A generic object-oriented model for representing computer network topologies

S. Pierre*, I. Gharbi

Mobile Computing and Networking Research Laboratory (LARIM), Department of Electrical and Computer Engineering, École Polytechnique de Montréal, C.P. 6079, Succ. Centre-Ville, Montréal, Que., Canada, H3C 3A7

Received 27 May 1999; revised 27 March 2000; accepted 8 June 2000

Abstract

This paper presents a generic model based on an object-oriented approach for representing computer networks in topological design. The modeling approach used is based on Rumbaugh’s OMT methodology and integrates four essential interdependent steps. The implementation of this model has resulted in a tool called DESNET, which is useful for several levels of users. DESNET allows to extract from a database an already captured network, to capture and store a new network, and to cancel an existing one. It also allows the user to select routes through a network, to assign flows and capacities to links, and to evaluate network performances. DESNET can be considered as a practical tool, which can be used by network designers for topological design, link sizing and capacity planning purposes. © 2001 Elsevier Science Ltd. All rights reserved.

Keywords: Object-oriented approach; Topological design; Generic model; Modeling tool; Network representation; Capacity planning

1. Introduction

A computer network is a set of equipments (switches, computers, routers, etc.) linked to each other by communication links. The network topology refers to the way in which the nodes are linked to each other and to the capacity of links, which represent the means of transmission between the nodes [1–3].

The topological design of computer networks essentially consists of finding a topology that minimizes communication costs. It takes into account a certain number of constraints including delay and reliability [4–8]. Location of the nodes, traffic matrix, maximum allowable delay, available capacity options and their costs, constitute the set of data. The network design problem raises three kinds of questions: what is the usefulness of designing a model for representing computer networks? how these models could be used to improve the network performances? and what degree of maneuverability these models must have?

The purpose of the network representation is to find the most efficient way to model networks in order to make easier their design and analysis. To achieve this objective, the representation must be based on the use of data structures sufficiently efficient (i.e. neither congested nor complex) for minimizing the execution times or optimizing the data storage process. A good representation facilitates the implementation of both analysis and design algorithms [9].

There exist two types of representation: external and internal. The external representation of a network deals with aspects related to data display. In a design and analysis tool, external representation plays the important role of taking into account the data supplied by the user, controlling their integrity and organization, storing in a manner that facilitates their use and update. At the level of this representation, the user specifies the number of the nodes, their location, the choice of links and related characteristics, as well as the traffic requirements. Internal representation has the purpose of facilitating the implementation of design and analysis algorithms. It also makes the latter efficient in terms of memory consumption and execution times.

Ag Rhissa et al. [10] describe two types of network representation: the inheritance tree and the content tree. The inheritance tree defines the hierarchical organization of the network’s components and the administrative element included in these components. Such a representation can be transformed into an object structure whose route is a superclass, which contains information regarding all the objects. The content or instance tree defines the physical or logical relationships between the different elements of the network;
it allows to identify, in an unequivocal manner, an element from its positioning in the tree. This tree can also be transformed into an object structure and allows to model the relationships between different objects in the form of an information management tree. The content tree essentially supplies information regarding the management of configurations [11].

Users still suffer from difficulties while they manipulate the design and representation models available in the marketplace. Their work is often constrained by the lack of flexibility of such models. Hence, it is necessary to provide for a new generic representation model. By generic model, we refer to the capacity of representing a variety of types of networks, aspects related to their performance and management. Therefore, the following questions can be raised: what are the functions that a representation model could offer to the user? what data structure one could use in order to be able to take into account all the aspects of a computer network under design? and how to implement and validate such a model by taking into account the diversity of topologies, structures and management mechanisms associated with the concept of network?

This paper presents a generic model based on an object-oriented approach for representing computer network topologies. Section 2 analyses some tools of network representation available both in the marketplace and literature. Section 3 characterizes computer networks by identifying the essential elements which the generic model, proposed in Section 4, must take into account. Section 5 presents implementation and execution details of DESNET, a representation tool that supports our generic model.

2. Analysis of network modeling tools

The representation of networks is always considered as a secondary aspect of topological design. Most models and tools were rather devoted to the aspects of routing, evaluation and optimization of performance [12,13]. In these models, a network representation is often limited to capture incomplete data and display the network in a graphical mode. Tools and models presented in this section are not contradictory to these observations.

