Multimedia Network Simulation with a Fluid Flow Model

***Priscila Solís Barreto* [1]** ***Tiago Flor Bento*[2]** ***Paulo H. P. de Carvalho*[2]**

[1]Department of Computer Science, [2]Department of Electrical Engineering, University of Brasilia, Brazil

Summary
This work presents the modeling and design of a multimedia network simulation tool based on the fluid flow model. The tool uses continuous fluid flows instead of individual packets for network simulation and proposes their use to model multimedia network traffic. The main goal of the simulation using this approach is to produce a fewer number of events to reduce computational costs for large and complex networks simulations. Several simulations were executed with the proposed tool and the results show that the performance and quality of service values are similar to the ones obtained in the packet event based simulations. This shows that the proposed tool may be useful in the simulation of large and complex networks with multimedia traffic.

Key words:
fluid model, multimedia, network simulation, traffic.

1. Introduction

The multimedia networks had in the last years a very considerable growing. The complexity involved in the analysis of these networks make difficult the application of the traditional analytical methods. As an example, the multimedia traffic results when integrating, over a single communication channel, a diverse range of traffic sources [1,2] with different statistical properties.

Different observations of operating networks confirm that the aggregated traffic in the cores, i.e. the multimedia traffic, has a self-similar and multifractal nature [3,4,5,6,7,8,9] which results from multiplexing different sources, that significantly differ in their traffic finite buffer systems when compared to the Markovian traffic sources. Nevertheless, when analytical models may be used to obtain approximation to these processes, the network simulation task is used to complement the model validation process [7].

The simulations at the packet level has been broadly used to evaluate different network characteristics [7,8]. Even when this simulation technique may be very useful and well-accepted to measure and plan network performance, it may become very costly due to the size and complexity of the simulated networks [9,10].

It is well known that the packet based simulators are tools that with very acceptable degrees of precision may model a network and provide information about its elements. But in the practical field, this type of simulation has a computational limitation, even for memory capacity or for process time for large and complex networks [7,9,12].

With the goal to develop more efficient tools for simulation, different techniques have been tested in the last years. These techniques may be divided in three groups [13]:

(a) models based in the computational power, that increase the simulation efficiency with the use of grids and parallel processing;

(b) models based on simulation technologies that use more efficient algorithms;

(c) simulation models which are based in increasing the simulation abstraction level, to reduce the number of events and promote efficiency gains.

This work focus in the development of a tool that is in the simulation models category.

The fluid model treats the traffic as a continuous fluid flow instead of discrete packets. A fluid simulator records the changes in the fluid rate in the source and the queue, while a packet simulator records the events of all the packets in the system [9,10,14]. For this, is expected that the necessary number of events to represent the same traffic source may be less in a fluid simulation.

This paper is organized in the following parts: in section 2 are presented the main characteristics and theoretical concepts for the comprehension of the fluid model. In Section 3 are presented the details and conceptual premises of the proposed tool. In Section 4 are shown experiments and comparative results with other network simulators. Finally, in Section 5, the conclusions and future works of this research are discussed.

2. The Fluid Model

A simulation model is event oriented when the simulation process advances according to the occurrence of events in variable time slots. In a time oriented simulation model,

Manuscript received June 5, 2010
Manuscript revised June 20, 2010

the simulation process advances in uniform time slots. One of the most used methods for network simulation is based on packet discrete events. This method may model a system with a very accurate precision, but as a counterpart, the computational cost may be very high due to the large number of generated events [8,12,13,16].

The main idea in the fluid model is to make an abstraction of individual packets. For the case of packets based simulation, the simulator has to treat all the packets events in the network, while in the fluid simulation, it is only necessary to treat the rate change events in different network nodes.

The abstraction takes place when, packet flows with little time slots separations are considered in the same fluid flow with a constant fluid rate. Little time variations among packets are not considered and in this way, the number of events is reduced.

In a packet event simulator, the number of events is proportional to the number of packets produced by the traffic source, while in a fluid simulator; the number of events is proportional to the number of rate transitions [13]. Figure 1 shows an example of the packet abstraction for a fluid flow model.

2.1 Fluid Queue Model

One important difference between a packet queue and a fluid queue is that in the last one, more than a flow may share the queue service rate. Simultaneously, different flows may arrive from different sources.

