A Multiconstraint-based Real-time Routing Scheme using Simulation Methodology

Saurabh Mittal, Wenji Wu, Bernard P. Zeigler, IEEE Fellow
Department of Electrical & Computer Engineering
University of Arizona, Tucson, AZ 85721
Email: wenji@u.arizona.edu, {saurabh, zeigler}@ece.arizona.edu

Keywords: MCP, Routing, Simulation, DEVS, ASON, Networking

Abstract

In this paper, we propose a multiconstraint-based real-time network routing scheme using simulation methodology. We first discuss how to resolve the multi-constrained path (MCP) problem with a discrete event simulation method. Our proposed Path Selection using Simulation Method (PSSM) can work with a wide class of constraints that properly include the additive constraint assumption required by most graph-search algorithms for the MCP problem. Then we show how to implement the PSSM algorithm with DEVS (Discrete Event System Specification). Finally we discuss the application of PSSM method into Automatically Switched Optical Network (ASON), developing a multiconstraint-based real-time routing scheme using simulation methodology.

1. INTRODUCTION

Recently multiconstraint-based routing [1][2][3] has received lots of attention both in industry and academia. Multiconstraint-based routing tries to find a path between a source and a destination node (router or terminal) satisfying multiple independent path constraints. The multiconstraint-based routing problem has found many applications in different fields. For example, in multimedia, applications usually require communication paths to meet stringent requirements on delay, delay-jitter, cost, and/or other quality of service (QoS)[4] metrics. In an Automatically Switched Optical Network (ASON) [5], a successfully established optical connection should meet customer- specified requirements placed on power budget, dispersion, and signal-to-noise ratio etc.

Multiconstraint-based routing can be formalized as a multi-constrained path problem (MCP), which is NP-hard. Different heuristic algorithms have been proposed to address the problem [1][2]. In all of these algorithms, it is assumed that the network state information (network topology/resource information) is temporarily static and has been disseminated throughout the network using the underlying link state protocols (e.g., OSPF). In each network router, a link state database about the network is built up from which a directed graph can be developed. The directed graph represents the network. Each node stands for a router, and each edge represents a physical connection. Various methods have been developed to search the graph, trying to find a feasible path to meet the multi-constraint requirements. However, almost all of these graph-searching algorithms are based on the assumptions that all metrics of interest are additive; i.e. for each metric of interest, a routing path's metric is the sum of each of its constituent links' values [1][2]. Apparently, this assumption is limited and not general enough. For example, the bit error rate of a routing path is not the sum of each of the constituent link's bit error rate.

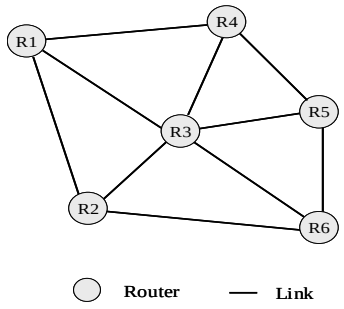

Figure 1: Computer (Telecommunication) Network

In this paper, we propose a heuristic algorithm (PSMM) for the MCP problem. Compared to those graph-search algorithms, our proposed PSMM algorithm does not require the additive constraint assumption; it works with a wide class of constraints. Therefore, our algorithm has significant practical use. To illustrate, we apply the PSSM method to ASON, developing a multiconstraint-based real-time routing scheme using simulation methodology. The paper is organized as follows: In section 2, we present path selection using a simulation method (PSSM) algorithm. Section 3 describes PSSM's DEVS (Discrete Event System Specification) implementation. In section 4, we discuss how to apply PSSM to implement a multiconstraint-based real-time routing scheme. Section 5 concludes this paper.

2. PATH SELECTION USING SIMULATION METHOD (PSSM)

2.1 Definitions and Notations

Given a general computer (telecommunication) network as shown in Figure 1, which consist of routers and links. Let R_i denote a router i; $L_{i,j}$ denotes a link between router R_i and R_j; Each link $L_{i,j}$ has k attributes: $w_1, w_2, \ldots, w_k$; for the purpose of generality, k attributes are independent of each other.

Let $w_1^{i,j}$ denotes link $L_{i,j}$ attribute's w_1, thus for a link $L_{i,j}$, its attributes are: $w^{i,j} = (w_1^{i,j}, w_2^{i,j}, \ldots, w_k^{i,j})$. Well-known metrics include bandwidth, delay, jitter, cost, loss probability, etc. Different metrics may have different features. A router might also have attributes, but a router's attributes can always be combined with its outgoing links' attributes.

