CHAPTER 2 Applications

Applications are the predominant sources of traffic in the network. It is the traffic generated by applications that loads the network, makes demands on the bandwidth and the underlying network technology, and creates load on the servers. For the optimum performance of an application, you must ensure that the network and server infrastructure are designed to meet the requirements of the application. Likewise, it is necessary to design applications so that they minimize their impact on network and server resources. To do this, it is necessary to first create an accurate model of the application.

To be an accurate representation of the application, an application model should have the same traffic characteristics in terms of the size of the packets generated, the rate at which they are generated, the transport protocol over which it runs (e.g., TCP, UDP, fiber channel, etc.), the number of simultaneous connections, timeouts, retransmissions, failure and recovery, and so on. Together these characteristics create a run time traffic pattern of the application.

Each application has its unique traffic pattern, or signature, and therefore creates its characteristic load on the network and servers. Thus, the concept of application modeling is to capture the traffic pattern of the applications of interest.

This section describes the following topics:

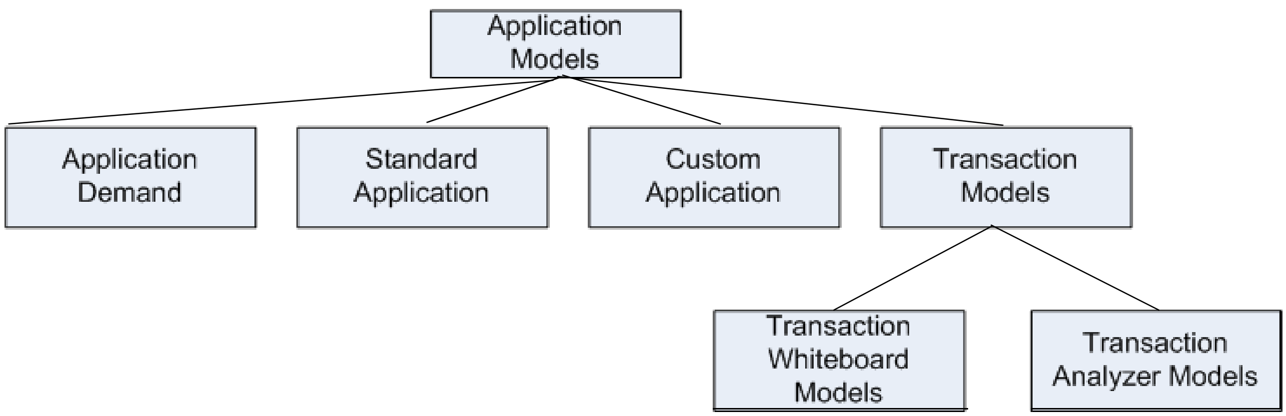
Note: Flow Analysis is not supported for analyzing applications.

Modeling Applications

All application modeling requirements are not equal in terms of the accuracy that they demand from the model. They can vary across the spectrum from a very basic need to generate a certain number of packets-per-second application load to actually reproducing the detailed message exchanges, including some of the actual application's logic embedded into the application model.

For this reason, Riverbed Modeler provides several application configuration models and modeling frameworks, each targeted to speeding up model development for specific modeling requirements. These solutions are summarized below, and it is important to understand the advantages, limitations, and use cases, from a high level, before selecting one solution. Even though each solution is targeted for certain use cases, there is overlap among them, and it is possible that more than one solution can meet your modeling requirements.

Figure 17 Application Modeling Solutions



Modeling Application Demands

Application demands require minimum configuration and are the fastest way to introduce application layer traffic into the simulation. Application demands represent generic client-server applications, with the client sending request packets to the server and the responses returning. For each request, there is exactly one response, unless you disable responses. There are attributes on the application demand that allow you to control the size of the request and response packets, the rate at which the requests are generated, transport protocol, and type of service (ToS).

Note: Flow Analysis is not supported for application models.

Advantages

Application demands let you introduce client-server application traffic into the network quickly. Traffic representation can also be toggled between purely discrete, purely background, or a fractional mix of the two.

Limitations

Application demands do not represent traffic patterns generated by specific applications (FTP, Email, etc.) and are restricted to two-tier (client-server) applications. Application demands are also not integrated with the Advanced Server Models.

Use Cases

Application demands are useful when your goal is to create generic application traffic quickly, while the actual focus of your study is some other component or protocol in the network.

For example, you may be attempting to study how a certain routing protocol or service queue behaves under the impact of different traffic loads. In this case, application demands let you create the load required to reach your goal. Application demands are also beneficial for studying end-to-end connectivity in the network, to perform reachability analysis, and to troubleshoot routing issues.

For more information, see Application Demands on page MC-14-736.

Using Standard Application Models

Standard application models include preconfigured models of the following commonly-used network applications:

Table 2 Standard Application Models

Available Application	Description
FTP in Application Modeling	File transfer
Email in Application Modeling	Sending and receiving email
Peer-to-Peer File Sharing in Application Modeling	P2P file sharing
Remote Login in Application Modeling	Rlogin (telnet)
Database in Application Modeling	Database queries and updates
HTTP in Application Modeling	Web browsing
Mobile User Applications in Application Modeling	User applications for smartphones and other mobile devices
Print in Application Modeling	Print job submission
Voice in Application Modeling	On-Off voice model
Video Conferencing in Application Modeling	Video conferencing involving image exchanges

These models are shipped as part of the standard model suite. Each of these models has attributes that relate to a specific standard application and generate an appropriate traffic pattern.

Note: Standard application models have default values that represent typical application behavior. To customize the behavior to represent your own network, you must modify the values, as described in Standard Applications Supported by Application Models.

For example, the FTP model has attributes for file size, upload/download ratio, and so on, while the Voice model has attributes for encoder scheme, talk/silence duration, and other voice-related attributes. Similarly, from a behavior perspective, the FTP model uploads/downloads files directly over the user-configurable transport protocol (TCP/UDP, for example), while Voice runs on RTP streams and can use SIP signalling.

Advantages

Standard application models enable you to create representative models for commonly-used network applications quickly. The attributes and statistics for these models are targeted for the specific application they represent and are simple to understand and configure.

Limitations

Standard application models represent common network applications in a generic way and do not represent any specific vendor's implementation of the application. In actual networks, each application type has specific vendor implementations that create different traffic patterns.

Standard application models represent only two-tier (client-server or peer-to-peer) applications and cannot model multi-tier applications such as WebService or enterprise resource planning (ERP). Standard application models are not integrated with the Advanced Server Models.

Use Cases

Standard application models are useful for studying traffic generated by a typical user or to model an office environment in which most traffic is due to the usage of standard network applications such as FTP, Email, HTTP, and so on. These models provide the necessary traffic generation patterns for performing capacity planning and response time studies.

Using the Custom Application Model

The custom application model is more of an application modeling framework than a true model. It lets you define your own, custom traffic pattern for your study instead of using a more generic model. The custom application model breaks down the application into smaller components known as *tasks* and *phases* and lets you configure each detail of how and when the application sends requests and responses, how it sets up and reuses connections, and how much time is spent in server processing, for example.

Advantages

The custom application model has no predefined behavior and follows only the specifications you configure. It is a flexible model that lets you configure it to behave in a variety of ways.

The model supports multi-tier applications, so there is no limit to the number of tiers over which you can configure an application to run. Custom application model is integrated with the Advanced Server Models.

Limitations

The custom application model provides a high level of configurability, which means that you must specifically configure each attribute to represent the behavior you want to model. Though it is possible to capture some basic application decision-making logic in a custom application model with attribute configuration, full support for programmable logic is not supported.

Use Cases

The custom application model is useful for modeling multi-tier, complex applications in which the architecture, in terms of its transactions, is known and in which there is limited logic that can alter the application traffic pattern. The model is also useful for modeling application-layer signaling protocols, which usually have well-defined behavior.

Transaction Analyzer Models

SteelCentral Transaction Analyzer can generate a model of an application by analyzing a live network packet trace capture from the application. Based on the capture files, this module makes conjectures as to the actual message exchange pattern of the application, complete with message sizes, timing sequence, and dependencies. You can import Transaction Analyzer models into the discrete event simulation environment and play back the models to create the same traffic pattern as you captured in the live environment.

Note: This section pertains only to the DES features of Transaction Analyzer. Transaction Analyzer has other important features that are outside the scope of this document.

Advantages

A Transaction Analyzer model is as close as you can get to creating an exact model of your existing application. Because Transaction Analyzer makes no assumptions about the application behavior and plays back the messages exactly as it capture them in the live environment, the resulting traffic pattern very closely replicates the actual application behavior.

Transaction Analyzer models require minimal configuration, because, unlike the custom application model which requires you to configure the application behavior, a Transaction Analyzer model already contains the details of the application behavior. This reduces the possibility of user errors in configuration. Transaction Analyzer is also integrated with the Advanced Server Models.

Limitations

The starting point for Transaction Analyzer is an application packet trace capture from a live network; therefore, Transaction Analyzer can only be used for applications that are implemented and operational in the live network. Transaction Analyzer cannot be used for applications that are in the design phase or are not fully implemented.

Typical applications generate slightly different traffic each time a transaction is executed. For example, a database query may generate different traffic patterns, depending on the search criteria. A given trace represents one instance of the transaction and the traffic pattern associated with that instance and cannot encompass every possible variation.

You can measure throughput in a Transaction Analyzer file using the AppDoctor statistic “Network Throughput (Kbits/sec)”. You can also generate throughput statistics from a discrete event simulation. However, AppDoctor and a discrete simulation might produce different throughput results—even with the same network conditions and assumptions—because they interpret the size of ethernet packets differently.

Transaction Analyzer calculates the packet size as follows:

<*size_of_ethernet_packet*> = <*header_size*> + <*data_size*>

A discrete event simulation calculates the packet size as follows:

<*size_of_ethernet_packet*> = <*preamble_size*>
+ <*header_size*> + <*CRC_size*> + <*data_size*>

You can deploy one or more traces in a scenario (in the Project Editor, choose Protocols > Applications > Import from AppTransaction Xpert > as Discrete Traffic...). This operation creates a custom application that includes one task for each packet trace. These tasks will not reuse the same TCP connection during a simulation; each task initiates its own, separate TCP connection. If you want to simulate an application that uses the same TCP connection, you must capture the entire transaction in one capture operation. This will ensure that the entire transaction is modeled in one Transaction Analyzer model file (rather than in multiple files).

Use Cases

Transaction Analyzer is useful for creating an accurate representation of the traffic pattern of your existing application. It eliminates guesswork in reproducing traffic patterns, since it can deduce those patterns from the packet capture from your live network.

Transaction Whiteboard

Transaction Whiteboard (part of Transaction Analyzer) provides another way to generate an application model that can be used in simulations. With it, you can draw the traffic pattern on a network as a series of interdependent messages, then run a simulation using the resulting application model.

Note: You must have a license for Transaction Analyzer to edit Transaction Whiteboard files.

Transaction Whiteboard is a fully developed message editor, so you can

- Specify the message sizes
- Specify the processing time on the tier when a message is received
- Create connections on which the messages are sent
- Create message blocks, and so on

Note: This section pertains only to the DES features of Transaction Whiteboard. Transaction Whiteboard has several other important features that are outside the scope of this document.

One of the most powerful features of Transaction Whiteboard is that it provides a Python interface for embedding logic scripts in the model. The behavior of the model can be controlled dynamically at run time by introducing Python scripts at strategic places in the Transaction Whiteboard model.

Advantages

Transaction Whiteboard can be used to model applications that are in the design phase or have not yet been fully implemented. The full Python interpreter and the standard Python libraries are available for writing arbitrarily complex scripts. Since Transaction Whiteboard can also import Transaction Analyzer packet traces and render them editable, you can redesign and modify existing applications.

Limitations

Because Python is an interpreted language, executing the Python scripts in an Transaction Whiteboard model can potentially slow down a simulation. However, for most models, this might be negligible, because the time spent executing the Python scripts is relatively small compared to the overall simulation time. This limitation does not apply to models that do not contain any Python scripts.

Use Cases

Transaction Whiteboard is an application design environment for modeling and studying different application designs or design modifications, including application logic and module hierarchies, before implementing them.

Representing Application Traffic

There are two ways to represent traffic:

- Discrete Traffic (also called "explicit traffic")
- Background Traffic (also called "analytical traffic")

This section gives a high-level explanation of these traffic types as they relate to application modeling.

Note: In addition to using purely discrete or purely background traffic, you can use a mixture of the two (hybrid). By using hybrid, you can model a percentage of the traffic as individual packets, while the remainder of the traffic is modeled as background traffic.

Discrete Traffic

When the traffic generated by the application is modeled as individual packets, it is called discrete traffic. This means each and every packet generated by the application is modeled and each packet traverses all of the queues, processors, and buffers on each switch and router through the entire path from the source node to the destination node.

The advantage of this approach is that it results in a high fidelity simulation that can capture all the transient phenomena in the network. It can give highly accurate results and detailed statistics.

The disadvantage is that when the traffic volumes grow large (i.e., several gigabits per second), simulating it at the granularity of a packet becomes expensive. The simulation may take a long time to complete, even on a powerful machine.

Background Traffic

To overcome the disadvantages of discrete traffic, you can use background traffic. In this case, instead of modeling each packet, you can bundle the packets together into an aggregate, abstract entity called a "flow". A volume of traffic represented by a flow is characterized by two main attributes: bits/sec and packets/sec. This lets you adjust the volume of traffic to a manageable scale.

Note: Although flows are discussed in this section in relation to background traffic, you can, in fact, configure a traffic flow with a mixture of background and discrete traffic.

For example, a single flow can represent a 1-Gbps or a 100-Gbps traffic volume. Even with a one-hundred-fold increase in a large traffic volume, a simulation will be faster, since the traffic is in a single flow. A given model can have several flows to represent different traffic flows between end nodes, applications, or application sessions. A flow specification can have step variations in time and does not have to be the same value across the duration of the flow.

Advantages

The advantages of the background traffic flow approach are that the flows can simulate large volumes of traffic in a small simulation run time, resulting in faster simulations. Internally, instead of creating a large number of packets and routing each individually through switches and routers, flows are modeled mathematically. Mathematical models and micro-simulation techniques are employed to compute the impact of the flow on the intermediate queues and buffers.

Disadvantages

The disadvantage of using background traffic flows is that, since the granularity of the traffic model is now an aggregate flow rather than a packet, the results are less detailed than discrete traffic. Some amount of randomness is built into the mathematical models, but overall it is a steady-state model. Transient phenomena are not captured in the simulation. Also, since mathematical models are based on certain assumptions, background traffic flows might not be reliable at congestion points in the network at which the buffer or link capacities are exceeded or close to fully-utilized.

Support for Traffic Types in Application Modeling

With respect to application modeling, all of the models described in the Modeling Applications can generate discrete traffic. Additionally, application demands and the Voice standard application models can generate both discrete and background traffic (and any fractional mix of the two) dynamically at run time. Transaction Analyzer and Transaction Whiteboard contain wizards that can, at the time of model development, automatically create traffic flows in the network, based on the traffic pattern in the Transaction Analyzer or Transaction Whiteboard models. They cannot do this dynamically at run time, however.

Simulating Application Traffic

Riverbed Modeler has two simulation engines—discrete event simulation (DES) and Flow Analysis. DES can run both discrete and background traffic, while Flow Analysis is targeted to background traffic flows. All of the application models described in Modeling Applications can run in DES.

Flow Analysis only considers the traffic represented using *IP demands*, so to run a Flow Analysis simulation for application traffic, you must create IP demands to represent the traffic. In Transaction Analyzer and Transaction Whiteboard, you can use the wizard to automatically create IP demands based on the traffic pattern of the models.

For voice-over-IP (VoIP) traffic, you can use the VoIP wizard, which can automatically create IP demands based on the specifications you enter in the wizard. Additionally, for voice, Riverbed Modeler provides a VoIP Readiness Assessment Module, which simplifies the process of using Flow Analysis to design VoIP networks. For more information on this feature, see Using VoIP Readiness Assessment.

You can also create IP demands by importing traffic from third-party tools such as NetFlow, nGenius, Sniffer, and so on. For Transaction Analyzer and Transaction Whiteboard, there is a specialized simulator called Transaction Analyzer QuickPredict, which can run a high-level simulation of the Transaction Analyzer trace. In this the network topology is modeled in terms of effective bandwidth and latency.

Getting Started with Application Modeling

To configure a workstation or LAN to model the behavior of a user or group of users, you need to describe their behavior. A user's behavior or "profile" can be described by the applications he or she uses and how long and often the applications are used throughout a given period of time.

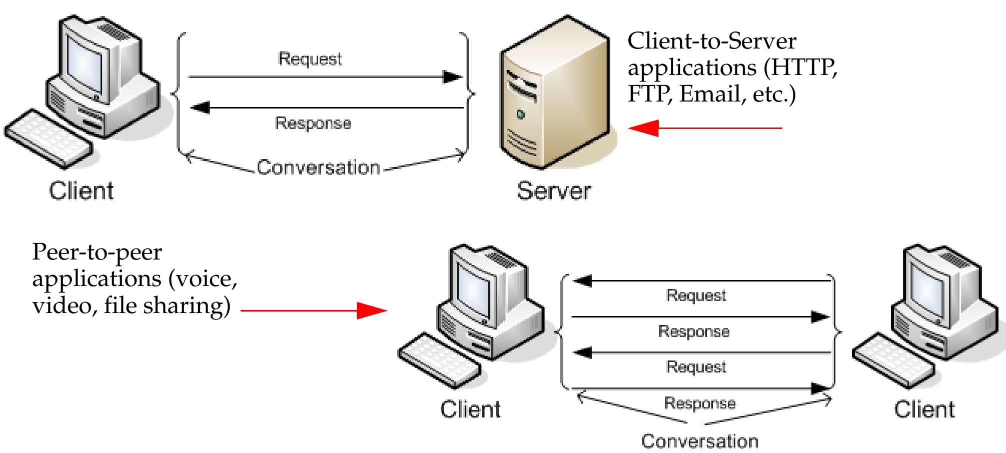
Modeler provides "global" objects for defining profiles and applications. The advantage of using a global object is that once you have defined the profiles and applications, you can reuse them across the entire topology. These global objects are portable entities that are defined independent of each other and of other objects. Therefore, you can copy and paste global objects from one project or scenario to another and reuse the same profiles and applications.

When you understand the standard network applications and the custom application, you should be able to determine which option best describes the behavior of the application whose activity you wish to model. There may be more than one application model you can use for a distinct scenario. The compatibility of an application model will likely depend on the objective of your simulation study. Some examples are included with application descriptions to assist in your understanding of this model.

Standard Applications Supported by Application Models

Depending on their underlying networks, application architectures may differ. Standard network applications are implemented in a two-tier architecture in which the client issues a request and a server or client receives the request and returns a response. This request-response exchange typically happens within one "conversation" between the client and the server, or between a client and another client. In this chapter, we use the term "conversation" to represent a sequence of activity between a client and a server within the context of a given application. A conversation includes a pattern of data exchanges, typically defined in a statistical manner that repeats over time.

Figure 18 Application Architecture



Standard Applications Workflow

When modeling standard applications, such as Email or FTP, configure the applications on the Applications Config node, then define the profiles in the Profile Config node, and finally deploy the applications and profiles to network objects. The workflow for building standard applications is as follows:

1) Configuring Individual Applications for Application Modeling

2) Configuring the Profiles

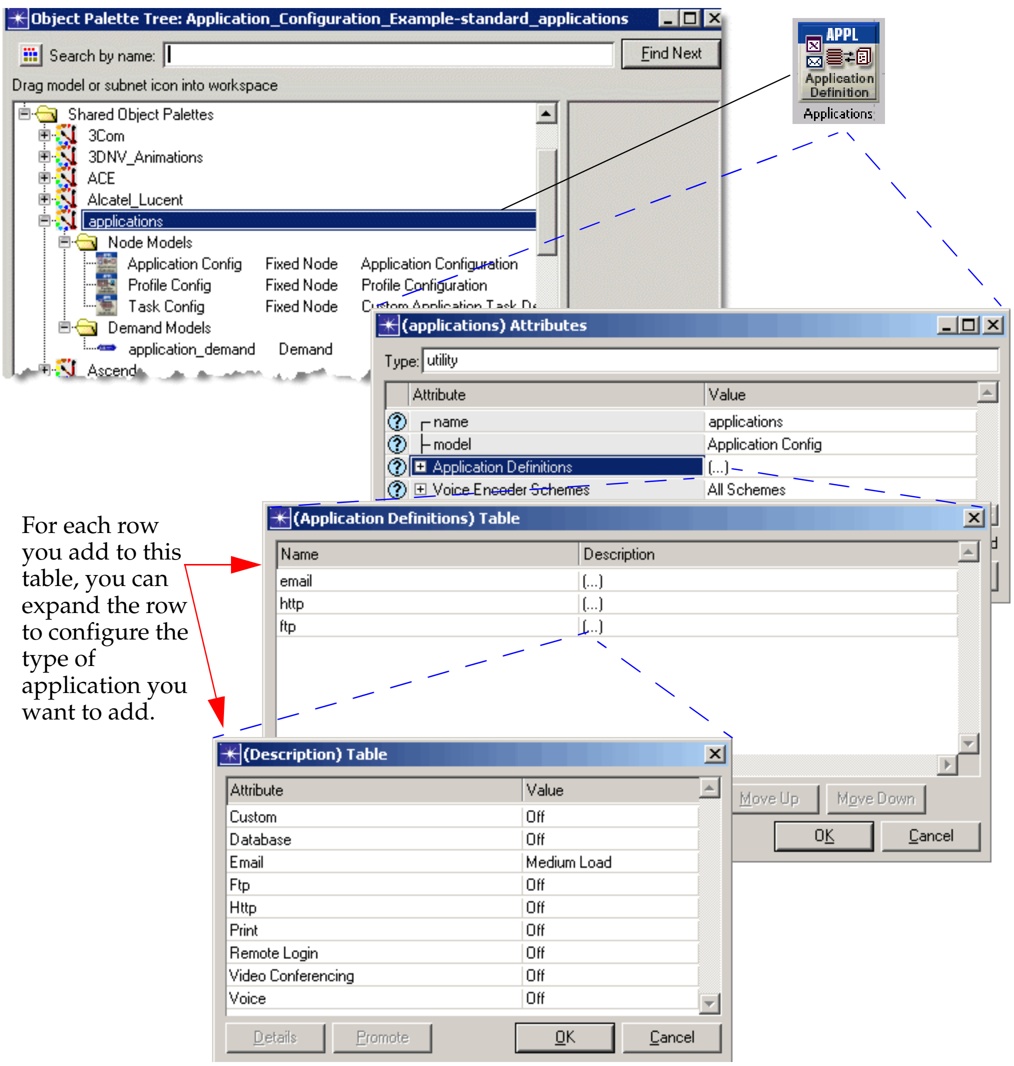
3) Deploying the Applications

4) Run a simulation

Configuring Individual Applications for Application Modeling

To begin configuring an application, drag and drop an Application Config node object from the Applications Object Palette to the Project Editor.

