Practical Worksheet 5   OPNET   Frame Relay Network Performance

Overview

Frame Relay is a connection-oriented, unreliable technology based on virtual circuits. A virtual circuit must be set up between a source and destination before any data can be exchanged. Most Frame Relay service providers support Permanent Virtual Circuits (PVC), which are long-lived connections, as opposed to Switched Virtual Circuits (SVC), which are set up at call time. When a PVC is set up, parameters which govern the behaviour of the PVC are agreed upon. These include the Committed Information Rate (CIR), committed burst rate (Bc) and the excess burst rate (Be). The CIR is the rate (in bits per second) which the service provider guarantees to the customer. The committed burst rate specifies how quickly that data may be sent. For instance, a customer may contract for a CIR of 64 Kbps and a Bc of 32 Kbits/100 ms. In this case, the customer may send 64 Kbits per second, but no more than 32 Kbits in any 100 ms period. Frame Relay service providers will accept additional data above the CIR and transmit it if there is excess capacity in the network at that time. The amount of excess data that they will carry is defined by the excess burst rate. Any data frames that are sent in excess of the CIR are marked with the Discard Eligible bit set. These frames are discarded first in the event of congestion.

Objective

To examine the behaviour of a Frame Relay PVC as contract parameters are varied.

Build the Simulation Model

Start up OPNET IT Guru Academic Edition.

First, it would be useful to have a look at this Web page for information on Frame Relay: http://www.protocols.com/pbook/frame.htm. Next, read the information provided by OPNET regarding Frame Relay configuration.
Select the **File** tab => **Open**. Choose the **Frame_Relay** project
(C:\PROGRA~1\OPNET\11.0a\models\std → example_networks → Frame_Relay)
and click on **OK**. Read through the descriptions of Required Node Models and Configuration (double-click on the icon to see the description).
Select **File** => **Close** (Don't save) when you have finished.

Select the **File** tab => **New...**
Choose Project and click on OK.
Change the Project Name to **xx_WAN_Frame_Relay** (**xx** = your initials). Set the **Scenario Name** to **CIR64** and click on **OK**.
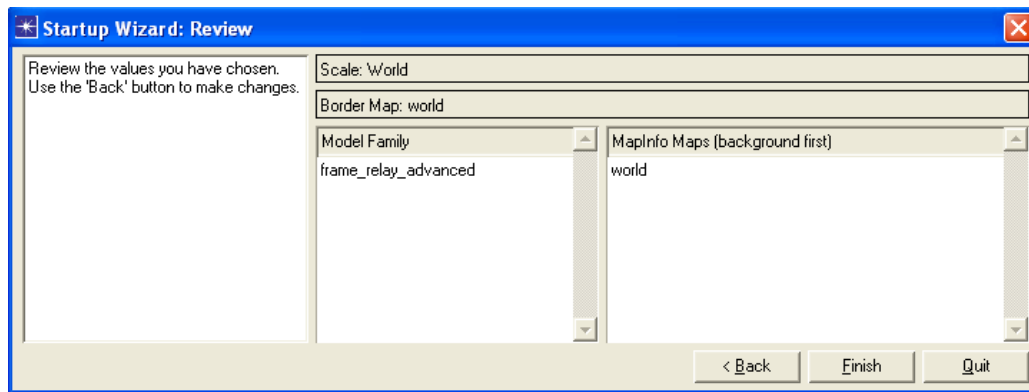In the **Initial Topology** window, select **Create Empty Scenario** and click on **Next**.
In the **Choose Network Scale** window, select **World** and click on **Next**.
In the **Choose Map** window, choose **usa** and click on **Next**.
In the **Select Technologies** window, scroll down and include the frame_relay_advanced model family and then click on Next.
In the **Review** window, click on **Finish**.