Dutta and Mitra [14] have designed a hybrid tool that integrates formal models for specifying problems (Lagrangian formulation), and heuristic models for evaluating and optimizing network performances. This leads to the most needed flexibility in both the processing of data and application of constraints. For this purpose, this tool first divides the network into a core part (Backbone) and several other access networks. After this procedure, the designers limited their work to the topological design of the backbone only. They have considered as data the following items: the location of the nodes, traffic between each node pair, allowed maximum delay, reliability and cost requirements. From this information, the tool decides: the topological configuration, routing, flow and capacity assignment. All this is done by taking into account the constraints of the problem.

In this approach, the network representation has been practically neglected for the benefit of other aspects of topological design. This is explained by the fact that the objective of the designers was essentially the optimization of network performances. Therefore, the resulting tool lacks both universality and interactive capabilities.

AUTONET [15] is a tool for analyzing the performance of wide area networks (WAN) that interconnects local networks. It was produced by the American firm Network Design and Analysis. It is among the few tools that pay special attention to network representation. In fact, AUTO- NET provides for the users, sophisticated means to design network and perform modification and adjustments required at any moment. It also offers a user-friendly interface suitable for various levels of users.

One of the inconveniences of AUTONET remains the fact that it is dedicated exclusively to the modeling and representation of WAN, without taking into account other types of networks. Thus, the local networks, which could be linked to WAN, are considered as forming a single entity, and therefore, their specific characteristics are neglected. The other inconvenience is that the capture of data is essentially oriented into performance evaluation; this is not necessarily compatible with the requirements of network representation. As a result, these data are incomplete and insufficient for adequately representing networks. Another weakness of AUTONET is that it only allows graphical representations without taking into account other modes of network representation. Here again, the main problem is one of lack of universality.

COMNET III [16] is the latest version of a series of tools produced by CACI Products Company. This tool simulates and analyses the performance of telecommunication networks, and offers the advantage of modeling and representing all types of network. COMNET III also integrates sophisticated graphical display options allowing users to represent networks with a greater flexibility. Among the features of the new release 2.1 of COMNET III are graphical software and transaction diagrams, new routing strategies, selected reports, view by spreadsheets, simulate/trace dialogue, simulate/animate dialogue, and so on. In addition, COMNET Predictor version 2.0 has been introduced in April 1999; it is an automated capacity planning tool that allows on-the-fly simulation of WANs and LANs. It is available to users of all levels, from beginners to experts.

Like AUTONET, COMNET III has some inconveniences; for instance, its data capture is incomplete. In terms of universality, COMNET III is certainly better than the other tools previously mentioned. However, it could be improved by both introducing other representation modes than graphical representation, and by adding data control procedures and user-customized representations.

OPNET (Optimized Network Engineering Tools),
produced by MIL3 [17], is a simulation and analysis tool, which allows, among other things, the simulation of large networks and detailed modeling of related protocols. OPNET integrates several characteristics such as the specification of graphical models, dynamic simulation based on events, integrated data analysis, hierarchical and object modeling, ability to verify quality of service capabilities for devices designed to carry voice and multimedia traffic over ATM, Frame Relay and IP networks.

OPNET integrates three important domains: specification, simulation and analysis. The specification domain offers several editors that allows the user to specify graphically the different components of a network. It also supplies an editor for programming finite states and specification parameters. The simulation domain is a library of procedures that facilitates the processing of packets to be transmitted. Finally, the analysis domain supplies the users with several tools including those that collect, graphically represent, and analyze data.

3. Characterization of computer networks

A computer network is characterized by different structures, types, topologies, components, performance indexes and management strategies. Regarding structures, one can distinguish between the distributed and centralized networks. In centralized networks, terminals or workstations are related to a single data source called server through all communications pass. Conversely, distributed networks are characterized by the multiplicity of routes that link sources to destinations.

Generally, three types of network should be distinguished: Local Area Network (LAN), Metropolitan Area Network (MAN) and Wide Area Network (WAN). A LAN is a local communication system connecting several computers, servers and other network components; it makes possible high-speed data transfer (1–100 Mbps) through short distances but on wide areas including organization, campuses, firms, and so on. A MAN essentially serves a large city. It also regroups several computers with more or less reduced data throughput and allows to link several enterprises in one city. WANs are used to connect several cities located at some 10 km between each other. Usually, the data throughput of a WAN is inferior to 10 Mbps.