Definition 1: In a network, a path is an alternating sequence of routers and links. For simplicity, it is defined as $P = (R_1, R_2, ..., R_{n-1}, R_n)$; it has k attributes: $w_1, w_2, \ldots, w_k$, which are denoted as $w^P = (w_1^P, w_2^P, \ldots, w_k^P)$. The k attributes might not be additive.

We have:

$$w_1^P = f_1(w_1^{1,2}, w_1^{2,3}, ..., w_1^{n-1,n}),$$
$$w_2^P = f_1(w_2^{1,2}, w_2^{2,3}, ..., w_2^{n-1,n}),$$
$$\ldots$$
$$w_k^P = f_k(w_k^{1,2}, w_k^{2,3}, ..., w_k^{n-1,n}),$$

Here, $f_1, f_2, \ldots, f_k$ are general functions, which are defined upon attributes $w_1, w_2, \ldots w_k$.

Definition 2: Given a computer/telecommunication network, a source router, a destination router, and k constants $r_1, r_2, \ldots, r_k$ represented by a vector $r = (r_1, r_2, ..., r_k)$, multiconstraint-based routing is to find a path P from source router to destination router such that $w^P \le r$, i.e. $w_1^P \le r_1$, $w_2^P \le r_2, \ldots, w_k^P \le r_k$.

Property 1: The path has the concatenation property. As shown in Figure 2, there are paths $P_1 = (R_1, R_2, \cdots, R_{n-1})$ and $P_2 = (R_1, R_2, \cdots, R_{n-1}, R_n)$. Now, if P_1's k attributes are available: $w^{P_1} = (w_1^{P_1}, w_2^{P_1}, \ldots, w_k^{P_1})$. Then path P_2's k attributes can be calculated as follows:

$$c(P_2) = c(P_1) + c(L_{n-1,n}),$$
$$w_1^{P_2} = f_1(w_1^{P_1}, w_1^{n-1,n}),$$
$$w_2^{P_2} = f_2(w_2^{P_1}, w_2^{n-1,n}),$$
$$\ldots$$
$$w_k^{P_2} = f_k(w_k^{P_1}, w_k^{n-1,n}).$$

By property 1, a path's attributes can be calculated recursively link by link. Our PSSM algorithm is based on this property.

2.2 How the PSSM Algorithm Works

Different from the traditional graph searching based heuristic algorithms; we propose a heuristic algorithm for the MCP problem using a discrete event simulation method. The idea of path searching using a simulation method resembles current practice. As is well known: before a route is set up in a wide-area network and turned over to business operation, experiment test data are usually sent across the route to test whether the route meets the multiconstraint requirements. If the test data's quality (bit-error-rate, SNR, etc) can't meet the requirements, this is taken to mean that the route can't meet the requirements. The experimental route's quality is assessed by evaluating the quality of the experiment data that pass over it.

Similarly, we try to find a path that can meet the multiconstraint requirements by sending data from source to destination using a simulation-based search method. Simulation-based path searching using a simulation method proceeds as follows: first, based on the network topology and resource information, a network simulation model is developed; when given the path's source, destination and its QoS requirements, a simulation is run by flooding the packets from source to destination. Each packet carries a frame with the following information: Destination, Path Constraint Requirements, Traveled path (The path that the packet has traveled), and Traveled Path's Attributes. By property 1 expressed in previous section, we can calculate each packet's Traveled Path's attributes link by link during

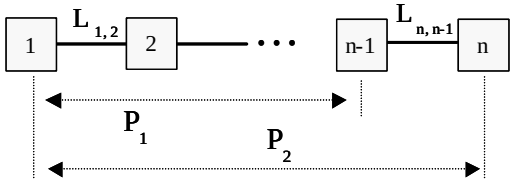

Figure 2: Path's Concatenation Property

the process of the packet's traveling to the destination. It could happen that before a packet reaches the destination; the attributes of the path it has traversed would have exceeded the constraint requirements. If so, the packet will be dropped. For each link, a "transmission delay" is assigned. When the first packet arrives at the destination, its traveled path is the one that meets the multiconstraint requirements; simulation stops. It could happen that there is no any path will meet the multiconstraint requirements in the network. The simulation will never derive a qualifying path. To watch over this case, the simulation will be given a maximum "STOP Simulation Time", after which, no matter whether a path is derived or not, simulation stops.