First, we will configure a bursty data application to generate traffic over our frame relay network. The File Transfer Protocol (FTP) will serve our purposes.
Select an **Application 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 **Bursty Application**. Expand the **Application Definitions** attribute and set the **rows** attribute to **1**. Expand the **row 0** attribute and set the **Name** attribute to **FTP_Bursty**. Expand the **Description** attribute and edit the value of the **Ftp** attribute. Set the **Inter-Request Time (secs)** attribute to **exponential(0.1)**. Set the **File Size (bytes)** attribute to **constant(1000)**. The application you have now defined will transfer an average of ten 1-KB files every second. The exponential distribution is often used to model waiting times so we can use it here to model the time between file requests. Click on **OK** to close the window.
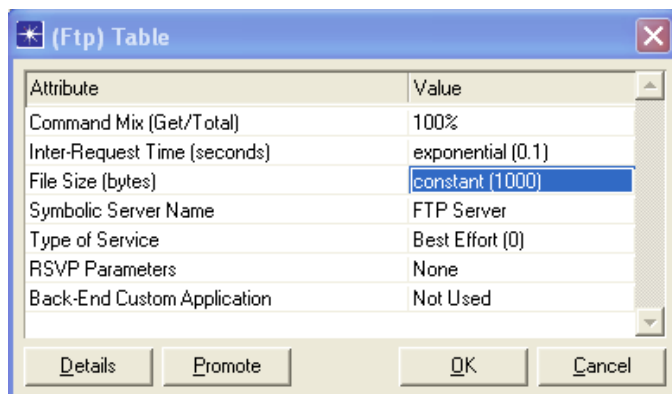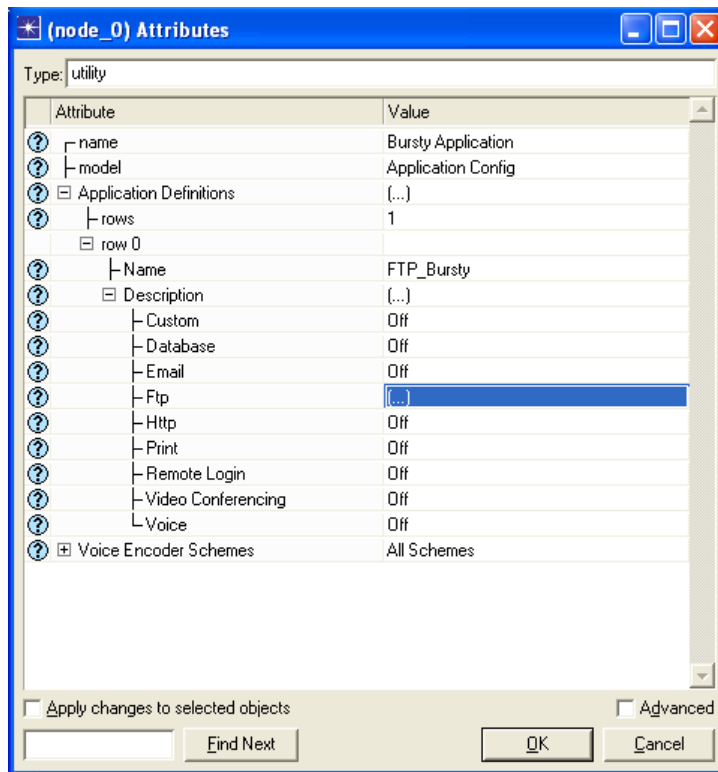
Select a **Profile Config** object from the Object Palette and place it in the project workspace. Right click on the object and choose **Edit Attributes**. Set the **name** attribute to **Bursty Profile**. Expand the **Profile Configuration** attribute and set the **rows** attribute to **1**. Expand the **row 0** attribute and set the **Profile Name** to **FTP_Bursty_Profile**. Expand the **Applications** attribute and set the **rows** attribute to **1**. Expand the **row 0** attribute and set the **Name** to **FTP_Bursty**. Set the **Repeatability** attribute to **Once at Start Time**. Click on **OK** to close the window.
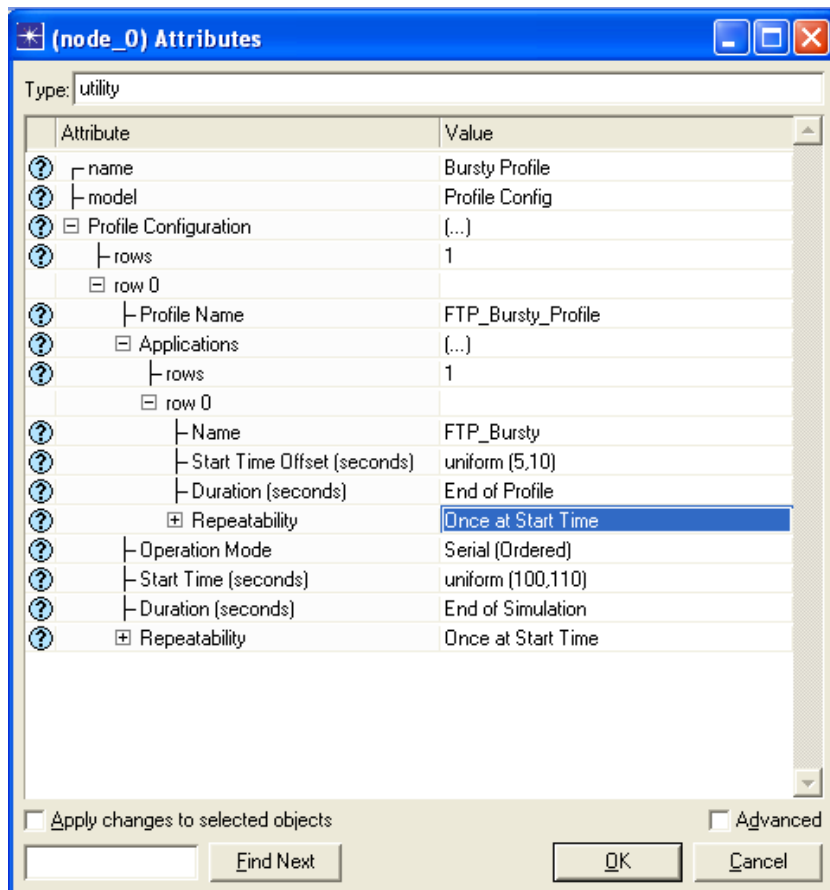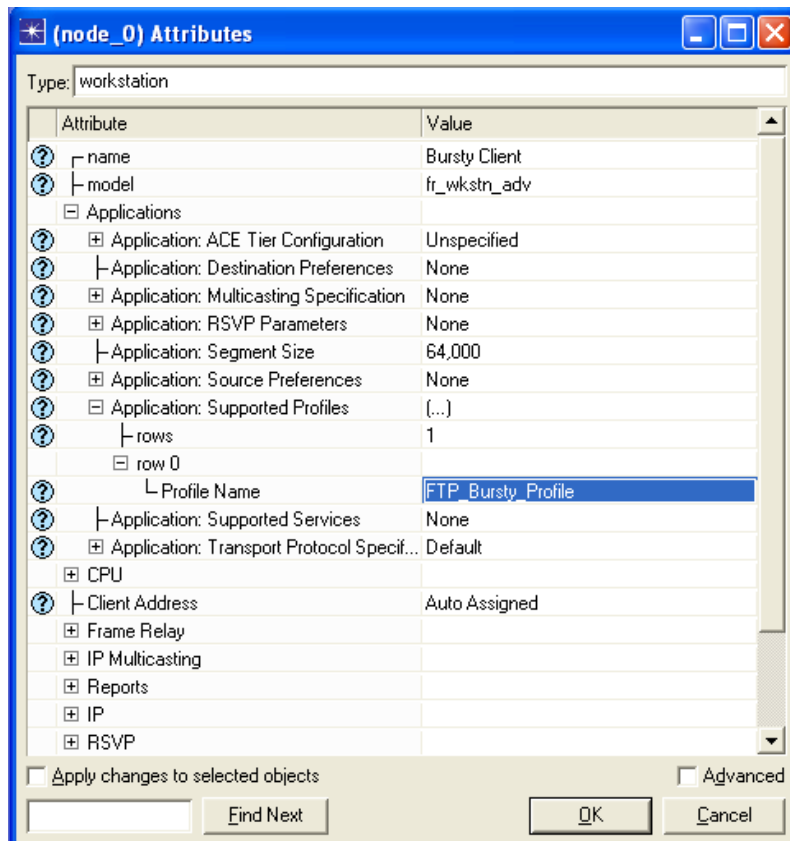
Select an **fr32 cloud** object from the Object Palette and place it in the project workspace. Right click on the cloud and choose **View Node Description**. The cloud represents a WAN that consists of Frame-Relay-capable switches and that supports up to 32 Frame Relay connections.
Right click on the cloud and select **Set Name**. Set the name to **Frame_Relay_Cloud**. Click on **OK** to close the window.

