Proceedings of the 1991 Winter Simulation Conference
Barry L. Nelson, W. David Kelton, Gordon M. Clark (eds.)

GENERIC APPROACH TO LAN MODELING

Prabhudeva Kavi

COMDISCO Systems Inc.
1310 Wakarusa Drive
Suite A
Lawrence, Kansas 66049

Victor S. Frost
K. Sam Shanmugan

University of Kansas
Telecommunications and Information Sciences Laboratory
2291 Irving Hill Drive
Lawrence, Kansas 66045

ABSTRACT

Predicting the performance of a local area network (LAN) can be a difficult task. Studies exist on performance of particular LAN protocols, but they make assumptions which may not apply to real networks. Additionally, they often ignore the issues of interconnected LAN configurations. It is possible to create a simulation model of a specific LAN configuration, but a typical implementation is focused on the configuration of interest and not easily extendable. This paper presents a methodology, implemented using the Block Oriented Network Simulator (BONeS), which overcomes these limitations. This methodology provides flexibility in designing and testing both LAN protocols and interconnected LAN configurations. Several underlying protocols have been implemented using the guidelines of this methodology, including Carrier Sense Multiple Access with Collision Detect (CSMA/CD), Token Ring, the Fiber Distributed Data Interface (FDDI), and Logical Link Control (LLC). Other protocols (e.g. Token Bus) implemented following the guidelines should work with the current models. After the protocol stacks are defined, various LAN configurations can be specified using the graphical interface in BONeS. After the topology has been specified, the methodology encourages iteration of traffic and protocol parameter settings. The simulation results are collected and plotted as a function of the iterated parameter for analysis.

1 INTRODUCTION

Local area networks (LANs) are proliferating at a steadily increasing rate. The design and management of these LANs requires a capability to predict the performance of various LAN configurations. Although extensive studies have been made on individual protocols such as CSMA/CD [IEEE85a], Token Ring [IEEE85b], and FDDI [FDDI86], it is difficult to directly apply the results from those studies to predict the performance of a real LAN. There are two primary reasons:

• The studies make specific assumptions about topology, traffic characteristics, and network parameters that are difficult to change. These assumptions often do not match the settings of interest to the users. Further, these results are often difficult to extrapolate.

• The studies only focus on the performance of a specific protocol of interest. However, a LAN often consists of segments of different protocol types (i.e. Ethernet and Token Ring) connected with routers or bridges.

The purpose of this paper is to present a methodology, hereafter called the Generic Approach, for the modeling and analysis of interconnected LAN configurations. In addition to overcoming the two deficiencies listed above, this Generic Approach has the following advantages over previous approaches to simulation models.

1. It provides guidelines that assure compatibility at layer boundaries. The LLC sublayer is above the MAC sublayer in the OSI Stack [Tane88]. Any valid LLC model should therefore be able to interact with any valid MAC sublayer model, such as Ethernet, Token Ring, or FDDI.

2. It is extendible. If a protocol of interest is not currently implemented (e.g. Token Bus), then an implementation following the guidelines of the Generic Approach guarantees that it can interact with higher layer protocols that follow its guidelines (i.e. LLC).

3. It encourages various levels of modeling abstraction. For example, whether an implementation of Ethernet is abstract or detailed is independent of its ability to interface with LLC.

4. It allows design iterations in protocol implementation. It is possible to take an abstract model of Ethernet and modify it into a more detailed model without affecting its interaction with LLC. This allows for multiple representations of the same physical entity.

5. It permits definition of an arbitrary network topology. After the protocol stack is defined a LAN can be created by naming nodes and specifying how they are connected. Separate segments can be connected using bridges or routers. Revisions to the topology definition are easily incorporated.

6. It provides a flexible method for specifying traffic patterns and protocol parameters. After the topology has been specified, traffic patterns can be specified using the

node names. Traffic rates and protocol parameters can be iterated to find how the LAN's performance is dependent upon a particular parameter.

The organization of this paper is as follows. Section 2 shows the ramifications of using the OSI Reference Model for creating simulation models and contrasts it to the typical approach for simulation models. Section 3 illustrates, using an example, how the features in the Block Oriented Network Simulator (BONeS^{TM}) [Shan90] were used to implement this Generic Approach. This is followed by Section 4, which describes the underlying protocols which are currently implemented following the guidelines of this methodology. The final section summarizes the Generic Approach and lists directions for future extensions.

2 BACKGROUND

2.1 OSI Reference Model

The OSI Reference Model [Tane88] divides the functions of a computer network into layers. Each layer provides a set of services.