As shown in [13], the total arrival fluid rate in a queue with N fluid sources, with a FIFO scheduling discipline and a service rate equal to c is shown in (1), in which $a_k(t)$ is the fluid arrival rate of the k^{th} queue source.

$$a(t) = \sum a_k(t) \quad (1)$$

The queue size dynamic may be described by the recursive relation as showed in (2) for $\tau_i \leq t \leq \tau_{i+1}$, in which τ is the time in which the aggregated arrival fluid rate changes.

$$q(t) = \max(0, q(\tau_i) + (a(t) - c)(t - \tau_i)) \quad (2)$$

Consider a queue with size B between 0 and ∞. The fluid quantity in the buffer in a time slot t will be 0 or the fluid quantity in the buffer plus the fluid quantity that overpass the queue service rate since time slot τ_1 is given by $a(t)$-c.

For a queue with a finite buffer size $B<\infty$, fluid losses occur when the buffer is saturated in certain time slot and the aggregated arrival rate is greater that the service rate, i.e. $a(t) > c$. This case is shown in (3)

$$q(t) = \min(B, q(\tau_i) + (a(t) - c)(t - \tau_i)) \quad (3)$$

2.2 Ripple Effect

A change in a flow rate that feeds a queue induces a change in the initial fluid flow rate of all the others flows that share the queue at the same time. This produces an increment in the number of events that will be executed during the simulation because each new flow will be a new event.

The above situation occurs due to the change in the output rate of the flow or due to the rate alteration of the others flows that share the queue.

The number of changes in the output processes will be greater than the number of changes in the input processes and this will propagate throughout the route. Since the output flow affects all the flows in the same route, the change in the fluid rate will continue to increase. This phenomena is the Ripple effect that may turn the fluid simulation more complex than the discrete event packet case. The results presented in [13] show a network with serving politic WFQ (Weighted Fair Queueing) which has the Rippley effect in a lower degree than the FIFO (First In First Out) queue.

2.3 Fluid Flow Source

Consider a group of packet flows that are randomly separated. To make the abstraction of this flow into a fluid flow, a time slot is chosen. The time slot defines a new block configuration that has an inter-block time slot definition. A fluid is a set of grouped packets in which, the beginning and end among flows is defined by the inter-block time slot which must be chosen before the abstraction process

In this way, the inter-block time slot will be the smaller time among packets that belong to the same fluid. The next packet that will be generated with a bigger time slot than the inter-block time slot will belong to the next fluid. In this way, a fluid traffic flow may be generated from a packet traffic file. In Figure 2, the above concept is illustrated.

The Markov model with two states on/off may be used for the modeling of a packet source, as showed in Figure 3. The transition rate from the state on to the state off and vice-versa is represented by λ and μ, respectively. The time of each state is exponentially distributed. When the source is in the on state the packets are generated with a y rate.

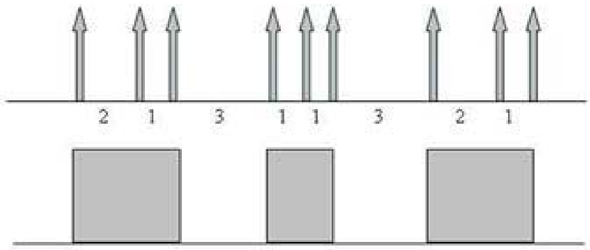

Fig. 2 Fluid Flow Modeling

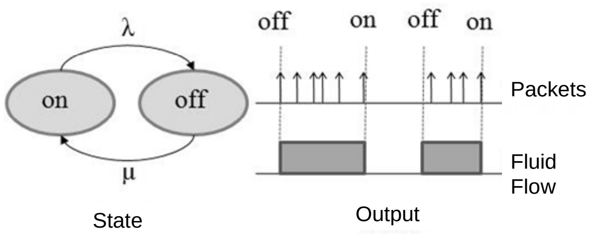

Fig. 3 Packet Source

The packet size may be random or constant. For fluid sources the idea remains the same. When the source is in the on state, the fluids are generated as flows at a constant rate. When the source is in the off state, the flow is interrupted. As the packet flow generation, the fluid flows may also follow a certain probability distribution.