Select an **fr_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 Frame Relay protocol. We will now set up the client to use the bursty FTP application that we defined.
Right click on the station and choose **Edit Attributes**. Modify the **name** attribute of the device to **Bursty Client**. Edit the **Application: Supported Profiles** attribute.
Insert a row and set the **Profile Name** to **FTP_Bursty_Profile**. Click on **OK** to close the window.

Select an **fr_server_adv** 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 also supports the Frame Relay protocol. We will now set up the server to support the Bursty FTP application that we defined.

Right click on the device and choose **Edit Attributes**. Modify the **name** attribute of the server to Bursty Server. Edit the **Application: Supported Services** attribute. Insert a row and set the **Name** to **FTP_Bursty**. Click on **OK** to close the table window. Click on **OK** to close the window.
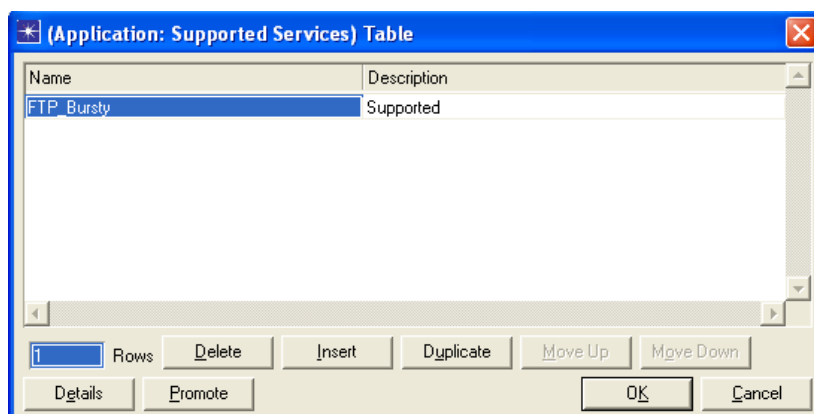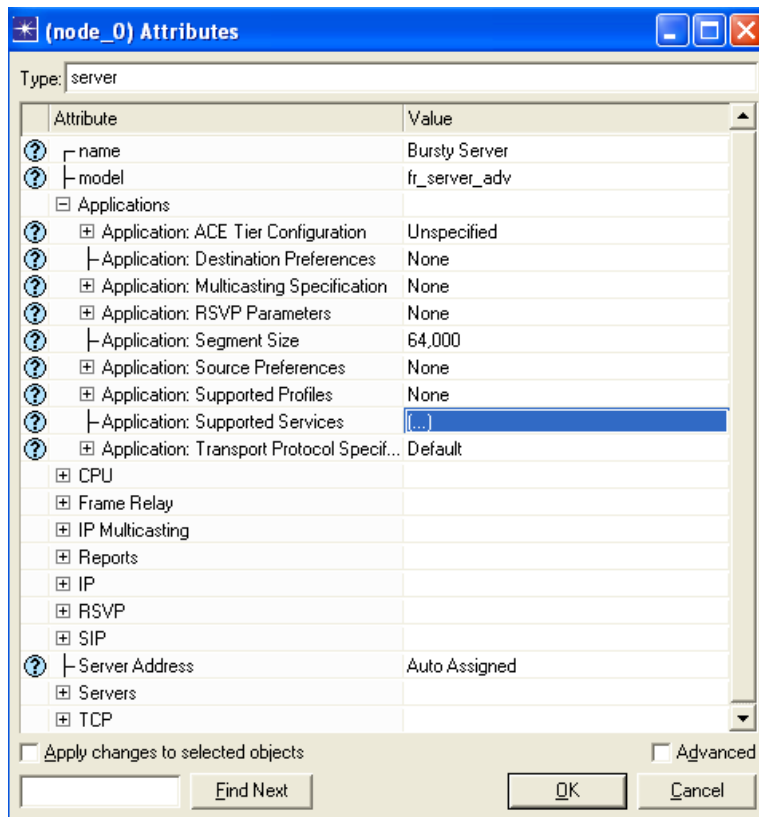
Select two **fr8_switch_adv** objects from the Object Palette. Place one next to the client and one next to the server. Right click on one of the switches and choose **View Node Description**. Note that the switch takes action when it becomes congested. It will discard the frames marked with the Discard Eligible Bit first, but may even discard normal frames. Click on the close window icon to close the window.
Right click on the client's switch and select **Set Name**. Set the name to **Client Switch**. Click on **OK** to close the window.
Right click on the server's switch and select **Set Name**. Set the name to **Server Switch**. Click on OK to close the window.