3. IMPLEMENTATION OF PSSM WITH DEVS

3.1 DEVS Background

The DEVS (Discrete Event System Specification) formalism [6] provides a means of specifying systems in a systematical and mathematical way. DEVS emphasizes the separation of two classes: model and simulator. The *Model* is a set of instructions for generating data comparable to that observable in a real system. The behavior of the model is the set of all possible data that can be generated by faithfully executing the model instructions. The *Simulator* exercises the model's instructions to actually generate its behavior.

The DEVS formalism provides a framework for the construction of hierarchical models in a modular fashion. DEVS models are built using a set of basic models called **atomic**, which can be combined to form **coupled** ones. In a DEVS, input events arriving in time are handled somewhat in the manner of interrupts by the modeler-specified external transition function and result in an immediate change in state. This function determines the state transition and how long to stay in the new state. At the end of this duration, the model outputs an event determined by the output function and transits to a new state determined by the internal transition function.

3.2 PSSM DEVS Simulation Model

When the network topology and resource information are available (can be obtained by underlying link state protocol or manual configuration), a simulation network model can be easily developed in DEVS. As is well known, there are three types of entities in a real network: Router, Link, and Packet. In the simulation network model, there are also three types of corresponding models:

- **Packet Model**, which models the behaviors of a real packet in the network.

A Packet is an entity having the following atttributes:

i.	*src (Source Address)*	*int*
ii.	*Dest (Destination Address)*	*int*
iii.	*reqAttr1 (Requirement for attribute 1)*	*double*
iv.	*reqAttr2 (Requirement for attribute 2)*	*double*
v.	*Attr1 (Attribute 1)*	*double*
vi.	*Attr2 (Attribute 2)*	*double*
vii.	*Travel_path (The path the packet has traveled)*	*string*

The values of Attr1 and Attr2 may be considered as analog parameters that contain information about the quality of service offered by the path under evaluation, traversed by this packet. The values of ReqAttr1 and ReqAttr2 are the constraint requirements for Attr1 and Attr2 respectively. The information collected by the packet is used to determine if its traveled path can meet the requirements. If the traveled path's accrued attribute values exceed the requirements, the packet is discarded during transit by the link or the routers. Only the 'fittest' packet arrives at the destination thereby setting up the path. Note that any finite number of attributes may be employed without changing the basic approach.

- **Router model**, which models the behaviors of a real router.

Each router consists of ports as depicted in Figure 3a. It shows a router having Id number 2 in the network. It receives packets from Routers with Id 1 and 3 and forwards packets to Router with Id 5. The functionality is depicted in Figure 3b and its DEVS description is as follows:

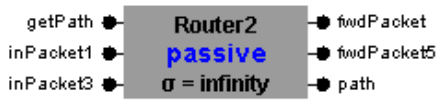

Figure 3a Router Model with inports and outports

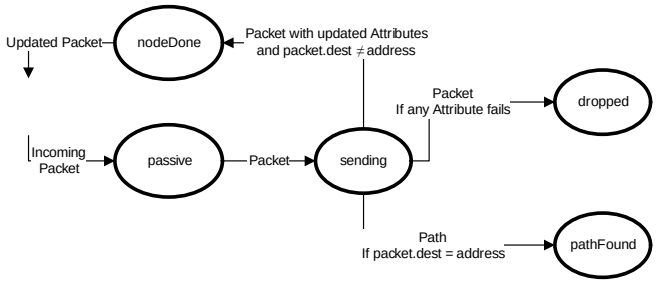

Figure 3b Router Model Finite State Transition Diagram

$Router_{DEVS}$ = (X, Y, S, δ_{ext}, δ_{int}, λ, t_a)
Where,
//pcktu = {src, dest, reqAttr1, reqAttr2, attr1, attr2, path}
InPorts = {"getPath", "inPacket"+nodeNum} where,
$X_{getPath}$ = {pcktu},
$X_{inPacket+nodeNum}$ = {pcktu},
where nodeNum = {1,2,3…n}

X = {(p,v) | p ε Inports, v ε X_p} is the set of input port and value pairs;

