IDEA : Integrated Design Environment for Assessment of ATM Networks*

Ricardo Fricks† Steve Hunter‡ Sachin Garg§ Kishor S. Trivedi
fricks shunter sgarg kst
@ee.duke.edu

Center for Advanced Computing and Communications
Department of Electrical and Computer Engineering
Duke University
Durham, NC 27708

Abstract

With the increased attention ATM is receiving to meet the needs of a wide variety of applications, tools are needed to help a network designer focus on the design at hand, rather than to spend time exhaustively learning the tools themselves. This is the concept behind IDEA. This paper introduces "modeling engines" chosen to be integrated into IDEA and presents a user interface for networking design. The first objective for IDEA is to demonstrate its usefulness for dependability modeling. Later objectives include incorporating performance and performability modeling.

1. Introduction

Emerging networks of today will be used for a wide variety of applications: teleconferencing, interactive multimedia, medical imaging and client server applications with each usually being *data sensitive* or *delay sensitive*. Data sensitive applications require that every piece of information transmitted across a network be correct, while the amount of time that it takes for the actual transmission may vary. Delay sensitive applications may tolerate some errors in data transmission, but the time that it takes for the data to be transmitted must be bounded. Client-server traffic is usually data sensitive while applications integrating voice and/or video are delay sensitive.

*This work was supported by a CACC core project funded in part by NASA Lewis Research Center and by the National Science Foundation under grant number EEC-9418765.

†Supported in part by a GTE fellowship.

‡On leave from IBM Corp., Research Triangle Park, NC.

§Supported in part by an IBM Fellowship.

Another characteristic of a network application is the amount of bandwidth it requires. Determining this requirement for voice and video may be straightforward, but determining what it should be for transferring a file depends on the user's delay tolerance. For example, a user continually receiving image files would desire more than a 14.4 Kbps link for network access. It is possible for the response time to be undesireably high even if enough bandwidth is available due to the latency of transmission.

Major issues with Gigabit networks are the relationship between latency and bandwidth and the amount of data in transit on a high-speed link at any given time [10]. Consider a Gigabit link that spans the United States. It would take approximately 15 ms for the beginning of a data transmission to reach from one coast to the other and could be followed by 15 million bits of data already in transit. This is the data that the receiving node must be able to immediately forward or do a temporary store and forward, but due to memory costs and the amount of processing power required, the temporary store and forward option currently is not feasible.

To help address these and other issues, Asynchronous Transfer Mode (ATM) is receiving increased attention and has become the transport mode of choice for broadband integrated-service networks [13]. One difference between ATM and some of the current networking techniques (e.g., routing, bridging) is the concept of establishing a virtual circuit or connection. That is, before a source node can transmit data, it must first set up a connection with a destination node. The source sets up this connection by passing an address of the destination, the amount of bandwidth required and the other Quality of Service parameters using a signaling protocol. The ATM forum has standardized Q.2931 as the signaling protocol and is described in the ATM UNI specification [1, 2]. When a connection is established, it is assigned a Virtual Path Identifier/Virtual Circuit Identifier which is unique for every Virtual Channel Connection.

Due to ATM's wide acceptance, not only are performance issues important (such as those mentioned above), but the dependability of networking services is also becoming more important [16]. Therefore, since different performance and dependability attributes can have varying levels of importance depending on the situation, a formal methodology is needed to optimize a design for specific requirements.

1.1. Need of an Integrated Design Environment

The use of analytic modeling tools is limited at present in industry involved in ATM development despite the availability of a large set of performance/dependability/performability modeling tools. Tools based on discrete event simulation do find a limited use; however, they are usable only for performance modeling. A designer needs both performance and reliability modeling tools that allow an assessment of the tradeoff between the two measures. For a survey on modeling tools, the reader is referred to [18].