Select four **FR_T1_int** links from the Object Palette and use them to connect the client to its switch, the server to its switch and the two switches to the Frame Relay cloud. T1 speed is 1.5 Mbps.
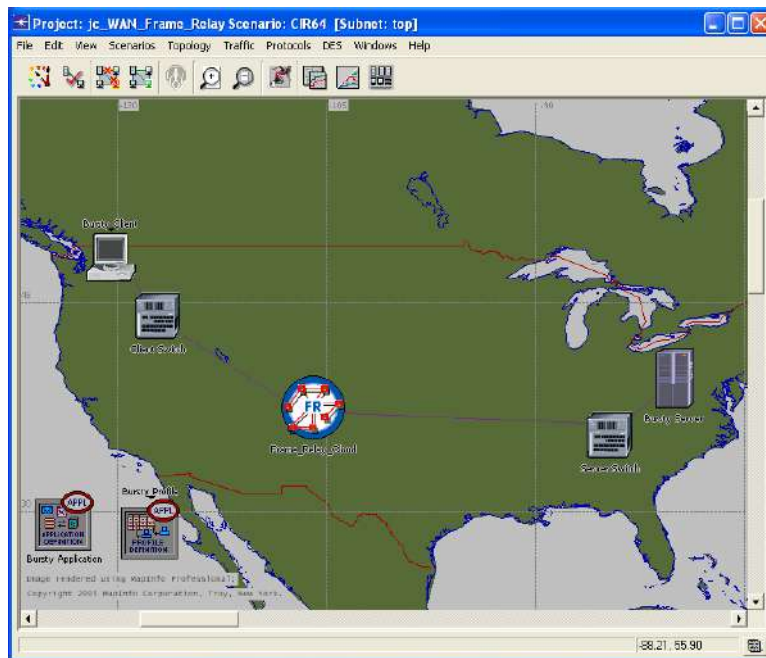
Finally, we can set up a permanent virtual circuit (PVC) between the client and the
server so that they can communicate. Select an **FR PVC Config** object from the
Object Palette and place it in the project workspace. Right click on it and select **Set
Name**. Set the **Name** to **PVC Config**. The **FR PVC Config** object must be placed in
the workspace before any PVCs may be set up. Click on **OK** to close the window.

Select an **fr_pvc** object from the Object Palette and use it to connect the client to the
server. When you right click to finish placing the PVC, choose **Abort Demand
Definition**.

Right click on the PVC. Choose **Edit Attributes** and expand the **Contract
Parameters** attribute. These parameters constitute the agreement between the end
systems (client and server) and the Frame Relay service provider. Set the **Outgoing
CIR (bits/sec)** attribute to **64000**. Set the **Outgoing Bc (bits)** to **64000**. Set the
Outgoing Be (bits) to **32000**. The Committed Information Rate (CIR) is the amount
of traffic that the service provider has agreed to carry. The committed burst size (Bc)
indicates how much data can be sent in a given interval. In this case, the interval is
one second, so the CIR = Bc. The excess burst size (Be) is the amount of traffic that
the service provider may accept in addition to the committed rate, if network
conditions allow. Excess data may be discarded if the service provider's network
becomes congested. Click on **OK** to close the window.
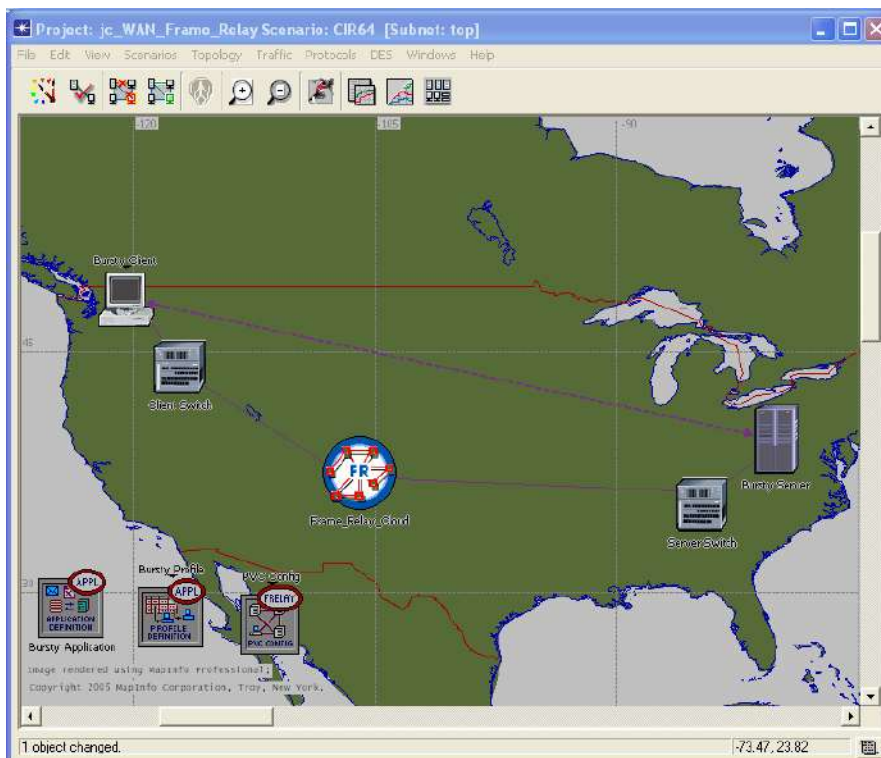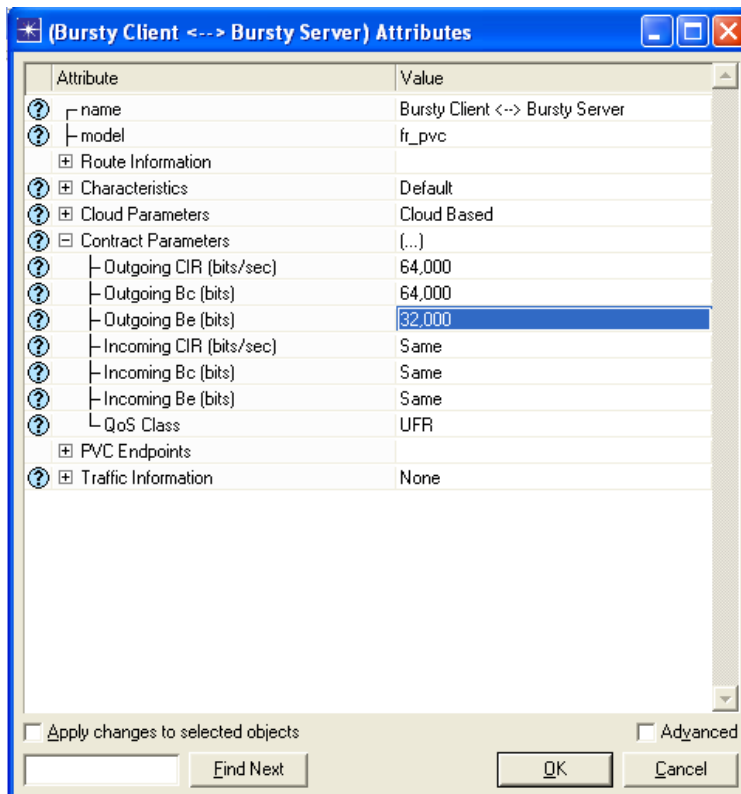
Configure and Run the Simulation