The topology of a network provides information on its physical characteristics, that is, the way in which its nodes are linked to each other; it also indicates the capacity of each link in the network. If communication is established between two nodes through a direct link, one can speak of a point-to-point link; every network consisting of a point-to-point link is called a point-to-point network. Conversely, if communication is rather broadcasted from one node to several nodes, it is case of a multipoint link; every network consisting of multipoint links is a multipoint network or general broadcasting network.

Point-to-point networks can have a star, tree, ring, or mesh topology. In a star topology, all nodes are related by a point-to-point link to a common central node called the star center. All communications placed in this type of network must go through this node. In a tree topology, the network has a directory structure that is hierarchically structured. The principal node of this topology through which all applications pass is called tree root. In this structure, the common link takes the form of a cable (with several branches) to which one or several stations are attached. In a ring topology, all the nodes are related to form a closed ring, which, in its turn, takes a point-to-point form. A mesh topology is formed by a number of links such that each node pair of the network is linked by more than one path. Generally, a mesh topology is used for WANs.

In multipoint networks, there are two types of topologies: the bus topologies and the ring topologies. In a bus topology, each network node is set linearly on a cable, which constitutes a common physical link. The information is transmitted by any node through the entire bus in order to reach the other nodes of the network. In a ring configuration, all the nodes are set on a closed circuit formed by a series of point-to-point links. These nodes form a ring. The information within the ring is transmitted in one sense.

The main components of a network are nodes and links. The nodes represent more or less advanced units, which could be terminals, servers, computers, multiplexers, concentrators, switches, bridges, routers, or repeaters. Each of these units has its own attributes. In the framework of topological design, the most relevant factors are: cost, capacity, availability, compatibility and reliability [9].

The cost of a node includes purchasing and maintenance, as well as the cost of related software. The capacity of a node refers to the speed of its processor, the size of both programs and available memories. Availability is defined by the percentage of time during which a node is usable, or else, as the probability that a node could be available in a given instant. The compatibility of a node can be defined as the concordance between the types of traffic that the node manages and the types of links to which that node could be attached.

The reliability of a node is considered as the probability that it correctly functions in given conditions. This means that the node is not a subject of repair nor of any intervention other than the one predicted in the technical manuals.

A link or a transmission support refers to a set of physical means put in place in order to propagate electromagnetic signals that correspond to messages exchanged between an emitter and a receiver. There exist several types of transmission support. Each type has its distinct physical features, the way it carries data and its realm of use. Most known and used transmission supports are: twisted pairs, coaxial cables, electromagnetic waves, fiber-optic and satellite links.

Like the nodes, each type of link has its own characteristics, which are attributes that could taken into account during the topological design process. The most important
Fig. 1. Characterization of computer networks.
attributes are: length, capacity, cost, flow, delay, utilization rate and reliability [18,19]. The length of a link is an important attribute, particularly used for determining both the network cost and the shortest paths between node pairs. The capacity of a link refers to the maximal quantity of information that a link can carry per time unit. The cost of a link includes rental, installation, and maintenance. The flow of a link is defined as the effective number of bits that circulate through a link per unit of time. The average delay of a link is the average time taken by any packet to travel between one end to another of a link. The utilization rate of a link is defined as the ratio flow-to-capacity of this link.

The most usual indexes for measuring the performance of a network are: response time, data throughput, stability, ease of extension, information security, reliability, availability and cost. Response time can be defined as the time delay between the emission of a message by a node and the receipt of an answer. An efficient data throughput is the useful average quality of information processed by a unit of time and evaluated in given conditions and time period. The stability of a network refers to its capacity to absorb a realistic traffic pattern; it depends on the way the system is built.

Network stability refers to the number of tasks that the system performs, and to the time needed to process each of these tasks in different instants, while the network is functioning. For a given network, the ease of extension represents the capacity of this network to accept new users, or its capacity to be modified without structural changes. Information security includes both the information protection, which restricts users’ access to a part of the system, and the notion of information integrity, which depends on the capacity of the system to alter or lose information.