The basic reason behind this is that the analysis needed in the process of ATM network design and development involves a plethora of activities ranging from dependability, performance and performability modeling at the component, node and network level. Issues such as assessment of potential traffic, ease of network management, etc. also need to be considered. Currently available tools, however, are mostly restricted to a certain domain with no built-in support for assessment of ATM design. A designer has to look for a suitable tool every time he/she has to solve a particular problem. This in itself may not be so troublesome. Upon selection, a more daunting task presents itself - that of translating the problem into one which the tool understands. Needing to know the internals of every tool to be able to obtain the desired output results in loss of precious time and resources, which we know, is crucial in the competition that prevails in today's information technology industry.

The primary purpose of the IDEA project is to integrate off-the-shelf modeling tools into an user friendly environment geared towards the analytical assessment of ATM network designs.

IDEA integrates many existing off-the-shelf tools for performance, reliability and performability modeling providing a common front end to a user. In this way, the problem description is specified in the same language which in turn shall have both textual and graphical user interface. The front end in the IDEA package provides an environment which is specifically geared towards assessment of ATM design. Moreover, a database is incorporated to contain parameter values, real data, and previously assessed designs (submodels) to be able to perform validation and encourage the philosophy of hierarchical modeling.

1.2. Related Work

There have been but few attempts in the past to integrate different tools by providing a common environment and to the best of our knowledge, none geared specifically towards assessment of ATM design. SHARPE combines different reliability and performability modeling paradigms such as fault trees, reliability graphs, stochastic Petri nets and queuing networks in a single tool. An effort in the direction of integrating off-the-shelf tools is made in the SDM [14] project. It defines an integrated reliability modeling environment for a heterogeneous hierarchical system description, thus providing a common description language for system description, which translates into suitable code understood by CAFTA, SETS, HARP, EHARP, SHARPE or SPNP.

The rest of the paper is organized as follows. Section 2 addresses a design process for ATM networks and the scope of the IDEA package. Section 3 details the first phase of the *IDEA* project. Section 4 covers software architecture issues. Section 5 provides a current implementation with an example and Section 6 concludes with future steps to be taken.

2. Network Design Process

As shown in Figure 1, there are several phases within the design process, the first being the *requirements* phase. It is during this phase that requirements are gathered for a system or network. The terms listed below can be thought of as attributes of a design (sometimes called design factors) that the requirements can be mapped directly or indirectly into.

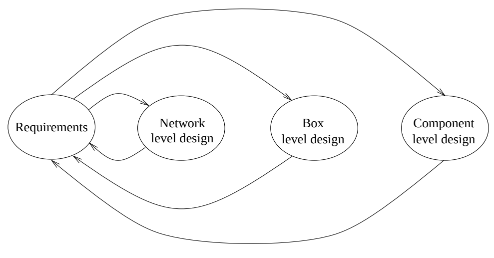


Figure 1. Design Process

- **Performance Measures**: The measures used for the network performance can vary depending on perspective. For example, from the perspective of an end user, response time is of interest and it can include the protocol stacks of a workstation. However, from the perspective of network management, link utilization may be more important. We consider the following measures.
  - *Capacity*: Determining the desired capacity or bandwidth of a network is often associated with *capacity planning* which requires analysis of existing network traffic and forecasting of expected network load at future points. The capacity needed should be sufficient to handle the aggregate traffic streams of simultaneous users. The works on the topic of *equivalent capacity* or *effective bandwidth* [9] has received significant attention among the industry as well as the research communities.
  - *Latency (Response time)*: The latency of a network is the length of time required for data to be transmitted by a source and received by a final destination. It is primarily due to propagation delay of links and buffering at nodes. Related to latency is *jitter*, which is the variation of latency over time, and *skew*, which is the difference in time between two related things, such as video and the corresponding audio.
  - *Loss probability*: For ATM, this is the probability that a cell does not reach its destination due to buffer overflow or transmission errors.
- **Dependability Measures**: Dependability is used as a catch-all term for measures which include the following:
  - *Reliability*: The reliability $R(t)$ of a network is its ability to maintain failure-free operation in the interval $(0, t)$. The mean time to failure (MTTF) is given by $\int_0^\infty R(t)\,dt$.
  - *Serviceability*: The serviceability of a network is the ability to service or repair the system in a timely manner when a component or network failure occurs.
  - *Availability*: The availability is often expressed as *instantaneous availability*, $A(t)$, and/or the *steady-state availability* (i.e., $\lim_{t\to\infty} A(t)$). The instantaneous availability, $A(t)$, is defined as the probability that the network is operational at time t. It allows for one or more failures to have occurred during the interval $(0, t)$.
- **Performability**: Recognizing that for a network with degradable performance, it is important to combine performance and dependability into composite measures, sometimes called performability measures [12].
- **Cost**: Cost should always be taken in consideration as options for a design are being compared or while making tradeoffs.