Select the **DES** tab => **Choose Individual Statistics...**
Expand the **Global Statistics** item and the **Frame Relay** item and select the **Delay (sec)**, **Delay Variance** and **Residual Error Rate** statistics.

Expand the **FTP** item and select the **Download Response Time (sec)** and **Traffic Received (bytes/sec)** statistics.
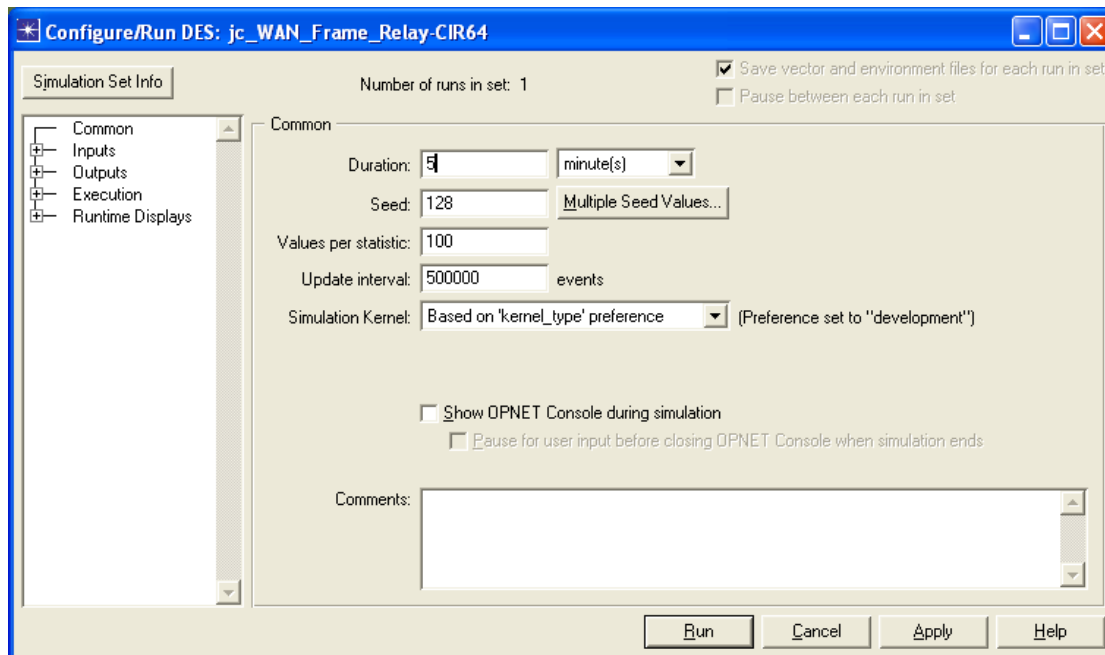
Expand the **Node Statistics** item and the **Frame Relay PVC** item and select the **BECN Status**, **DE Status** and **FECN Status** statistics.