The average delay of a network, also called end-to-end delay, is the average time taken by a packet to travel from a source to any destination within a network. The average delay $T$ is a parameter, which essentially depends on the flows and the capacities of the links. According to Kleinrock [20], it can be expressed as follows:

$$T = \frac{1}{\gamma} \sum_{i=1}^{m} \frac{f_i}{C_i - f_i}$$

where $f_i$ and $C_i$, respectively, represent the flow in packets/s and the capacity in bps of the link $i$, and $\gamma$ the total traffic or load of the network.

The reliability of a network can be defined as its ability to continue functioning when some of its nodes or links fail. It is often linked to the notion of network connectivity [3,21]. The availability of a network refers to the probability that this network is usable during a period of time, given redundancies, breakdown detection and repair procedures, as well as reconfiguration mechanisms.

Networks could assume several functions that could differ by their importance and use. Among these functions, one can mention: routing, flow control, congestion control, configuration management, performance management, error and breakdown handling, security and accounting information management. Fig. 1 synthesizes the different characteristics of computer networks, corresponding to the generic model of representation that we propose in the next section.

4. Generic model for representing networks

Fig. 2 schematizes our modeling approach. Inspired by Rumbaugh’s OMT methodology [22], this approach integrates four independent and essential steps: definition and analysis of needs, construction of the object generic model, validation of the model, implementation of the model.

4.1. Definition

The purpose of this step is to regroup all necessary and useful data to process the different steps of topological design: routing, flow and capacity assignment, cost, transmission delay, link utilization, etc. To determine the routing within a network, it is necessary to know the traffic between each node pair. To assign capacities to the links, it is
necessary to know the capacity options available in the marketplace, as well as the link flows. To calculate the cost of a link, it is necessary to know the length and the capacity of this link. In order to calculate the transmission delay on a link, it is necessary to know the global traffic in the network, as well as the flows and capacities of the links. To calculate the utilization of a link, it is necessary to know its flow and capacity. To display a network, it is necessary to know the name of nodes, their type and location, as well as the list of links.

4.2. Construction of the generic object model

The construction of the object model integrates seven steps: identification of classes of object, preparation of a data glossary, identification of associations between objects, identification of attributes, organization and classification of classes of objects, verification of access paths, and finally the iteration of the object model.

4.2.1. Identification of classes of objects

From the description of the problem dealt with in this paper, we have sorted the following list of classes: network, node, link, representation, matrix, graph, list, distributed and centralized.

4.2.2. Preparation of a data glossary

Here is the data glossary corresponding to the above-mentioned classes:

- Network: Specific by its name and number of nodes, Network is a basic class of the model from which the majority of other classes are generated and manipulated.
- Node: It is an access and processing point in a network; it represents the main component of the network. There are, at least, three nodes in a network.
- Link: It is an important component of the network. One link must be attached to two distinct nodes, except for bus or ring topologies.
- Traffic: This is one of the most important aspects. It refers to the quantity of information exchanged between the node pairs.
- Representation: To each network corresponds one representation, which depends on nodes, links and their characteristics.
- Matrix: It is a type of representation. There exist two types of matrix: adjacent matrix and incident matrix.
- Graph: It is a type of representation.
- List: It is a type of representation.
- Centralized: It is a network structure that does not induce routing considerations because each node pair of this network is linked by a single path.
- Distributed: This is a network structure that induces routing considerations because of the multiplicity of paths that link each node pair of such a network.

4.2.3. Identification of associations between objects

Here is the final list of associations that we have drawn from our description of the network representation problem:

- to a network corresponds a representation;
- a network has a centralized structure;
- a network has a distributed structure;
- a network is composed of many links;
- for a given network there is pattern traffic;
- a representation is of matrix type;
- a representation is of graph type; and
- a representation is of list type.

4.2.4. Identification of attributes

Having regrouped and filtered all the possible attributes, we have made a list of attributes for each class. It should be noted, however, that certain classes have no attributes:

- network: name, number of nodes, type, configuration and structure;
- representation: type;
- node: name, type, Cartesian coordinates and cost;
- link: beginning node, end node, type, delay, capacity, cost and length; and
- traffic: beginning node, end node, and traffic.