The importance, in relation to each other, and bounds of each attribute should be determined for the network as part of the requirements phase. This resulting information is used to enter the *network level design* phase. Note that the requirements phase is reentered after exiting other phases. This could be useful in verifying that the requirements of that phase have been met; alternatively, it may be learned that the requirements are too strict.

Network level design involves determining potential topologies for the network by taking into consideration applications to be used, distances to be traveled, technologies available, etc. When exited, the requirements phase is passed through before entering the *node level design* phase.

It should also be pointed out that the design process shown is iterative. That is, once you exit a phase it may need to be revisited to modify one or more of the above attributes. For example, if it is desired to have a highly available product, it should be determined whether failures should be very infrequent or periodic failures are acceptable as long as they are repaired quickly.

The node level design phase addresses each box or node in the network to determine what best meets requirements. The attributes listed above are also relevant to this phase. When exited, the requirements is passed through and the *component level design* phase entered.

Since component level design primarily deals with individual components such as cards, this phase may require only the collection of information that relates to the attributes above.

3. Functional Specification

The first phase of implementation of IDEA will implement a complete dependability modeling environment for ATM networks, and will establish the foundation for the other phases. The output of this implementation is named **IDEA-kernel** because everything else shall be developed around it.

As a whole the project will emphasize the integration of off-the-shelf software components (data base managers, graphical interface builders, etc.) to speed-up the development process and to enhance the quality

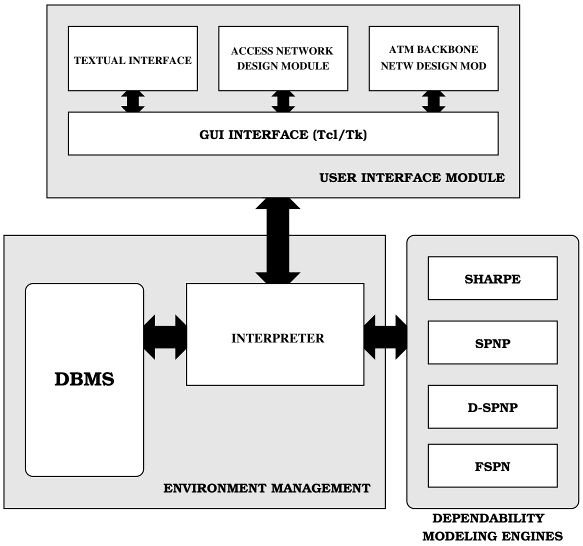


Figure 2. Functional Specification.

of the final product. The kernel of the integrated environment is designed based on three functional pillars (see Figure 2): user interface, dependability modeling engines, and environment management modules. This organization allows for incremental expansion of the environment with the addition of new tools or integration of support algorithms (modeling/optimization).

In the first stage, here considered, IDEA will only provide interface with human operators, although automatic data acquisition is under consideration for implementation in later versions of the environment. The *User Interface Module* is composed of two major components: a textual interface (TI) and a graphical user interface (GUI). The TI component should support a dependability modeling description language adapted for network modeling at both levels (network access and ATM backbone). The GUI should implement all the expected functionality of a computer-aided design interface for two-dimensional designs, including features like levels of zooming, panning, scrolling, macro functions, extensive report capabilities, etc.

