# SIMULATING LARGE-SCALE NETWORKS WITH ANALYTICAL MODELS

Matthias Scheidegger, Florian Baumgartner, Torsten Braun
Institute of Computer Science and Applied Mathematics
University of Bern, Neubrückstrasse 10, CH-3012 Bern
Email: (mscheid|baumgart|braun)@iam.unibe.ch

## KEYWORDS

Queueing Systems and Network Models , Stochastic Simulation, Analytical and Numerical Simulation.

## ABSTRACT

Discrete-event simulation of computer networks has significant scalability issues, which makes simulating large-scale networks problematic. We propose a high-level abstraction modeling network domains, inter-domain links and traffic with highly scalable analytical models, which is much more efficient but slightly less accurate than node-by-node models. Thus, simulation scenarios containing several ISP networks become feasible. We also propose a way to combine this modeling approach with traditional packet-based simulators and present some preliminary evaluation results of the concept.

## INTRODUCTION

In traditional packet-based simulators networks are modeled in terms of nodes and links with individual capacities and delay characteristics. When simulating large Internet topologies this approach quickly becomes problematic, due to the huge amount of events to be processed. Many approaches to this scalability problem have been proposed, each with slightly different application ranges. Parallel simulation (Chandy and Misra 1981; Ammar et al. 1999) is probably the most prominent one, but there are also approaches such as fluid flow simulation (Yan and Gong 1999; Liu et al. 1999; Liu et al. 2001), time-stepped hybrid simulation (Guo et al. 2000) and packet trains (Ahn and Danzig 1996), amongst others. Scalability in network simulation is generally achieved by reducing the level of detail of the simulation scenario or of the simulation algorithm. Carefully chosen, such abstractions of the simulated network can significantly reduce the complexity of large-scale simulations. In this paper we propose a model that aims for far more efficient simulations than traditional approaches while still giving a good approximation of real network behavior.

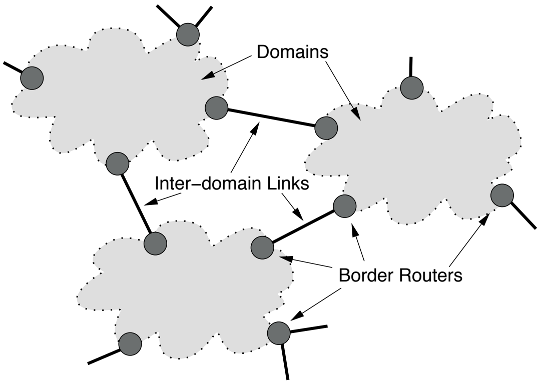

Figure 1: The basic modeling view

This model is based on the assumption that, over certain time spans, networks like the Internet can be divided into areas where congestion is negligible, interconnected by bottleneck links. We treat congestion free areas as black boxes, which we call *domain models*. Modeling congestion free areas has the advantage that we can neglect packet losses and excessive queuing in large parts of the network and restrict the model to quasi-stationary delay behavior. Apart from its scalability advantage this approach is primarily useful to model network areas of which we do not know the exact topology. Domain models can be based on empirical cumulative distribution functions (ECDFs) to simulate the delays of packets crossing the domain. The ECDF is chosen depending on the ingress and egress nodes on which the packet enters and leaves the domain, respectively. A big advantage of this concept is that delay measurements from a real network can be directly used to configure a domain model.

The bottleneck links between two domains of a simulation scenario are represented by *inter-domain link models*. Here, packet loss and queuing delay are simulated. The basic parameters of an inter-domain link model are similar to those of a link in a packet-based simulator. Nonetheless, inter-domain link models are not event-driven but rely on parameters like offered load and link capacity. Figure **??** shows this modeling view.

One useful partitioning scheme is to model autonomous systems (ASs) as domains, and their border links as inter-domain links. This partitioning is reasonable since the ingress routers of an AS may police flows to prevent congestion inside the AS. Moreover, the interior links usually have bigger capacities than inter-AS links, and internal routes may be changed to distribute traffic load.

Further components of this model system are the *application traffic models* concerned with traffic load. They serve as scalable models for large aggregates of application traffic like VoIP, Video, HTTP, etc. They take the form of a function that yields the load generated by the traffic aggregate given a (monotonously rising) point in time. By combining domain, inter-domain link and application traffic models we create a *multi-domain model*. Multi-domain models can be viewed as an equivalent to simulation scenarios in packet-based simulations. A simulator could be written based solely on these models. However, for several reasons it is desirable to combine these models with packet-based simulation. The behavior of an individual flow is easier to describe as a packet-based model, and many protocol and application models already exist for packet-based simulators. Furthermore, a combination of fine grained packet-based simulation and coarse grained analytical models could be very useful in scenarios like a multi-site virtual private network. We refer to this combination of analytical models and packet-based simulation as *hybrid simulation*.