4.2.5. Organization and simplification of classes of objects

From the classes identified in above steps, we have discovered two inheritance cases:

- The association between the Representation class on one hand, and the classes of Matrix, Graph and list on the other hand, can be transformed into an inheritance relation, given that each of these classes represent a type of representation.
- The association between the Network and the Centralized and Distributed classes can be transformed into an inheritance relation, given that each of the latter classes represent a network structure.

4.2.6. Verification of access paths

This step consists of following the paths through the object model’s diagram in order to verify if they lead to significant results. In the case of our model, we have undertaken several verifications that gave positive results. Here is an example.

In order to represent a network, one should use the inheritance relation between the Representation and Network classes. It is through the object network that the system recovers the information required on the nodes and links that constitute the network. This information has been gathered using the link between the Network and Node classes n times, and the link between the Network and Link classes m times.
4.2.7. Iteration of the object modeling

This last step consists of repeating the process several times, because an object model is rarely correct for the first time.

4.3. Validation of the object model

This phase has a particular importance in our approach. It allows to test the working of the object model, built from the precedent phase, and to verify its dynamism. The purpose of validating a model is to adjust it, according to the integrity constraints between the objects and the occurrence of events. To undertake this phase, we were inspired by Rumbaugh’s functional modeling [22]. This has led us to add a certain number of operations to the classes.

There are several ways of identifying the operations of a class:

- The operations of the object model: Operations on the structure of objects include the writing and reading of the value of attributes and associative links. These operations do not need to appear explicitly in the object model, but they are induced by the presence of attributes. Here are some examples of operations deducted from our object model:
  - **Network Class**: `Capture()` and `Open()`;
  - **Link Class**: `Read()` and `Calculate_length()`;
  - **Graph Class**: `Display_node()`, `Display_link()`, `Display_bus()`, and `Display_ring()`;
  - **Matrix Class**: `Display_adjacency_matrix()`, `Display_incidence_matrix()`;
  - **Distributed Class**: `Calculate_cost()` and `Calculate_delay()`.

- Operations and actions in the states: The actions and activities of the state diagrams (dynamic model) can be functions. These functions have an interesting calculation structure. They have to be defined as operations. For example, the **Network** class has an activity that allows verifying the connectivity of a network. This activity can be defined in the model as a `connectivity()` operation.

- The operations of functions: each function of the dataflow diagram (functional model) corresponds to an operation on one or several objects. Here are some examples of functional operations, which we have identified:
  - **Network Class**: `Delete()` and `Add_node()`;
  - **Distributed Class**: `Calculate_routing()`, `Undertake_flows()`, `assign_flows_capacities()`;
  - **Node Class**: `Modify()`;

Fig. 3. Generic object model resulting from validation.
Link Class: Add(), Delete() and Modify();
Bus Class: Modify();
Ring Class: Modify();
Traffic Class: Add(), Delete() and Modify().

- Shopping list operations: sometimes the behavior of objects in the real-world suggests certain operations. Meyer [23] calls these operations “shopping lists” because they do not depend on a particular operation, and therefore they are not subject to a particular arrangement. In our application, we have identified two operations of this type: the Matrix Routing() of the distributed class and that of the operation Number_of_link() of the topological configuration class.

Most models require more than one operation to be complete. We have proceeded in a repetitive manner to prepare a first approximation of the solution, which we have refined by successive iterations. Fig. 3 presents the generic model resulting from both the adding operations and iterations of the three first steps of our approach.

5. Implementation

DESNET (DESign of NETwork) is the name of the tool emerging from this generic model. Developed in Visual Basic 4.0 for “Pentium 100” micro-computers in Windows 95 environment, it offers the user the possibility of processing several design aspects such as routing, flow and capacity assignment, calculation of network’s cost and delay. It integrates data manipulation interfaces that are understandable and accessible by any user who has a minimum knowledge of topological design.

5.1. Description of functions

DESNET provides the user with a large number of functions. Its flexibility and simplicity facilitate adding several functions without affecting the robustness and reliability of the system. For organizational reasons, we have decided to regroup the functions offered by DESNET in three classes according to their type, extent, and domains of application. We have defined a class for the definition of networks, another to represent them and another for their handling.