Through the GUI the user shall be able to retrieve/input/modify the network link and node hardware stochastic dependability models and corresponding parameters (e.g., failure rate, failure distributions, and associated parameters), or data base view and the parameters shall be able to be saved in the database or corresponding device library.

The *Dependability Modeling Module* converts the ATM network into the most appropriate stochastic model such as Markov or Semi-Markov processes, reliability block diagrams, fault trees, etc. Network dependability characteristics are then evaluated based on the switch architecture, network topology and other applicable results.

For the first release, the nodal analysis is restricted to single stage core switches, including shared bus, memory, and cross point switch. In addition, capability is also needed in order to model the entire switch node. Notice that some core switches are in fact a multi-stage interconnection of small switching units. Furthermore, the network traffic control mechanism is not included in the first release specification as the control algorithm has not yet been determined by a standards body.

The *Environment Management Module* acts as a general service module for the other two. Some of the features available at the user interface level need strong support from the data management level, which is not only responsible for providing an uniform interface with the modeling engines, but also to keep a data base (actual and historical) of designs and of additional information necessary to smooth the operation of the whole environment.

4. Software Architecture

The functionality of the IDEA kernel is made available through the assistance of four software modules and three independent tools (termed *auxiliary tools*), as shown in Figure 2. At first, the kernel will allow only dependability analysis, but later performance and performability analyses will also be available through the common environment. Due to its flexibility, availability and quality, the initial auxiliary modeling tools selected are SHARPE, SPNP, D-SPNP, and FSPN. The aggregate of these modeling packages constitutes the dependability modeling engines of IDEA kernel, and establishes the fundamental orientation adopted in the overall design of IDEA: "*IDEA is not supposed to be another modeling package per se, but it is to integrate off-the-shelf modeling packages in a single environment, sharing a common user interface oriented towards ATM network design.*"

The integration of the modeling engines is provided through an interpreter that maps one intermediate language implemented by the user interface into the input languages of the auxiliary tools (e.g., CSPL for ths SPNP package). The interpreter concept allows for continuous growth of the modeling engine module, either in the number of supported tools or their characteristics. New modeling packages shall be easily integrated, providing freedom for each individual installation of the environment to select tools more appropriate for their particular use.

Additional automation implemented in the interpretation process is the automatic selection of the auxiliary package(s) most suitable for a particular problem. The selection process can be overridden by the user. This is an additional feature implemented to allow IDEA to accommodate several distinct types of users with different modeling skills. If the user has a strong modeling background he or she will be able to select and work directly with his or her choice of auxiliary tool, but even in this case IDEA provides a superior interface than the individual auxiliary packages themselves. The interpreter also provides a bidirectional interface with the auxiliary tools. Not only mapping the intermediate language into the auxiliary packages' input languages, but also it takes results from several packages and maps them into a common interface to present to the user through tables and appropriate plots.

Another important part responsible for implementing the expected functionality of the Environment Management Module is the data base management system (DBMS). The DBMS assists the interpreter as the intercommunication medium for various tools of the environment, besides being the information source for all of them. The DBMS adopted for the IDEA kernel is an off-the-shelf product allowing good performance, excellent interface and report capabilities.

The GUI will also provide a textual interface, necessary if we wish to accommodate several categories of users. Careful design of the textual interface is necessary since this will be the intermediate language, that the interpreter will understand.

In the subsections that follow we provide further details regarding off-the-shelf tools that will be integrated in the IDEA kernel.

4.1 Data Base Management System

The DBMS module is responsible for storing network design data (e.g., trunk parameters and switch/router data). Besides, it also serves as a common interface and I/O communication platform for various tools such that the output of one tool can readily serve as input to the rest of the modules. Each module (mathematical engine or other tools) output can be written into the DBMS in the format that other modules will be able to use as input without modification.