OutPorts = {"fwdPacket"+nodeDest, "path"}
where,
$Y_{fwdPacket+nodeDest}$ = {pcktu}
where ,nodeDest = {1,2,3 … n}
Y_{path} = {pcktu}

Y = {(p,v) | p ε OutPorts, v ε Y_p} is the set of output port and value pairs;

S = {"passive", "sending","nodeDone", "pathFound"}

//adr: = node address,
δ_{ext}((phase,σ, tc, adr), e, (p,v)) =
("pathFound", inf, v),
if phase="passive" &
(p="getPath" or p="inPacket"+nodeNum)
& pcktu.dest == addr
("sending", 0, v),
if phase="passive" &
(p="getPath" or p="inPacket"+nodeNum
& pcktu.dest ≠ addr
δ_{int}(phase,σ) =
("nodeDone", inf), if phase="sending"
("passive", inf), otherwise

δ_{conf}(phase,σ) = δ_{ext}(δ_{int}(phase,σ))

λ(phase,σ) =
("fwdPacket"+nodeDest, pcktu),
if phase="sending" & pcktu.addr
≠address
("path", pcktu),
if phase="pathFound" &
pcktu.dest == address
null, otherwise

t_a(phase,σ) = σ

- **Link Model**, which models the behaviors of a physical link in the network.

Figure 4a present an abstract level of the link that goes from Router Id 3 to Router Id 2. It is modeled using DEVS as follows:

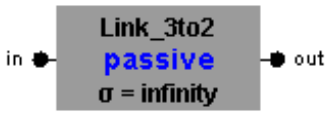

Figure 4a Link model with inport and outport

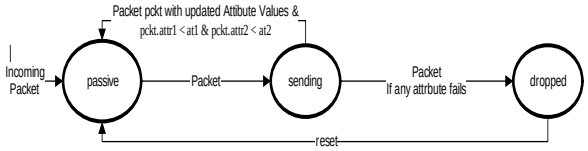

Figure 4b Link Model Finite State Transition Diagram

$Link_{DEVS}$ = (X, Y, S, δ_{ext}, δ_{int}, λ, t_a)
where,
//at1=reqAttr1 and at2=reqAttr2
InPorts = {"in"}, where X_{in} = {pcktu}

X = {(p,v) | p ε Inports, v ε X_p} is the set of input port and value pairs;

OutPorts = {"out"}, where Y_{out} = {pcktu}

Y = {(p,v) | p ε OutPorts, v ε Y_p} is the set of output port and value pairs;

S = {"passive", "sending", "dropped"}

//at1: =ReqAttribute1, at2: = ReqAttribute2
δ_{int}(phase,σ) = ("passive", inf), if phase = "dropped"
("passive", inf), if phase = "sending"

δ_{ext}((phase,σ), e, (p,v)) =
("sending", 0.05, v),
if phase="passive" & pcktu.attr1 < at1
& pcktu.attr2 < at2
("dropped", 1, v), otherwise

δ_{conf}(phase,σ) = δ_{ext}(δ_{int}(phase,σ))

λ(phase,σ) =
("out", pcktu),
if phase="sending" & pcktu.attr1 > at1
& pcktu.attr2 > at2
null, otherwise
t_a(phase,σ) = σ

4. APPLICATION OF PSSM IN REAL NETWORK

Since computing power has increased, and price has decreased greatly in the last few years, it is now economically feasible to include a DEVS simulation engine within a router. PSSM can be employed to derive real-time routing scheme.

One potential application of the PSSM methodology is the Automatic Switched Optical Network (ASON). ASON is an optical network that has dynamic optical channel connection and configuration capability. ASON's optical connections are established End-to-End. For example, as shown in Figure 6, when an optical connection request comes at an

edge optical router (source), the source edge router should find a feasible path that meets the QoS constraints [7] (e.g.

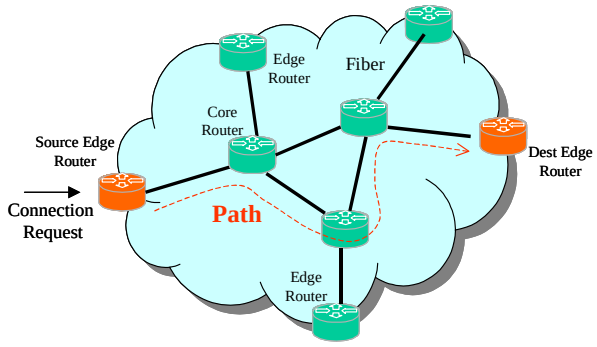

Figure 6: An ASON Network

optical power, optical signal-to-noise ratio (OSNR), and optical dispersion etc) to the destination edge router. Then the source edge optical router will send out signaling packets along the path to establish the connection.