Figure 19 Applications Config Node



Right-click on the application config node object, and select Edit Attributes... to configure specific parameters. In the following section, the attributes for each type of standard application are described. You can configure these additional parameters for each selected application. These standard applications include

- FTP in Application Modeling

- Peer-to-Peer File Sharing in Application Modeling
- Email in Application Modeling
- Remote Login in Application Modeling
- Video Conferencing in Application Modeling
- Video Streaming in Application Modeling
- Database in Application Modeling
- HTTP in Application Modeling
- Mobile User Applications in Application Modeling
- Print in Application Modeling
- Voice in Application Modeling

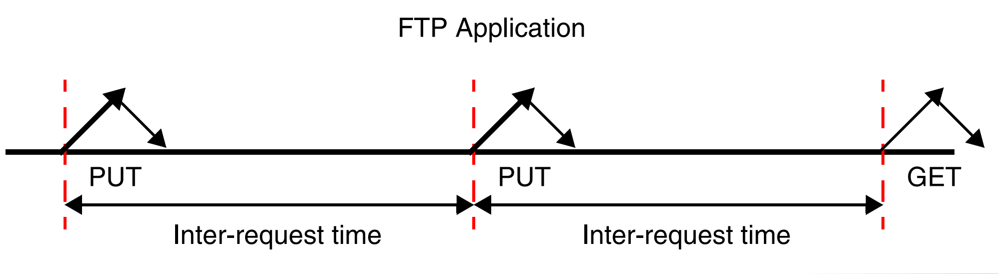
Note: You can use measurements from your live network to determine the settings for your application model. Each application description that follows includes a section called, “Using Measurements for Configuration” to describe ways to use collected measurements.

FTP in Application Modeling

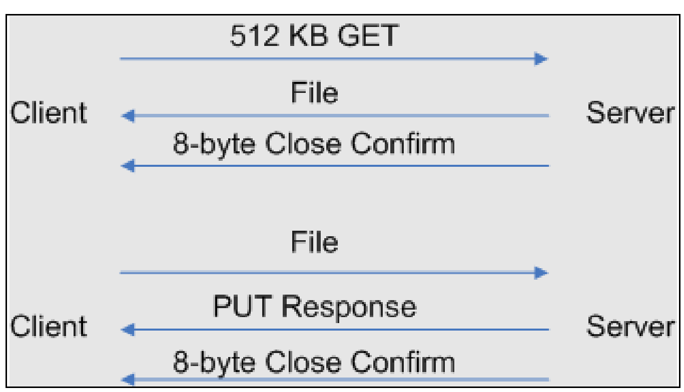
An FTP application enables file transfers between a client and a server. FTP has two basic commands for transferring a file: “get” and “put”. The “get” command triggers the transfer of a file from a remote server. The “put” command sends a file to a remote server.

Figure 20 FTP Application Message Exchange

How FTP works:



How FTP is modeled in Riverbed Modeler:



For each “put” or “get” file transfer, the FTP application model sends two messages: a control message and a data message. The control message represents a confirmation response for a “put” operation or the request for a “get” operation. The data message represents the file being transferred. You can configure the size of the data message, but the size of the control message is 512 bytes, by default. The effects of the control messages are more noticeable when you are modeling the transfer of small files, but have less impact on the results for large file transfers.

For connection-oriented transport protocols such as TCP, the model opens a new transport connection for each file transfer. The model does not include separate channels for data and control traffic—the data and control messages for a file transfer use the same TCP connection. TCP is the default transport protocol for this application.

Some of the important FTP attributes are listed in the following table.

Table 3 FTP Application Model Attributes

Attribute	Description
Command Mix (get/total)	Ratio of “get” (download) commands to the total number of commands (sum of “gets” and “puts”).
Inter-Request Time (seconds)	Time between subsequent file requests.
File Size (bytes)	Size of a file being transferred.
Symbolic Server Name	Symbolic name of the file server to which the client connects. For more information, see Edit Destination Preferences.
Type of Service	Quality-of-service parameter used to assign a priority to the traffic generated by this application.
RSVP Parameters	RSVP parameters for making bandwidth reservations.
Back-End Custom Application	Points standard application to custom application, allowing complex modeling of web applications, web caching, email, etc. **Note:** You must configure the custom application before pointing a standard application to it in this manner.

Model Limitations

The rate at which files are requested is independent of the responses received, that is, the second request can leave without the first response being received.

To represent a single user generating FTP application traffic, a single row of this attribute must be configured. If multiple rows are configured, each row represents an independent FTP session; that is, one of multiple users generating FTP application traffic simultaneously on the same machine.

Using Measurements for Configuration

To obtain the total number of FTP connections opened by a user, position a protocol analyzer on a segment shared by the client. For each FTP session, there is one control connection and one data connection. To obtain the number of file transfers, divide the total number of FTP-related connections by two.

The ratio of FTP “gets” to “puts” is judged from the direction of the data connections:

- An FTP “put” issued at the client results in a data connection from the client to the server.
- An FTP “get” issued at the client results in a data connection from the server to the client.
- The file transfer rate = (number of file transfers) / (length of traffic capture). The Inter-Request Time is the inverse of the file transfer rate.

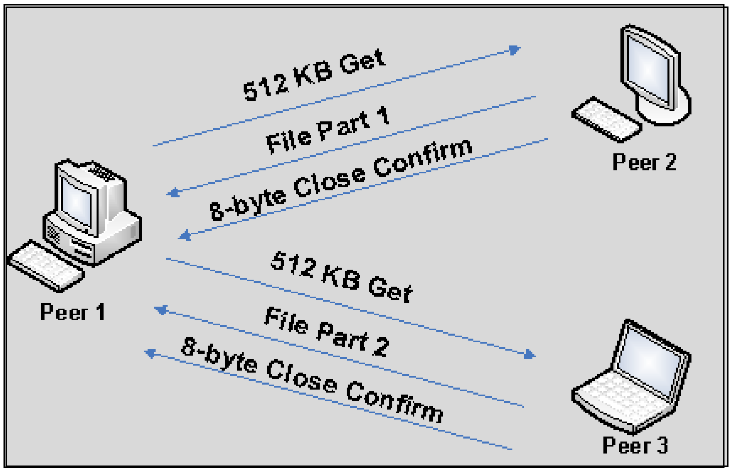
- The average file size = (total number of bytes) / (number of file transfers).

Peer-to-Peer File Sharing in Application Modeling

Peer-to-peer (P2P) file sharing over a network eliminates the need for centralized servers, allowing all computers to communicate and share resources as equals. That is, peer-to-peer file sharing is a client-to-client protocol without the concept of a server. Music and video file sharing are examples of network applications that rely on peer-to-peer technology.

Figure 21 Peer-to-Peer File Sharing Message Exchange

How P2P File Sharing is modeled in Riverbed Modeler



Note: The Application_Configuration_Example project, included with Riverbed Modeler, contains a pre-configured scenario (“peer_to_peer_file_sharing”) that demonstrates a simple configuration to enhance your understanding of this application. Example projects can be found in the following directory: <reldir>\models\std\example_networks.

Some of the important peer-to-peer file sharing attributes are listed in the following table.

Table 4 Peer-to-Peer File Sharing Model Attributes

Attribute	Description
Inter-Request Time (minutes)	Time between subsequent file requests.
Requested File Size (bytes)	Defines the size of the requested file.
File Popularity	Defines the popularity of the files (i.e., the number of peers that have the requested file).
Leecher Probability	Defines the probability that a node that runs the Peer-to-Peer File Sharing application will be a leecher. A leecher is a node that only downloads files from other peers and doesn't contribute any files to the peer-to-peer system.

Peer-to-peer file sharing differs from HTTP and FTP downloads in that a peer may request a portion of a file (for example, a movie file) from multiple peers rather than just one. Simultaneous requests for portions of a file are supported.

To configure peer-to-peer file sharing, create an application definition in the Application Definition config node using the Peer-to-Peer File Sharing application, and then configure one or more profiles that include the application you defined. The nodes on which these profiles are deployed will form the peer-to-peer file sharing community.

Note: You do not need to specify a node that supports this application under its “Supported Services”; because peer-to-peer file sharing is modeled as a client-to-client application.

During a simulation of peer-to-peer file sharing, Riverbed Modeler does the following:

- Determines the peers participating in a file sharing operation by using the profile(s) assigned to each peer
- Determines the number of peers from which to retrieve equal portions of a given file by using the File Popularity attribute
- Determines the likelihood of a peer being a download-only client, or *leecher*, by using the Leecher Probability attribute, described in Table 4. For example, if you have 10 peers in a network, with a Leecher Probability value of 0.2, then two of the peers will not share files but will download files only.

Special Situations

File Popularity Value Greater than the Number of Nodes

If the computation based on the File Popularity setting returns a number that is greater than the number of nodes running the peer-to-peer file sharing application, the file popularity value is set to the total number of available nodes running the peer-to-peer file sharing application (minus the leecher nodes and the node requesting the file). In this case, the peer will contact all other peers for the download of that particular file. When the node is searching peers for the next download, a new popularity value will be calculated again based on the attribute’s setting.

Unreachable Peer for Download

If a peer that is selected for download of a file portion is unreachable, the peer originating the file search will try to contact any remaining available peers that have not yet been contacted for that file search instance. If all available peers have been contacted for that file search instance, a peer that has already been contacted will be selected at random. This means that for that file search instance, the randomly-contacted peer will have to return two parts of the file.

Peer Becomes Unreachable During Download

If one of the selected peers becomes unreachable during a download, the peer originating the file search requests the remaining part of the file from another available peer. The priority is given to peers that have not yet been contacted for that file search instance. If all of the available peers have been contacted for that file search instance, an already contacted peer will be selected at random.

Note: If a node running the peer-to-peer file sharing application is unreachable by another peer at any point, the unreachable peer will still be considered as a candidate peer in all consecutive file search instances and will not be permanently flagged as unreachable.

Model Limitation

Many peer-to-peer file sharing protocols have clients known as trackers that keep track of file distributions. Trackers are not modeled in Riverbed Modeler.

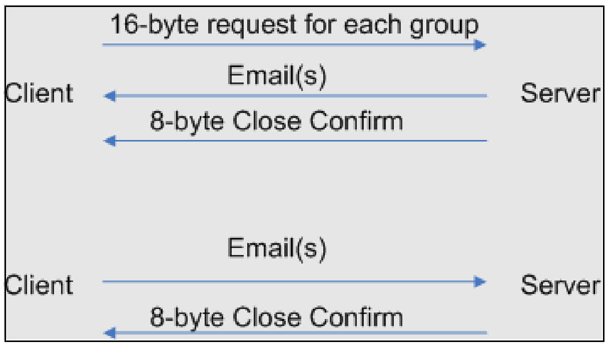
Email in Application Modeling

The default transport protocol used in the email application model is TCP, i.e., messages are sent and received using TCP. Modern email packages use a combination of SMTP (Simple Mail Transfer Protocol) and POP (Post Office Protocol). Both SMTP and POP use TCP as the underlying transport. SMTP transfers an email from the client to the mail server.

The following figure shows how Email is modeled in Riverbed Modeler.

Figure 22 Email Application Message Exchange

How Email is modeled in Riverbed Modeler



Some of the important email attributes are listed in the following table.

Table 5 Email Application Model Attributes

Attribute	Description
Send Interarrival Time (seconds)	Time between e-mails sent from the client to the server.
Send Group Size	Number of e-mail messages grouped before transmission.
Receive Interarrival Time (seconds)	Time between e-mails received from the server at the client.
Receive Group Size	Number of e-mail messages grouped before reception.
E-Mail Size (bytes)	Average size of an e-mail message.
Symbolic Server Name	Symbolic name of the server to which the client connects. For more information, see Edit Destination Preferences.
Type of service	Quality-of-service parameter for assigning priority to this application's traffic.
RSVP Parameters	RSVP parameters for making bandwidth reservations.
Back-End Custom Application	Points standard application to custom application, allowing complex modeling of web applications, web caching, email, etc. **Note:** You must configure the custom application first, before pointing a standard application to it in this manner.

Example: Server-Based Email Application

When a client sends an email, the email is stored on the server. The client polls the server on a regular basis, and receives e-mail destined to it. You can model this architecture easily by configuring the email Standard Application Model, which provides a Send Interarrival Time attribute and a Receive Interarrival Time attribute.

Note that the interarrival time, which is the time between successive messages, is the inverse of the rate. Inter-arrival times are configurable on an individual basis for each client. Send and receive inter-arrival times are independent. This means that a client can be a frequent sender of messages but an infrequent recipient.

Model Limitations

The send rate is independent of the receive rate. Also, if TCP is used as the transport protocol, there is only one TCP connection open from the client to the server. Messages are sent and received through this single TCP connection.

Using Measurements for Email Configuration

You can use the following measurements to configure the Email application.

- Use a protocol analyzer to determine the protocols supporting e-mail.
- To obtain the send interarrival time and the receive interarrival time, place a protocol analyzer on a segment shared by the client, configure the analyzer's filters to capture e-mail traffic exclusively, and monitor the e-mail exchange between the client and the server.
- The total bytes transferred from the server to the client (and vice-versa) are obtained.
- The average e-mail size = (total number of bytes)/(number of e-mails exchanged).

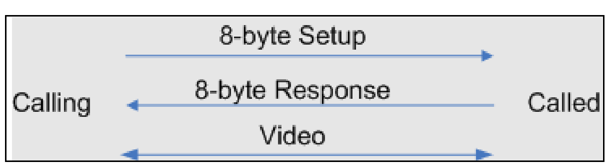
Note: The number of e-mail messages exchanged can be obtained by requesting the user to quantify e-mails sent/received when determining the send and receive rates.

Remote Login in Application Modeling

A Remote Login application models a remote login scenario. Users login to different machines, interact with the operating systems of the remote hosts. The commands they enter and the feedback they receive generate traffic on the network. TCP is the default transport protocol. The following figure shows how remote login is modeled in Riverbed Modeler.

Figure 23 Remote Login Application Message Exchange

How Rlogin is modeled in Riverbed Modeler



Some of the important Remote Login attributes are listed in the following table.

Table 6 Remote Login Application Model Attributes

Attribute	Description
Inter-Command Time (seconds)	Time between commands issued within a remote login session.
Terminal Traffic (bytes/command)	Average amount of data transferred per command.
Host Traffic (bytes/command)	Average amount of data returned in response to a command.
Symbolic Server Name	Symbolic name of the server to be contacted. For more information, see Edit Destination Preferences.
Type of Service	Quality-of-service parameter for assigning priority to this application's traffic.
RSVP Parameters	RSVP parameters for making bandwidth reservations.
Back-End Custom Application	Points standard application to custom application, allowing complex modeling of web applications, web caching, email, etc. **Note:** You must configure the custom application first, before pointing a standard application to it in this manner.

Model Limitations

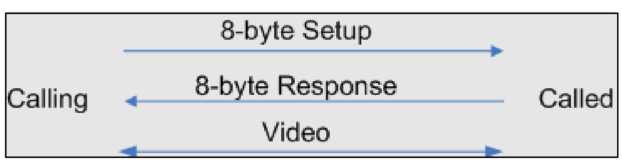
Traffic within each login session (modeled as an independent TCP connection) is represented by the host traffic and terminal traffic attributes. Host traffic is defined as the traffic being received at the client. Terminal traffic is defined as the traffic being sent by the client.

Video Conferencing in Application Modeling

A video conferencing application lets users transfer streaming video frames across the network. UDP is the default transport protocol used for video conferencing. The following figure shows how video conferencing is modeled in Riverbed Modeler.

Figure 24 Video Application Message Exchange

How video applications are modeled in Riverbed Modeler



Some of the important video conferencing attributes are listed in the following table.

Table 7 Video Application Model Attributes

Attribute	Description
Frame Interarrival Time Information >	
Incoming Stream Interarrival Time (seconds)	Time between frames generated within a video conferencing session from the destination.
Outgoing Stream Interarrival Time (seconds)	Time between frames generated within a video conferencing session from the source.
Frame Size Information	
Incoming Stream Frame size (bytes)	Average size of an incoming video frame.
Outgoing Stream Frame size (bytes)	Average size of an outgoing video frame.
Symbolic Destination Name	Symbolic name of the destination to which the client connects. For more information, see Edit Destination Preferences.
Type of service	Quality-of-service parameter for assigning priority to this application's traffic.
RSVP Parameters	RSVP parameters for making bandwidth reservations.

Model Limitations

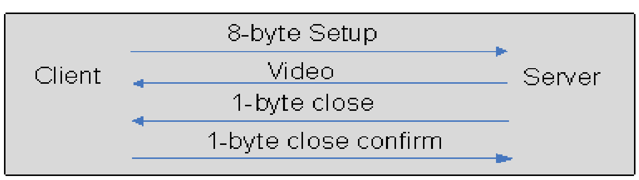
If TCP is used as the transport protocol (not the default), each conference opens up an independent TCP connection.

Video Streaming in Application Modeling

With the video streaming application support in Riverbed Modeler, you can capture the video traffic produced by a given video streaming software (for example, VLC) into a file, and then use the file in Riverbed Modeler to simulate the same traffic in a scenario. The following figure shows how video streaming is modeled.

Figure 25 Video Streaming Message Exchange

How video streaming is modeled in Riverbed Modeler



Some of the important video streaming attributes are listed in the following table.

Table 8 Video Streaming Model Attributes

Attribute	Description
Video File Name	Name of the GDF file containing the traffic capture. Inside the file, the first line specifies the size of the packets, and the remaining lines specify the packet inter-arrival times in microseconds.
Symbolic Server Name	Symbolic name of the server sending the video streaming packets.
Type of service	Quality-of-service parameter for assigning priority to this application's traffic.
RSVP Parameters	RSVP parameters for making bandwidth reservations.

The following procedure describes how to capture video streaming traffic from an application, using VLC as an example.

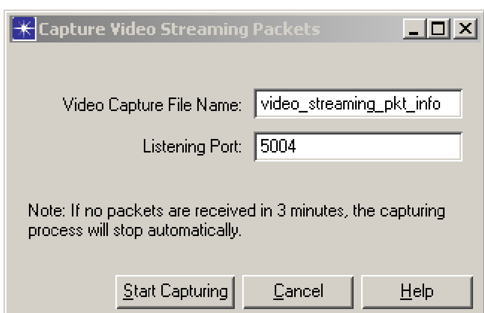
Procedure 6 Capturing Video Streaming Traffic

1. Prepare to capture the video streaming traffic.

1.1. Choose Protocols > Applications > Capture Video Streaming Packets.

➢ The following dialog box appears.

Figure 26 Capture Video Streaming Packets Dialog Box



Riverbed Modeler will append the GDF file extension to the filename you specify.

1.2. Specify a filename to which to write the packets in the Video Capture File Name field.

1.3. Specify a port on which the video streaming application is communicating in the Listening Port field.

1.4. Click the Start Capturing button when you are ready to listen for traffic on the specified port.

1.5. The video capture file will be stored in the primary directory (that is, the first directory listed) in your model directories (mod_dirs) preference. The file should be kept either in the primary model directory or in the same directory as the scenario that uses the file resides; otherwise, the application model will not be able to find the file during simulation. For more information about configuring and using model directories, see mod_dirs on page EI-2-48.

1.6. Go to the video streaming application and start sending traffic.

Note: Always start the capture before you start sending the packet stream so that no packets are missed.

2. Send traffic from the video streaming application.

Note: These steps are based on the use of VLC. If you use another program, field names and order may be different.

2.1. From the video streaming application, specify the following:

- Video streaming protocol

Note: RTP / MPEG Transport Stream is supported

- IP address of the destination machine (i.e., the machine running Riverbed Modeler)
- Destination port (which is specified in Modeler as the "Listening UDP Port" (see Figure 26, above)
- Time-to-live. This is required if the video streaming application is on a different machine than Riverbed Modeler. Increase the number to at least the number of hops between the source and destination addresses.
- (optional) Transcoding scheme (can be used to model different video codecs)

2.2. Click Stream when you are ready to start sending packets.

3. To stop capturing traffic, either stop the transmission of the video stream in the video streaming application or cancel the Capture Video Streaming Packets process in Riverbed Modeler. Otherwise, capturing will stop automatically three minutes after the end of the video.

End of Procedure 6

Note: A video capture file ("video_streaming.gdf") is shipped with Riverbed Modeler as part of the "Application Configuration Example" project, in the "standard_applications" scenario. You may examine the contents of this example file and study the example project scenario to see how video streaming is modeled.

Collecting Statistics for the Video Streaming Application

Note the following statistics that are available for the Video Streaming application.

- Global Statistics > Video Streaming—Use global statistics to see traffic for all video streaming servers and receivers in the scenario
- Node Statistics > Client Video Streaming—Use these statistics to see the packet delay, delay variation, and traffic received per client.
- Node Statistics > Server Video Streaming—Use these statistics to see the traffic sent per video streaming server.

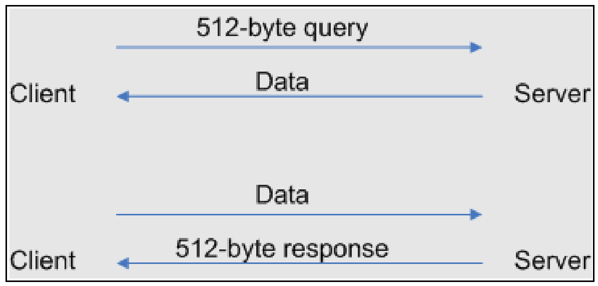
Database in Application Modeling

Database operations are divided into two categories: (1) a database entry, and (2) a database query. A database entry results in a fixed amount of data being written into the database. A database query results in the client issuing a query, and the server responding with some data. The default transport protocol for the database application is TCP.

The following figure shows how database transactions are modeled in Riverbed Modeler.

Figure 27 Database in Application Message Exchange

How Database applications are modeled in Riverbed Modeler



Some of the important database attributes are listed in the following table.

Table 9 Database Application Model Attributes

Attribute	Description
Transaction Mix (Queries/Total Transactions)	Ratio of queries to the total number of transactions (queries + updates).
Transaction Interarrival Time (seconds)	Time between transactions.
Transaction Size (bytes)	Average size of an entry or a response to a query.
Symbolic Server Name	Symbolic name of the server the client contacts. For more information, see Edit Destination Preferences.
Type of Service	Quality-of-service parameter for assigning priority to this application's traffic.
RSVP Parameters	RSVP parameters for making bandwidth reservations.
Back-End Custom Application	Points standard application to custom application, allowing complex modeling of web applications, web caching, email, etc. **Note:** You must configure the custom application first, before pointing a standard application to it in this manner.

Model Limitations

Requests within a transaction run independently of responses. The second request can be initiated before the first one completes.