This arrangement has two benefits. First, human intervention can be minimized during the network design process. This will reduce the required expertise of network design engineer's capability in understanding and manipulating the internals of the design tools. And finally, various design tools from different vendors can be easily integrated as part of the environment. In view of the scarcity and diverse expertise of ATM network designers today, this is especially important for deploying the ATM network design tools in a short time frame and yet be able to utilize the state of the art methodology in the field.

4.2 SPNP

SPNP (Stochastic Petri Net Package) has been developed by Ciardo et.al. [7]. The model type used for input is a *stochastic reward net* (SRN) SRNs incorporate several structural extensions to GSPNs [3] such as marking dependencies (marking dependent arc cardinalities, enabling functions, etc.) and allow reward rates to be associated with each marking. The reward function can be marking dependent as well. They are specified using CSPL (C based SRN Language) which is an extension of C with additional constructs for describing the SRN models. Whereas CSPL exploits the full power of C and makes the SRN specification very flexible, it also makes it imperative that the user knows C. There is no interactive interface, but a graphical interface is near completion. The user can either specify the SRN graphically or type in the CSPL file.

SRN specifications are automatically converted into a Markov reward model which is then solved to compute a variety of transient, steady-state, cumulative, and sensitivity measures. Standard measures such as average number of tokens in a place, average throughput of a timed transition, probability (transient or steady-state) that a place is not empty, and probability that a timed transition is enabled can be computed. More "customized" measures can be requested by means of user defined reward functions. For a given reward function, expected value of the function, expected accumulated reward over a finite time interval, and time-averaged expected value can be computed. For SRNs with absorbing markings, mean time to absorption and expected accumulated reward until absorption can be computed. For details on how to use SPNP software, the reader is referred to [5]

4.3 SHARPE

SHARPE (Symbolic Hierarchical Automated Reliability/Performance Evaluator) was originally developed in 1986 by Sahner and Trivedi. The main feature of SHARPE is that hybrid and hierarchical models can be easily constructed [15]. The overall system model may consist of several submodels of possibly different types. The model types allowed are fault-trees, reliability block diagrams, reliability graphs, Markov chains, semi-Markov chains, single and multi-chain product form queuing networks, GSPNs, and series-parallel task graphs. For example, the reliability of a system can be modeled by a reliability block diagram where reliability of each block is computed by solving a Markov chain. If a single Markov model for such a system was constructed, then it could have a state space of exponential size. Hierarchical modeling alleviates the problems of model largeness and stiffness to a large extent.

SHARPE can compute distribution function (symbolic in t), mean, and variance of time to failure of the system. User may opt for numeric solution in which case a numeric value of the distribution function at a given time t is computed. If failure probability or availability (instantaneous or steady-state) is specified for each component, then system failure probability or system availability is output. Specialized solution algorithms are used for different model types.

4.4 D-SPNP

One of the key difficulties in using state space based models like SRNs is the size of the underlying state space. As a single workstation tool, SPNP, SHARPE etc. are limited to small problems whose state space can be stored within the workstation's memory. One possible solution to the problem is to parallelize the generation of the underlying Markov chain. Ciardo et. al. from College of William and Mary have developed an efficient algorithm for distributed state space generation. A prototype distributed engine has also been developed for generating the underlying state space associated with SPNP models. This prototype is referred to as the D-SPNP. The current implementation of the engine can be run on either a set of networked workstations or on a SP/2 multiprocessor. For details on D-SPNP, please see [6].

4.5 FSPN

Another approach to tackling the state space explosion is to use models based on continuous flow rather than discrete movement of tokens. By incorporating this idea into the SPNs, Kulkarni et. al. [17] have developed a modeling approach using Fluid Stochastic Petri Nets (FSPN). The use of FSPNs is particularly suited to modeling performance issues in ATM networks as the traffic always consists of small cells of the same number of bits. Fluid flow theory, although not in the framework of Petri-nets has already been effectively used in ATM network performance modeling [4]. FSPNs and corresponding solution engine will enable the analyst to easily specify and automatically generate/solve fluid models. For details on how such models are specified and solved for desired measures, the reader is referred to [8].