Click on **OK** to close the window.
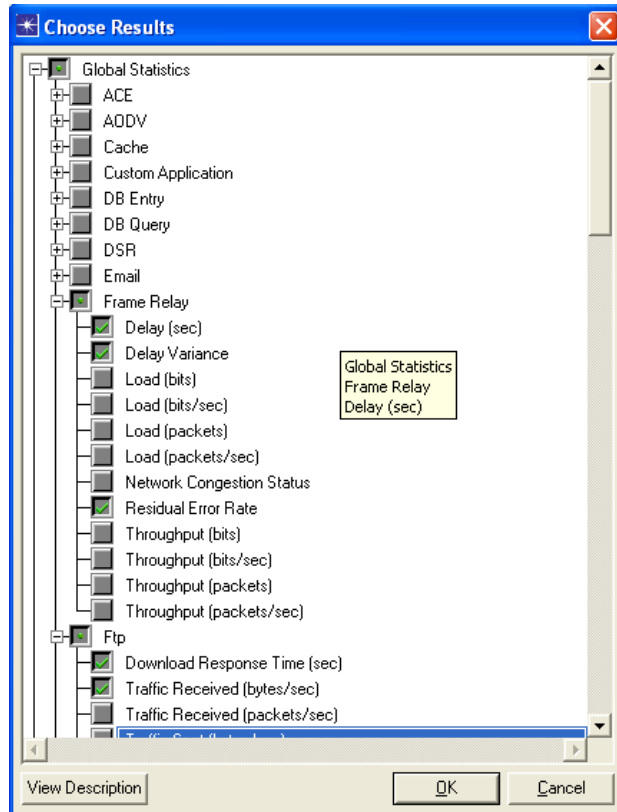
Select **DES => Configure/Run Discrete Event Simulation...**

Under **Common**, set the **Duration** to **5** and the unit to **minute(s)**. Click on **Apply** and then close the window.

Duplicate the Scenario

We are now going to duplicate the scenario and configure the PVC to support a higher data rate. This will allow us to compare the performance across the two PVC rates. Choose **Scenarios** => **Duplicate Scenario** and name the new scenario **CIR128**.

Right click on the PVC and expand the **Contract Parameters** attribute. Set the **Outgoing CIR (bits/sec)** attribute to **128000**. Set the **Outgoing Bc (bits)** to **128000**. Leave the **Outgoing Be (bits)** unchanged at **32000**. Click on **OK** to close the window.
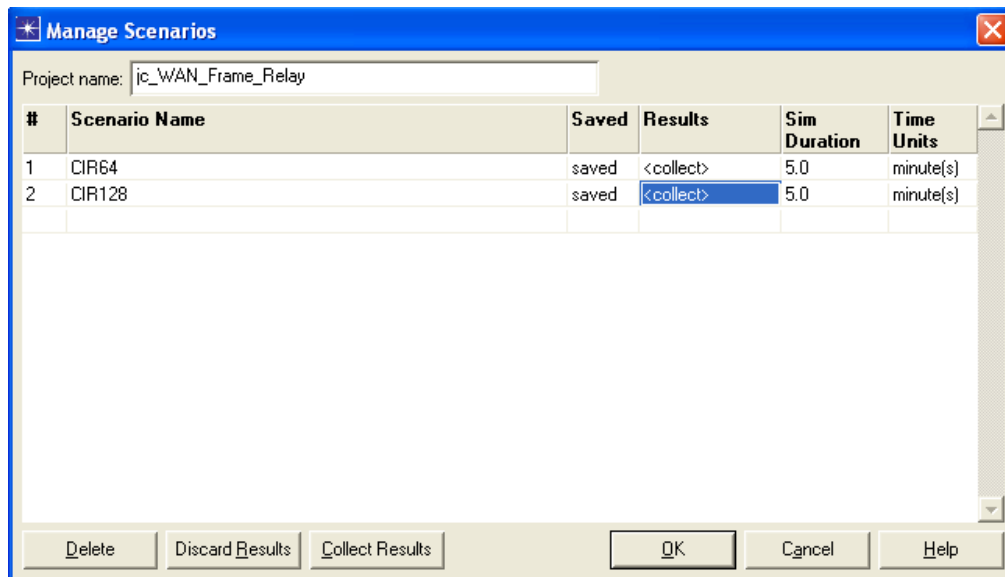
Run the Simulation