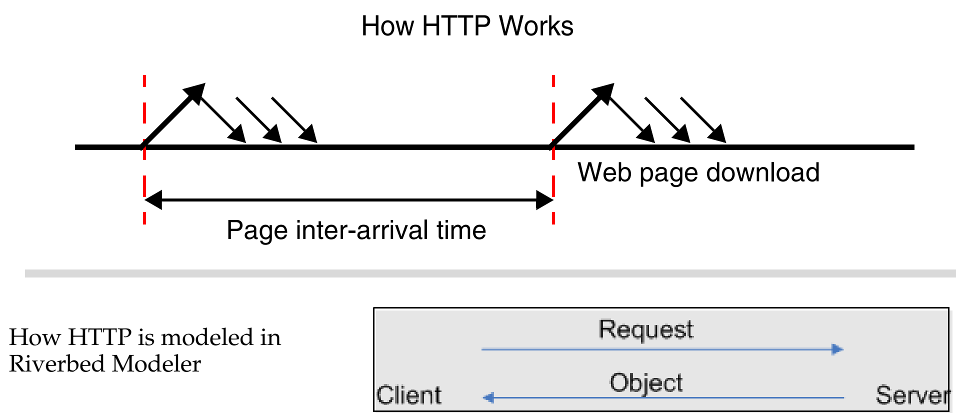
There are two types of transactions in the standard database application: an entry/response transaction and a query/response transaction. In the entry/response transaction, the size of the entry is determined by the Transaction Size attribute and the size of the response is fixed at 512 bytes. For the query/response transaction, the query size is fixed at 512 bytes and the response size is determined by the Transaction Size attribute.

HTTP in Application Modeling

The HTTP application models Web browsing. The user downloads a page from a remote server. The page contains text and graphic information, and sometimes video. These page elements are collectively referred to as “inline objects”. TCP is the default transport protocol for HTTP. Each HTTP page request may result in opening multiple TCP connections for transferring the contents of the inline objects embedded in the page. The number of concurrent TCP sessions is determined by the application configuration.

The following figure shows the requests and objects returned in an HTTP transaction. An HTTP request is sent for each inline object. The size of the request packet is determined by the setting of the HTTP attribute.

Figure 28 HTTP Application Modeling



A video object contained in a web page may be

1) downloaded automatically in its entirety, along with other inline objects,

2) streamed (downloaded) upon the viewer’s request, after a random delay, in its entirety and played upon the start of the download, or

3) streamed as live video upon viewer's request, after a random delay

In the model, the characteristics of the inline objects that comprise the web pages are set using the attribute “Page Properties” (described in Table 10), which has these sub-attributes.

- Automatically Loaded Page Objects—Specifies properties for all of the objects in the page that are automatically loaded, which includes automatically loaded videos (such as advertisements). Therefore, this attribute group specifies settings for videos described by case 1, above.
- Streamed Video Properties—Specifies properties for video that does not automatically load when a user opens the page. The video type can be “Live” (like web-TV, as in case 3 above) or “On Demand” (like playing a movie from an online archive, as in case 2 above).

Some of the important HTTP application attributes are listed in the following table.

Table 10 HTTP Application Model Attributes

Attribute	Description
HTTP Specification >	
HTTP Version	The name of the supported HTTP version, 1.0 versus 1.1.
Max Connections	Maximum number of simultaneous TCP connections that HTTP can spawn.
Max Idle Period (seconds)	Maximum idle time after which a connection is torn down.
Number of Pipelined Requests	The number of HTTP requests that can be buffered together in a single application message. This attribute is only used for HTTP version 1.1.
Request Size	Specifies the size, in bytes, of a single requested object.
Page Interarrival Time (seconds)	Time between subsequent pages that a user browses.
Page Properties >	
Automatically Loaded Page Objects >	
Object size (bytes/object)	Average size of an object. The first row in the table defines the page itself. All other rows define object on the page. The value for this field is a probability distribution or a choice of pre-defined object sizes for images or videos contained in a page.
Number of Objects (objects/page)	Number of inline objects contained in a page.
Location	The symbolic name of the server on which the object is located. For more information, see Edit Destination Preferences.
Streamed Video Properties >	
Video Existence Probability	Probability that a video object is contained in the page. A page can have at most one streamed video. Predefined special values are “All Pages Include a Video” or “No Video”.
Play Start Time Offset	The time difference, in seconds, between the time the download of the page was started and the time that the user clicked “Play” to start the video
Video Type	Specifies attributes for video that does not pre-load when a user opens the page. The video type can be “Live” (like web-TV) or “On Demand” (like playing a movie from an online archive). • For “Live” video, the generated traffic is calculated based on the value of “Frame Size” and the value of “Frame Inter-arrival Time” distributions for the specified duration of the video. • For “On Demand”, the HTTP server sends back a file sized equal to the mean frame size (“Frame Size”) times the video duration (“Video Length”) divided by the mean frame interarrival time (“Frame Inter-arrival Time”) **Note:** For “On Demand”, you may not set the duration of the video to “Until Page Closed”.

Table 10 HTTP Application Model Attributes (Continued)

Attribute	Description
Video Length	Defines the duration of the video • For Video Type "Live", this attribute specifies for how long the live video will be viewed. A live video will be assumed to be finished when the page is closed, regardless of the length specified with this attribute. **Note:** For live videos, you can use the special value "Until Page Closed". • For Video Type "On Demand", the value of this attribute is used while computing the sizes of the video objects.
Server Selection >	
Initial Repeat Probability	Probability that a user would request the next page from the same server.
Pages Per Server	Number of pages accessed consecutively on the same server.
RSVP Parameters	RSVP parameters for making bandwidth reservations.
Type of Service	Quality-of-service parameter for assigning priority to this application traffic.

Using Measurements for Configuration

The above attributes require measurement of user behavior. For example, to configure the page interarrival time, a typical user's browsing activity should be monitored. A corporate firewall may be a good source of information about average page downloads and the number of bytes transferred. There are many applications that monitor Web activity and provide summary statistics on corporate Internet usage.

Typically, the number of inline objects reflects the amount of graphical elements on a page.

Example: Intranet Application

An IS department has deployed an intranet application that lets employees use services such as the company directory, and marketing information. Each employee has a browser installed on her desktop. A central web server serves the intranet site. The web-browsing application is two-tier with the browser issuing requests to the web server and the server returning pages, text or images.

The HTTP application model will represent the above architecture appropriately because it allows you to do the following:

- Select the average number of pages downloaded and the average number of objects per page
- Choose the average page size and the number of servers accessed to download various objects on the page

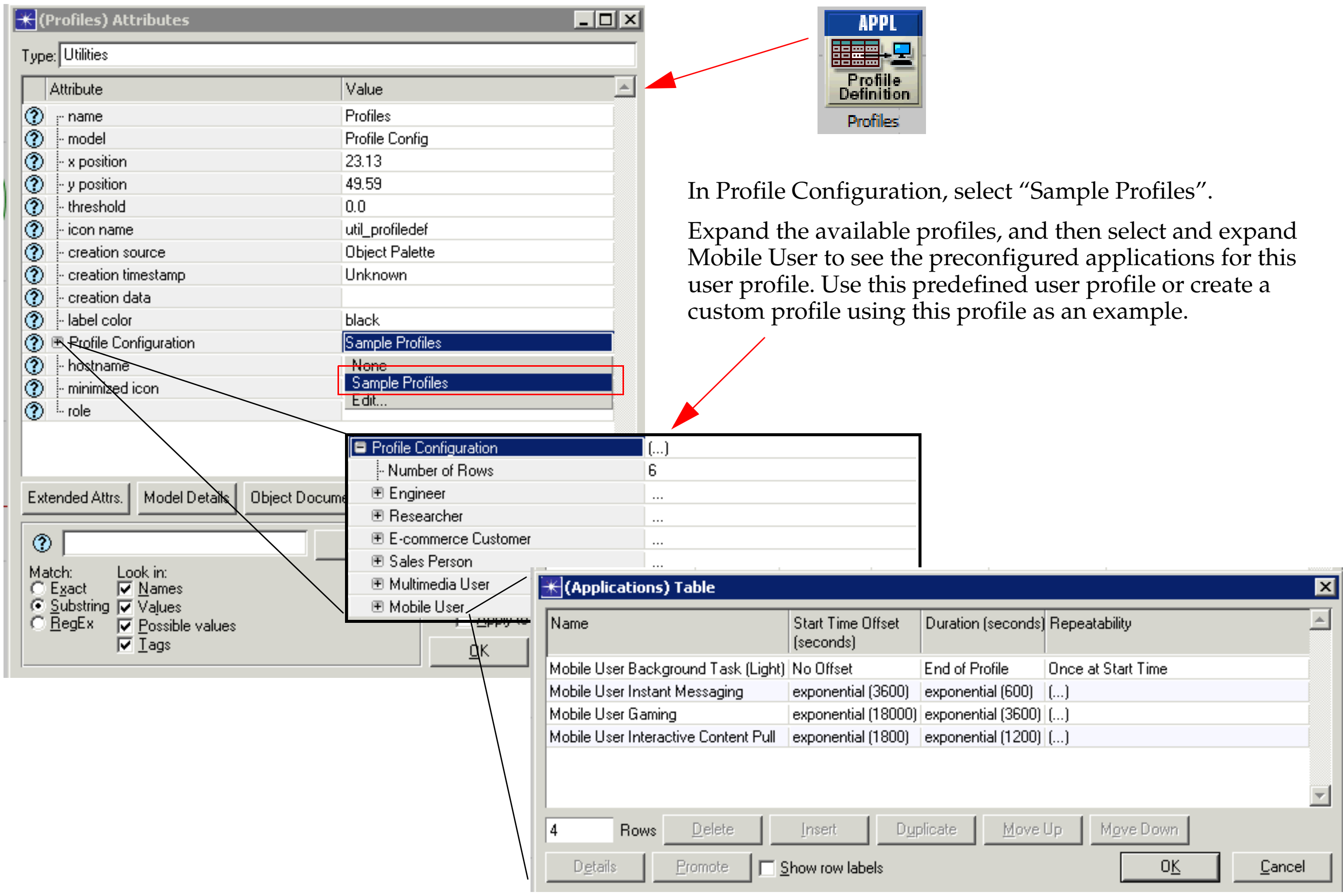
The number of TCP connections that can be opened simultaneously by the browser can be specified.

Mobile User Applications in Application Modeling

A Mobile User traffic profile, based on the technical report TR 36.822 (published by 3GPP), is available to let you model a variety of mobile user applications popular on cell phones and tablets. This traffic profile represents a typical mobile device user running applications that perform interactive content pull, instant messaging, Internet gaming, and device background tasks.

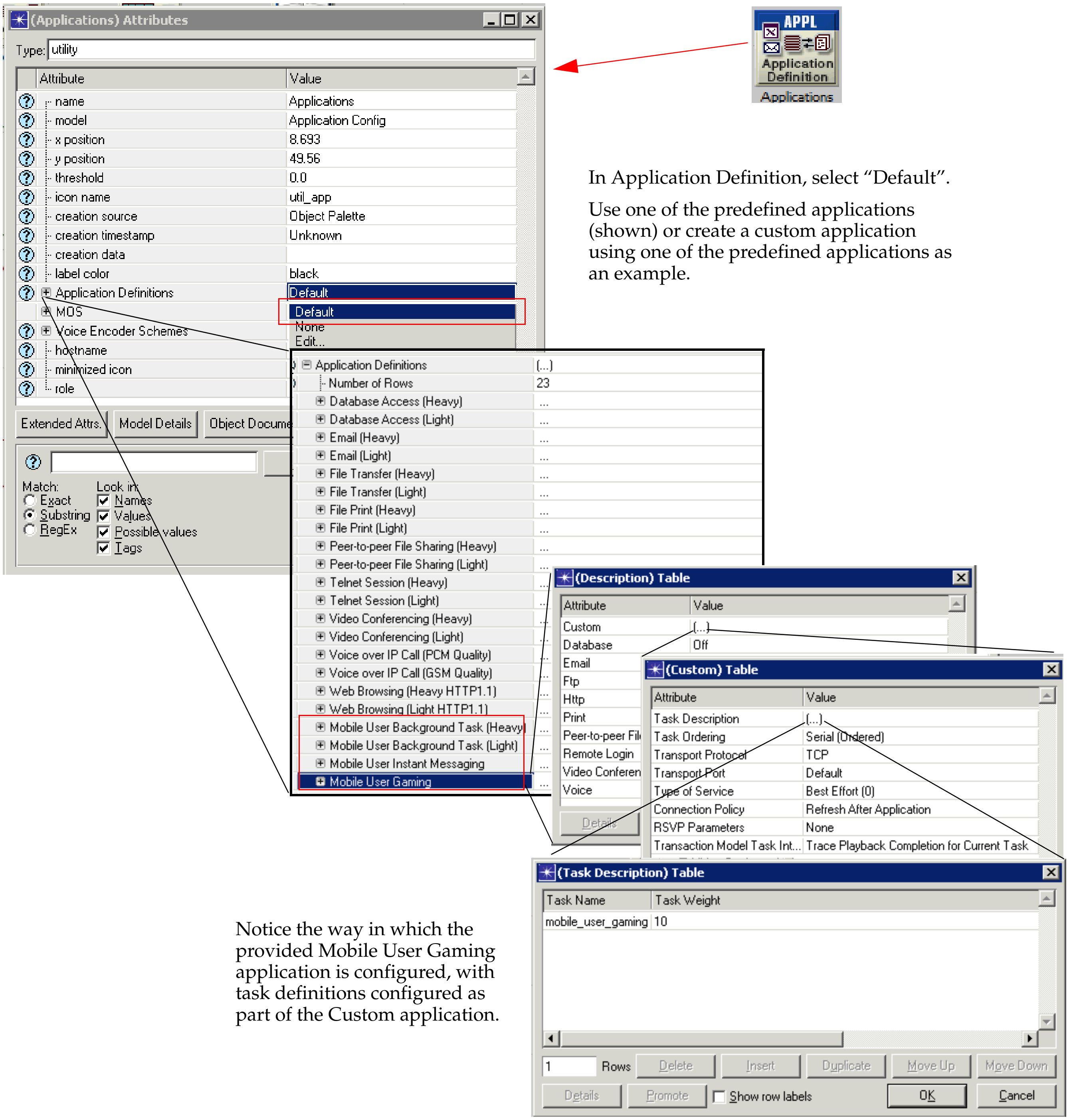
This predefined Mobile User traffic profile is shown in the figure below. You can use this profile as a starting point to create your own characterization of the mobile users in your network model.

Figure 29 Mobile User Profiles



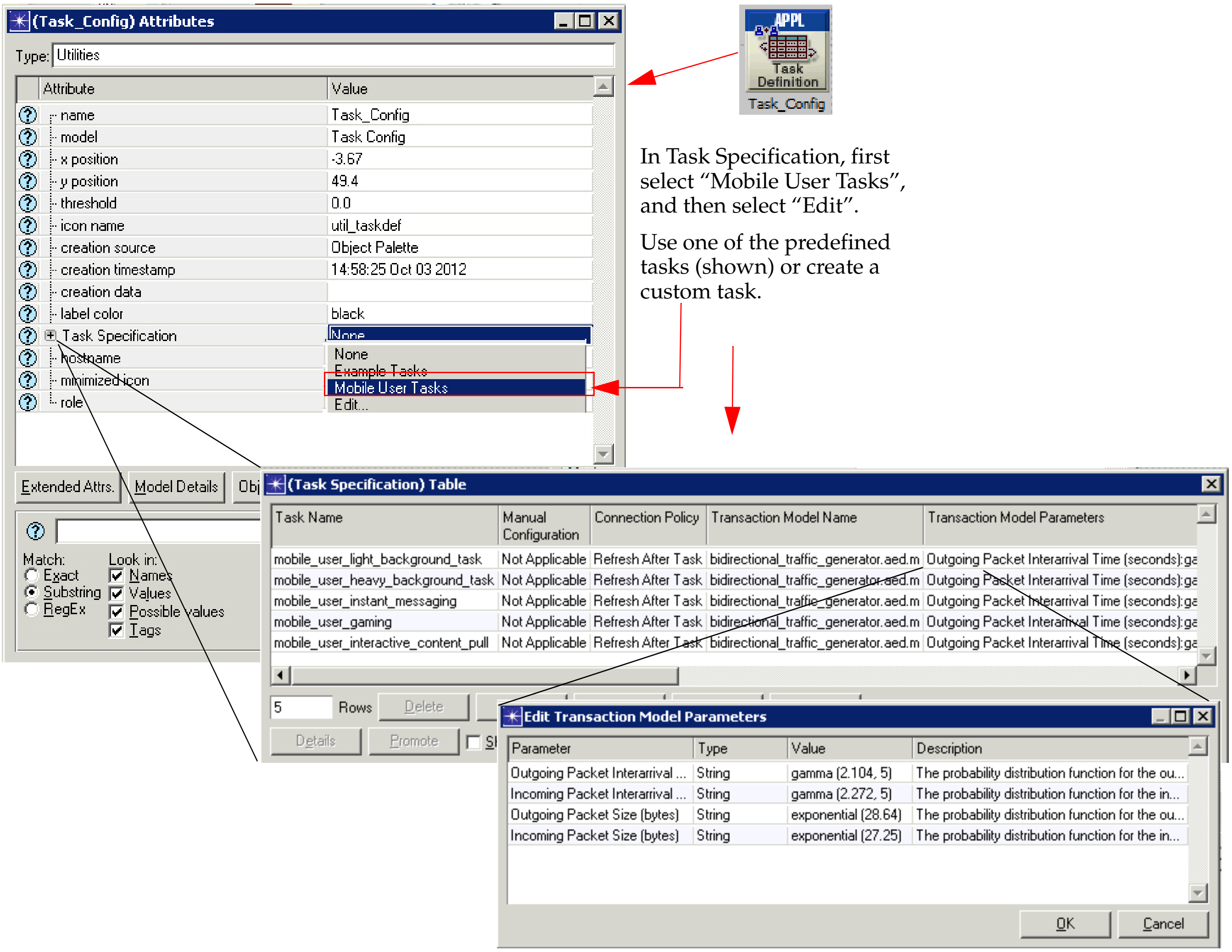
The applications that are part of the predefined traffic profile are configured in the Applications configuration node. To see the predefined applications, choose “Default” in the Application Definition utility node, as shown below. The mobile user application definitions, which are generated using the custom application model, will be part of that list that comprises the default application definitions.

Figure 30 Mobile User Applications



The tasks that are used in the default mobile user application definitions are specified in the Task Definition utility node.

Figure 31 Mobile User Tasks



As we can see from the figure above, all the mobile user tasks are based on a simple Transaction model called bidirectional_traffic_generator. This model generates packets in both directions (client -> server, server -> client) based on the following input parameters: Outgoing Packet Inter-arrival Time, Incoming Packet Inter-arrival Time, Outgoing Packet Size, and Incoming Packet Size.

These parameters are different for each of the mobile user tasks, and their values are based on actual values collected in real networks in various field studies, as reported in 3GPP TR 36.822. In particular, for each parameter of each task, one representative curve was selected from the plots presented in 3GPP TR 36.822, a probability distribution function (PDF) that describes the curve was derived and set as the value of the Transaction model parameter. Since the traffic in both directions is generated using statistical models, these tasks do not follow a request-response concept.

You can customize the values for these tasks either directly or by duplicating the task with a new name, and then modifying the new task.

Collecting Statistics for Mobile User Applications

Note the following in regard to selecting and collecting statistics for Mobile User applications.

- If you want to collect statistics for each Mobile User Application separately, choose the statistics under "Transaction Analyzer Model" group.
- The "Transaction Analyzer Model > Task Response Time" statistic is not applicable, as discussed above, so it is not collected.

Several example projects contain mobile user applications. For example, the WLAN project, included with Riverbed Modeler, contains a pre-configured scenario ("WLAN_backbone_config_ex") that demonstrates a simple configuration to enhance your understanding of the deployment of mobile user applications in a network.

Note: Example projects can be found in the following directory: <reldir>\models\std\example_networks.

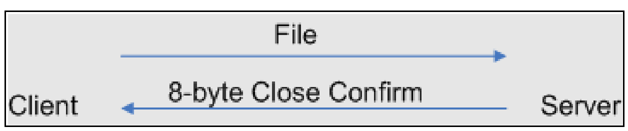
Note: To learn more about defining custom applications, including creating or editing profiles like the one provided for Mobile User applications, please see Custom Application on page STM-2-34.

Print in Application Modeling

A print application allows the user to initiate print jobs. TCP is the default transport protocol used for this application. Each print job creates a new TCP connection with the printer, as shown below. A close confirm message completes the transaction.

Figure 32 Print Application Message Exchange

How Print applications are modeled in Riverbed Modeler



Some important print application attributes are listed in the following table.

Table 11 Print Application Model Attributes

Attribute	Description
Print Interarrival Time (seconds)	Time between print jobs issued by the user.
File Size (bytes)	Average size of the file sent for printing.
Symbolic Printer Name	Symbolic name of the printer. For more information, see Edit Destination Preferences.
Type of Service	Quality-of-service parameter for assigning priority to this application's traffic.

Voice in Application Modeling

A voice application enables two clients to establish a virtual channel over which they can communicate using digitally encoded voice signals. UDP is the default transport protocol used for this application. The voice data arrives in spurts that are followed by a silence period. Encoding schemes can be specified for the voice-to-packet translation.

Internally, the voice packets are sent over real-time protocol (RTP) streams. No special configuration is needed for RT, as RTP does not send explicit packets. By default there is no signaling in Voice; however, it can be set to H323 or SIP. See “Signaling” in Table 12.

Some important voice application attributes are listed in the following table.

Table 12 Voice Application Model Attributes

Attribute	Description
Silence Length (sec)	Silence length for the incoming and outgoing calls along with the associated distributions.
Talk Spurt Length (sec)	Length of a talk spurt for the incoming and outgoing calls along with the associated distributions.
Symbolic Destination Name	Symbolic destination name of the client. For more information, see Edit Destination Preferences.
Encoder Scheme[1]	Encoding scheme in effect at the client.
Voice Frames per Packet	Number of voice frames that can be sent in a single packet.
Type of Service	Quality-of-service parameter for assigning priority to this application's traffic.
RSVP Parameters	RSVP parameters for making bandwidth reservations.
Traffic Mix	Proportion of traffic that should be modeled analytically instead of discretely. Higher amounts of analytic traffic decrease simulation run times but may limit the statistics that can be collected. Setting this attribute to all background has the same effect as configuring this traffic as an IP traffic flow.
Signaling	Specifies the method used for establishing and tearing down a voice call. Can be H.323 or SIP.
Compression Delay (seconds)	Specifies the delay in compressing a voice packet.
Decompression Delay (seconds)	Specifies the delay in decompressing a voice packet.

1.Encoder schemes are managed through the top-level attribute “Voice Encoder Schemes”.

To set the parameters for the voice encoding you want to use, add encoding schemes in the Voice Encoder Schemes table.