5. Example

We now illustrate how far the objectives of the proposed tool are met through an example. Assume that the task is to support a client-server application over an ATM network. Given the quality of communication requirements expressed in terms of the reliability and availability of service, the designers have come up with a solution of using 4 switches in addition to a gateway. To connect these switches the topology shown in Figure 3 is proposed. We shall refer to this topol-

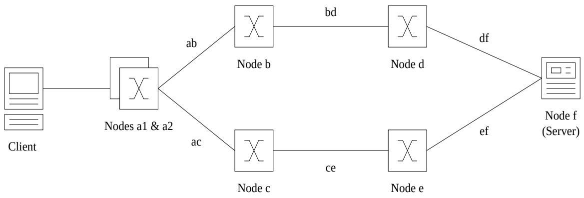


Figure 3. Network Configuration : Topology 1

ogy as "net-1". Note that a virtual path connection between the client and the server has two choices of underlying physical paths given by `a-ab-b-bd-d-df` and `a-ac-c-ce-e-ef`. Thus even if a switch or a link in a particular path is down, thus causing that physical path to fail, the virtual path can be maintained by using the other route. To provide more alternative paths and therefore more reliability, the designers want to explore the possibility of putting another link `be` between switches `b` and `e`. Now, four alternate physical routes are available for establishing a virtual path between client and server. These routes are `a-ab-b-bd-d-df`, `a-ac-c-ce-e-ef`, `a-ab-b-be-e-ef` and `a-ac-c-ce-e-be-b-bd-d-df`. We shall refer to this topology with additional link `be` as "net-2" shown in Figure 4.

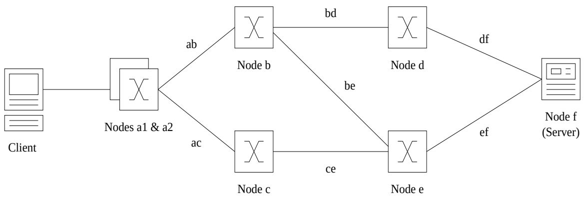


Figure 4. Network Configuration : Topology 2

Our objective is to model the two alternatives and predict the gain in the availability of a connection. We assume that there is a single facility to repair any failed switches and links. For dependability modeling of systems having components with dependent repair, state space models and specifically, stochastic reward nets are most appropriate. For details on power hierarchy of dependability model types, we refer the reader to [11]. IDEA provides support for creating (editing, debugging and correcting) and executing SRN models. In the current implementation, a textual editor is used to specify the SRN which will be replaced by an easy-to-use GUI, nearing completion.

Figure 5 shows the stochastic reward net model for net-2. The reward function is specified in terms of boolean functions which describe whether a virtual path is established or not. This net is coded in CSPL [7] using the editor support provided by IDEA.

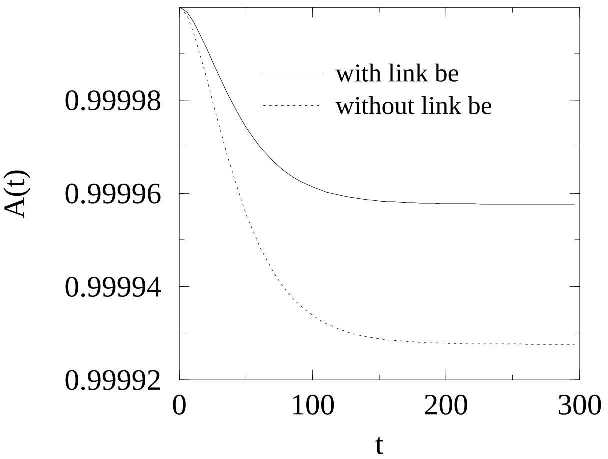


Figure 6. Availability comparison.