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

Inspect and Analyse Results

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

Select and expand the **Global Statistics** item and the **Ftp** item and select the **Download Response Time (sec)** statistic. Use **As Is** mode to view this statistic. Click on **Show** to see a more detailed graph. The statistic shows how long each download took and there will be one point for each file downloaded. You can see that the downloads took much less time in the CIR128 case and that the variation in download times was much less. This is to be expected since there was more capacity available to the application in the second scenario. Click on the close window icon and choose to Delete the panel. Click on the statistic again to disable the preview.
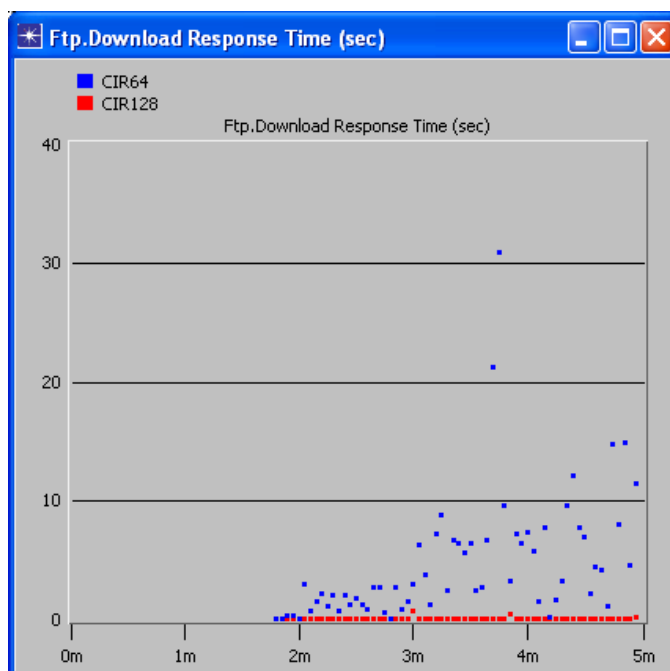
Expand the **Traffic Received (bytes/sec)** statistic and view using **As Is** mode. Click on **Show** to see a more detailed graph. This statistic shows the total amount of FTP traffic received by both the server and client. You can see that more traffic is received in the CIR 128 case and that there is more variation than in the CIR64 case. Click on the close window icon and choose to **Delete** the panel. Click on the statistic again to disable the preview.
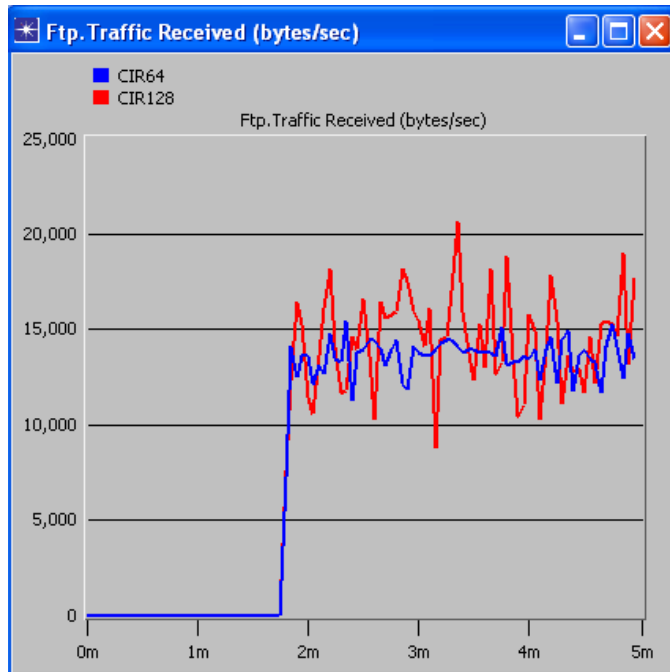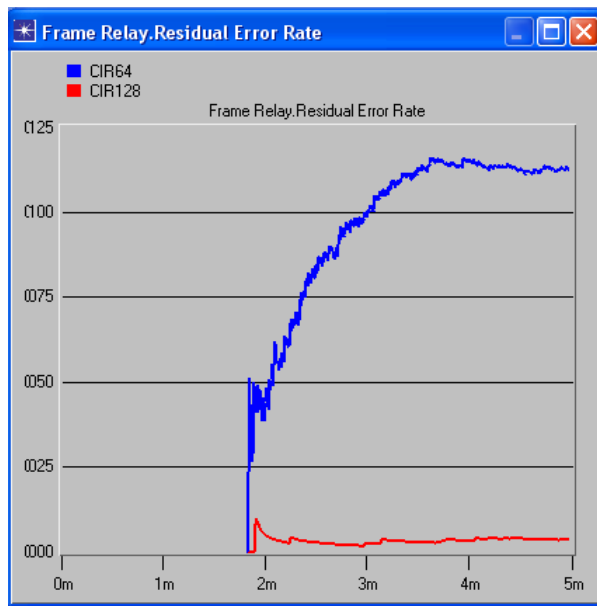