Model Limitations

The following is a list of important notes on this model:

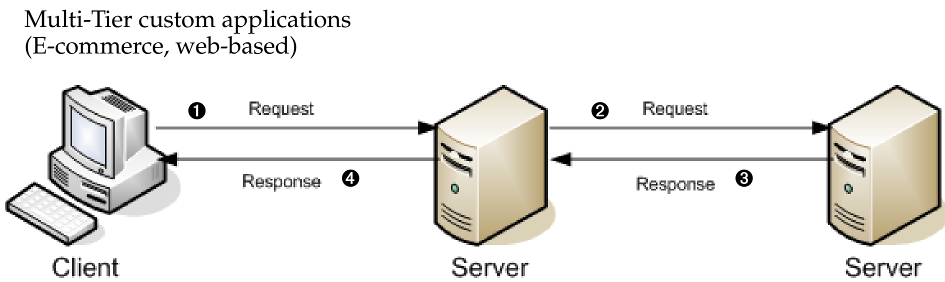
- A voice call is established from one client to another, no server is modeled for voice conversations.
- The parameters for both the incoming and outgoing directions of the voice call are configured on the client.

- The silence length refers to the pause between talk spurts, the source client may stop talking when the destination is talking.

Custom Application

The custom application is an application modeling framework that you can use to model a broad class of applications. It can be used when the application of interest does not correspond to any of the standard applications. The custom application provides attributes that allow you to configure various aspects of the application in detail.

Figure 33 Application Architecture



Custom application modeling is useful for modeling complex multi-tier applications. In a single, successful transaction, multiple processes happen on the same logical level either on the same machine or on more than one machine.

A custom application can be used to represent any number of tiers. Some examples in which custom application is applicable are described later in this section.

Note: An example project, “Custom_Application_Examples”, is included with the software. To access an example project, go to <reldir>\models\std\example_networks.

Custom Application Architecture

You can program the Custom Application to generate network and server loads according to specified patterns. The Custom Application has the following characteristics:

- More than two hosts can be involved in data transfers and processing. You can specify the sequence of hosts involved in each task. The sequence can begin with a workstation or server, end with a workstation or server, and can use any collection of workstations or servers in between.
- Client-Server interaction is task-based. A task consists of many interactions between a client and a server or between successive servers.

The basic task consists of many “phases”. Each phase consists of a data transfer and/or a processing event, which can occur at any end device. This end device becomes a tier for the application. Subsequent phases are typically set up to occur in a chain, where the destination of one data transfer phase becomes the source of the next data transfer phase. The entire task is complete only when the last phase of the task has been completed. A task need not terminate at the originating node, though this is a common case. Wherever the task completes, a “Response Time” statistic is measured to record the time separating the initiation of the task (e.g., when a simulated user presses the “OK” button) to its completion (e.g., when the “hourglass” disappears).

Custom Application Examples

Networked Calendar Application

A networked calendaring application is used to schedule meetings and to book resources for those meetings. When a calendar entry is made, the client application contacts the server separately for each staff member or resource involved in the entered event. As each new data item is received, schedule conflicts are generated and analyzed at the server side. The server then responds to the client, indicating that it is prepared to process the next item. Once all information has been received and processed by the server, the final entry is made, and a confirmation is returned to the client.

A custom application is a good choice to represent it because no standard calendaring application is provided. Another reason to choose the custom application model in this instance is that the request-responses are serial in nature, i.e., the next request is made only when the previous response is completed.

Three-Tier Customer Relationship Management Application

A sales department uses a proprietary GUI to create, browse, and edit customer information stored in a single database. The application is three-tier with a client submitting requests to an application server. The application server submits queries to the database server, organizes the returned information, and sends formatted text information back to the client for display. Of specific interest to this case study are searches performed to display a collection of records for clients responding to common criteria (e.g., all clients located in Virginia).

The application is programmed as follows:

1) The client submits a completed form to the application server.
2) The application server processes the form and generates one or more queries to the database server, one query at a time.
3) The database server responds to each query, and the application server accepts and accumulates the returned records.
4) The application server organizes the records and generates content which is sent to the client.
5) The client parses the content and displays a form with appropriate information in each field.

The custom application is the most applicable model for the following reasons: (1) there is no corresponding standard application; (2) there are more than two tiers; and, (3) on each of its tiers, the application consists of a series of blocking (i.e., serial) requests, followed by responses.

General Data Traffic from a LAN

A number of users on a LAN segment generate a variety of data traffic, i.e., they transfer documents and images to and from one or more servers. The rate and size of these documents can be measured using a protocol analyzer. A recommended approach to doing this is to use the analyzer to monitor the behavior of typical users over a sufficient period of time. Relative to capturing all of the users' traffic, this method reduces the volume of data to analyze.

Since the application model is being used to represent the traffic of an entire LAN, the various documents and/or images are transferred independently. These can occur at any time, and without blocking for each other's completion. The Custom Application corresponds well to this behavior. Furthermore, the Custom Application can be scaled easily to represent many similar users, all users adhering to the same profile while acting independently.

Custom Applications Workflow

When modeling custom applications, such as web-based business, the steps are a little more complex.

1) Define the Application and all of the steps in the transactions.

2) Define the Phases of a Task.

3) Configure the Tasks in the Task Config node.

4) Configure the custom application, as described in Configuring Individual Applications for Application Modeling, selecting “Custom,” instead of any of the standard applications, and listing the tasks that constitute this application.

5) Define user profiles, as described in Configuring the Profiles.

6) Run a simulation.

Define the Application

Before you can model an application, you must first understand the way the application works and how the transactions take place. The information you collect during this step helps you configure the tasks in your custom application model. For breaking down the application, use the following process:

1) Identify the Tiers

2) Identify the Independent Tasks

3) Identify the Task Phases

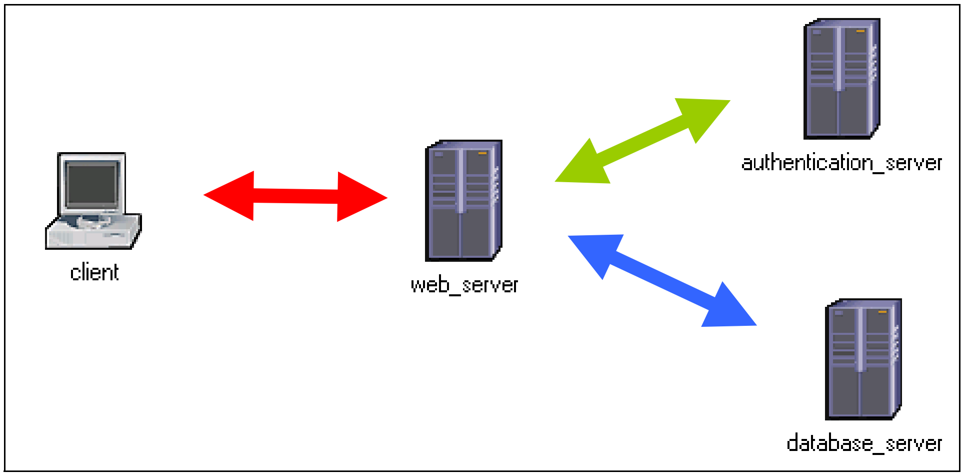
4) Breakdown Each Phase

5) Determine the Parameters

Identify the Tiers

Identify the tiers of the application you wish to model. This means identifying the entities involved in the application. In the following figure, the tiers involved include the client, the web server, the authentication server, and the database server.

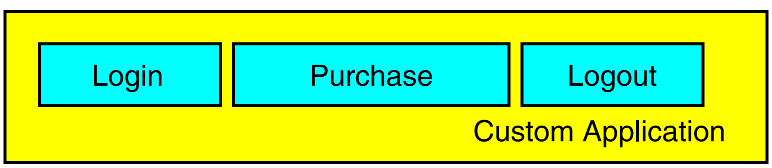
Figure 34 Application Tiers



Identify the Independent Tasks

The tasks in an application include all steps involved in the transactions. This can include, as in the case of an E-commerce application, a login, a purchase, and a logout. Some applications are much more complex. Be sure to consider all of the steps in the transactions you wish to model.

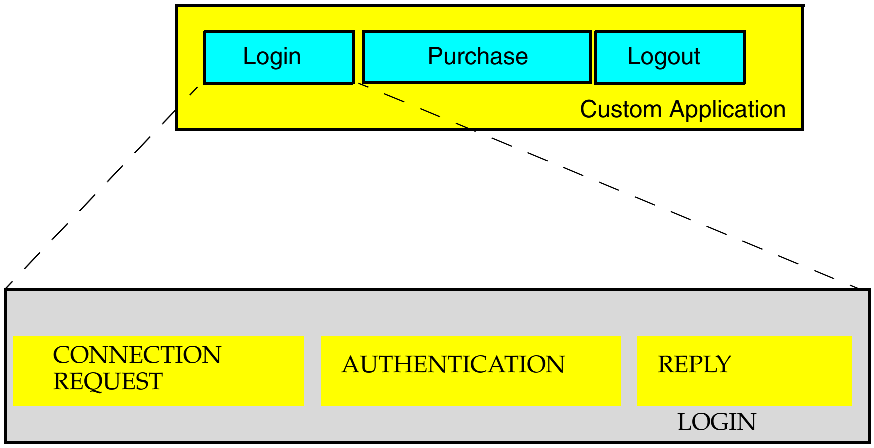
Figure 35 Tasks in an Application



Identify the Task Phases

Each task comprises phases. In the following figure, the task "LOGIN" may include a connect message from the client, an authentication to the authentication server, and a reply from the authentication server, either granting or denying access to the client.

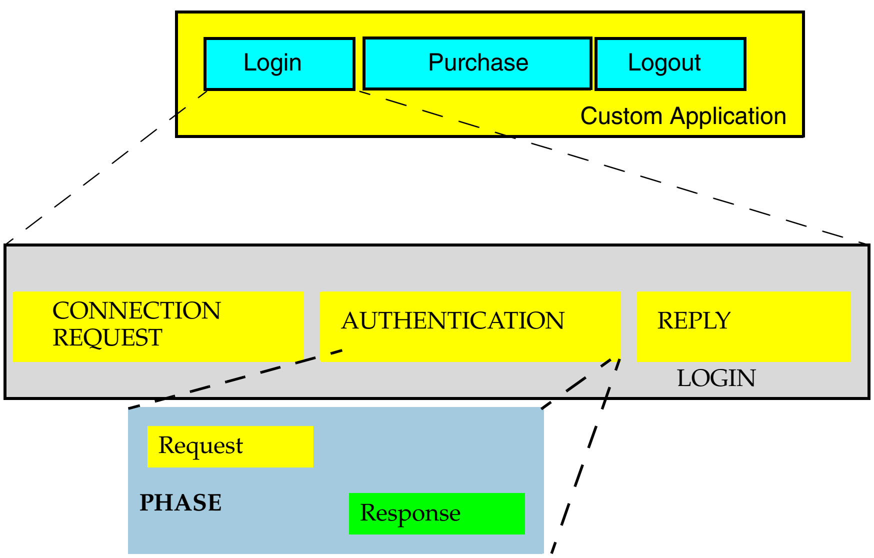
Figure 36 Phases in a Task



Breakdown Each Phase

As a task is made of phases, each phase is made of a series of requests and responses. Before you can completely model a complex application, you must have a good understanding of the request-response patterns and the sequence in which each occurs.

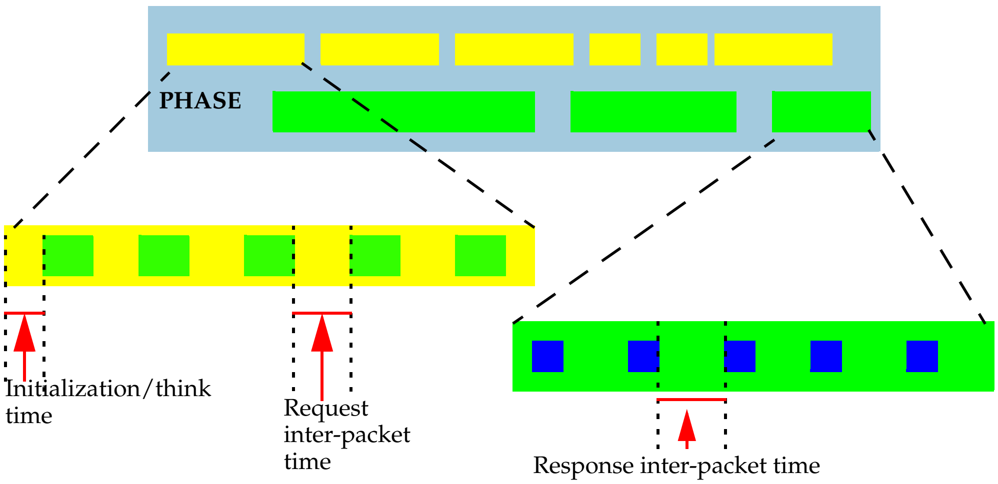
Figure 37 Request-Response Patterns in a Phase



Determine the Parameters

When you have determined the request-response pattern, you must next determine the specific parameters of the patterns. For example, is there a wait time between the processing of requests? Are there time-out parameters to consider? The following figure shows the parameters of an example pattern.

Figure 38 Parameters of Request-Response Patterns



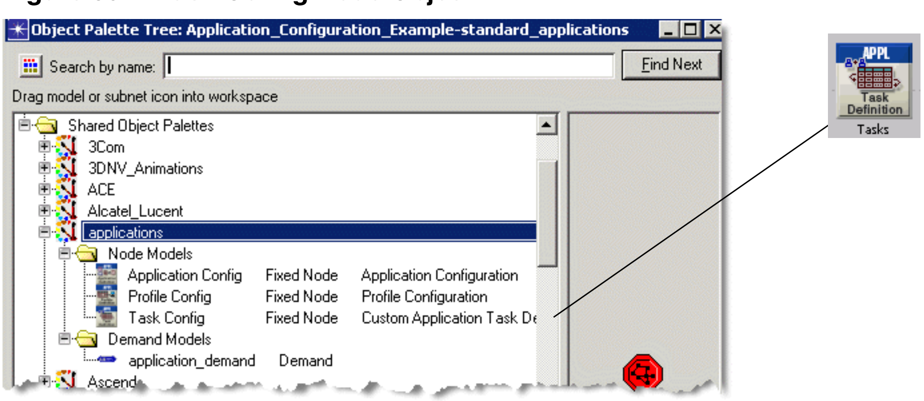
When you have finished breaking down the application you want to model, continue on to configure the tasks.

Understanding Tasks and Phases

The Custom Application makes use of "global objects" in its configuration, providing two advantages:

- It provides a single place in which to specify information that can be shared by many objects within the network model.
- Large amounts of configuration information can be copied from network to network simply by copying and pasting global objects.

Figure 39 Task Config Node Object



Custom applications are configured using two global objects: the Application Config object and the Task Config object. Tasks function as the building blocks of the custom application and are defined in the Task Config object. Once the tasks have been defined, they are used to build (define) the custom application in the Application Config object. Once defined, the custom application can be used in the profiles that are deployed to the individual workstations or LANs using the application.

Phases of a Task

Before configuring the tasks, described in Procedure 7, you must understand how to configure the phases of each task. The previous section described how to break an application into tasks and a task into phases. Data transfer phases consist of network activity and processing. Each data transfer includes the exchange of many application data units from a source address to a destination address. An exchange involves a sequence of requests (from the source to the destination) and responses (in the other direction).

Figure 40 Serial and Concurrent Request/Response Patterns

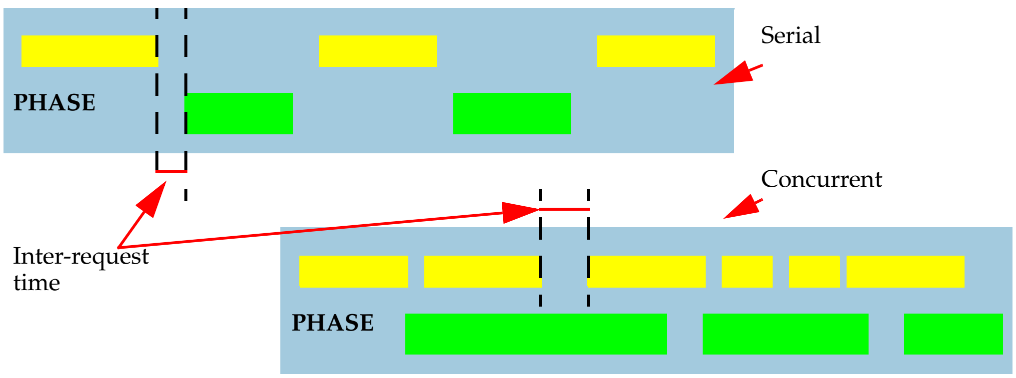
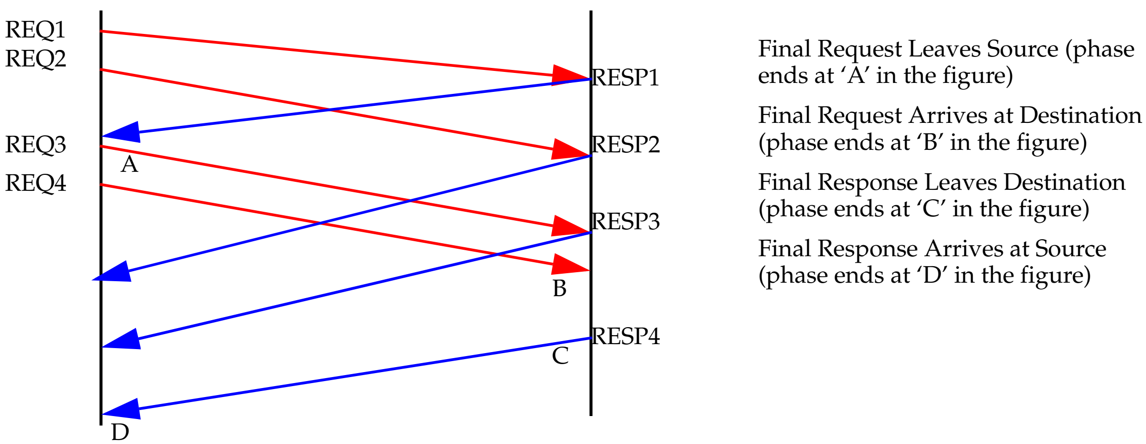
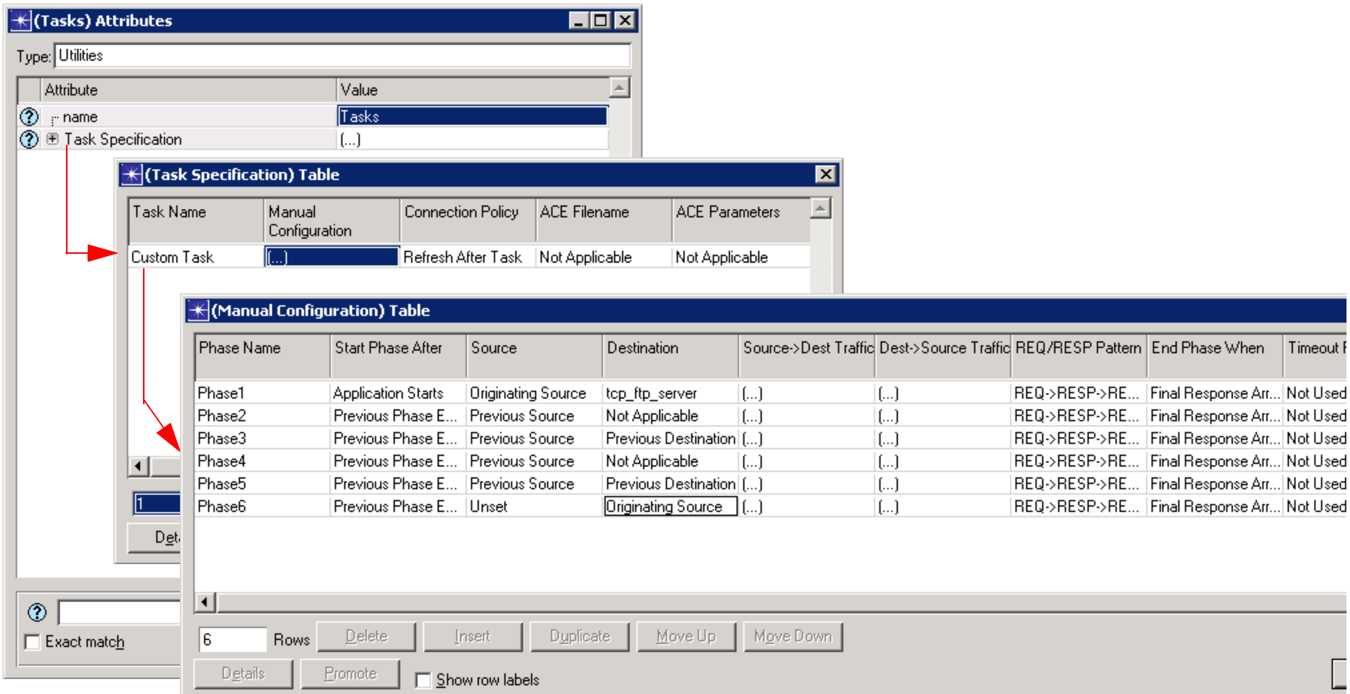


Figure 41 Specify When Phase Ends



The following figure shows a completed Manual Configuration Table for a task with 6 phases. Note that the second and fourth phases, Transaction #2 and Transaction #4, are processing phases and have the Destination attribute set to Not Applicable. With the exception of the first phase, each phase begins after the final request from the previous phase arrives at its destination.

Figure 42 Phase Configuration



Configure the Tasks

Procedure 7 describes the steps involved in manually configuring tasks. You can also use Transaction Analyzer to record the task details, and then specify the Transaction Analyzer filename in the Task Specification Table.

Each phase you configure in a task has the following attributes, shown in Figure 42 and listed in the following table.

Table 13 Task Phases

Field Name	Description	Usage
Phase Name	Specifies a user-defined name that can also be associated with a statistical group name	The statistical group name is optional but can be used to associate multiple phases with a statistical group to measure the time it takes to complete each phase. Results are available after a simulation, as with any DES results.
Start Phase After	Specifies when a phase begins	The start of a phase can be • Independent—The source does not depend on the completion of another phase, so you can set "start phase after" to "Application Starts". • "Previous Phase Ends" is used to specify a sequential phase execution. Each phase starts after the one before it ends. • "*Phase name*"—Set the field to a specific phase, selectable from the drop-down menu, to indicate that this phase begins when the named phase ends.
Source	Specifies the symbolic name of the node that acts as a source for the corresponding phase	• Originating Source—The phase starts on the node that supports the profile containing this custom application. • Previous Source—The node that was the source of the previous phase acts as the source. • Previous Destination—The previous phase's destination acts as the source for this phase.
Destination	Specifies the symbolic name of the node that acts as a destination for the corresponding phase	Similar to "Source" attribute: • Originating Source • Previous Source • Previous Destination • Not Applicable (for processing phases) If the Destination attribute for a processing phase is denoted as "Not Applicable", then the Dest -> Source Traffic attribute is ignored. If the Destination attribute for a processing phase is set to a multicast address, the Request Processing Time attribute in the Dest -> Source Traffic attribute is used, but the traffic information is ignored and a DES log entry is made.