The fluid rate generated at the source is constant but it may change when the fluid passes a network element. Then, different levels are generated for the fluid rate which produces variations in its intensity.

2.4 Fluid Molecules

During the abstraction process may be lost information about each individual packet. The concept of fluid molecules is used to diminish this loss.

The fluid molecules are emitted by the sources. These elements do not have a volume and do not affect the queues. They are mainly used to measure the quality service during the simulation [13]. The fluid molecules are discrete objects which are generated by the source and traverse all the network elements.

One important molecule characteristic is that it can work in all the network elements that traverse. Always when a molecule arrives to a network element, it can be manipulated and its actions are defined within its programming elements.

3. The Fluid Flow Simulator

The built simulation tool in this work has a modular library of network elements, a simulation core and a system for collecting results. The Java language was used for programming.

The tool base level is the Java virtual machine. The second level is the simulation core, which is mainly a manipulation mechanism for discrete events, responsible for the simulation execution, time management and process execution of network elements. The core was constructed from a SSF (Scalable Simulation Framework) [21]. The main goal of this framework is to support high performance simulations [9,21, 22].

The network objects set was created base on basic elements, divided in five categories as proposed in [14, 23]. The categories are: fluid flow sources, collectors, queues, switches and links.

3.1 Traffic Source

A on/off traffic source produces fluid flows at a constant rate when it is in the on state. The events are generated during the transitions between states, The on/off periods as well as the fluid intensity in each on period is defined by an extra component called source behavior. Figure 4 illustrates the class diagram used for the on/off source.

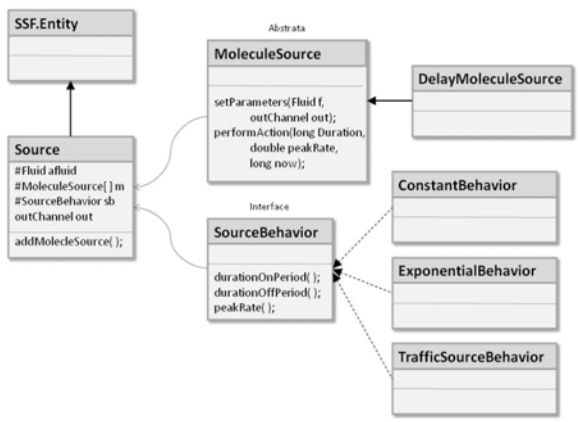

Fig.4 Source Class Diagram

The source behavior may be built according to certain characteristics. The time slots of each state may be constant, may have a predefined values set or may follow a random distribution. Then, there are three ways of source behavior: a) constant behavior: the consecutive on/off periods follow a constant value; b) exponential behavior; the time slots follow a random exponential distribution; c) behavior based on a network traffic file which approximates a discrete packet source into a fluid flow source.

The molecules sources are object introduced within the traffic sources. A traffic source may have more than one molecule source, but it may be always associated to the same fluid flow. The output rate of the molecules source is built as in [13].

3.2 Collectors and Links

A collector is the final point for a fluid flow or molecule, Each collector may have added one or more objects to calculate results. One collector example is the delay collector which is used to compute the end-to-end delay values for one or more fluid flows. To illustrate better the collector implementation, Figure 5 shows its class diagram.

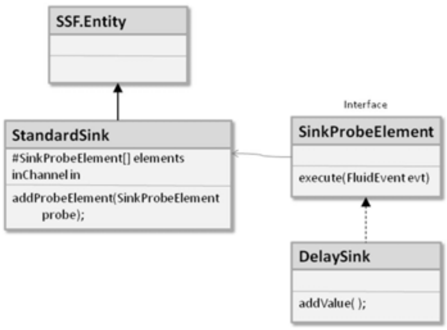

Fig. 5 Collector Class Diagram

3.3 Queue

The fluid queue is defined for transmission capacities $c > 0$ and a maximum buffer size $B \geq 0$. For a certain time value t, the queue has three internal variables which characterize its state: the total arrival rate $a(t)$, the total output rate $d(t)$ and the buffer occupancy $q(t)$.