- Network definition: This class of functions regroups all the applications which act as a single entity. It represents the entrance in the system. The functions that compose it allow the user for choosing the network to be processed and permit the introduction of a new network in the system, or the deletion of an existing network. Obviously, this class has a function allowing for quitting the system.
- Network representation: The class of functions that describe network representation has the role of representing the network and its components in several manners, according to the user’s needs. The latter has the choice between a graphical representation, matrix or list, or a graphical representation of the optimal path between each pair of nodes chosen by the user.
• **Network handling:** This is the most important class of functions offered by DESNET. As illustrated in Fig. 4, this class regroups the most specific functions of the tool. Certain functions of this class are accessible at any moment, when a network is open or captured: this is the case of handling topological configuration, nodes, links and traffic of a network. There are also certain other functions, which are accessible once the routing is calculated: this is the case of flow and capacity assignment and calculation of network performances.

As shown in Fig. 5, DESNET’s working logic rests on two interdependent steps. The first step is the access to data, while the second is reserved to their processing. The execution of the first step is a required condition for the second to take place. DESNET is provided with a mechanism that prevents the user to go directly to the second step without going through the first. The user can be satisfied by the execution of the first and quit the system without being obliged to go the second step, but cannot do the contrary.

5.2. **Examples of DESNET’s working**

To better present DESNET, we have undertaken to process two network examples. Through these examples, we put to the fore the main features and functions offered by the system.

5.2.1. **Example 1**

Let us consider a network, which has 9 nodes representing 8 American cities and 12 links. The coordinates of the nodes are drawn from Kershenbaum [9] and represented in Table 1; they have been fixed according to a reference point on the upper left part of the screen. As shown in Fig. 6, this network has a distributed architecture — it is a WAN with a mesh topology. For this example, we have decided to assign ourselves the flows and capacities to the links. After the capture of data related to the network and its topological configuration, we have started to capture the network’s

<table>
<thead>
<tr>
<th>Node</th>
<th>NYK</th>
<th>LSA</th>
<th>CHI</th>
<th>DAL</th>
<th>BAL</th>
<th>SFO</th>
<th>MIA</th>
<th>DEN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abscissa</td>
<td>92</td>
<td>12</td>
<td>64</td>
<td>56</td>
<td>86</td>
<td>2</td>
<td>99</td>
<td>36</td>
</tr>
<tr>
<td>Ordinate</td>
<td>12</td>
<td>98</td>
<td>26</td>
<td>90</td>
<td>25</td>
<td>90</td>
<td>89</td>
<td>61</td>
</tr>
</tbody>
</table>

![Fig. 5. DESNET running (working) logic.](image-url)
nodes. Fig. 7 illustrates the capture of the first node (NYK); one should note the presence of the principal node domain, reserved to network configured in star or tree. In both cases, it is important for the system to know the main node of the star and the root of the tree. Because our network has a mesh configuration, this field is automatically deactivated by the system.

Fig. 8 represents the screen that captures a link. One should note on this screen that the domain Cost is not activated, because the cost of a link is calculated automatically by the system and should be captured by the user. Similarly, for the nodes, DESNET does allow the user to capture the same link twice. After the capture of nodes and links, the user has the following choices: to capture the traffic, to confirm the capture or to quit the system. For this example, we have chosen the capture of the traffic matrix. We have supposed that the traffic matrix is uniform with a constant value of 5 packets/s; the average size of packets is equal to 1000 bits. Fig. 9 represents the capture of traffic between the nodes NYK and BAL.

Having confirmed the capture of these data, we can then access the data processing part. For example, we described the working of the network representation part, as well as the handling functions of nodes and links. Fig. 10 shows the graphical representation of the network studied by Kershenbaum [9]. The numbers that appear in this figure besides the links represent their length in kilometers. Table 2 shows the flow and the capacity of each link of this network. As a
performance measure, the average delay calculated using relation (1) is 68.88 ms.

Furthermore, we have added to this network, two new links, called SEA and CANADA5. The node SEA has 5 and 20 as coordinates, while the coordinate of CANADA5 are 45 and 15. During the addition of one or several nodes, DESNET allows the user to choose the network’s new name and the precise number of nodes needed to be added. The addition of a node is realized exactly in the same manner to that of capturing a node.

To link two nodes added to a network, we have decided to add four links, which are in this case the links between SEA and SFO, SEA and CHI, SEA and CANADA5, CANADA5 and NYK. As shown in Fig. 11, the addition of a link is considered as data capture. In effect, all control operations regarding the network’s connectivity and integrity of data used by DESNET during the capture of links are also applicable on the addition. The user has the right to add as many links as she/he likes.