Select and expand the **Global Statistics** item and the **Frame Relay** item and select the **Delay (sec)** statistic. Use **time _average** mode to view this statistic as this mode will make it easier to see the difference in performance. Click on the **Show** button to see more detail. The statistic shows how long each frame took to be delivered. You can see that the delay is greater in the CIR64 case and that the delay is growing over time. This is due to longer delays within the switches due to full buffers. Click on the close window icon and **Delete** the panel. Click on the statistic again to disable the preview.
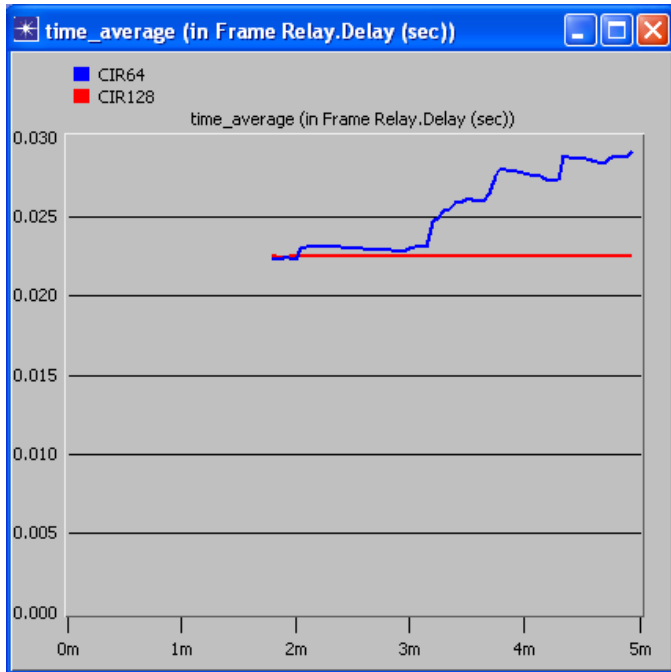
Expand the **Delay Variance (sec)** statistic and view using **As Is** mode. Click on **Show**. This statistic shows the variance in frame delivery delay. The CIR64 case shows much more variability and the variability is growing over time. Click on the close window icon and **Delete** the panel. Click on the statistic again to disable the preview.
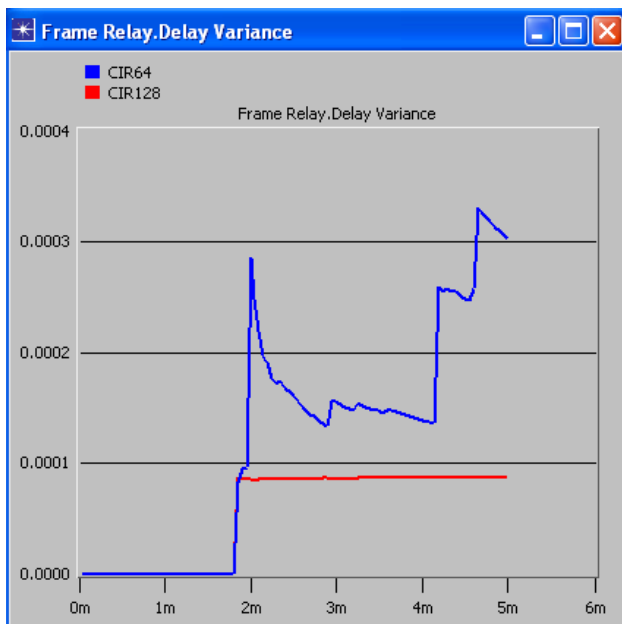
Expand the **Residual Error Rate** statistic and view using **As Is** mode. Click on **Show**. This statistic shows the percentage of frames which are discarded due to congestion or corruption. You can see that many more frames are discarded in the CIR64 case and that the percentage is growing as the switch becomes more and more congested. 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 **Bursty Client** item and the **Frame Relay PVC** item. Select the **DE Status** statistic and view using **As Is** mode. Click on **Show**. This statistic shows the number of frames that were received by the client that had the DE bit set. When many frames are received with the DE bit set, it indicates that the source of the frames was frequently sending faster than the contracted CIR. You can see that far more frames are marked with the DE bit in the CIR64 case. This makes sense since the application produces the same amount of traffic in both scenarios, but the CIR 128 scenario provides more capacity for carrying the traffic. Click on the close window icon and **Delete** the panel. Click on the statistic again to disable the preview.
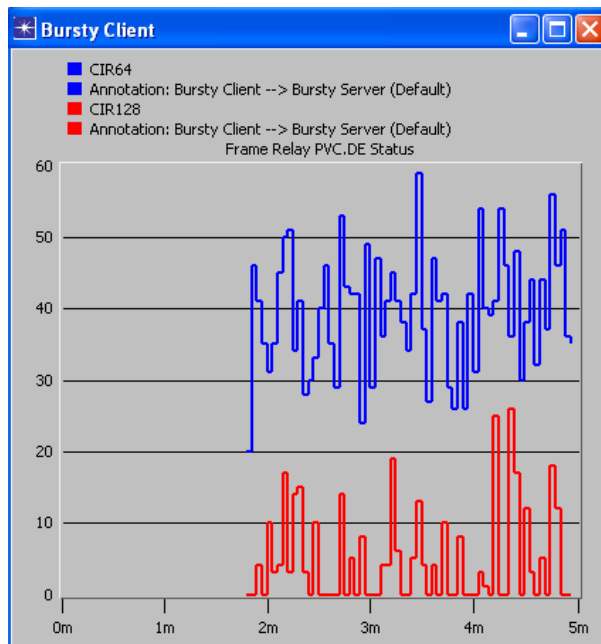
The **BECN Status** or **FECN Status** statistics show how many frames were received with the BECN or FECN bits set. These bits are set by the switch when it is

congested. If you view these statistics, you will see that no frames were received with the bits set. The offered traffic is not sufficient yet to cause congestion at the switches.

Save your model and close all windows.

Questions

1. How much traffic is being generated by the Bursty Server? You can calculate this data rate using the **Inter-Request Time** and **File Size** parameters you set when configuring the bursty application.

2. Given the traffic level calculated in Question 1, explain how this traffic can be carried given the speed of the links used to connect the devices and the contract parameters of the PVC connecting the client and server.

3. Duplicate the CIR64 scenario and replace the TI link between the client and its switch with an **FR_DSO_int** link. Remember that DSO speed is 64 Kbps. Rerun the simulation and inspect the **BECN Status** and **FECN Status** statistics for both the **Bursty Client** and **Bursty Server**. How do these compare to the CIR64 scenario results? Explain your results. Explain the usage of the BECN and FECN bits in the Frame Relay header. Also inspect the Residual Error Rate statistic. How does it compare to the CIR64 case?

4. Frame Relay service providers often offer Frame Relay PVC contracts called *zero CIR*. In these cases, *all* packets are marked discard eligible. Zero CIR service is usually less expensive, but provides fewer guarantees.

Duplicate the CIR64 scenario and edit the PVC attributes. Set all the **Contract Parameters** to zero. Rerun the simulation and compare your results to the CIR64 results. What differences did you find? What would be the effect on the Traffic Received at the Bursty Client if you added another client with a CIR of 64 Kbps to the zero CIR case? Explain your answer.

5. In Question 3, some frames are discarded due to congestion at the switch. The congestion can be relieved if the traffic source transmits frames less frequently. Frame Relay systems provide traffic shaping algorithms to restrain the traffic source from sending too quickly. Duplicate the DS0 scenario from Question 3 and edit the Bursty Server attributes. Expand the **Frame Relay Traffic Shaping** attribute and set the **Status** to **Enabled**. Click on the question mark to the left of each of the traffic shaping attributes to find out what they are for. Calculate values for these attributes that will result in an uncongested network; in other words, restrict the data rate to less than the speed of the DSO line (64 Kbps). Write down the function of each attribute, the value you calculated and an explanation of the calculations. Run the simulation and gather statistics to show that you achieved the goal of reducing congestion (and corresponding frame discards).