An event begins to receive its input in a time $t=\tau$ and it will only appear in the output after the $q(\tau)$ queue size consumption, i. e. in time queue, as shown in (4). Always when an event arrives to the queue, the buffer occupancy is calculated using this equation.

$$t_{queue} = \tau + \frac{q(\tau)}{c} \quad (4)$$

The total buffer occupancy since the last update and the time slot are both used by the queue prove elements. Always when there is some loss in the buffer, the prove elements receive three values: the loss volume, the time slot when the losses begin and its duration. The lost fluid volume in a certain period is calculated with (5).

$$L = \int_{\tau_n}^{\tau_{n+1}} [a(t) - c]\,dt \quad (5)$$

3.4 Switch

The switch node is an element that has an input channel and one or more outputs. When an input event arrives, it retransmits the event in one of its output channels with delay equal to zero. It uses a switching table to decide which will be the output channel.

4. Experimental Results

4.1 Fluid Traffic Modeling

A packet flow randomly separated is abstracted and results in a fluid flow. The abstraction time slot Δt between consecutive packets must be chosen in a way in which it reflects the traffic characteristics. Then, the Δt value is chosen as the least separation between packets that belong to the same flow.

Initially, for our experiments, a network traffic file was generated and the simulation parameters are the following:

Each packet has 200 bytes. A repetition pattern among packet slots was used to ease the analysis. The time slot between the first and second packet was 100 ms, between the second and the third it was 200 ms and this pattern is the same for the following packets.

The network used to verify the accuracy o the proposed tool is shown in figure 6. The main characteristics of this network are the following:
(a) The network is composed by a FIFO queue, a collector and three traffic sources.
(b) The sources received the fluid flows abstracted from the packet traffic file.
(c) The difference between the three sources is a delay equal to 50 ms for F2 and 100 ms for F3, both of them according to F1.

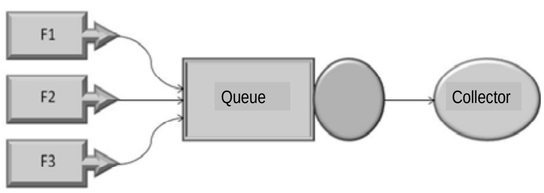

Fig.6 Simulated Network

The packet traffic file was abstracted with three different time slots Δt: 100 ms, 200 ms and 300 ms, as shown in figures 7, 8 and 9 respectively.

In all the figures, the horizontal axis show the time slot as a 100 ms unit. The vertical axis show the transmission rate with the unit equal to 1 kbps. The upper bar shows the fluid flow rate for flow 1, the middle bar stands for flow 2 and the lower bar stand for flow 3..

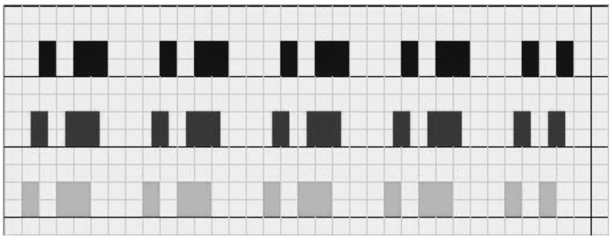

Fig. 7 Δt=100ms

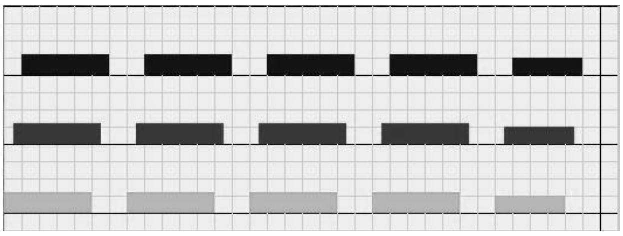

Fig. 8 Δt=200ms

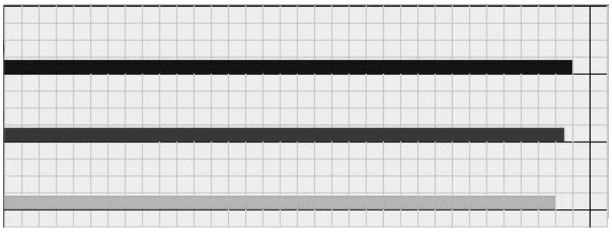

Fig. 9 Δt=300ms

Each traffic flow passes through a FIFO fluid queue with infinite buffer and transmission rate equal to 2 kbps. The total volume that traverses the queue is 18 kB and the performance value collected was the mean buffer occupancy.