In our ASON routing scheme, an edge optical router is expected to have an embedded DEVS engine, as shown in Figure 7. The router has the slot-based router architecture, which consists of multiple network interfaces (NIC), an interconnecting Backplane, and a routing module. The interconnecting Backplane interconnects the routing module and network interfaces. In the routing module, the routing protocol is a link state protocol, like OSPF [8]. It disseminates and floods link state advertisement (LSA) to obtain the network topology and resource information, and build the link date database. Based on the link state database, a DEVS simulation network model would then be developed. When an optical connection request comes, PSSM is employed upon the DEVS simulation network to

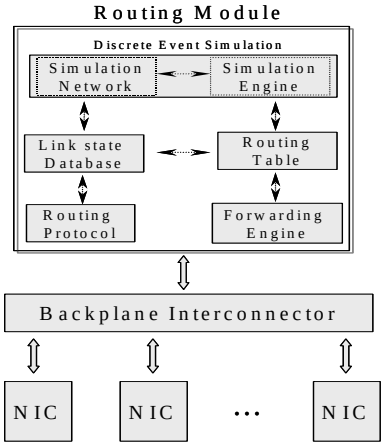

Figure 7 An Edge Router with DEVS Simulation engine

derive a feasible path from the source to the destination edge router.

Since an ASON's core router doesn't search for paths for optical connection requests, a core router does not have a DEVS simulation engine. But both the edge and core optical routers run link state routing protocols, to build routing tables in order to forward ASON signaling packets.

Monitoring in Real-time

The PSSM method will give accurate results only when the link state database within the router reflects the current state of the network. Obviously, the performance depends upon this correct information. The underlying routing protocol usually updates the link state database in two ways:

- **Periodic method**: Each of the routers updates its topology base periodically. Each router sends link-state advertisements (LSA) to every other router to get information about the state of the network
- **Quantized method**: The router updates its topology base and link state metrics only when a new requirement arrives or there is change in the configuration of the network (both router and link-attributes). Since the links are multi-attribute and have more than one constraint, a quantum is defined, based on which LSA are triggered. It works on the basis of threshold-mechanism.

Clearly, the periodic approach generally entails more traffic than the quantized approach since it continues to flood the network irrespective of any network-change. The quantized approach is much more adaptive. The quantum can be defined based on network characteristics and the updates are made when the network undergoes a significant change, as defined by the size of the 'quantum'.

5. SIMULATION EXPERIMENT

The entity models discussed in section 3.2 have been implemented in the DEVS simulation network model as shown in Figure 8. This figure shows a general layout of how a real-time simulation will be conducted. This simulation network can be run as a part of real-time system using model-continuity principles [9], which allows hardware-in-loop (HIL) simulation. As explained in Section 4 that the topology is created using Link-state database, we layout the simulation network using this real-time information. Now, this simulation network is required to come-up with a path from a certain source towards a certain destination allowing multi-constraint criteria. We give an external input to the simulation network, to the source node specifically, asking about the path towards a particular node. This node then executes the DEVS definition and floods the network with the request packets. These packets travel through node by node accruing the path as they reach the

destination. The first packet that reaches the destination terminates the simulation and the path that it traversed is communicated out of this simulation to the Routing table (Figure 7). In the figure below a request is made to Router Id 0 to find a path to Router Id 5. The simulation comes up with a path traversing Routing nodes with Id 0, 1and 2 with finally reaching ID 5. The path traversed (in red) is shown in the packet that is communicated to the Routing table. Also, notice the link (Link_4to5) which shows its state as 'dropped' implying that it has dropped the packet during the execution. Once the simulation has found the path, it resets itself so that the routing nodes that are in state "nodeDone" are ready for a next external request.