One effect of this layered architecture is that boundaries are defined which specify how one layer communicates with the layers above and below it. As an example, the *network* layer can make a request to the *data link* layer to transmit a frame, error-free, to another node's network layer. It does this by giving the data link layer a specified structure of information, called a request. In this example, this information includes the Source and Destination address, and the Data to be transmitted.

Note that the information transmitted between layers does not specify the protocol the data link layer should use to implement its services. For example the data link layer may use Stop & Wait, Sliding Window--Go Back N, or Sliding Window--Selective Repeat to guarantee that the frame is delivered properly.

2.2 Previous Simulation Approaches

The structure of past simulation models often did not follow the OSI model. A typical approach in studying a specific protocol (i.e. Ethernet) uses traffic generators which are tightly coupled to the portion of the model that simulated the Ethernet protocol.

Admittedly this was done for a good reason. It is easier to design and implement a model for Ethernet if one does not have to worry about creating an interface for Ethernet which is compatible with other MAC level models such as FDDI, Token Ring, and Token Bus.

However, this practice has one glaring drawback. It leads to simulation models that are useful only for studying the specific protocol. If a model is implemented to study Ethernet and it is later found that the bottleneck turns out to actually be LLC, it can take a large amount of work to modify the simulation model.

2.3 Reason for Following OSI Model Structure

The first key in the Generic Approach is that it follows the structure of the OSI Reference Model. This automatically gives the first advantage listed above, compatibility at layer boundaries. Additionally, following the structure of the OSI model in combination with other BONeS features gives the user the remaining advantages of the Generic Approach. This is described in the next section.

3 DESCRIPTION OF BONeS

BONeS is a graphical discrete event simulation environment optimized for network simulation. Some of the advantages listed above are available from the framework of BONeS. Others are implemented using its "tool building" capabilities to create the underlying structures required by the Generic Approach. What follows is a description of how the various parts of BONeS have been used to build an example network using the Generic Approach. The example used to illustrate the BONeS Generic Approach consists of an FDDI ring containing five nodes.

The network of interest is composed of five nodes that are connected using FDDI. The layout of the network is given in Figure 1-1. The performance metric

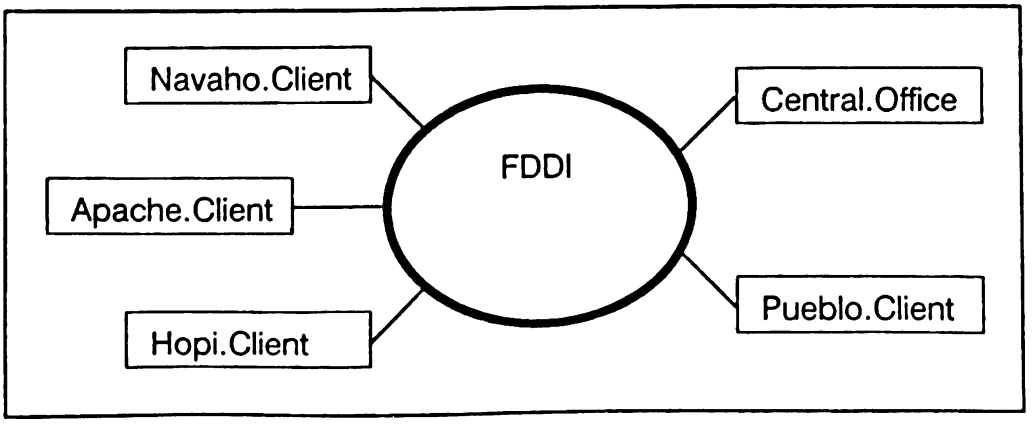

Figure 1-1 Topology for the Five Node Example

of interest in this case is FDDI throughput as a function of the LLC's Acknowledgement Timeout Period for a given Window Size.

3.1 Data Structure Editor

Different data structures are used to represent the information that flows among the various portions of a computer network. For the FDDI example, the two data structures are LLC frames and FDDI packets. Each of these structures is composed of several fields. To be able to effectively model any of these protocols requires the capability to define and manipulate these structures within the modeling environment. BONeS allows these structures to be defined through the use of its Data Structure Editor (DSE).

The DSE can also be used to define data structures that match the information exchanged at layer boundaries in the OSI model; these data structures are called service primitives [Tane88]. An example of a service primitive is a MA-UNITDATA.request which is a request from the LLC to have a MAC sublayer transmit a packet. The parameters included with the request are shown below.

MA-UNITDATA.request (Source, Destination, Data, Priority, Service Class)

The BONeS implementation of this data structure is shown in Figure 1-2. One additional field, Length, is required in the simulation model to calculate the transmission times on the medium.