During the simulation process, the output of each queue was collected. The results for each traffic flows 1, 2 and 3 are shown in figures 10, 11 and 12 respectively. The result of the mean buffer occupancy is shown in Table 1.

In Table 1 the results show that for a more distributed traffic type, there is a trend to have a lower buffer occupancy even when the its occupancy duration is longer. This was a expected results since for a low arrival traffic rate, the queue serves a bigger portion of traffic and a lower portion needs to be buffered.

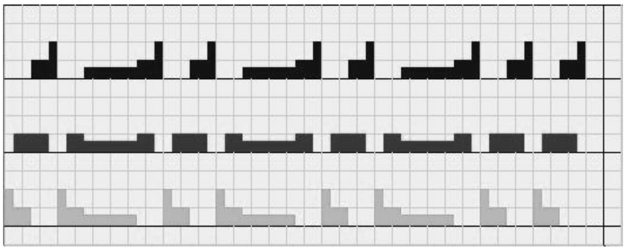

Fig.10 Output for traffic flow 1

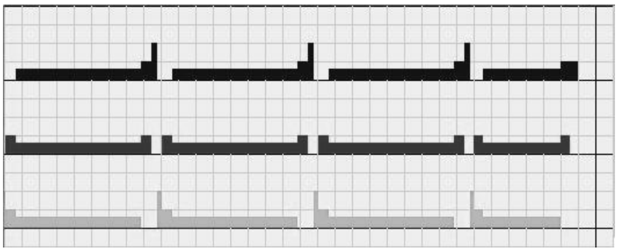

Fig. 11 Output for traffic flow 2

Fig. 12 Output for traffic flow 3

Table 1. Mean Buffer Occupancy

Source	**Mean Occupacy (B/s)**	**Total Occupacy (B)**	**Duration (s)**
Traffic 1	2007,14	14050	7,0
Traffic 2	1869,69	13648,75	7,3
Traffic 3	1653,98	12404,86	7,5

4.2 Packet Simulator Comparison

Consider a simple network model with an unique infinite queue and one input packet CBR(Constant Bit Rate) traffic source F1 as shown in Fig. 13.

Fig. 13 Simulated Network

The source generates packets with size equal to 20 bytes, during 150 secs time interval. For this particular case, three simulations with different packet transmission rates were executed using the NS-2 (Network Simulator 2): 1050 kBps, 2100 kBps and 8400 kBps.

The results are shown in Tables 2, 3 and 4 respectively for each one of the above flows.

Table 2. Results for 1050 Bps

Simulation	Rate	Events	Mean Delay
NS	1050 Bps	23580	3,77
Fluid Simulator	1050 Bps	9485	3.76

Table 3. Results for 2100 Bps

Simulation	Rate	Events	Mean Delay
NS	2100 Bps	47147	82,36
Fluid Simulator	2100 Bps	9485	82,57

Table 4. Results for 8400 Bps

Simulation	Rate	Events	Mean Delay
NS	8400 Bps	188625	553,91
Fluid Simulator	8400 Bps	9494	555,85

The results show that when the transmission rate increases, also the mean delay increases. For the three cases, the simulation results for both simulators appear to be compatible.

Another important observation is that the number of events generated by the packet simulation tool is proportional to the number of packets. For the fluid simulation, when the transmission rate increases, the number of events remains almost the same which is a very important advantage in terms of computational performance.

4.3 Fluid Simulator Comparison

The goal of this section is to compare the performance of our proposed simulator with another fluid simulator. The chosen simulator is the FluidSim [14].

The simulated network is shown in figure 14. The guidelines for this simulation are the following:

(a) The traffic sources F1-F20 are *on/off*, exponentially distributed with a 10 ms mean time between *on/off* periods.

(b) All the traffic sources have a mean rate of 7 Mbits/s and a peak rate of 15 Mbits/s.

(c) The two queues are identical with service rate of 149,76 Mbits/s and infinite buffer size.

(d) An aggregated flow with ten input sources in queue 1 and the same number of input sources in queue 2. The other traffic sources arrive exclusively in queue 1.

