculate an answer. Look for the size of the request, the size of the response, and the delay between requests.
2. Based on your answer to Question 1, and the results from your simulation, how much overhead did UDP impose? In other words, what percentage of the bytes sent on the links was used for packet headers and control packets? How much overhead did TCP impose?
3. Duplicate your scenarios and modify the **Dest --> Source Traffic** in the **Task Specification** for both UDP and TCP scenarios so that the **Packets per Response** is **constant(1000)** instead. This configuration corresponds to an application which sends a small request and then receives a large file in response. Calculate the size of the response. Now rerun the simulation for both scenarios. How much slower is the **Task Response** time when using TCP than when using UDP? How large must the responses be before the TCP response time is less than 10% longer than the UDP response time?
4. Switch to the small packet request-response scenario. Duplicate the scenario and edit the attributes of the **ip32_cloud** to modify the **Packet Discard Ratio** to **1%**. Rerun the simulation. How does error affect the response time for UDP and TCP? Explain your results.
5. Investigate the operation of TCP. Explain how connection setup is accomplished. Explain how connection teardown is accomplished. Draw diagrams to illustrate your explanations.