DSF Window (Browse Mode)

Name: MAC-Data.Req — 7 of 7 Fields Displayed
Super: MAC
Author: kavi
Date: 1-Jan-1991 15:56:47
Revision History: YES
Documentation: YES

ADD FIELDS | SCROLL UP | ORDER FIELDS | COPY FIELDS | SCROLL DOWN | DELETE FIELD

Field	Type	Subrange	Default Value	Doc.
Instance Identifier	ROOT-OBJECT	N/A	N/A	YES
Source	INTEGER	-- Infinity	N/A	YES
Destination	INTEGER	-- Infinity	N/A	YES
Data	ROOT-OBJECT	N/A	N/A	YES
Length	INTEGER	>= 0	N/A	YES
Priority	INTEGER	>= 0	0	YES
Service Class	MAC-Service Class	All Values	Asynchronous	YES

Figure 1-2 MAC-Data.Req Data Structure

Similar data structures have been created for most MAC level and LLC level service primitives. Table 1-1 shows the primitives that have been implemented using the DSE.

Table 1-1 Service Primitives Implemented as BONeS Data Structures

MAC-Data.Req (source, destination, data, length, priority, service class)
MAC-Data.Ind (source, destination, data, length, priority, service-class, reception status)
MAC-Data-Status.Ind (source, destination, data, length, priority, service-class, transmission status)

DL-Unit.Req (source, destination, data, length, priority, service class)
DL-Data.Ind (source, destination, data, length, priority, service-class)

DL-Conn.Req (source, destination, priority, service-class)
DL-Conn.Ind (source, destination, priority, service-class, virtual circuit)
DL-Conn.Resp (source, destination, priority, service-class, virtual circuit)
DL-Conn.Conf (source, destination, priority, service-class, virtual circuit)
DL-Data.Req (virtual circuit, data, length)
DL-Data.Ind (virtual circuit, data, length)
DL-Disc.Req (virtual circuit)
DL-Disc.Ind (source, destination, virtual circuit, reason)
DL-Flow.Ind (virtual circuit, amount)

There are three important ramifications of defining these service primitive data structures:

1. Communication between adjacent layers is restricted to using these data structures.
2. The service primitive specifies the service that the layer is to provide, but does not specify that a particular protocol be used to provide that service.
3. The service primitives does not specify the detail at which a simulation model of a protocol must be implemented.

An example will illustrate the importance of these requirements. Suppose a simulation model of LLC is to be created. The first effect specifies that its communication with other layers is restricted as shown in Figure 1-3.

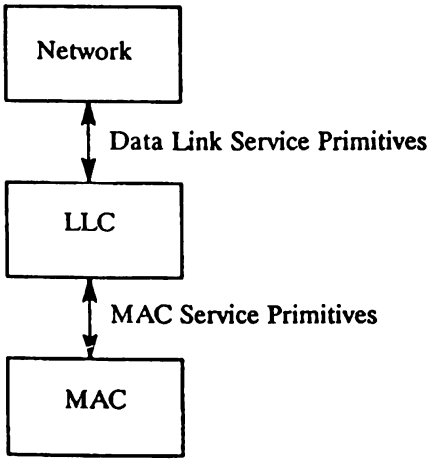

Figure 1-3 Communication of LLC with other layers

The second requirement specifies that the LLC sublayer can use any protocol to implement its services. Although the current 802.2 standard specifies using a Sliding Window--Go Back N algorithm, for test purposes a Sliding Window--Selective Repeat algorithm can be implemented and simulated.

The final feature specifies that the simulation model of LLC can be implemented to any desired level of detail without affecting its ability to work with either the MAC or Network level models. If a detailed model of the MAC sublayer is viewed to be more important, it is possible to "implement" LLC essentially as something which just takes requests from the Network layer, and simply repackages them as requests to the MAC sublayer.

In summary, these BONeS data structure definitions give the modeler three of the advantages listed earlier.

3.2 Block Diagram Editor

The Block Diagram Editor (BDE) in BONeS is used to create the models of protocols which manipulate these data structures. These models are created by interconnecting modules (or blocks) that perform specific tasks, i.e., the input data structure to a module is modified by the block to produce an output data structure. Blocks are interconnected by arcs or paths, data structures flow along these paths. Three features of the BDE have been used in the Generic Approach.

- The ability to create modules which implement specific functions. For example, a module can be built which implements a traffic generator.
- The hierarchical capability of the BDE. Once the underlying modules have been built, the BDE allows these modules to be used in a higher level module, such as a node. The number of levels of hierarchy allowed is unlimited.
- The node naming capability. Once a node module has been built, a SYSTEM can be created with many nodes which form a LAN. Each node in the network can be given a unique name.