After adding the nodes, it is convenient to add the traffic between these new nodes and the remaining network nodes. In order to maintain the traffic matrix uniform, the traffic between each new node pair is kept equal to 5 packets/s; the average size of packets is still 1000 bits. Fig. 12 represents the network obtained after the addition of SEA and CANADA5 nodes. The new network contains 10 nodes and 16 links. The CANADA5 node is represented by an icon different to that of the other nodes, in order to indicate that the latter is, in fact, a local network attached to a larger distributed network. Table 3 shows the flow and the capacity

![Fig. 8. Data capture of the link BAL–MIA.](image1)

![Fig. 9. Data capture of the traffic between NYK and BAL.](image2)
of each link of this new network. As a performance measure, the average delay calculated using relation (1) is now 55.39 ms.

DESNET allows the user, by means of a Zoom technique, to graphically represent local and metropolitan networks when these networks are part of a wider network. It is sufficient that the user click twice on the icon that represents the node in question. Fig. 13 graphically represents the local network CANADA5, which is made of four nodes (city1, city2, city3 and city4) and has a ring topology.

5.2.2. Example 2

We have chosen as a second example the GTE network, with 25 links and 12 nodes representing American cities. The coordinates of nodes are represented in Table 4 [9].

Having opened the network, DESNET permits the user to see all its features and to modify some amongst them. As shown in Fig. 14, the GTE network is a mesh-distributed network. From the screen that appears in this figure, the user can access the nodes, links and traffic; he/she can consult, modify, delete and add links. There is another faster way of doing these same operations, directly from the welcoming interface. This alternative allows for faster access and gives exactly the same results as the first procedure.

Apart from the handling of nodes, links and traffic,

<table>
<thead>
<tr>
<th>Link</th>
<th>Flow (kbps)</th>
<th>Capacity (kbps)</th>
</tr>
</thead>
<tbody>
<tr>
<td>NYK–BAL</td>
<td>70</td>
<td>100</td>
</tr>
<tr>
<td>LSA–SFO</td>
<td>40</td>
<td>50</td>
</tr>
<tr>
<td>LSA–DEN</td>
<td>90</td>
<td>100</td>
</tr>
<tr>
<td>CHI–DAL</td>
<td>30</td>
<td>60</td>
</tr>
<tr>
<td>CHI–BAL</td>
<td>60</td>
<td>100</td>
</tr>
<tr>
<td>CHI–SFO</td>
<td>30</td>
<td>50</td>
</tr>
<tr>
<td>CHI–MIA</td>
<td>10</td>
<td>19.2</td>
</tr>
<tr>
<td>CHI–DEN</td>
<td>20</td>
<td>50</td>
</tr>
<tr>
<td>DAL–DEN</td>
<td>40</td>
<td>50</td>
</tr>
<tr>
<td>BAL–MIA</td>
<td>20</td>
<td>50</td>
</tr>
<tr>
<td>BAL–DEN</td>
<td>40</td>
<td>50</td>
</tr>
<tr>
<td>MIA–DEN</td>
<td>40</td>
<td>50</td>
</tr>
<tr>
<td>ANADA–SEA</td>
<td>40</td>
<td>50</td>
</tr>
</tbody>
</table>

Table 3: Link flows and capacities for the network of Fig. 12
Fig. 11. Addition of a link between SEA and CHI.

Table 4

<table>
<thead>
<tr>
<th>City</th>
<th>NYK</th>
<th>PHI</th>
<th>WDC</th>
<th>ATL</th>
<th>DAL</th>
<th>PHO</th>
<th>LSA</th>
<th>SFO</th>
<th>SEA</th>
<th>DEN</th>
<th>CHI</th>
<th>DET</th>
</tr>
</thead>
<tbody>
<tr>
<td>Abscissa</td>
<td>86</td>
<td>96</td>
<td>90</td>
<td>73</td>
<td>60</td>
<td>33</td>
<td>14</td>
<td>11</td>
<td>15</td>
<td>34</td>
<td>55</td>
<td>67</td>
</tr>
<tr>
<td>Ordinate</td>
<td>10</td>
<td>30</td>
<td>52</td>
<td>52</td>
<td>84</td>
<td>81</td>
<td>72</td>
<td>33</td>
<td>11</td>
<td>29</td>
<td>10</td>
<td>5</td>
</tr>
</tbody>
</table>

Fig. 12. New network graphical representation.
DESNET allows the user to undertake several other operations in the network, the most important among these is routing. In fact, DESNET displays, for each node pair, the optimal path, its length, as well as the link through which it passes.