As was shown in [14], the simulation was executed 30 times. The results for delays and mean buffer occupancy are shown in Table 5 and Table 6 respectively. The results appear to very close for the proposed simulator and for the FluidSim. This leads to the conclusion that the proposed simulation tool has equivalent results and has the advantage that is platform independent and may be extended to treat different traffic types.

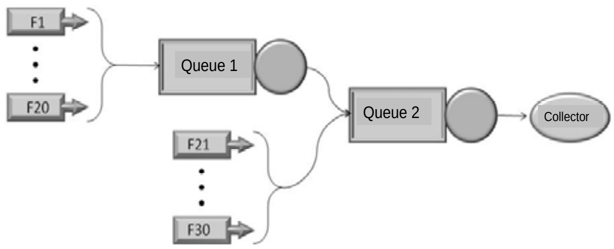

Fig. 14 Network for Fluid Simulation Comparison

Table 5. Delay comparison between fluid simulators

Sources	Mean Delay (ms) Proposed simulator	Mean Delay (ms) FluidSim
1-20	3,46	3,488
21-30	0,76	0,78

Table 6. Buffer occupacy comparison between fluid simulators

Queue	Buffer Occupacy (kbit) Proposed Simulator	Buffer Occupacy (kbit) FluidSim
1	390,07	393,9
2	95,65	96,41

5. Conclusions and future work

In this work was presented the theoretical concepts for the development of a network fluid flow simulation tool. The proposed tool was also explained and discussed. Some problems related to the implementations were discussed, mainly the Ripple effect that may diminish the simulation performance and accuracy.

An abstraction model for packet flows was proposed to produce a fluid traffic flow. In this model, the packet flows are grouped in fluid flows according to a interval. The simulations show that for different time slot values the results may vary considerably

A set of experiments were done to evaluate the proposed tool and to verify its simulation results with packet event based simulators and fluid simulators. For the packet event simulator the NS2 was used and the results appear to be equivalent to the ones of the proposed simulation tool.

In both cases, the variation in the queue arrival rate produces the same variation in the mean delay value. But, the number of events for the proposed tool in this work remains almost the same for bigger arrival rate while it increases considerably for the packet event simulator. This difference may be crucial when simulating large and complex networks in terms of computational effort.

To validate the results' accuracy in the proposed tool, a comparative simulation was done with another network fluid flow simulator, the FluidSim. The results appeared to be equivalent.

As a future work, the influence of individual traffic flow parameters in the performance results appear as an interesting issue, mainly for the case of multifractal traffic flows.

References

[1] Rosen E., Viswanathan A., Callon R. Request for Comments: 3031. Janeiro 2001.

[2] Evans J., Filsfils C. Deploying IP and MPLS QoS for multiservice networks. ISBN-13: 978-0-12-370549-5, Morgan Kaufmann, 2007.

[3] W. E. Leland, M. S. Taquu, W. Willenger and D. V. Wilson. On the self- similar nature of ethernet traffic (extended version). In: IEEE/ACM Trans. on Networking, vol. 2, n. 1, pp. 1-15, 1994.

[4] Park K., Willinger W. Self-Similar Network Traffic and Performance Evaluation. John Wiley and sons. ISBN: 9780471206446, 2000.

[5] Sahinoglu, Z. Tekinay, S, On multimedia networks: self-similar traffic and network performance. In: IEEE Communications, Volume: 37, Issue: 1, pp. 48-52, 1999.

[6] Yusheng JI. Multi-Scale Internet Traffic Analysis Using Piecewise Self-Similar Processes. In: IEICE Transactions in Communications, E89-B: 2125-2133, 2006.

[7] Mark E. Crovella and Azer Bestavros. Self-Similarity in World Wide Web Traffic: Evidence and Possible Causes. In: IEEE/ACM Transactions on Networking, Vol. 5, pp. 835-846, 1997.

[8] Fujimoto, R.M.; Perumalla, K.; Park, A.; Wu, H.; Ammar, M.H.; Riley, G.F. *Large-scale network simulation: how big? how fast?”*. In: 11th IEEE/ACM International Symposium on Modeling, Analysis and Simulation of Computer Telecommunications Systems, pp.: 116 – 123, 2003.