An example illustrating all three of these concepts is shown in Figures 1-4 through 1-6. Figure 1-4 shows the top level view of the network described earlier. Node modules have been connected and named to form an FDDI LAN. The format allows names that match the IP naming or addressing convention. The purpose of node naming will become apparent in the description of the traffic generators in section 4.

Figure 1-5 is an expansion of one node module (all node modules in Figure 1-4 are identical). This illustrates how the module hierarchy capability in the BDE has been used to define a node. This particular node consists of a traffic generator communicating with the LLC protocol using the Data Link service primitives. The LLC protocol in turn communicates with the FDDI protocol using the MAC service primitives.

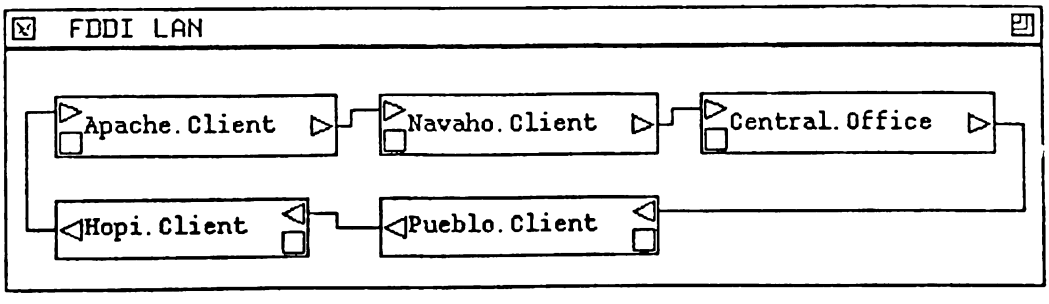

Figure 1-4 Five Node FDDI Network

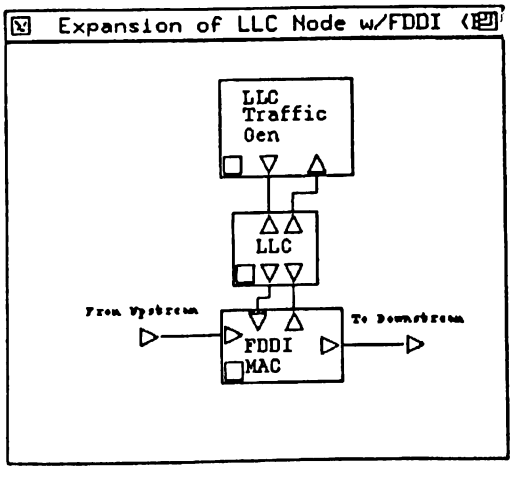

Figure 1-5 LLC Node w/FDDI Module

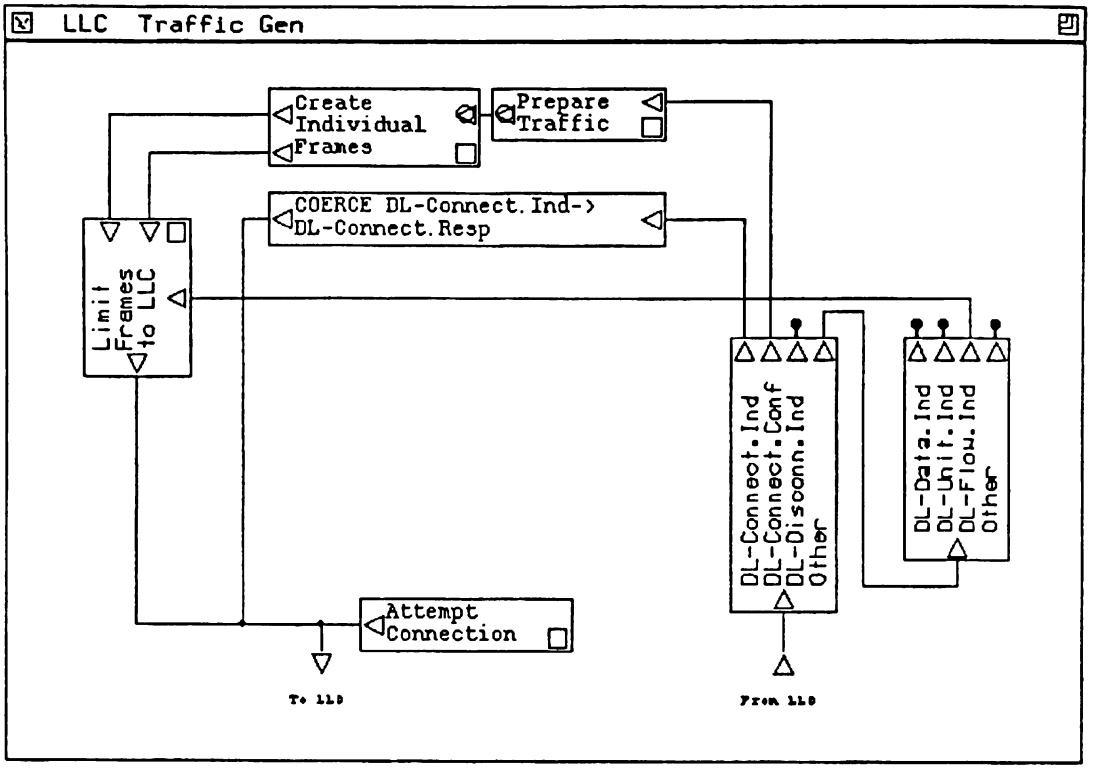

Figure 1-6 Expansion of Traffic Generator

Figure 1-6 shows the implementation of an LLC level traffic generator. The traffic generator is a low level module in that it contains submodules that implement the details of sending traffic using the LLC sublayer. Some of these submodules are BONeS library functions (e.g. COERCE..., which does a conversion from one data structure to another), while others (i.e. Attempt Connection) are again hierarchical block diagrams.

This hierarchical ability of the BDE offers two advantages. First, by allowing a module to be defined with a specific external interface (i.e. MAC service primitives), it completely isolates the internal implementation of the module from its ability to interact with other modules that have the same interface (Token Ring, FDDI, or Ethernet).

The BDE also allows revisions to a module's internal structure. If it's later found that the FDDI model is not sufficiently detailed, its internal structure can be modified without requiring any changes in either the node or system definition. When revisions are done at the protocol level, various levels of modeling abstractions can be investigated. When the revisions are at the system level, various LAN topologies can be investigated.

3.3 Simulation Manager

The Simulation Manager has two purposes in BONeS. First, it is where the system parameters (such as the traffic patterns or the propagation delay between nodes) are set. Numerical parameters can also be iterated so that the effect of modifying an independent variable can be seen on a dependent variable.