From Figure 6, it is clear that the availability with link `be` is increased over the case when `be` is not present. The decision to add the link is still to be made.

CASE 1: If net-1 does not satisfy the specified availability requirements and net-2 does, adding `be` is an obvious choice.

CASE 2: However, net-1 meets the basic requirements and adding `be` means improving the quality of service only, the decision needs to take into consideration the cost of adding `be`.

In the current implementation of IDEA, CASE 1 is easily evaluated but there is no support for cost analysis to evaluate CASE 2.

6. Conclusions

With ATM technology fast becoming a standard for B-ISDN, the need for quick, effective and accurate analysis of the design issues involved in deploying this technology is also growing. An ATM designer is faced with numerous options from which he needs to pick the best considering all (possibly conflicting) tradeoffs of performance, reliability, availability, safety, quality of service guarantee, and last but not least, cost. We proposed an integrated environment, which we call IDEA, to help the designer to make such a decision.

Using off-the-shelf modeling tools, a GUI based environment is being developed which eases the specification of the design problem and uses one or more of the tools provided to solve the model. In our current prototype, using the philosophy of incremental design and implementation, we have restricted ourselves to integrating SPNP, SHARPE, D-SPNP and FSPN in a single environment and provide easy to use GUI based support to do model specification, editing, debugging, execution and result generation for dependability modeling.

The IDEA-kernel project has met its initial objectives. The possibility of integrating diverse modeling tools through a common interpreter has been successfully demonstrated in the SDM environment [14]. A spin-off project of IDEA named "iSPN: an integrated environment for SPNP" is nearly complete. The purpose of iSPN is the careful design and implementation of a GUI for SPNP, crafting recent advances in user interaction into the design. The role of this project is central to IDEA since it establishes a template to be followed in all subsequent GUI developments. The implementation of the IDEA-kernel is now entering a new phase where the designed GUI will be adapted for other modeling tools and the integration experiment will be extended.

References

[1] ATM Forum, *ATM User-Network Interface Specification, Version 3.0*, Prentice Hall (ISBN 0-13-225863-3), September, 1993.

[2] ATM Forum, *ATM User-Network Interface Specification, Version 3.1*, July 21, 1994.

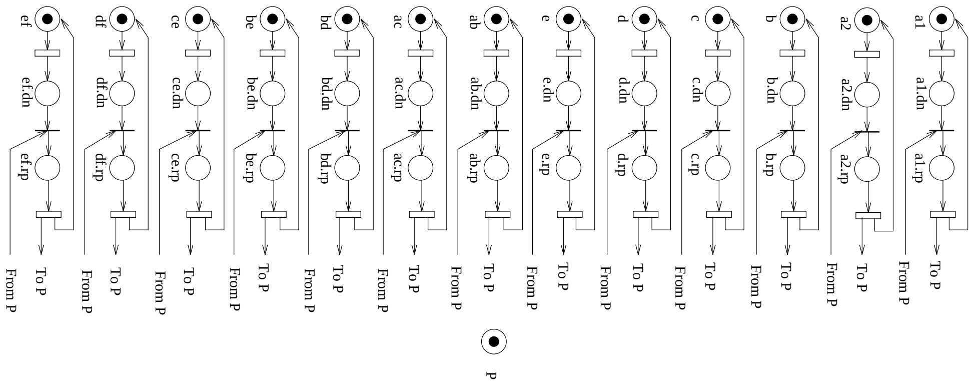
[3] M. Ajmone-Marsan, G. Conte, and G. Balbo. "A class of Generalized Stochastic Petri Nets for the performance evaluation of multiprocessor systems," *ACM Transactions on Computer Systems*, Vol. 2, No. 2, pp. 93-122, 1984.

[4] D. Anick, D. Mitra and M. Sondhi, "Stochastic theory of data-handling systems", *The Bell System Technical Journal*, Vol. 61, No. 8, pp. 1871-1894, October 1982.