Table 13 Task Phases (Continued)

Field Name	Description	Usage
Source -> Dest Traffic	Specifies the parameters for data transfer from the source to the destination. You may also set processing delays here.	• Initialization Time—Time spent in the beginning of the phase, before any packets are sent out. If no requests are sent, then this is a processing phase. • Request Count—The total number of requests from source to destination. • Interrequest Time (seconds)—The amount of time between requests. • Request Packet Size (bytes)—Specifies the number of bytes in each packet. • Packets Per Request—The number of packets sent for each request. • Interpacket Time (seconds)—Specifies the number of seconds between packets within each request.
Dest -> Source Traffic	Specifies the parameters for data transfer from the destination to the source. You may also set processing delays here.	• Request Processing Time (seconds)—Time to process a request before sending a response. • Response Packet Size (bytes)—Total size of a response packet. • Packets Per Response—Each response may contain multiple packets. Use this field to specify the usual number. • Interpacket Time (seconds)—Explicit time that server will spend for each response packet. If this is computed at the server: set to 'Use Server CPU'.
REQ/RESP Pattern	Specifies the scheduling of request/response sequences (see Figure 40).	Serial—Next request is not sent out until the response for the previous request arrives. Concurrent—Requests are sent out independent of arriving responses.
End Phase When	Specifies the condition which officially marks the end of this phase	See Figure 41.
Timeout Properties	Specifies the time limit on the duration of the phase and is useful for recovery from lost packets or aborted connections	• Set a number of seconds after which the phase completes, whether or not all transactions are complete within the phase. • After timeout, execute either next phase or next task.
Transport Connection	Specifies transport connection properties, such as the connection policy and number of total allowable connections, and the amount of buffer for application messages	Supported transport protocols: • TCP • UDP • ATM • FC • X.25 You can specify the type of transport, port number, and whether or not to reuse the connection or establish a new one.

The example below shows three tasks. Two, “Login” and “Logout”, have been manually configured, while the third, “Oracle_Three_Tier”, uses imported data. We will not discuss importing traces and Transaction Analyzer in this section; detailed information on Transaction Analyzer is available in the Transaction Analyzer documentation.

Procedure 7   Configuring Tasks

1. Open the Utilities object palette, and drag a Task Config object into the project workspace.

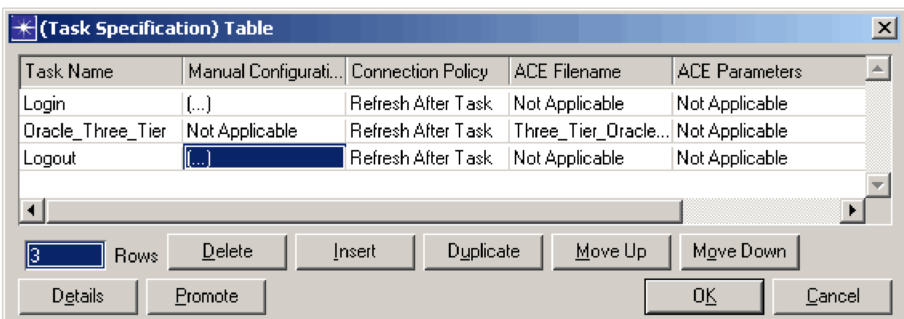
2. Right-click and choose Edit Attributes.

3. Select Edit… from the Task Specification drop-down menu.

4. Change the Rows value to the number of tasks you need in the Task Specification box.

5. List each task in a separate row of the table.

Figure 43   Task Specification Table



   5.1. To configure a task using imported data, simply list the file in the Transaction Analyzer Filename field.

   5.2. To manually configure a task, choose Edit… from the Manual Configuration field drop-down menu. In the Manual Configuration Table, change the Rows value to the number of phases in the task.

   Manually configuring a task involves detailing the task in its Manual Configuration Table where each row of the table represents a phase.

   For more information about the configuration of a phase, see Table 13.

6. Enter a name for each phase in the Phase Name column of the Manual Configuration Table.

7. Complete the table by specifying values for each attribute.

   For more information, see Table 13.

End of Procedure 7

Source Preferences on Nodes

The Application: Source Preferences attribute is an advanced attribute used to identify objects that act as the traffic initiating node for the first phase of a task. Typically, the source specified as part of the first phase of a task originates the task. For example, one node originates a task, such as Login.

When the Task Specification > Manual Configuration > Source attribute on the Task Config object is set to the symbol map “Originating Source”, the attribute resolves to the node on which the Profile is supported. In effect, you are identifying the node as the Originating Source.

For certain advanced applications, you may have more than one source that acts as an independent traffic initiating node. For example, if you have a remote node on which a given profile is supported, but which is not the traffic initiating node for the phase, you must set the Task Specification > Manual Configuration > Source attribute to a unique name other than “Originating Source”. Then set the Application: Source Preferences attribute on the *actual* traffic initiating node to the symbolic name you identified in the Task Config object.

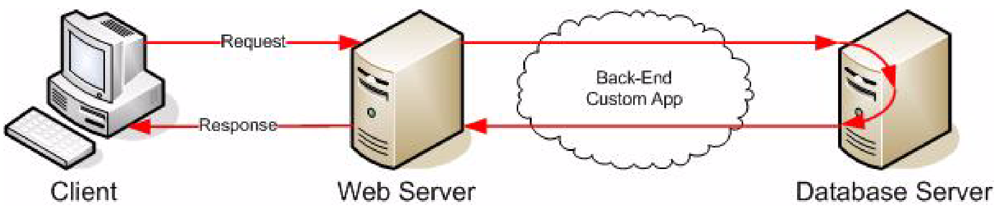
Note: If the Start Application When attribute in a particular phase is set to “Application Starts”, then the Source configured in this phase is a starter node.

Back-End Custom Application

Some custom applications represent not only a multi-tiered network but also multiple applications. As shown in the following figure, a modeled web application may involve a request arriving at the web server. Upon receiving the request, the web server triggers a second request to the database server. After the database server responds to the web server, the web server sends a response to the client. This secondary request-response cycle is modeled with Back-End Custom Application.

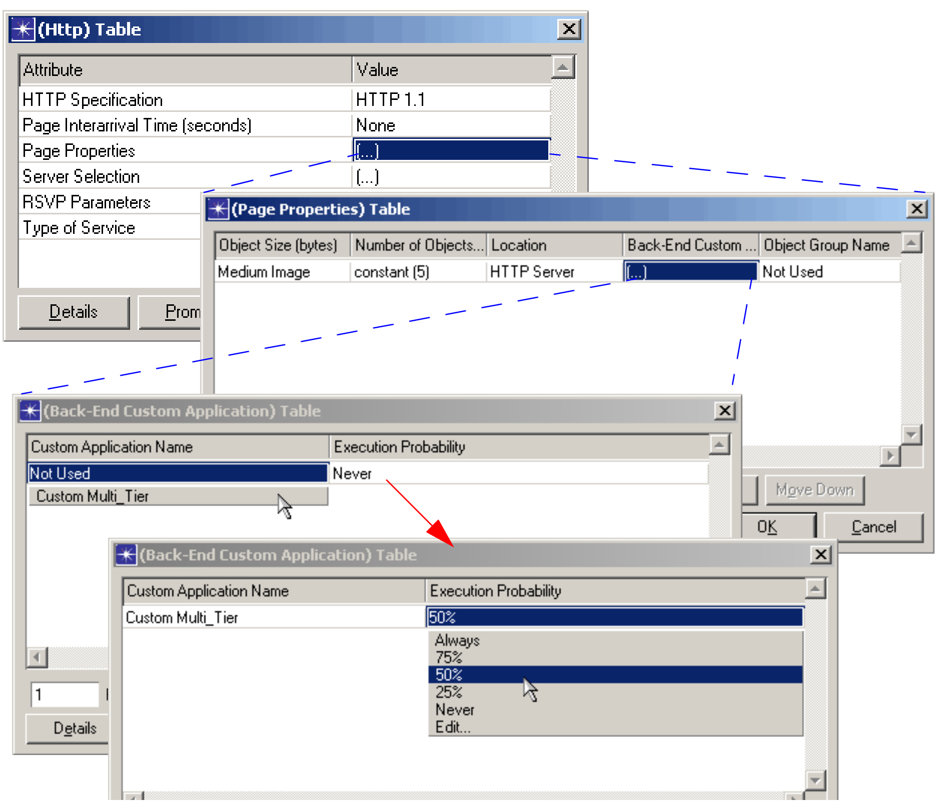
Figure 44 Using a Back-End Custom Application

Using back-end custom applications in a complex model



To configure a back-end custom application, you must define the custom application first. For the example shown above, you can then define your HTTP application, as shown, pointing to your custom application. The custom application is triggered after the HTTP model receives the request.

Figure 45 Configuring a Back-End Custom Application



Modeling Application Usage Patterns

The traffic pattern of an application is not the only determining factor for defining the traffic generated by the application. The following factors also influence the way you will model your traffic:

- How often the application is used
- Usage during each session
- Number of users
- Usage fluctuations

In considering how to model your traffic, take into account the different ways the users can use the application. For example, if you do a 1 MB file download, you will generate a smaller amount of traffic than if you do a 1 GB file download, even with the same FTP application. Consider the most common use cases when modeling the traffic.

Similarly, the duration and usage pattern of the application have a direct impact on the traffic generated. For example, if you browse the Internet several times a day, this is one usage pattern. If another person is constantly using Internet applications in the course of a day, that is a different usage pattern. The duration of use is also important. For the first example, you may model HTTP application usage for one hour, but for the second example, you may model eight hours of HTTP application usage.

In determining the usage pattern, you must also consider the number of users for a given application and whether they use the application concurrently or serially. For example, a sales office with 100 salespersons generates less traffic than does a sales office with 1,000 salespersons.

Traffic patterns tend to fluctuate over the course of a day. If you perform a trending analysis of your network, you may notice that your traffic is heavier early in the morning and after lunch than during the rest of the day. Knowing this information helps you accurately model your network traffic.

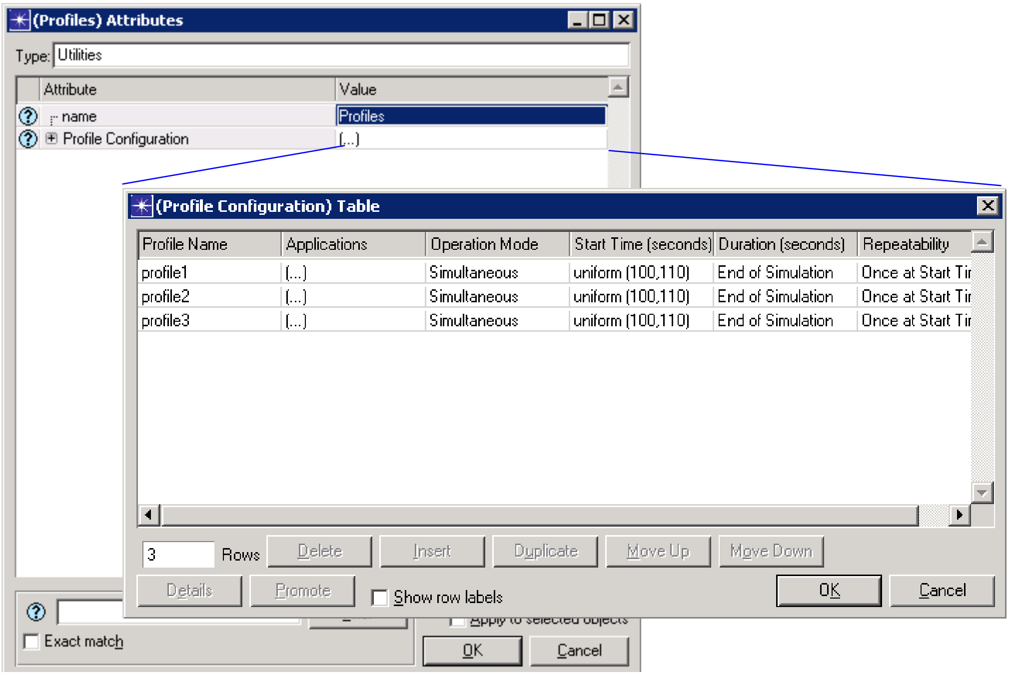
Building Profiles Based on Traffic Patterns

A profile captures the usage pattern of a set of applications used in your network. Essentially, this captures all of the applications in use, the time at which each application is started, the duration of use of an application, and the frequency of application use.

For example, if you build a profile to model the typical application usage of an engineer, you might include three applications: Email, HTTP, and FTP. A profile of a salesperson might include all three of these applications plus a sales database application. Each profile must specify when, how long, and how often each user type used these applications.

As with the application configuration object, you must configure profiles in a global object called a Profile Config node, shown in the following figure.

Figure 46 Configuring a Profile



The profile config node has attributes that let you configure timing, repeatability, and sequencing. These attributes are related, so changing one attribute impacts the others. Together the attributes determine the traffic pattern that is generated during the simulation. The following section explains these attributes.

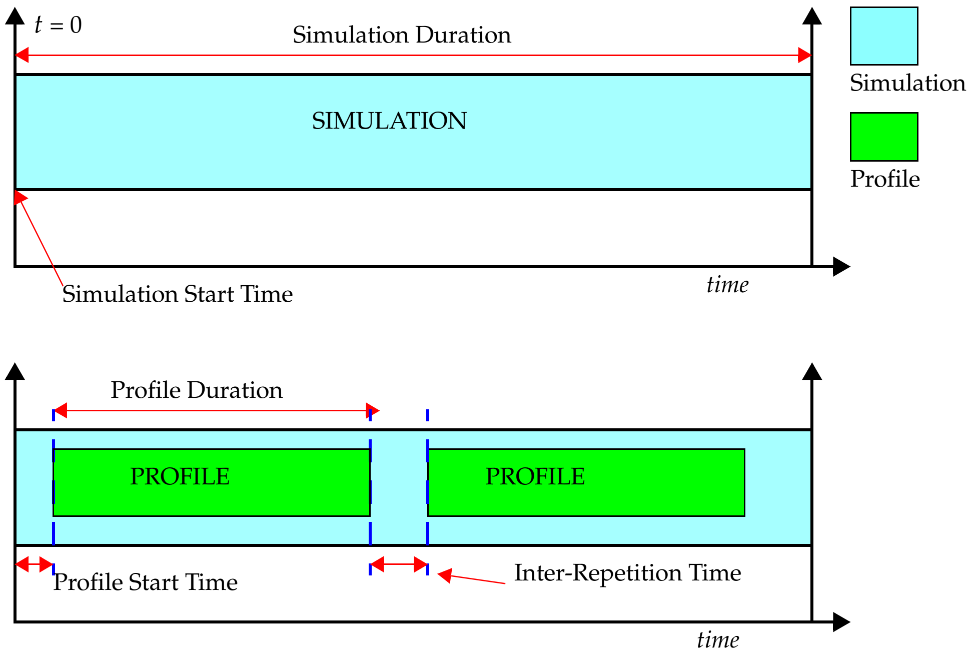
Understanding Timing, Repeatability, and Sequencing

This section describes the concepts of timing, repeatability, and sequencing in relation to applications and profiles within the Profile Definition utility node. Using the information you gathered about application usage and network traffic, you can configure your model accordingly.

- Application Timing describes the specific attributes related to configuring an application within a simulation profile.
- Profile Timing describes the specific attributes related to configuring a simulation profile.

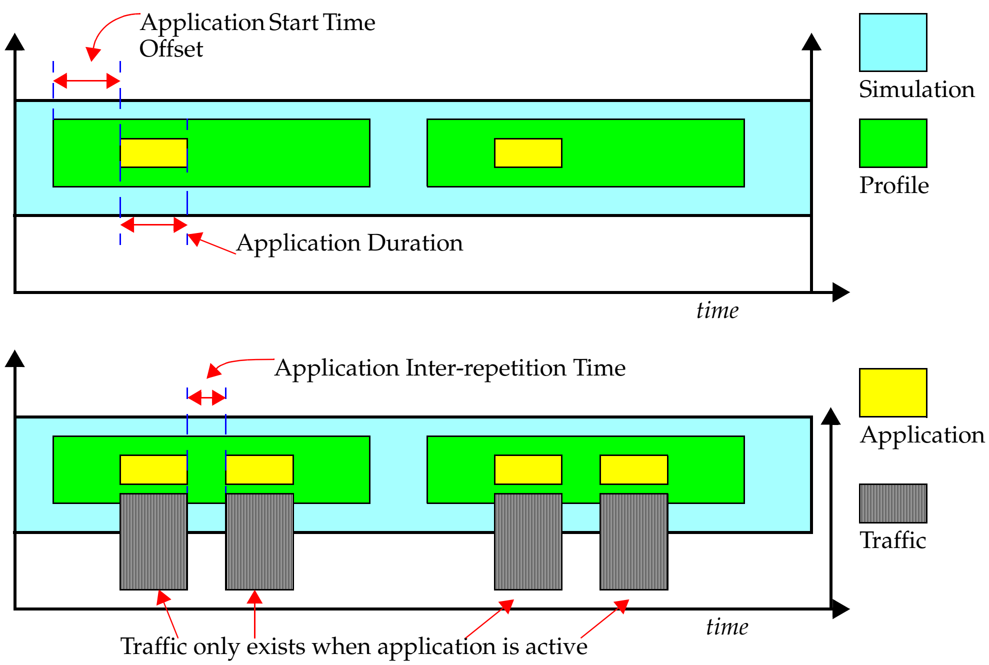
The following figure illustrates a simulation, with a start time and a certain duration. When you add profiles, notice the gap between the start of the simulation and the start of the profile. Each profile has its own duration, and there is a gap (inter-repetition time) between the repetitions of each profile.

Figure 47 Profile Time-Related Attributes



The following figure takes the illustration a step further. Within each profile, applications run. Each application has a start time offset, which determines when the application starts in relation to the profile start time. Each application has its own duration setting. Traffic is generated in the network model only when the application is active, therefore traffic duration equals application duration. The gap between repetitions of the application, if so configured, is the inter-repetition time.

Figure 48 Applications and Time-Related Attributes



The diagrams below depict possible profile configurations, in which the profiles can run serially or concurrently. Different profiles may run serially during a simulation, for example, Profile1, Profile2, Profile3, and so on.

Figure 49 Profiles Running Serially

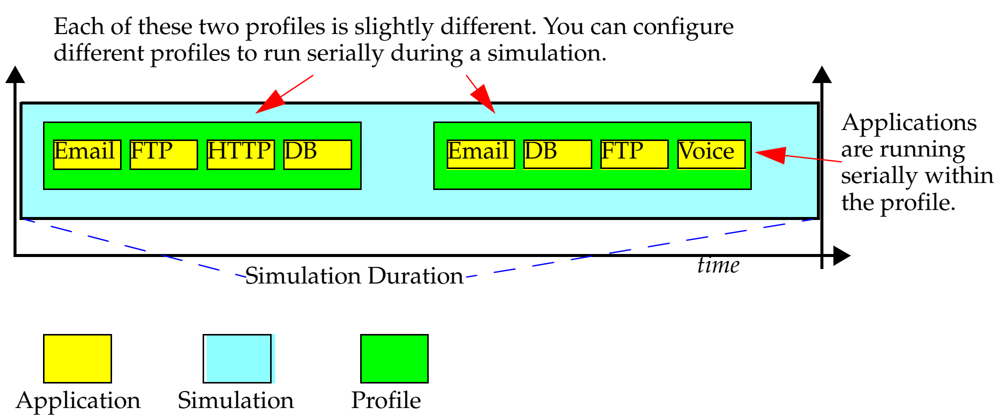


Figure 50 Profiles Running Simultaneously

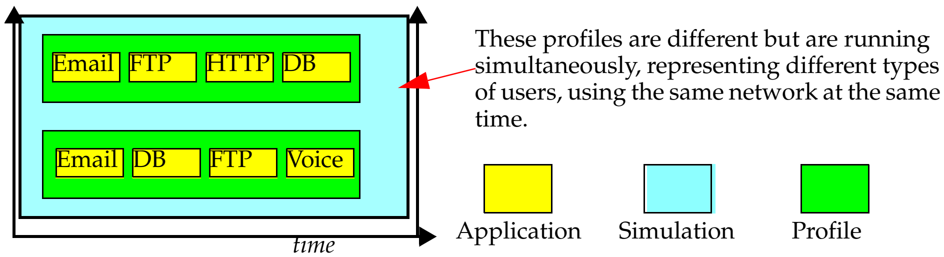
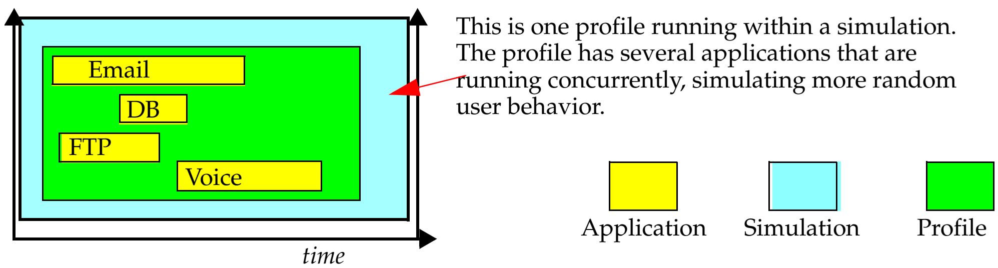


Figure 51 Applications Running Concurrently within Profile



Application Timing

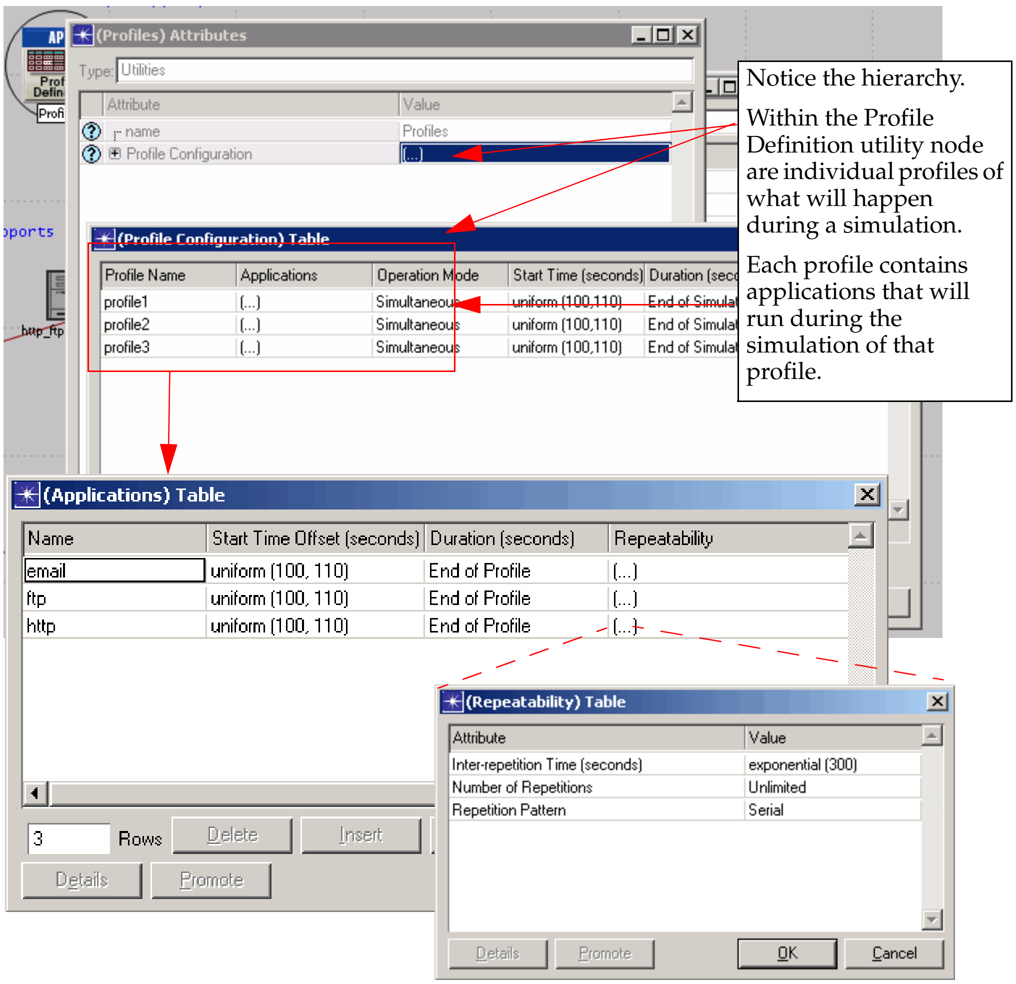
Applications contain many time-related attributes. These attributes determine such things as the start-stop times of the application and have an impact on how much traffic is generated. These time-related attributes include

Key Concept—Application timing is set per-application within individuals profiles in the Profile Definition utility node. Do not be confused by the use of the term “profile” within this section.