Various statistical results are collected through the use of "probes". Probes are attached to ports of BONeS blocks. Data structures flowing through these ports are manipulated by the structure of the specific probe and the results of this processing placed in a file. The information collected by the probes is further manipulated and displayed by the post processor. The simplest probe saves the entire data strucutre of each packet flowing by a probe. The Simulation Manager allows any point in the LAN hierarchy to have a "probe" attached.

In this example, the interest is in investigating the effect of varying the LLC's Acknowledgement Timeout Period on the overall FDDI throughput. To do so we iterate the value of that parameter over several values, while keeping the other parameters constant.

For a very small value of the Acknowledgment

Timeout Period, the FDDI throughput is expected to be less than optimal because an excessive number of small acknowledgement frames must be sent. However, if the Acknowledgment Timeout Period is too large, then the LLC must stop offering frames to the MAC until its received an acknowledgement for its outstanding frames.

The window size of the LLC was set to 16, while the propagation delay between nodes was set to 25 ms (which translates to a distance of approximately 5 km).

The traffic parameters were set so that all the ".Client" nodes make a connection to the Central.Office node and use that connection to send frames to and from the Central.Office. Note that while each ".Client" node manages only one connection, the Central.Office manages four independent virtual connections. The Central.Office can therefore have 16 outstanding packets on each of its four virtual circuits before having to stop sending frames to the MAC.

The offered traffic intensity in each node was set to the full channel capacity of the FDDI. This forced the LLC's flow control protocol to limit the number of frames actually transmitted. A probe was placed on the ring to collect the ring's throughput.

3.4 Post Processor

After a simulation has been run, the Post Processor is used to analyze the information collected by the probes. The post processor has access to probe data, probe parameters, and simulation parameters. It contains built-in analysis functions for performing statistical operations on the data, and can collect the results from different simulations, organize them according to a user defined metric, and display them graphically. The method of batch means is used to form confidence intervals in the BONeS Post Processor.

In this example, a plot of FDDI Throughput vs Acknowledgment Period is shown in Figure 1-7. The optimal Acknowledgment Timeout Period for this parameter set is approximately 1 ms.

4 ELEMENTS OF THE GENERIC APPROACH

This section describes the specific elements of the underlying protocols that have been implemented using the guidelines of the Generic Approach including those used in the previous example. These models allow nodes and LAN configurations that use these protocols to be created and simulated.