Having done the routing, the user can visualize the optimal path between all pairs of nodes in the network. For this purpose, DESNET makes available to the user a screen that permits to choose the two end nodes of the path to be displayed. The user can also display the optimal path, or cancel the operation and return to the welcoming interface. As we can see in Fig. 15, the links that form the optimal path between NYK and SEA cities are white, while the other links are black. The
optimal path between NYK and SEA passes through DET, CHI, and DEN cities. Having assigned simultaneously flows and capacities, DESNET displays all the links, their flows and their capacities, as shown in Table 5. The average delay of this network is 47.17 ms.

6. Conclusion

In this paper, we have presented a generic model for representing computer networks. The proposed modeling approach is inspired by the OMT methodology and integrates four essential interdependent steps. The first step consists of defining users’ needs. The second is about organizing and regrouping the data obtained during the first step, in classes and associations in order to build a generic model. The third phase validates this model by testing its functions and dynamism. At the end of this step, we have obtained an object model sufficiently generic to support the functions required for network topological design. The implementation of our model has led to DESNET, which emerges as a tool dedicated to network representation and capacity planning.

DESNET is made of two interdependent parts. The first has the role of helping the user to access data stored in a database. It offers the user the possibility of opening a network already captured, capturing new one and deleting another. The second part has certain functions that allows the user for handling the network’s components and the features of these components. DESNET also allows for graphical representation of networks. As a result, DESNET can be considered as a practical tool that can be used by network designers for topological design, link sizing and capacity planning purposes.

However, even though DESNET has several mechanisms...
for controlling the integrity of data, it does not allow the
user to control his or her choice of nodes. Therefore, future
research should be oriented towards implementation
mechanisms that allow DESNET to better assist the user
in her/his choice of terminals, concentrators, multiplexers,
servers and switches.

Acknowledgements

The authors would like to thank the anonymous referees
for their insightful comments on the first version of this
paper. This work has been supported in part by the Tele-
learning Network Centres of Excellence of Canada.

References

economical packet-switched computer networks. Artificial Intelli-
computer network topologies. IEEE Trans Man, Systems Cybernetics
of topological optimization. IEEE Trans Man, Systems Cybernetics
[4] Gavish B, Neuman I. Routing in a network with unreliable compo-
[5] Girard A, Sanso B. Multicommodity flow models, failure propagation,
and reliable loss network design. IEEE/ACM Trans Networking
economical configurations of elementary packet-switched networks.
networks under performance constraints. IEEE Trans Reliability
[8] Yokohira T, Sugano M, Nishida T, Miyahara H. Fault-tolerant packet-
switched network design and its sensitivity. IEEE Trans Reliability
IBM Thomas J. Watson Research Center, 1993.
modeling for fault management expert system. International Symposi-
um on Information, Computer and Network Control, Beijing, China.
February 1994.
globale des incidents pour un hyperviseur de réseaux hérogènes,
Actes des quatorzièmes journées internationales d’Avignon. AI’ 94,
[12] Conti M, Donatiello L. Simulation modeling of local and metropoli-
117–42.
[13] Pierre S, Beaubrun R. Integrating routing and survivability in fault-
317–27.
[14] Dutta A, Mitra S. Integrating heuristic knowledge and optimization
models for communication network design. IEEE Trans Knowledge
[15] AUTONET/Performance-3. Network Design and Analysis Corpora-
tion, VA, USA. 1995.
[16] COMNET III. COMNET III Predictor vs. 2.0. CACI Products
[19] Kubat P. Alternatives for reliable and quality telecommunication
[21] Pierre S, Eligbaoui A. A Tabu search approach for designing computer
network topologies with unreliable components. IEEE Trans Reliability
Modélisation et Conception Orientées Objet. Paris, France: Masson,
1995.
[23] Meyer B. Object oriented software construction. Hertfordshire,