- Start Time Offset—This attribute has two interpretations based on the value specified for the “Operation Mode”.
  - If the “Operation Mode” is set to “Simultaneous”, this offset refers to the offset of the first instance of each application from the start of the profile.
  - If the “Operation Mode” is set to “Serial (Ordered)” or “Serial (Random)”, this offset refers to the time from the start of the profile to the start of the first application. It also serves as the inter-application time between the end of one application to the start of the next. If an application does not end (e.g., duration set to “End of Profile”), subsequent applications won't start.
- Duration—The maximum amount of time allowed for an application session before it aborts. This is often used as a timeout. When set to 'End of Profile', the application will end when the profile duration has expired. When set to 'End of Last Task', the application will end when the last task of the application has completed regardless of task completion times.

- Repeatability—This compound attribute specifies the method for repeating the profile:
  - Inter-Repetition Time—Defines when the next session of the profile starts. For applications running serially, this determines the gap between the completion of one application and the start of the next instance, one application running after the other completes. For applications running concurrently, this determines the gap between the start of one application and the start of the next.
  - Number of Repetitions—Specifies the distribution name and arguments to be used for generating a random number of times this profile is repeated.
  - Repetition Pattern—Defines when the next session of the profile will start.

Figure 52 Time-Related Attributes



Profile Timing

All time-related profile attributes in the Profile Config node have the following characteristics:

- All timing is relative to simulation start time (t = 0), either directly or indirectly.
- Applications may repeat within a profile.

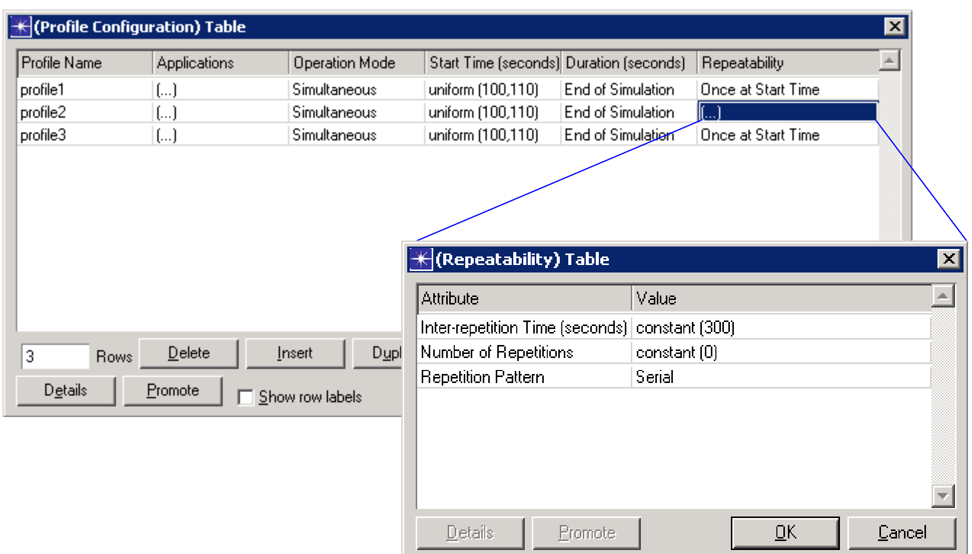
- Applications within a profile have an "Inter-Repetition" time. See Application Timing for more information.

Note: Review Figure 47, Figure 48, Figure 49, Figure 50, and Figure 51 to see how profiles and applications are related to each other and to simulations.

These time-related attributes determine such things as the start-stop times of the profile and have an impact on how much traffic is generated. These time-related attributes include

- Start Time—This is the time at which the profile starts during the simulation. Specify a pre-defined or scripted distribution.
- Duration—This field specifies the amount of time the profile is allowed to run during the simulation. You can choose to let it run until the end of the simulation or the end of the application duration, or select a distribution.
- Repeatability—This compound attribute specifies the method for repeating the profile:
  - Inter-Repetition Time—Defines when the next session of the profile starts. For profiles running serially, this determines the gap between the completion of one profile and the start of the next instance, one profile running after the other completes. For profiles running concurrently, this determines the gap between the start of one profile and the start of the next.
  - Number of Repetitions—Specifies the distribution name and arguments to be used for generating a random number of times this profile is repeated.
  - Repetition Pattern—Defines when the next session of the profile will start.

Figure 53 Time-Related Attributes

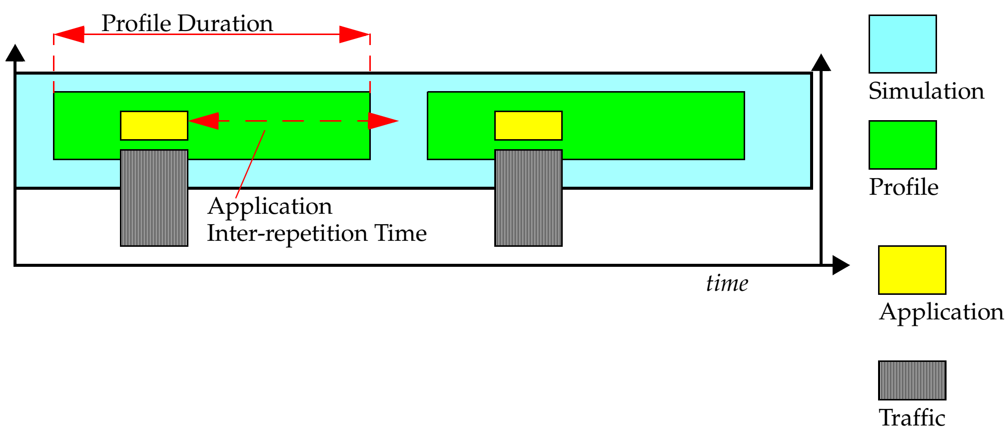


Understanding the Impact of Profile Misconfiguration

If a traffic profile is misconfigured, your simulation results will not be as expected. This section describes the potential impact of misconfigurations. The following three cases generate three significantly different amounts of traffic due to a single attribute configured differently.

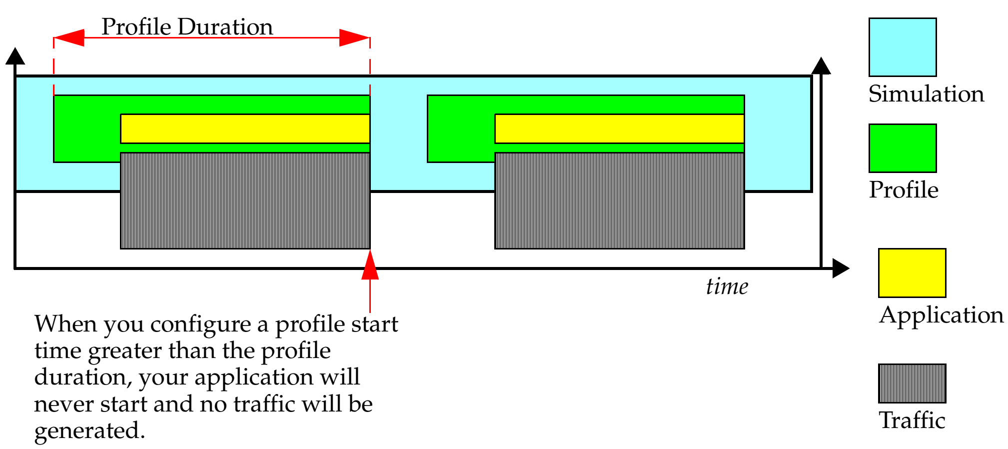
For example, if you configure the inter-repetition time for an application to a duration greater than the duration of the profile, your application will not repeat.

Figure 54 Applications and Time-Related Attributes



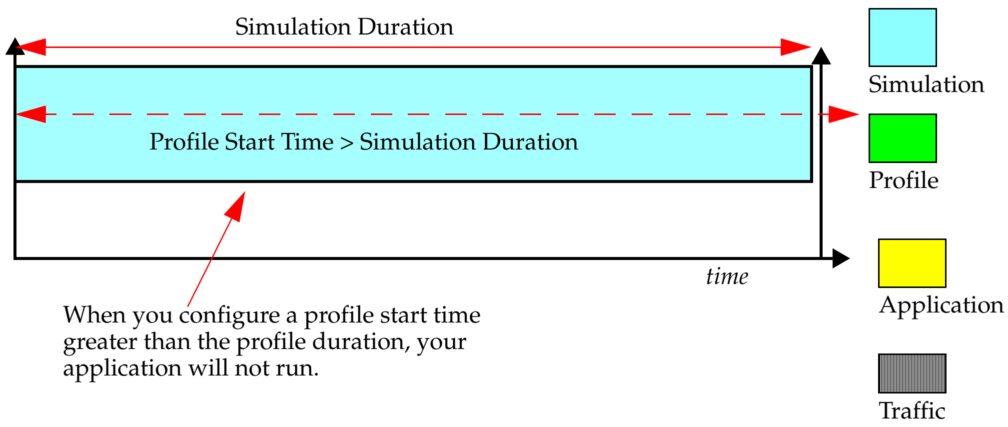
If you set the application duration to "End of Profile", your application will not repeat, since it will not end until the profile ends.

Figure 55 Applications and Time-Related Attributes



When setting the start time for the profile, be sure it is not greater than the duration of the simulation. For example, if the simulation runs 300 seconds, do not set a profile start time of 350.

Figure 56 Applications and Time-Related Attributes



Configuring the Profiles

Profiles describe the activity patterns of a user or group of users in terms of the applications used over a period of time. You can have several different profiles running on a given LAN or workstation. These profiles can represent different user groups. For example, you can have an Engineering profile, a Sales profile, and an Administration profile to depict typical applications used for each employee group.

Table 14 Profile Attributes

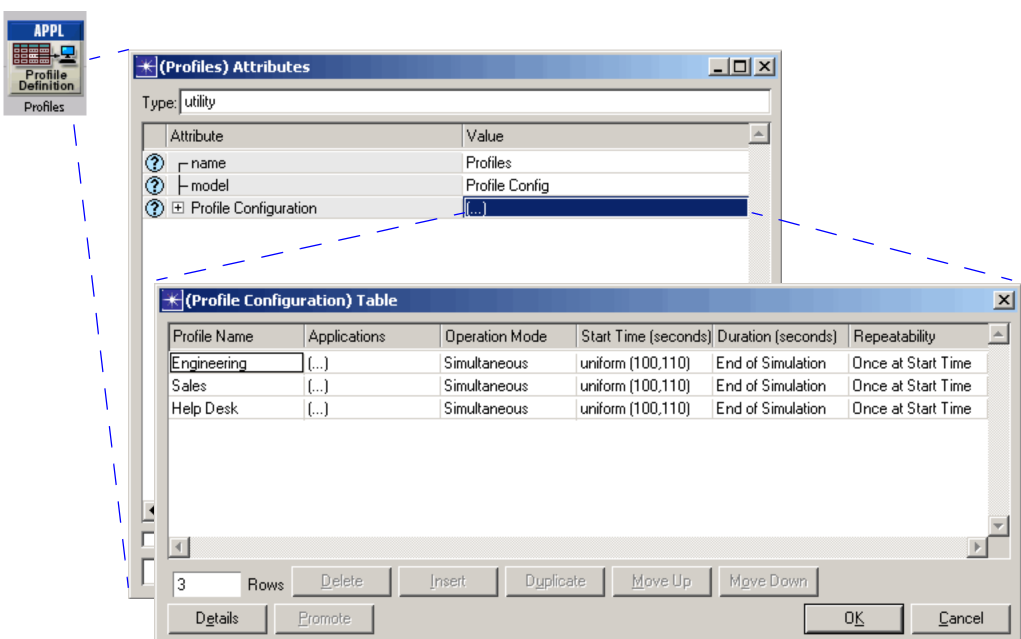
Attribute	Description
Profile Name	Specifies the unique name for this profile.
Applications	
Name	Specifies the name of the application configured in this row.
Start Time Offset	This attribute has two interpretations based on the value specified for the “Operation Mode”. • If the 'Operation Mode” is set to “Simultaneous”, this offset refers to the offset of the first instance of each application (defined in the profile), from the start of the profile. • If the “Operation Mode” is set to “Serial (Ordered)” or “Serial (Random)”, this offset refers to the time from the start of the profile to the start of the first application. It also serves as the inter-application time between the end of one application to the start of the next. If an application does not end (e.g., duration set to “End of Profile”), subsequent applications won't start. • For the special “scripted” distribution, specify a filename (*.CSV or *.gdf) containing the values required as outcomes. Values will be picked from this file in cyclic order.
Duration	The maximum amount of time allowed for an application session before it aborts. This is often used as a timeout. • When set to “End of Profile”, the application will end when the profile duration has expired. • When set to “End of Last Task”, the application will end when the last task of the application has completed regardless of task completion times. • For the special “scripted” distribution, specify a filename (*.csv or *.gdf) containing the values required as outcomes. Values will be picked from this file in cyclic order.
Repeatability	See Understanding Timing, Repeatability, and Sequencing.
Inter-repetition Time (seconds)	Defines when the next session of the application will start depending on the Repetition Pattern.
Number of Repetitions	Specifies the distribution name and arguments to be used for generating random session counts. “Unlimited” will allow new sessions to be created up until the “Application Stop Time” specification.
Repetition Pattern	Specifies the pattern in which this application repeats, serially or concurrently.

Table 14 Profile Attributes (Continued)

Attribute	Description
Operation Mode	Defines how applications will start. • Serial (Ordered)—They can start one after another in a ordered manner (first row to last row). • Serial (Random)—They can start one after another in a random manner. • Simultaneous—They can start all at the same time.
Start Time (seconds)	Defines when during the simulation the profile session will start.
Duration (seconds)	Defines the maximum amount of time allowed for the profile before it ends. • When set to "End of Simulation," the profile is allowed to continue indefinitely until the simulation ends. • When set to "End of Last Application," the profile is allowed to continue until the last instance of an application running as part of this profile ends. If the application repeatability is unlimited, the profile will end when the simulation ends. • Profiles that repeat should not have their duration set to "End of Simulation".
Repeatability	Specifies the parameters used to repeat execution of this profile. For more information, see Understanding Timing, Repeatability, and Sequencing.

Profiles can execute repeatedly on the same node. Riverbed Modeler enables you to configure profile repetitions to run concurrently (at the same time) or serially (one after the other). In the following figure, three different profiles are configured on the Profile Config node. Each profile contains one or more applications.

Figure 57 Profile Configuration



Profiles contain a list of applications. You can configure the applications within a profile to execute in any of the following manners, relative to the operation mode of the application itself:

- at the same time
- one after another—in a specific order you determine
- one after another—in a random order

In most cases, when describing the actions of a single user, the actions are serial because most people can only perform one activity at a time. However, when using applications that can perform non-blocking tasks, you can have more than one task running at a time. When describing the activities of a group of users, concurrency is common. Like profile repetitions, application repetitions within the profile can execute either concurrently or serially.

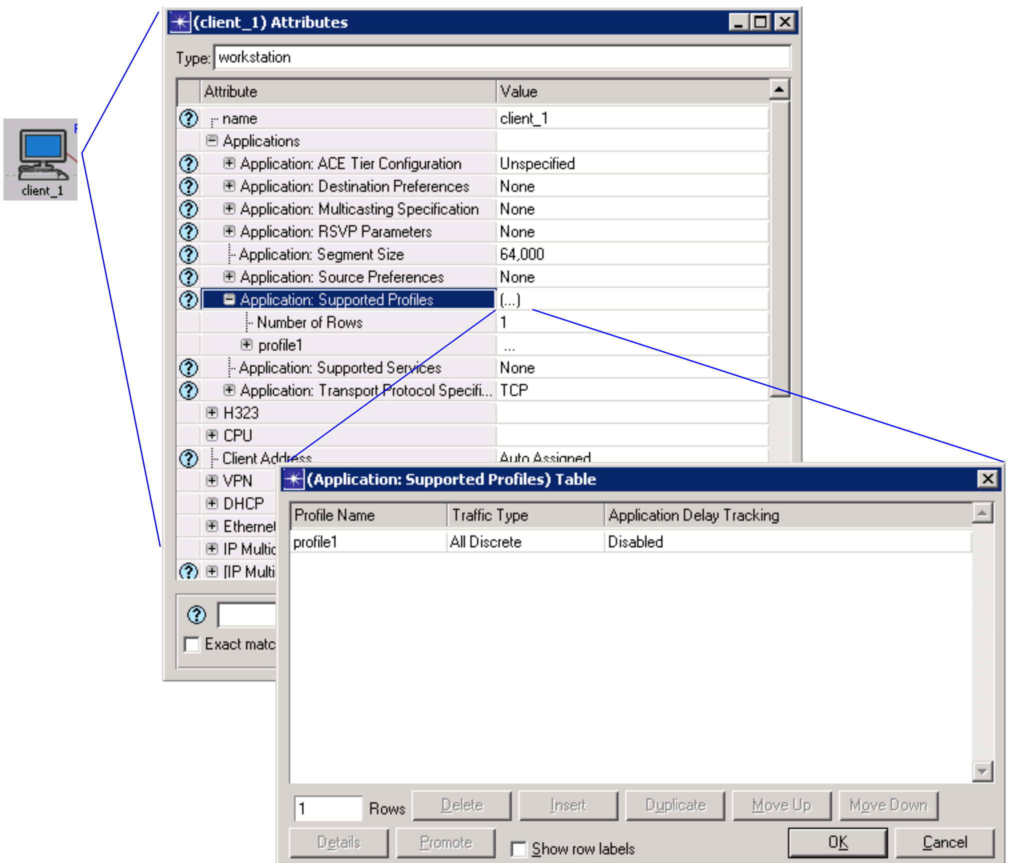
The Profile Definition Object defines all profiles that can be used within a scenario. Only profiles that have been defined in the Profile Definition Object can be applied to the workstations or LANs of a project and only applications that have been defined in the Application Definition Object can be used in profile definitions.

Deploying the Applications

After you have configured your applications, profiles, and tasks, you are ready to deploy these profiles on individual workstations, servers, and LANs. Typically, profiles are specified on workstations or LANs, since they are traffic sources. However, a profile may be deployed on a server if the server acts as a source for any application task. This implies that if you have configured a task table such that the server is an independent source, you should ensure that you support that profile on the server object. See Custom Application for details on the custom application task configuration.

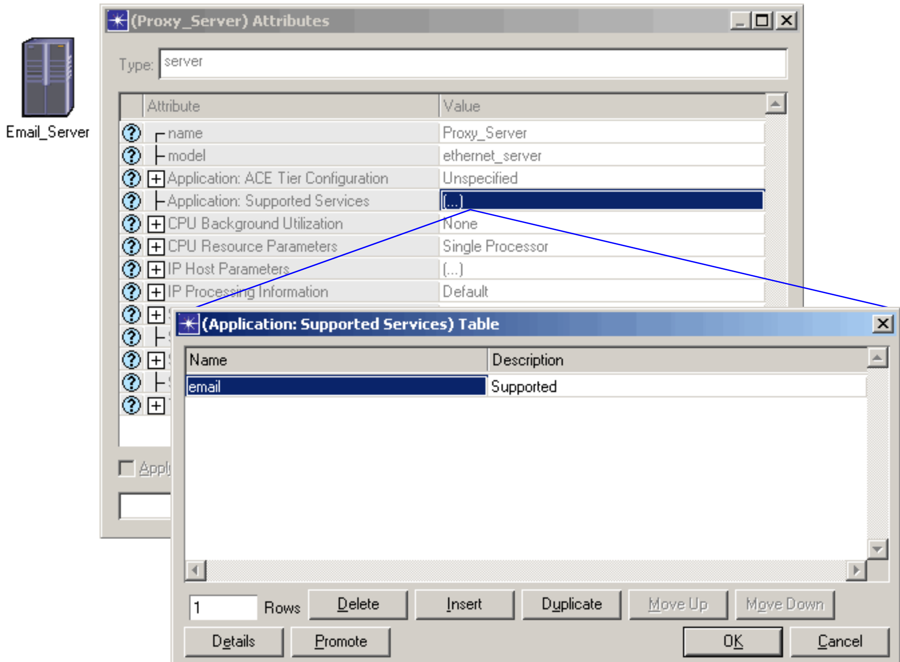
This section describes how to deploy applications and destination preferences, and how to add LAN nodes to your network model to represents clusters of workstations. Deploying applications with the Application Deployment wizard is an automated way to set node attributes. In the following figure, you can see that the "engineer" profile is deployed to the workstation. The wizard provides a way to deploy applications into many nodes in the model simultaneously.

Figure 58 Profile Deployed to a Workstation



In addition to deploying applications to workstations, you must deploy supported services to servers in your network model. For example, one server might support Email and HTTP services, while another server might support database operations. You can expedite this process by using the Application Deployment wizard.