4.1 Traffic Generators

Two models of traffic generators have been created. The first generates traffic for a MAC sublayer. This traffic generator simply generates packets at a specifiable Offered Rate to its destinations. It can interface with all implemented MAC protocols (FDDI, Token Ring and CSMA/CD).

The second generates traffic for the LLC sublayer. It is different from the MAC level traffic generator in that the LLC level traffic generator first attempts to establish virtual connections with its destinations. Once the connection is established, then traffic is generated to its intended destinations. The rate of traffic delivered to the LLC is the minimum of what is specified as the Offered Rate and what the LLC's flow control protocol allows.

Other parameters of both traffic generators are the Interarrival Distribution, the Mean Packet Length, and Length Distribution.

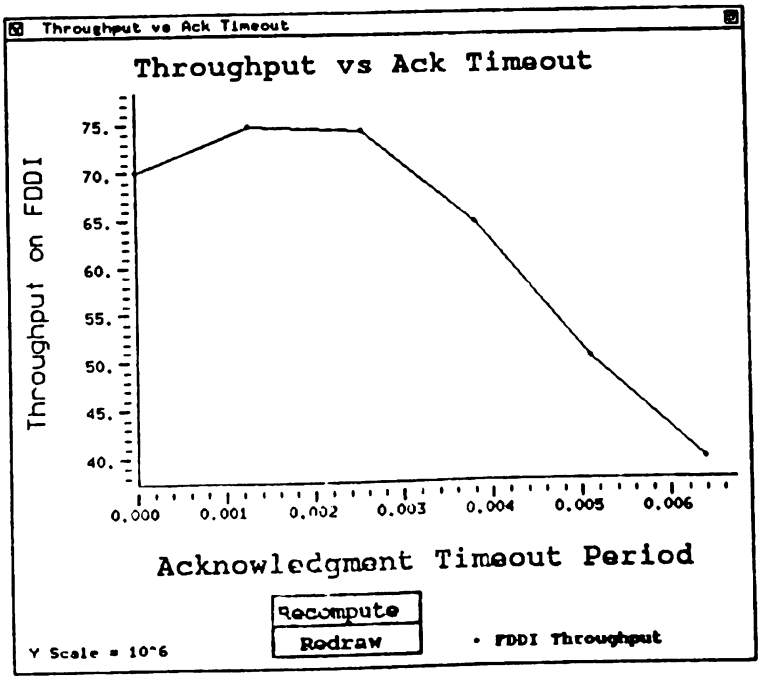

Figure 1-7 Throughput vs Acknowledgment Timeout Period

4.1.1 Node Naming

For both traffic generators, the destinations of the traffic are specified by giving the traffic generator a string, which acts as a wildcard. All node names in the network are compared to this string. If the strings match, then the matching node is one of the destinations of the traffic generator. Each packet created by the traffic generator randomly picks one of the nodes which match the destination string.

In the example given earlier, the traffic generator in the Central.Office was given "Client" as its destination expression, allowing it to send traffic to all other nodes in that particular network.

A second example illustrates the flexibility of the node naming convention. Suppose the system of interest was not a simple five node network but rather a network of interconnected engineering departments as in Figure 1-8.

Suppose that the traffic generation is to be uniformly distributed to all nodes in the Kansas University network, then the string "ukans.edu" would be entered for the traffic destinations parameter. Note that "edu" could have been used in this particular case, but would have been incorrect if the network contained nodes from more than one school.

In another case, the traffic pattern is being set on a node in the Electrical & Computer Engineering department. It's desired that the destinations of its generated traffic be distributed uniformly to other nodes within the department. The string for the destination would be set to "ece.ukans.edu". In a similar manner, a node can be set to generate all traffic to a specific node by giving either the full name (i.e. "volta.ece.ukans.edu"), or a portion of the name which is unique within the particular network (volta).

4.2 LLC

The LLC model supports both connectionless mode and connection oriented mode. It implements all primitives for connection establishment, data transfer, and connection termination. It is also capable of handling multiple independent virtual circuits concurrently. The LLC model can interface with any implemented MAC level model.

The parameters of the LLC are Number of Virtual Circuits Allowed, Window Size, Data Frame Timeout Period, and Acknowledgment Timeout Period. By default the parameters are set to match the 802.2 standard.