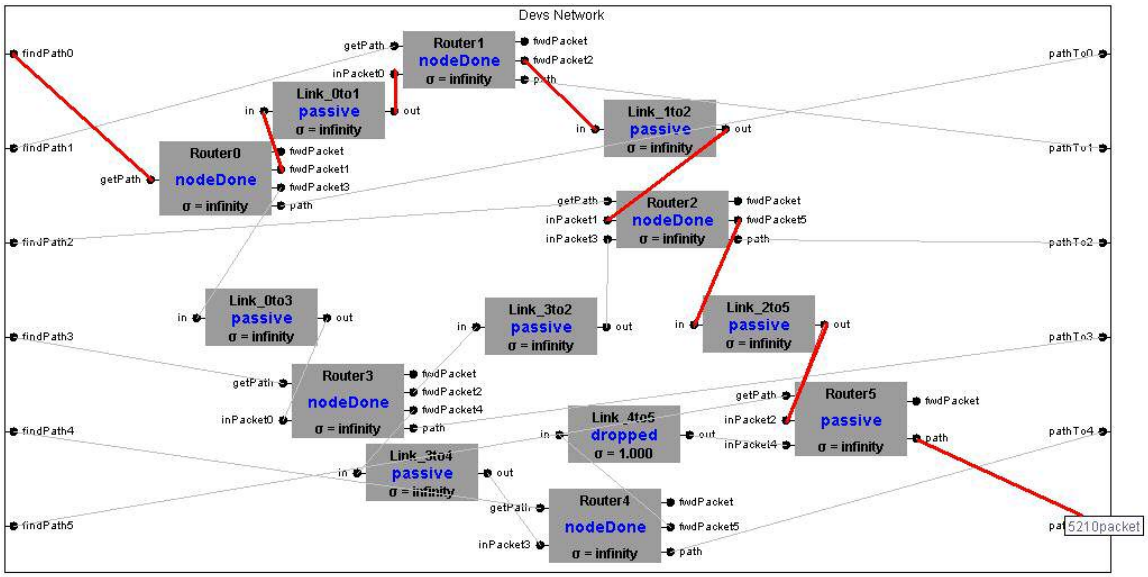

Figure 8: A simulation network showing the execution of PSSM algorithm

6. CONCLUSION

This paper has presented a simulation methodology to solve the MCP problem in the routing field. We have shown how the PSSM algorithm works, and discussed the implementation of PSSM with DEVS. Finally we apply PSSM to implement a multiconstraint-based real-time routing scheme using simulation methodology within the ASON context. We also intend to use model-continuity principles for testing PSSM algorithm in real-time systems.

ACKNOWLEDGEMENTS

This research has been supported in part by NSF Grant No. DMI-0122227, "Discrete Event System Specification (DEVS) as a Formal Modeling and Simulation Framework for Scaleable Enterprise Design".

REFERENCES

[1] F.A. Kuipers, T. Korkmaz, M. Krunz and P. Van Mieghem, Overview of Constraint-Based Path Selection Algorithms for QoS Routing, IEEE Communications Magazine, vol. 40, no. 12, December 2002.

[2] T. Korkmaz, M. Krunz and S. Tragoudas, "An Efficient Algorithm for Finding a Path Subject to Two Additive Constraints," Computer Communications Journal, Vol. 25, No. 3, pp. 225-238, Feb. 2002.

[3] J.M. Jaffe "Algorithms for Finding Paths with Multiple Constraints." Networks, 14:95--116, 1984.

[4] S. Chen and K. Nahrstedt, "An Overview of Quality of Service Routing for Next-Generation High Speed Networks", IEEE Networks, Vol.12, n. 6,1998.

[5] ITU-T Rec. G.8080/Y.1304, Architecture for the Automatically Switched Optical Network (ASON), November 2001.

[6] B.P. Zeigler, T.G. Kim, and H. Praehofer, "DEVS Framework for Modeling, Simulation, Analysis, and Design of Hybrid Systems," in Hybrid II, Lecture Notes in CS, P. Antsaklis and A. Nerode, Editors. Springer-Verlag, Berlin, pp. 529-551, 1996.

[7] J. Strand, A. L. Chiu, and R. Tkach, "Issues for Routing In the Optical Layer," IEEE Communication Magazine, pp. 81-87, Feb. 2001.

[8] J. Moy, "OSPF Version 2", Request For Comments: 2178, Network Working Group, April 1998

[9] Xiaolin Hu, "A Simulation based Software-development methodology for Distributed Real-time Systems", PhD Dissertation, University of Arizona 2004