[5] Ciardo, G., Muppala J. and Trivedi, K. S., "SPNP: Stochastic Petri Net Package", *Proc. Third Int. Workshop on Petri Nets and Performance Models (PNPM'89)*, Kyoto, pp. 142 - 151, 1989.

[6] G. Ciardo, J. Gluckman and D. Nicol, "Distributed state space generation of discrete-state stochastic models", to appear in *ORSA Journal on Computing*.

[7] G. Ciardo, A. Blakemore, P.F.J. Chimento, J.K. Muppala, K.S. Trivedi, "Automated Generation and Analysis of Markov Reward Models



Name	Function
G1	(#(a1.dn) == 1) ∧ (#(a2.dn) == 1)
G2	(#(b.dn) == 1) ∨ (#(be.dn) == 1) ∨ (#(e.dn) == 1)
G3	(#(bd.dn) == 1) ∨ (#(d.dn) == 1) ∨ (#(df.dn) == 1)
G4	(#(ac.dn) == 1) ∨ (#(c.dn) == 1) ∨ (#(ce.dn) == 1)
G5	G1 ∨ (#(ab.dn) == 1) ∨ (#(b.dn) == 1) ∨ G3
G6	G1 ∨ G4 ∨ (#(e.dn) == 1) ∨ (#(ef.dn) == 1)
G7	G1 ∨ G2 ∨ (#(ab.dn) == 1) ∨ (#(ef.dn) == 1)
G8	G1 ∨ G2 ∨ G3 ∨ G4

Figure 5. SRN model

using Stochastic Reward Nets", in: *Linear Algebra, Markov Chains, and Queuing Models*, Editors: C. Meyer and R. J. Plemmons, Vol.48 of *IMA Volumes in Mathematics and its Applications*, Springer-Verlag, 1992.

[8] G. Horton, V. Kulkarni, D. Nicol, and K. Trivedi, "Fluid stochastic Petri nets: theory, application, and solution", Technical Report 96-5, *Institute for Computer Applications in Science and Engineering*, 1996.

[9] Frank Kelly, "Notes on Effective Bandwidths", In: *Stochastic Networks: Theory and Applications*, Editors: S. Zachary, I.B. Ziedins and F.P. Kelly, Oxford University Press, 1996.

[10] L. Kleinrock, "The latency/bandwidth tradeoff in gigabit networks," *IEEE Communication Magazine*, pp. 36-40, April 1992.

[11] M. Malhotra and K. S. Trivedi, "Power-hierarchy of dependability model types", *IEEE Transactions on Reliability,* Vol. 43, No. 2, pp. 493-502, Sept. 1994.

[12] J.F. Meyer, "On evaluating the performability of degradable computer systems", *IEEE Trans. on Computers.* C-29(8):720-731, 1980.

[13] R. O. Onvural, *Asynchronous Transfer Mode Networks: Performance Issues*, Artech House, 1995.

[14] A. V. Ramesh, K. S. Trivedi, A. K. Somani, D. W. Twigg, U. R. Sandadi and T. C. Sharma, "Integrated reliability modeling environment", *submitted for publication.*

[15] R. Sahner, K. S. Trivedi and A. Puliafito, *Performance and Reliability Analysis of Computer Systems: An Example-Based Approach Using the SHARPE Software Package*, Kluwer Academic Publishers, 1995.

[16] *Special Issue on Dependability of Networking Services*, IEEE Communication Magazine, June 1993.

[17] K. Trivedi and V. Kulkarni, "FSPNs: Fluid Stochastic Petri Nets," *Proc. 14th International Conference on Application and Theory of Petri Nets*, Chicago, Jun. 21-25 1993.

[18] K. S. Trivedi, B. Haverkort, A. Rindos and V. Mainkar, "Techniques and tools for reliability and performance evaluation: problems and perspectives", *Computer Performance Evaluation: Modeling Techniques and Tools,* Lecture Notes in Computer Science 794, G. Haring and G. Kotsis (eds.), Springer Verlag, pp. 1-24, 1994.