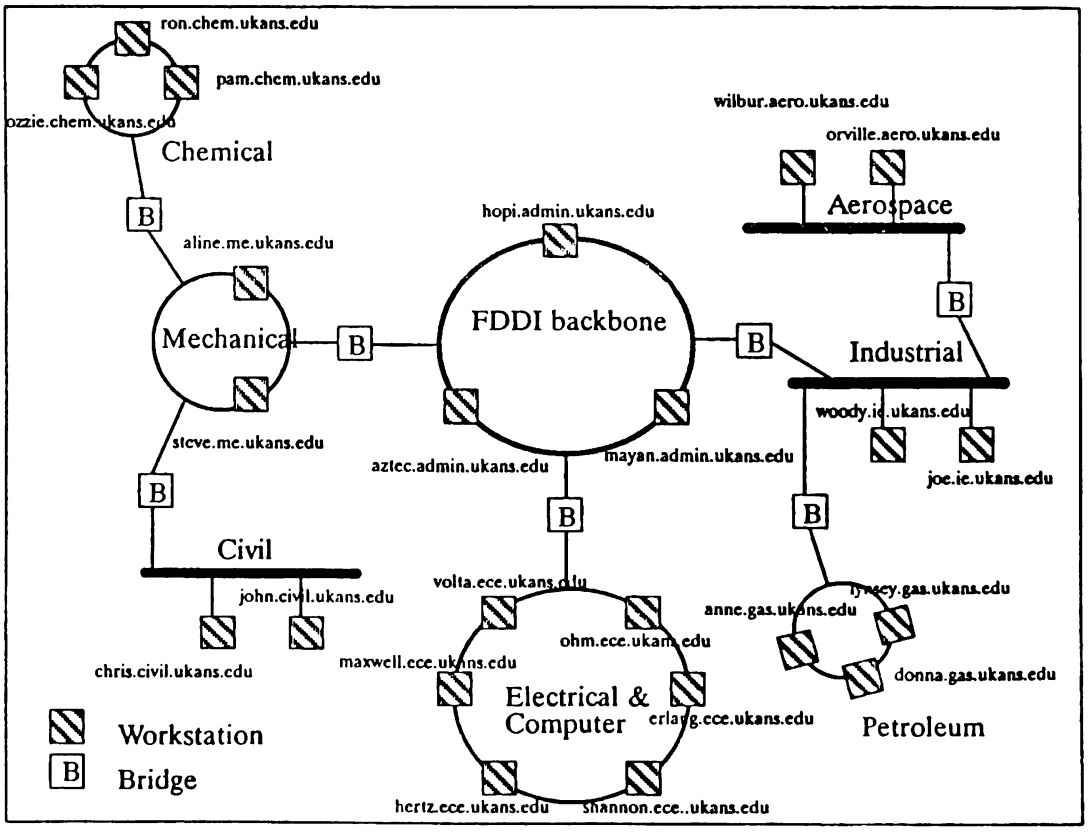

Figure 1-8 Example of Node Naming Using the Generic Approach

4.3 Token Ring

The Token Ring model implements both the Early Token Release and normal token operation. Features not currently implemented are the priority reservation mechanism and the ring monitoring functions. The parameters of the Token Ring model are Capacity, Token Holding Time, Number of Overhead Bits, Hardware Latency, and Propagation Delay (to next node). When applicable, the default values of the parameters match the 802.5 standard.

4.4 FDDI

A detailed model of FDDI has been implemented. Its capabilities include handling of both synchronous and asynchronous traffic, and implementation of the priority mechanism. The ring management functions are currently unimplemented.

The FDDI model has a large set of parameters which allows experiments with various values for ring Capacity, Synchronous Allocation, Target Token Rotation Time (TTRT), T_Pri(i) (amount of time for transmission of each priority), Hardware Latency, Propagation Delay (to next node), and number of Overhead Bits. Default values are used whenever possible.

4.5 CSMA/CD

A detailed model of 1-persistent CSMA/CD has been implemented. The parameters of this model are Capacity, Interframe Gap, Backoff Time, Slot Time, Backoff Limit, Attempt Limit, and Propagation Delay (to nearest neighbors). Again the default values follow the 802.3 protocol.

4.6 Bridge Models

Several simple bridge models have been developed. These models perform only a filtering operation to determine if a packet should be forwarded from one side of the bridge to the other.

The MAC level models described earlier are used in the transmitter sections of the bridge model. The bridge models currently implemented are:

- CSMA/CD - CSMA/CD
- CSMA/CD - FDDI
- CSMA/CD - Token Ring
- Token Ring - FDDI
- Token Ring - Token Ring

At this time, the bridge filter operation is "hard-wired". When placed in a LAN, the bridge must be told of the node names on each side of the bridge; wildcards can be used in the same manner as for the traffic generators. More advanced bridges which create their forwarding tables using either the Spanning Tree or Source Routing [Tane88] algorithms are in development.