Figure 59 Configuring Servers to Support Applications

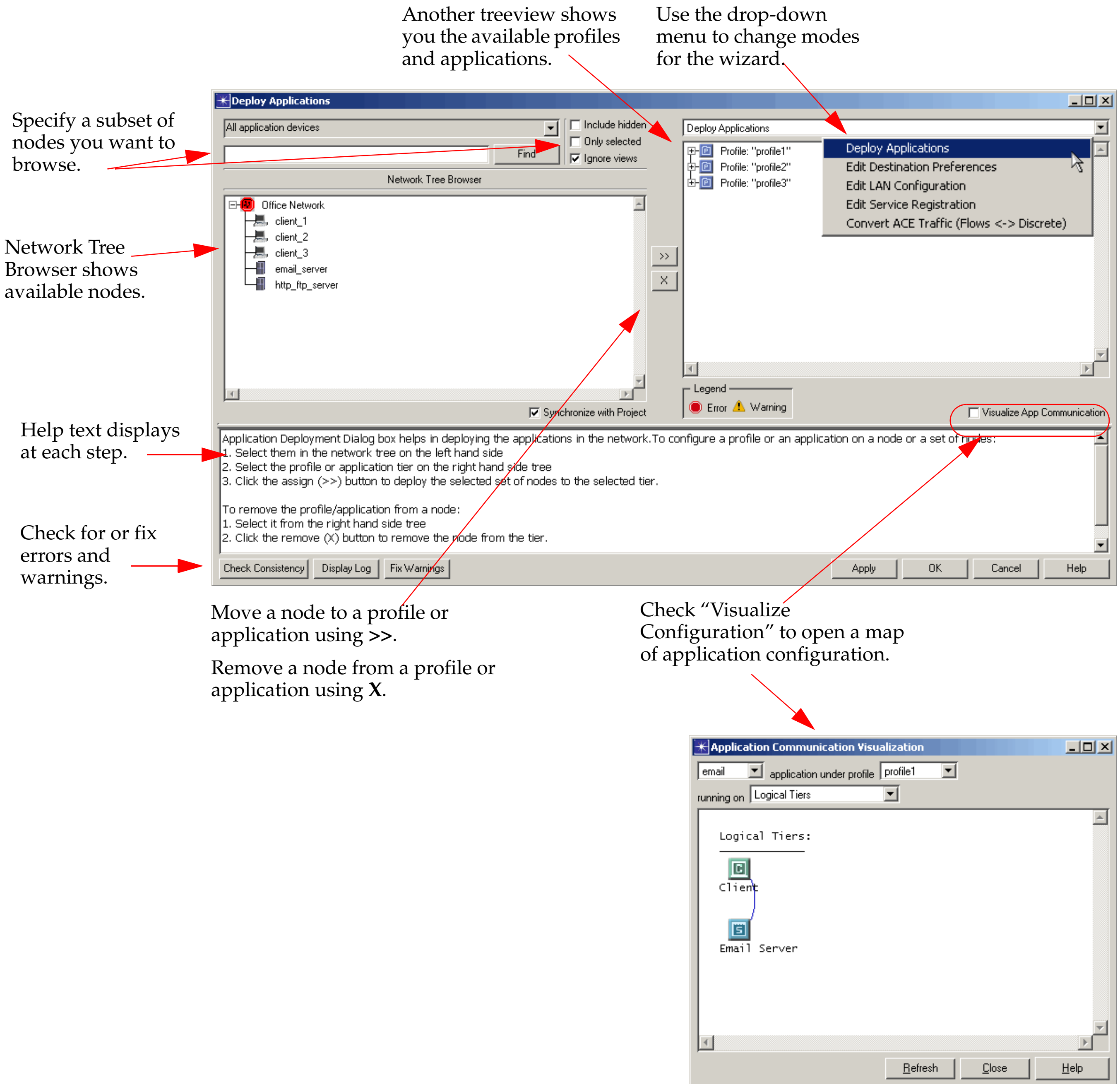


Application Deployment Wizard

The Application Deployment wizard automates the deployment of supported profiles and services to nodes in your network model. Using the Application Deployment wizard, you can

- view and modify the application deployment
- deploy profiles that are already defined in the profile configuration object
- populate the individual node attributes automatically

Figure 60 Application Deployment Wizard

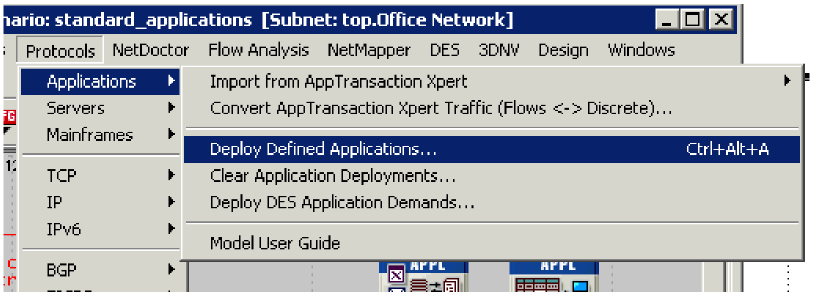


The following procedure describes how to use the Application Deployment wizard to deploy profiles quickly.

Procedure 8 Deploying Profiles with the Application Deployment Wizard

1. Open the Application Deployment wizard by choosing the menu selection Protocols > Applications > Deploy Defined Applications… or by the shortcut key combination Ctrl + Alt + A.

Figure 61 Open Application Deployment Wizard



2. Select the node(s) in the left treeview to which you wish to deploy a profile or application.

   Note: Remember to assign profiles to LAN nodes in your network also.

3. Select the profile or application tier in the right treeview.

4. Click the “>>” button to assign node(s) to the selected profile or application.

5. Check the consistency of the configuration, after you have assigned all nodes to the appropriate profiles, by pressing the Check Consistency button.

6. Correct any configuration errors identified during the consistency check.

7. Press the Fix Warnings button to automatically correct warning conditions.

8. Click Apply to apply the deployment to the network model, when you are satisfied with your configuration.

9. Click OK to leave the Application Deployment wizard.

End of Procedure 8

Servers are unique from workstations in many ways. Not only do they support a variety of services, they can also be made to model a wide variety of servers in your operational network. For example, if you have a dual processor system, you may want to increase the number of CPU resources in the server attributes. See the following procedure for more information.

Procedure 9 Configuring Specific Server Attributes

1. Right-click on the server node and Edit Attributes…

2. Set the Server Address attribute with the unique TPAL name.

3. Set CPU attributes.

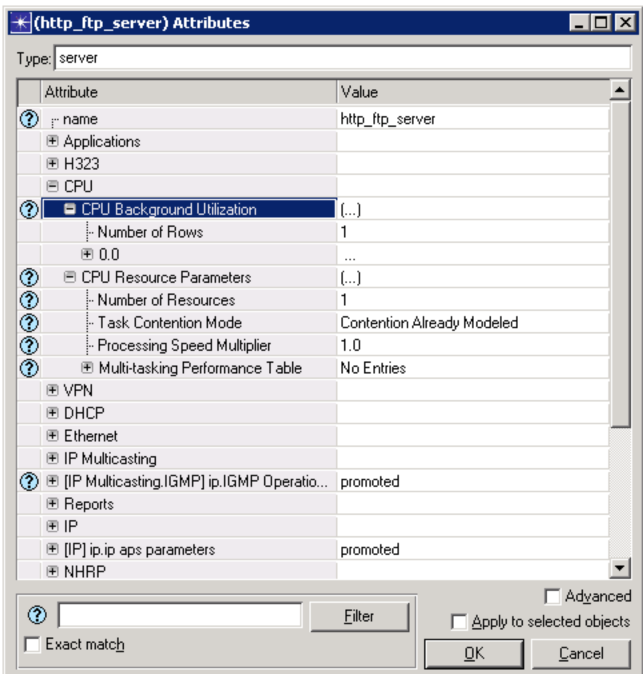
   3.1. Set CPU > CPU Resources Parameters.

"Number of Resources" and "Processing Speed Multiplier" both have a linear effect on the processing time of a job. For example, suppose by default "Number of Resources" = 1 and "Processing Speed Multiplier" = 1.0, completing a job in 5 seconds. If you increased "Number of Resources" to 2 and left "Processing Speed Multiplier" = 1.0, the same job will take 2.5 seconds to complete.

Another example: If "Number of Resources" = 1 and "Processing Speed Multiplier" is 2.0, the job will take 2.5 seconds to complete.

Note: Processing speed is based on the application in use. To find out the default processing speed for an application, expand Applications > Application: Supported Services and examine the Processing Speed (bytes/sec) attribute in the definition table for the specified application. The processing time required is based on the response size for FTP Get, Email Recv, Database Query application types and on the request sizes for the rest of the applications.

Figure 62 CPU Attributes



3.2. Set CPU > CPU Background Utilization. Add as many rows as are necessary to specify utilization percentages at a variety of start times within the simulation.

The CPU background utilization determines how much of the CPU processing power is available for the jobs. For example, if, in the absence of any CPU background utilization (0%), a job took 10 seconds to complete, then if the CPU background utilization is 50%, the same job will take 20 sec to complete (since it gets only 50% of CPU). If the CPU background utilization is 75%, the same job will take 40 sec to complete (since it gets only 25% of CPU).

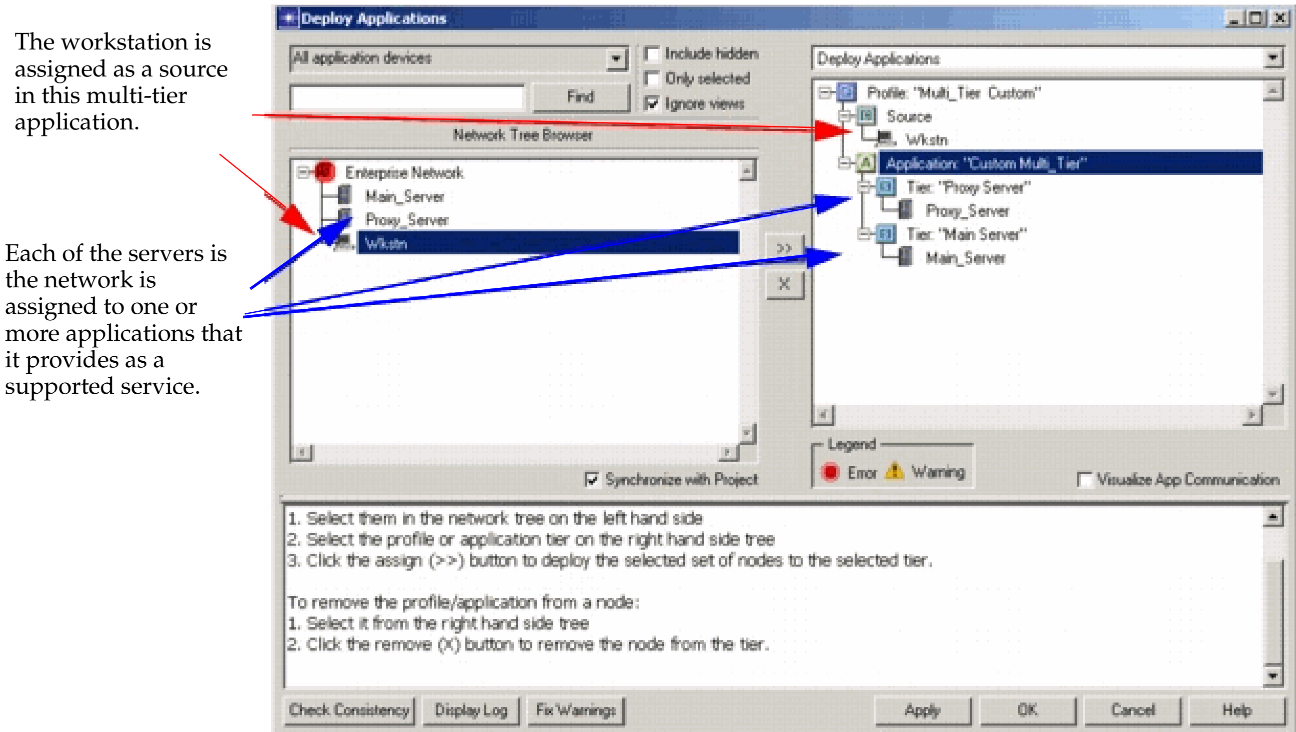
4. Set Task Contention Mode.

End of Procedure 9

Deployment Example

As described in Procedure 8, you can easily assign applications and profiles to any node in the network model. Here is a practical example of how the Application Deployment wizard is used:

Figure 63 Example of Custom Application Deployment



In the multi-tier custom application shown here, each layer of the profile represents a different level of communication. The workstation is the source of the application—when a user presses Enter to send a database request, for example. In the application itself, there are two servers: a proxy server and a main server. “Tier: Proxy Server” represents a layer of the custom application. In Figure 63 the node “Proxy_Server” is assigned to the tier so that our source will talk to this server at this level in the application.

Edit Destination Preferences

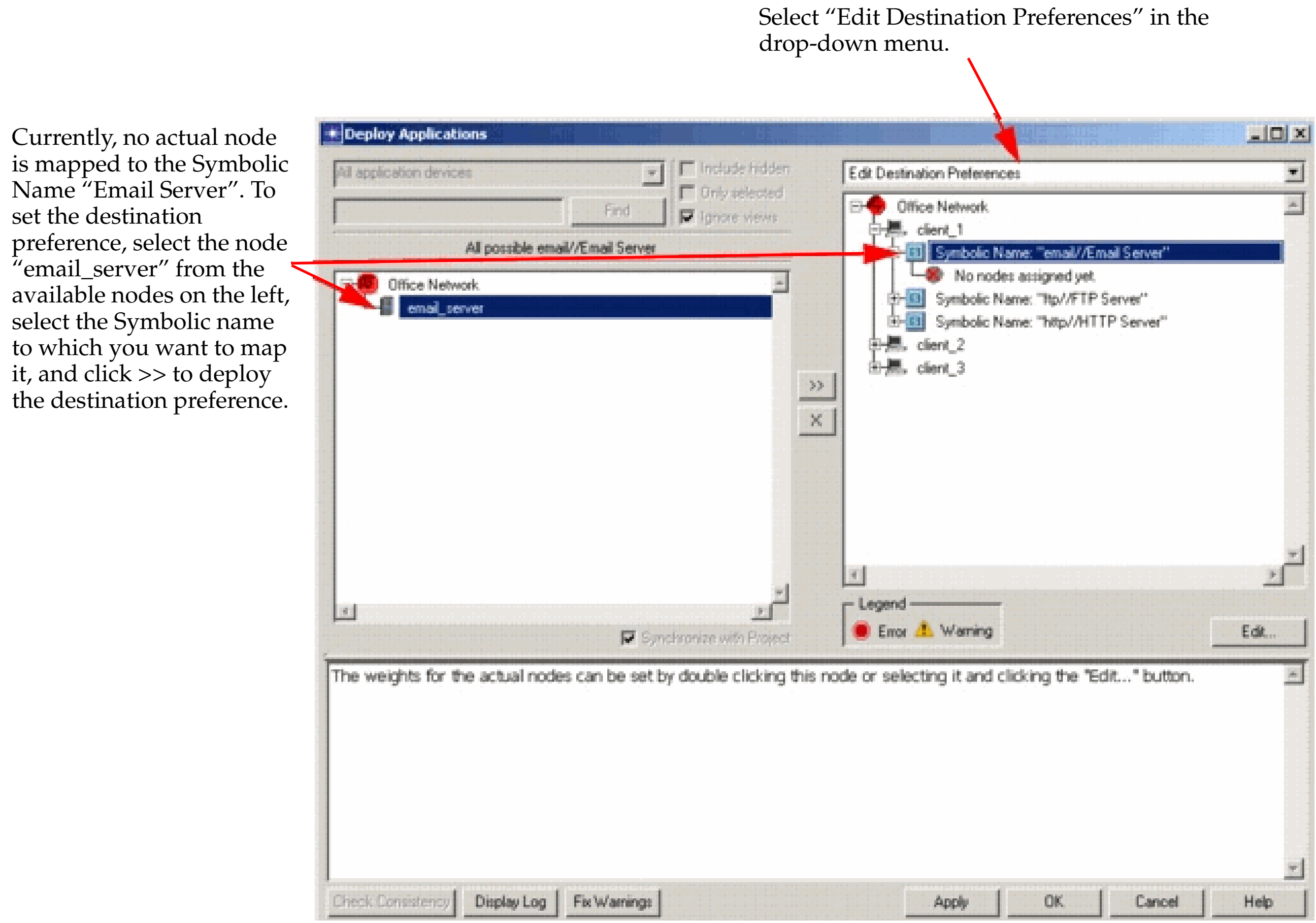
Each application uses a symbolic name to refer to a server. For example, if you have set the symbolic server name of the E-mail application to “Corporate_Email_Server”, you must resolve this reference so that the symbolic name refers to an actual server object on the network. The advantage of using symbolic names is that you can define an application once with a symbolic name. You can then resolve the symbolic name to different actual names on different workstations.

For example, you can resolve the Corporate_Email_Server described above as Server_192 for all engineers and as Server_198 for all sales people. When you set destination preferences, you assign a node to a symbolic name and set a weight value for its preferability to traffic.

Note: If you do not set any destination preferences, the server will be selected at random from the servers that support the application of interest.

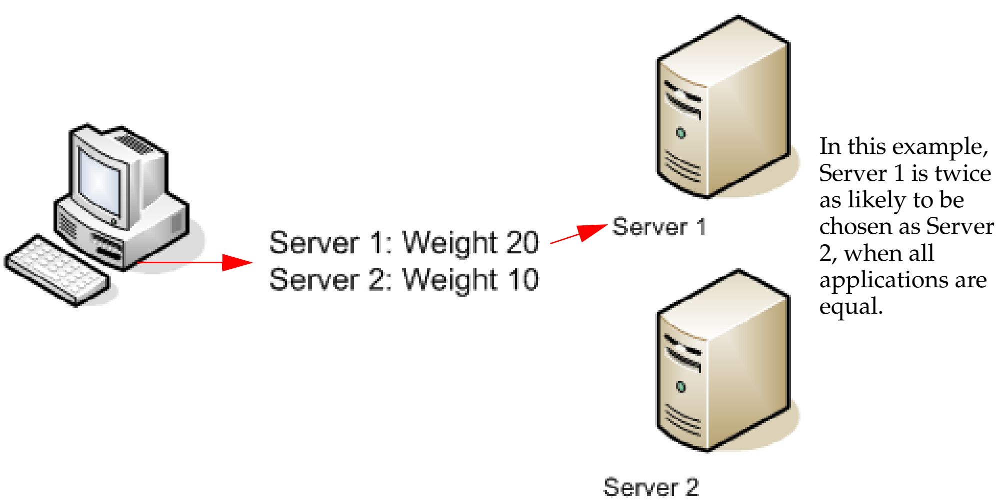
This is another task you can perform with the Application Deployment wizard.

Figure 64 Setting the Destination Preference with the Application Deployment Wizard



The default weight of a destination preference is 10. The higher the number, the greater the likelihood the server will be chosen during a simulation.

Figure 65 Effect of Server Weight on Destination Preference



For example, if you had five Email servers in your network model, you would map all five to the symbolic name "email/Email Server" in the example shown in Figure 64. Each node would be assigned a weight of 10, automatically. You can click on the "Edit…" button to set the weight to any number. If you want the traffic to use a particular server, set that preference to a higher weight than on other nodes.

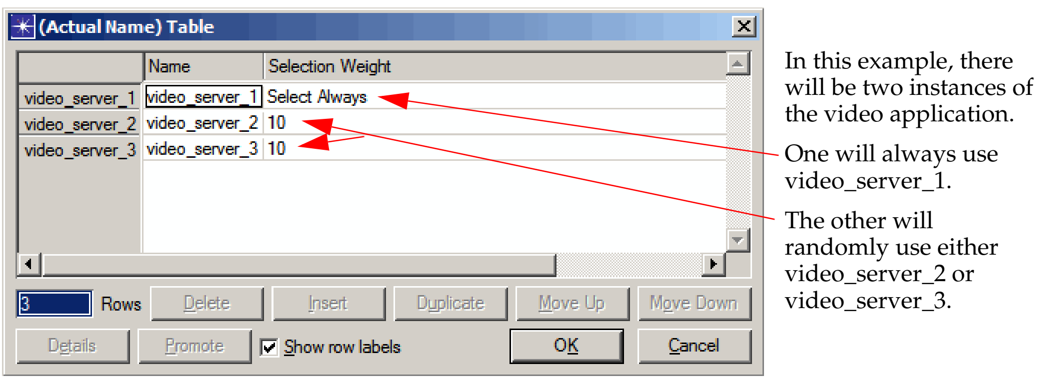
Using Multiple Servers for an Application

The usual method of setting Destination Preferences assigns only one server (chosen randomly) to an application. You might want to assign several servers, for example, to increase traffic. There are two symbol map values that can simplify the assignment of multiple servers to an application.

Normally, for each server, you add one row to the Application: Destination Preferences > Actual Name attribute and set Name in each row to a different server. Instead, if you want to create an application instance for every server that supports that application, you can simply add one row and set Name to "All Supporting Servers". Any additional rows will be ignored, and Selection Weight has no effect.

If you want to use only some of the supporting servers, add a row for each server and set Selection Weight to "Select Always". This will create one application instance for each of these servers (plus one additional instance with a random server, if there are also servers with numeric weights).

Figure 66 Effect of "Select Always" on Destination Preference

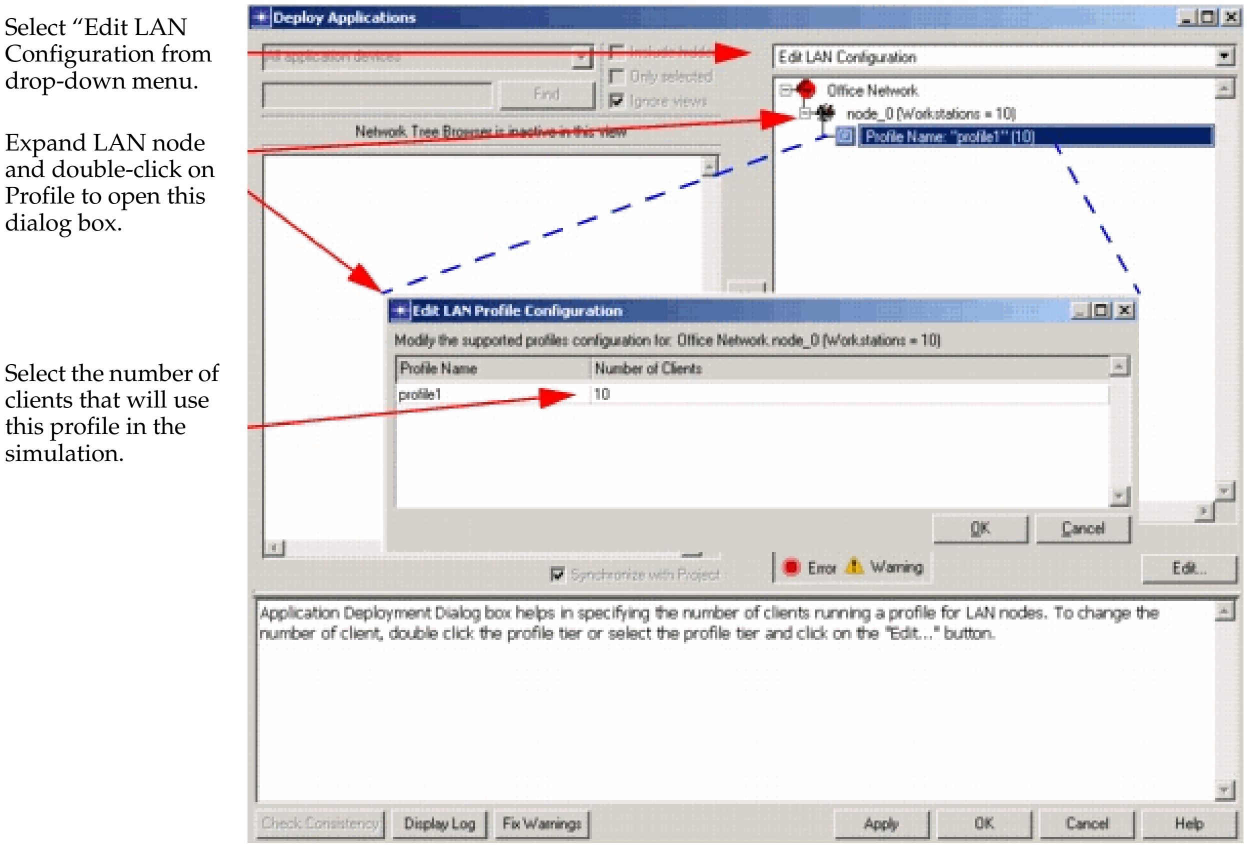


Note: These symbol maps are not available in the Application Deployment wizard. To use them, you must edit attributes on the desired nodes.