[9] Nicol, D. M., J. Liu, M. Liljenstam, and G. Yan. *Simulation of large-scale networks using SSF*. In Proceedings of the 2003 Winter Simulation Conference ed. S. Chick, P. J. Sánchez, D. Ferrin, and D. J. Morrice, 650–657. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, 2003.

[10] Nicol, D. M., J. Liu and M. Liljenstam. *Advanced concepts in large-scale network simulation*. In Proceedings of the 2005 Winter Simulation Conference ed. M. E. Kuhl, N. M. Steiger , F. B. Armstrong, and J. A. Joines, 153-166. Piscataway, New Jersey: Institute of Electrical and Electronics Engineers, 2005.

[11] Jones, K. G. and Das, S. R. *Parallel execution of a sequential network simulator*. In Proceedings of the 2000 Winter Simulation Conference ed. J. A. Joines, R. R. Barton, K. Kang, and P. A. Fishwick, 418-424, 2000.

[12] Liu, B., Guo, Y., Kurose, J., Towsley,D. , And Gong, W. 1999. *Fluid simulation of large scale networks: Issues and tradeoffs*. In Proceedings of the International Conferece on Parallel and Distributed Processing Techniques and Applications (Las Vegas, NV), 1999.

[13] Figueiredo D.R., Liu B., Guo Y., Kurose J., Towsley D. *On the efficiency of fluid simulation of networks*. Technical Report CMPSCI TR 04-82, Department of Computer Science, University of Massachusetts, Amherst, 2004.

[14] J. Incera, R. A. Marie, D. Ros, G. Rubino. *FluidSim: a tool to simulate fluid models of high-speed networks*. In: Performance Evaluation 44 – An International Journal, Elsevier Science B.V, p. 25-49, 2001.

[15] Fernandez, Elizabeth e Valadas, Rui. *Demonstrador Animado das Técnicas de Programação de Simuladores de Eventos Discretos*. Utilizando o MATLAB, Revista do DETUA. Vol. 2, N° 3, 1998.

[16] Liu, B., D. Figueiredo, Y. Guo, J. Kurose, and D. Towsley. *A study of networks simulation efficiency : Fluid simulation vs. packet-level simulation*. In Proceedings of the 2001 INFOCOM Conference, 2001.

[17] Java Technology. http://java.sun.com,.

[18] Sierra, Kathy e Bates, Bert. *Head First Java*. Editora Alta Books Ltda, 2005.

[19] Deitel, H. M. e Deitel, P. J. *Java™ How to Program*, Deitel & Associates, Inc, 6ª. Edição, 2004.

[20] SSFNet. http://www.ssfnet.org/homePage.html, acessado em 05 de dezembro de 2007, às 16:00.

[21] Cowie, James H. Scalable Simulation Framework API Reference Manual. Documentation Draft, 1999.

[22] MIRAI-SF Simulator 4.0 User Manual, *National Institute of Information and Communications Technology*. Document Version 4.0.0, 2006.

[23] Liu, Benyuan e Figueiredo, Daniel. *Queuing Network Library for SSF Simulator*. Dept. Of Computer Science, University of Massachusetts, 2005.

[24] Ros, David e Raymond. M. *Loss caracterization in high-speed networks through simulation of fluid models*. In: Telecommunication Systems 16, Kluwer Academic Publishers, p. 1,2, 73–101, 2001.

Paulo H. P. de Carvalho graduated in Electrical Engineering at University of Brasília (1988) , has a masters degree in D.E.A. d'Electronique en Comm. Optq. et Microondes at Universite de Limoges (1989), has a Ph.D in Doc. de l'Univ. de Limoges - Comm. Optiq. et Micro at Universite de Limoges (1993) and has a post-doctoral at Université de Limoges - Institut de Communications Optiques et Microondes (1995). Currently he is associated professor at the Department of Electrical Engineering, University of Brasília.

Priscila Solis Barreto graduated in Computer Engineering at the Universidade Francisco Marroquin (1991), has a master in Electrical Engineering and Computer Science at the Universidade Federal de Goiás (2000) and has a Ph.D in Electrical Engineering at the Universidade de Brasília (2007). Currently she is professor at the Department of Computer Science, University of Brasília.

Tiago Flor Bento is graduated in Electrical Engineering, University of Brasília. Currently he's finishig his master's degree in Electrical Engineering at the University of Brasilia.