5 SUMMARY

This paper presents a methodology for implementing LAN protocols so that they can become part of a protocol library. The elements in this library work can be connected together, allowing nodes to be defined. A LAN configuration can be tested by connecting nodes and specifying traffic patterns. The Generic Approach encourages both the testing of various levels of abstraction in protocol implementation, and various LAN configurations.

There are several directions for future work. First, more protocols can be implemented, such as Token Bus. Next, modeling abstractions for the currently implemented models can be considered to improve simulation execution times. Finally, interface definitions can be extended to the network layer to allow routers to be implemented.

REFERENCES

[Shan90] "Simulation of Communication Networks Using BONeS," Proceedings of Systems Design and Networks, Santa Clara, CA, March 1990, pp. 23-30.

[IEEE85a] IEEE Standards for Local Area Networks: Carrier Sense Multiple Access with Collision Detection (CSMA/CD) Access Method and Physical Layer Specifications, ANSI IEEE Std., 802.3-1985, (ISO / DIS 8802/3), IEEE, 1985.

[IEEE85b] IEEE Standards for Local Area Networks: Token Ring Access Method and Physical Layer Specifications, ANSI/IEEE Std., 802.5-1985, ISO/DP 8802/5, IEEE, 1985.

[FDDI86] FDDI Token Ring Media Access Control, Draft Proposed American Standard, X3T9.5/83-16, Rev.10, February 1986.

[Fros90] Frost, V. S. et al., "A Tool for Local Area Network Modeling and Analysis," *Simulation*, Nov. 1990.

[Tane88] Tanenbaum, Andrew S., *Computer Networks*, Prentice-Hall, New York, 1988.

AUTHOR BIOGRAPHIES

PRABHUDEVA N. KAVI was born in Karnataka, India in 1967. He received the received the BSEE degree from the University of Kansas in 1989, and is pursuing his MSEE.

In 1988, he joined Comdisco Systems in Lawrence, Kansas and studied both link level and network level simulation. He is currently a Systems Engineer at the Foster City, California division of Comdisco Systems. There he defines abstract modeling techniques for simulating network protocols.

Mr. Kavi is a member of Tau Beta Pi and Eta Kappa Nu.

VICTOR FROST was born in Kansas City, MO on March 6, 1954. He received the BS, MS and PhD degrees from the University of Kansas, Lawrence in 1977, 1978, and 1982, respectively. He jointed the faculty of the University of Kansas in 1982, where he is now an Associate Professor of Electrical and Computer Engineering and Director of the Telecommunications and Information Sciences Laboratory.

From 1974 to 1977 he was a Research Technician; from 1977 to 1978 he was a Research Engineer engaged in radar simulation and modeling research; and from 1978 to 1983 he was a Project Engineer conducting research in radar image processing, all at the University of Kansas Center for Research Inc., Lawrence. In 1981 he was a Visiting Scientist at the German Aerospace Research Establishment (DFVLR), Oberpfaffenhofen, West Germany. His current research interest is in the area of integrated communication networks and system analysis and simulation.

Dr. Frost has received a Presidential Young Investigator Award from the National Science Foundation, an Air Force Summer Faculty Fellowship, a Ralph R. Teetor Educational Award from the Society of Automotive Engineers, and a Miller Professional Development Award. He is a member of IEEE, Eta Kappa Nu, Tau Beta Pi.

K. SAM SHANMUGAN received the B. E. degree from Madras University, India, the M. E. degree from the Indian Institute of Science, Bangalore, India, and the Ph.D. degree from Oklahoma State University, Stillwater, all in electrical engineering.

He is currently the J. L. Constant Distinguished Professor of Electrical Engineering at the University of Kansas and a Senior Vice President of Comdisco Systems, Inc. Prior to joining the University of Kansas he has worked at AT&T Bell Laboratories and at Wichita State University. Dr. Shanmugan's research interests are in the areas of communication systems, signal processing and simulation. His group was responsible for developing the Block-Oriented Systems Simulator (BOSS) and the Block-Oriented Network Simulator (BONeS) which are used extensively for simulation of communication systems.

Dr. Shanmugan has published a number of technical articles and is author of two books, *Digital and Analog Communications Systems*, John Wiley and Sons, 1979, and *Random Signals, Noise and Filtering*, John Wiley and Sons, 1988. He has served as an editor of the IEEE Transactions on Communications from 1985 to 1988, and chaired the first IEEE Workshop on Computer-Aided Modeling and Analysis of Communication Systems in 1986. Dr. Shanmugan is a Fellow of the IEEE.