Edit LAN Configuration

When you use LAN nodes in your network model, you must deploy profiles to each one used in your simulation. After deploying the profile, select the number of workstations in the LAN model for your simulation.

Figure 67 Edit LAN Configuration



Deploying an Application from Transaction Analyzer Files (Manual Workflow)

This workflow describes how to add a Transaction Analyzer application task by configuring the Profile, Application, and Task definitions manually.

A Transaction Analyzer-generated scenario includes three configuration objects that define when and how a AppTransaction Xpert file is run during a discrete event simulation. Including a AppTransaction Xpert file in an existing network scenario involves the following three steps:

1) Include the AppTransaction Xpert file in the Task configuration object. This object contains information on each specific task to be run during a discrete event simulation. First add another row to the Task Specification table; then click in the Transaction Model Name field, and choose the AppTransaction Xpert file name from the pull-down menu.

2) Create an application definition in the Application configuration object. This object contains information about how a specific task should be run. Creating a definition is a two-step process:

   a) First create a row in the Application Definitions attribute table.

b) Then create a task description for this attribute by “drilling down” into the new compound attribute, like so: Description > Custom > Task Description. Create a row in the Task Description table, then click in the Task Name field and select the task from the pull-down menu. You can include multiple AppTransaction Xpert files in this table, and use the Task Weight and Task Ordering attributes to specify when and how the defined application runs the individual tasks.

3) Add the application to an application profile in the Profiles configuration object. An application profile includes a set of one or more application definitions (specified by the Applications attribute) and specifies when and how often these applications are run (specified by the Operation Mode, Start Time, Duration and Repeatability attributes). You can create a profile for your application definition (by adding another row to the Profile Configuration table) or add it to an existing profile.

Menu Operations: Applications

The Protocols > Applications menu includes several operations to help you configure applications in a network model.

Table 15 Application Menu Operations

Menu Option	Description
Import from Transaction Analyzer	Lets you specify Transaction Analyzer traffic as either discrete traffic or traffic flows through Transaction Analyzer dialog boxes.
Deploy DES Application Demands	Opens the Create Application Demands dialog box, which lets you specify multiple application demands. This functionality is described in Application Demands on page MC-14-736.
Deploy Defined Applications	Opens the Application Deployment Dialog in Deploy Applications mode. This lets you specify profiles and applications for nodes in the network model.
Convert Transaction Analyzer Traffic (Flows - > Discrete)	Opens the Application Deployment Dialog in Convert Transaction Analyzer Traffic (Flows -> Discrete) mode. This lets you select and convert flows quickly.
Clear Deployments	Opens a dialog that lets you choose to clear Transaction Analyzer traffic flow deployments, DES application demand deployments, or DES application deployments from the model.
Model User Guide	Opens this manual.

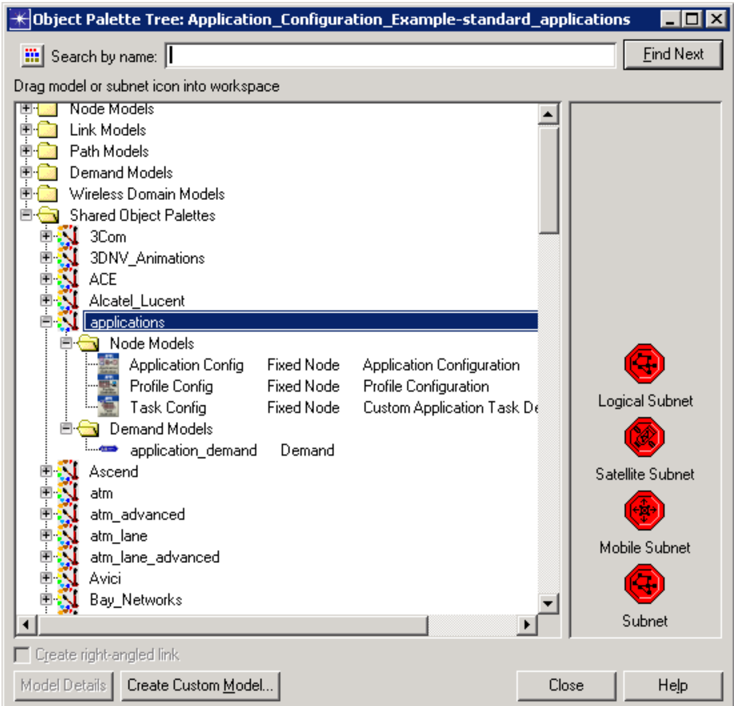
Applications Object Palette

The “applications” object palette contains the following models, as shown in the following figure.

- Application Config
- Profile Config
- Task Config
- Application Demand

These objects are described throughout this document.

Figure 68 applications Object Palette



Analyzing Applications

The model includes the following features that you can use to analyze applications in your model:

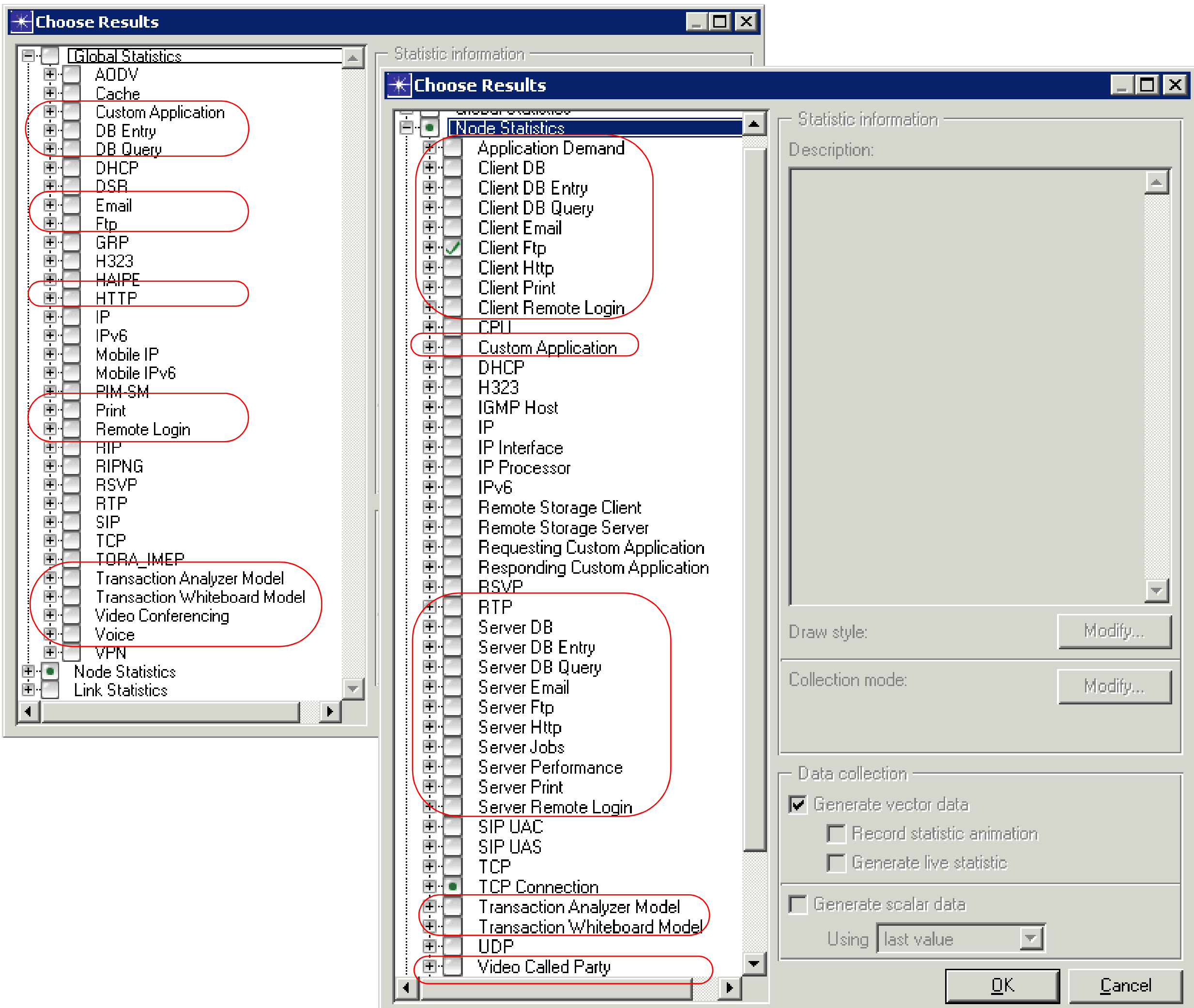
- Statistics (See Available Statistics: Applications.)
- User-defined reports (See Application Configuration Reports.)

Note: Flow Analysis is not supported for analyzing applications.

Available Statistics: Applications

The following global and node statistics are available for applications.

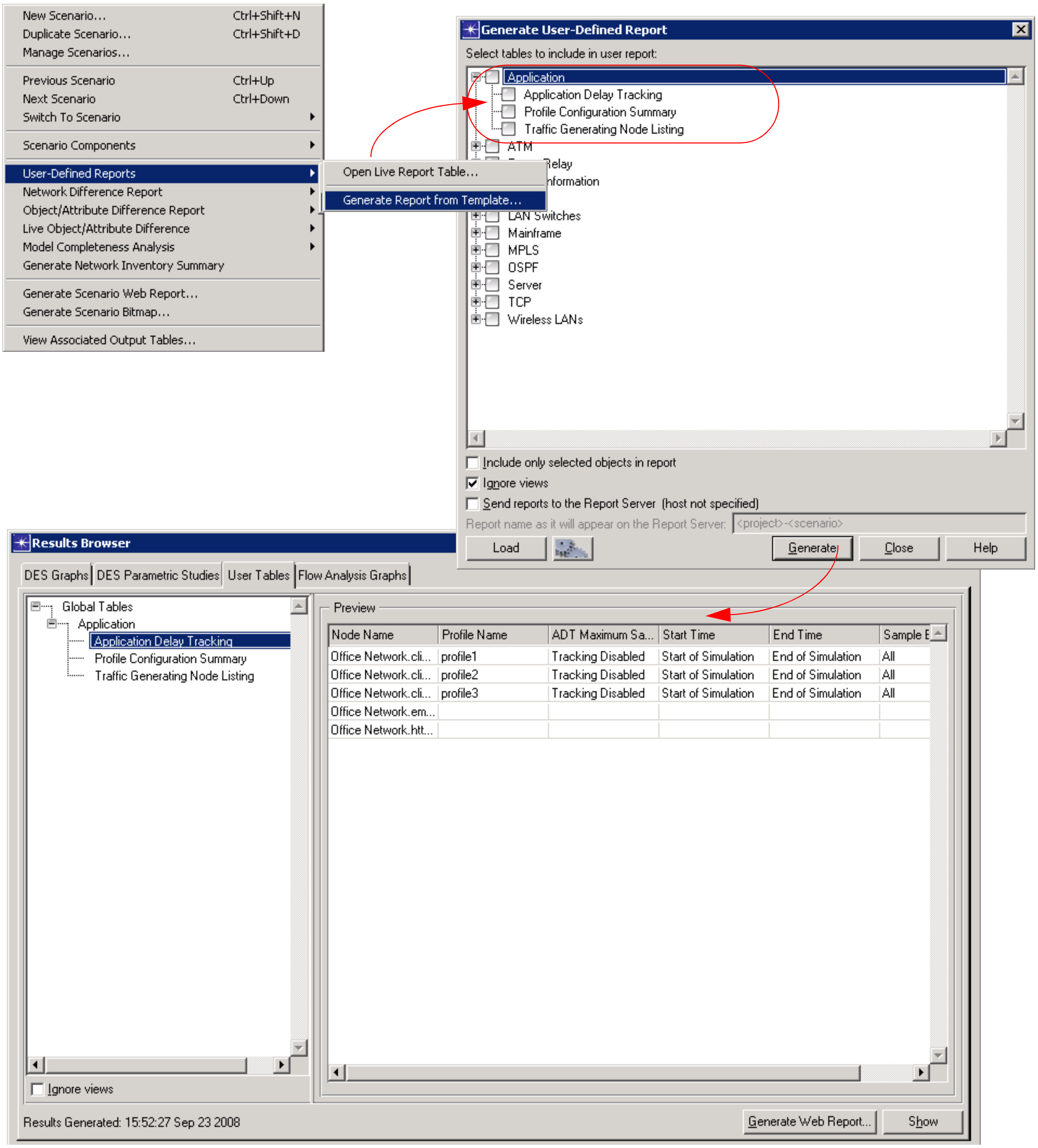
Figure 69 Application Statistics



Application Configuration Reports

The software can generate reports that list the applications, the profile configuration summary, and the traffic-generating nodes in the model.

Figure 70 Generating Application Configuration Reports



Troubleshooting Application Configuration

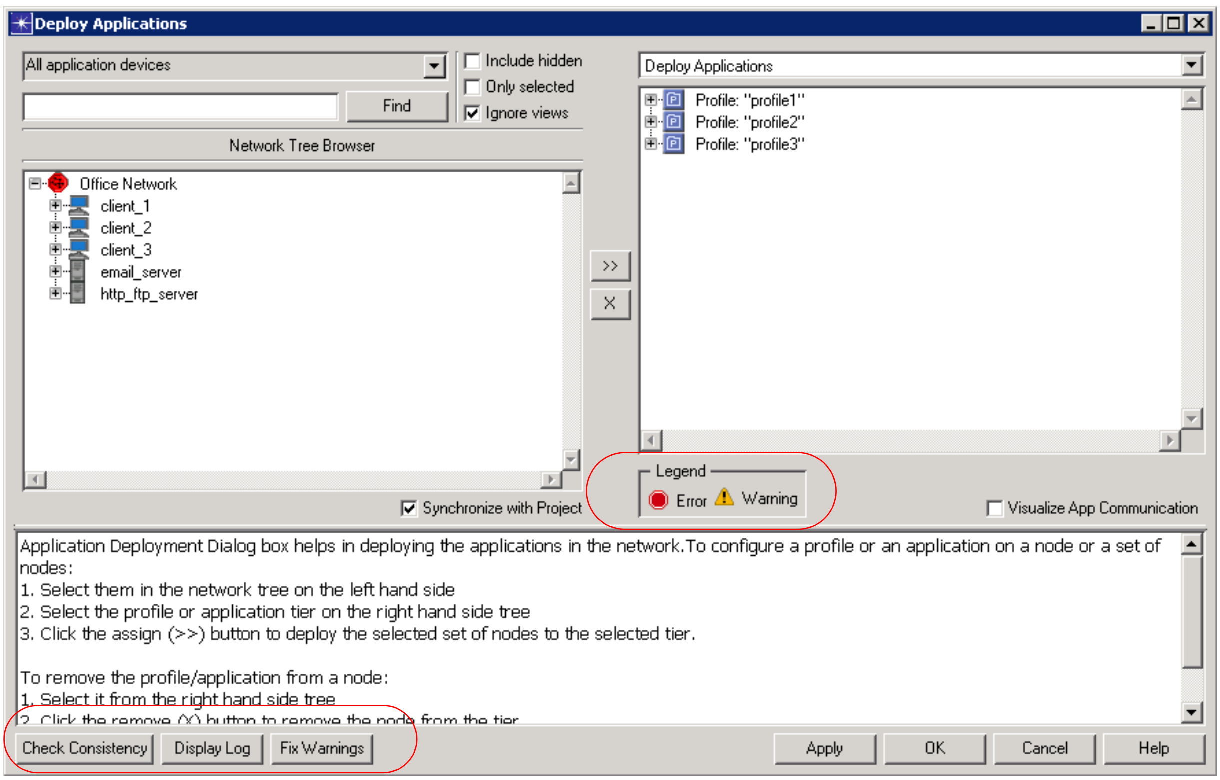
Debugging the Model

If you do not have traffic in the model or you do not have the traffic you expect, use these guidelines to assist you in debugging your model.

Application Deployment Dialog

The Application Deployment Dialog (ADD) provides error and warning messages to alert you to errors in your application deployment. As described in Procedure 8, when you use the Check Consistency button, ADD will show error or warning symbols next to profiles or applications that have been incompletely or incorrectly deployed. You can also choose to Display Log or Fix Warnings—two more ways to discover or troubleshoot problems. If you ignore errors or warnings here, your simulation may not run correctly.

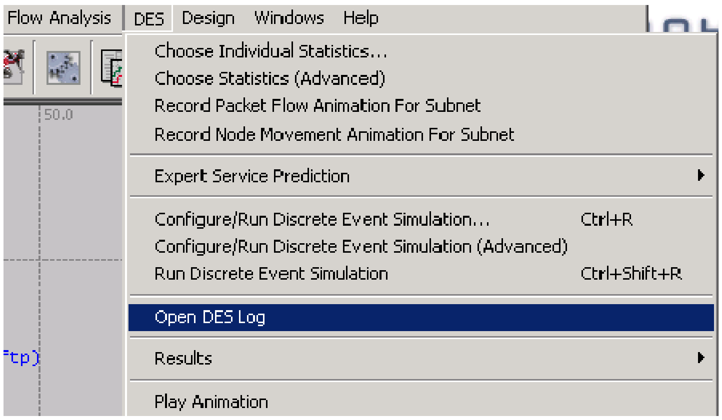
Figure 71 Application Deployment Dialog



DES Simulation Log

A valuable resource for troubleshooting is the DES simulation log. You can open this log from the DES menu, as shown. This log lists errors and warnings, which are timestamped, along with the node/object that caused the error. Using the messages in the log, you can narrow down the problems in your simulation. For more complex problems, this information is useful when calling technical support.

Figure 72 Opening the DES Simulation Log



Errors reported in the DES log display possible causes and solutions. Some of the errors you may find in the DES log include:

- Problems with Profile Repeatability—If you set the profile duration to "End of Simulation", the profile will not repeat. The reason is that the profile does not end until the simulation ends, therefore it cannot repeat.
- Application in Serial Mode will not start—If an application is running in serial mode and the application before it is set with a duration of "End of Simulation", the subsequent applications will not run.
- Profile does not run—If a profile is set on a client but is not defined, the profile will not run as expected. If a service is set on a client but is not defined on the server, the profile will not run as expected.
- Application Parameters not Defined—You will see an error if some of the application parameters are undefined along with a suggested solution.

Common Configuration Problems

Some problems that occur when trying to run the simulation are more difficult to pinpoint. Some of the configuration problems that commonly occur are

- If the profile start time plus the application start time offset are less than the convergence time for the routing protocol, your traffic may not have a route to its destination.

  Check to be sure that you take routing protocol convergence behavior into consideration when designing your simulations.

- If the profile start time plus the application start time offset are longer than the duration of the simulation, you will get unexpected results.

  This is a problem that often happens when using exponential distribution in profile start time or application start time offset.

- If the application duration is longer than the simulation duration, the application will terminate at the end of the simulation.

Isolate the Problem

Some things you can do to isolate difficult problems include the following:

- Use constant distributions to make profile scheduling deterministic.
- Collect application Traffic Sent/Received statistics in “All Values” mode. This will let you examine the exact time data was sent, as well as size of traffic.
- Check to see if the problem is caused by a network issue. Create an application demand between your client and server. Check to see if you can obtain application demand response time statistics.
- If you are working with a large network, try scaling down to a smaller subset.
- Choose Scenarios > User-Defined Reports to produce a report that lets you visualize attribute settings.
- Choose Scenarios > Object/Attribute Difference Report to compare the scenario to another scenario that works correctly.

Frequently Asked Questions: Applications

Additional information is available in the Knowledge Base section of the Support website (support.riverbed.com).

Can I make the simple CPU model non-linear?

Yes, you can make the simple CPU model simulate non-linear contention. By default, the simple CPU model uses linear interpolation. To make it non-linear, use the compound attribute
CPU > CPU Resource Parameters > Multi-Tasking Performance Table. Here is an example of how it works:

Suppose you set 2 rows in this table, as follows:

Table 16 Setting the CPU Multi-Tasking Performance Table

Scenario	Number of Tasks	Performance Fraction	Interpolation Mode	Result
1	5	1.0	Linear	Up to 5 simultaneous tasks contending. CPU performance is unaffected.
2	20[1]	0.1	Hold	Between 6 and 20 tasks contending. CPU performance degrades linearly from 100% to 10% with each additional task.

1.When more than 20 tasks are contending, CPU performance holds steady at 10%.

Does the FTP model support both data and control channels?

The standard FTP model included with Riverbed Modeler sends both data and command packets on a single channel.

How does segmentation and reassembly affect traffic in the application model?

At this time segmentation and reassembly (SAR) breaks down the application packet into segments, all at once, and passes them to TCP or UDP (depending on the application). Riverbed Modeler does not actively monitor the TCP send buffer to do any flow control at the application layer.

What is the difference between “Supported Services” and “Supported Profiles”?

Supported services are set on the server; supported profiles are set on the client. If you have a profile that supports FTP, Email, and HTTP, set the Supported Services on the server (i.e, FTP, Email, and HTTP), and set the Supported Profile on the client workstation.

What does it mean for a server to “support” an application?

If a server supports an application, the client workstation can send to and receive from the server this type of application traffic.

What happens if I do not set the Application: Destination Preferences attribute?

If the Destination Preferences attribute is not set, the destination is chosen randomly from all servers in the network model which support the given application. If there is no server which supports the service, the application will not start.

How do I know whether I should model my Email application using “Low Load”, “Medium Load,” or “High Load”?

The symbol maps for the Email application were selected based on a general understanding of low, medium, and high Email traffic loads. When you configure your application, however, you may not want to use the preconfigured application models but configure your own, based on the traffic generated by your own Email application.