The remainder of the paper is organized as follows: First, we describe how domain, inter-domain link and application traffic models are combined to multi-domain models. Then we go into further detail on domain and inter-domain link models, respectively. Another section discusses the combination of analytical models with packet-based simulation, and in the end we present some preliminary evaluation of the concept and give a conclusion.

## MULTI-DOMAIN MODELS

The purpose of a multi-domain model is to organize and control domain models, inter-domain link models and application traffic models to form a single analytical model. Thus, the basis of a multi-domain model is a set of such models and their parameters, e.g. delay characteristics for domain models and the link capacity for inter-domain link models. In order to combine these models to a multi-domain model additional information is required. The topology of a multi-domain model is a directed graph, where the domain and application traffic models are the vertices and the inter-domain link models are the edges (links are always simplex, duplex links are modelled using two simplex links). Accordingly, standard ways to represent graph topologies can be used, e.g. vertex and edge tables. Routing information is required to map the load generated by application traffic models to the correct inter-domain links. Each route is stored as a sequence of inter-domain links, which again can be implemented using tables. In combination with the topology this is sufficient to resolve all models along a routing path.

### Multi-Domain Load

While the inter-domain link models simulate the effects of network load on a single link, it is the task of multi-domain models to simulate the distribution of network load among its inter-domain links. This is the basis for estimating packet loss and delay behavior in the modeled network. Note that unlike packet-based simulators where events trigger an update of the system, we do not have to update unless we want to inspect the system's state.

Given we want to inspect the system at simulation time $t$. Let $P_s$ be the routing path (a sequence of inter-domain links) of the traffic originating at source $s$ (an application traffic model), and let $s(t)$ be the load generated by the traffic aggregate at time $t$. The processed load of an inter-domain link $L$ depending on the offered load $\lambda$ is written as $L(\lambda)$. Here, inter-domain links take the role of a function with $0 \leq L(\lambda) \leq L$. The computation of $L(\lambda)$ is described later in this document. If link $L_i$ directly follows link $L_j$ on a path we call $L_j$ a predecessor of $L_i$. Now, we calculate the processed load along a path $P_s = \{L_1, L_2, \ldots\}$ using the sequence

$$s(t), \quad L_1(s(t)), \quad L_2(L_1(s(t))), \quad \ldots$$

until a link on the path has more than one predecessor, or until the path ends. Then, we start over with the next path, and so forth, until all paths either have ended or have reached a link with more than one predecessor. Now we can return to the first path. The offered load $\lambda_{L_i}$ on the link in question is given by the sum of the processed loads of all predecessors. If the last calculated element of the path's load sequence was $\lambda$ we can now calculate the next element with

$$\lambda' = \frac{L_i(\lambda_{L_i})}{\lambda_{L_i}} \lambda \,.$$

We continue with this procedure until all paths have been followed to their end and the offered and processed loads of all inter-domain links are known.

The above algorithm may be optimized in several ways. First, when updating the system we only have to pursue changes in the offered load as far as they make a difference for the whole system. For example, if a traffic model overloads the first link on its path on one update, any additional load in the next update will influence only this first link. The processed load of this link stays the same. Furthermore, changes in the offered load may be marginal, in which case we can ignore this change at the cost of reduced accuracy. However, in order not to accumulate errors we then have to force updates in regular intervals.

### Multi-Domain Loss and Delay

Based on the load distribution calculations above, the delay distributions and packet loss ratios of a multi-domain model's paths can be found. The packet loss ratio along the path $P = L_1, \ldots, L_n$ is given by

$$1 - \prod_{i=1}^{n} \left(1 - \frac{\lambda_{L_i} - L_i(\lambda_{L_i})}{\lambda_{L_i}}\right)$$

where $\lambda_{L_i}$ is again the offered load on link $L_i$.

Delays along a path are similarly modeled. The time it takes for a packet to traverse a domain or an inter-domain link can be described as a random variable. Let $\delta_L$ be the random variable of the delay caused by inter-domain link $L$, and let $\delta_{L,K}$ be the random variable of delay in the domain between the inter-domain links $L$ and $K$ ($\delta_{L,K}$ is only defined if $L$ is a predecessor of $K$). Then the delay distribution on the routing path $P = L_1, \ldots, L_n$ is given by

$$\delta_P = \sum_{i=1}^{n} \delta_{L_i} + \sum_{i=1}^{n-1} \delta_{L_i, L_{i+1}}$$

In a simulation we need to generate random values accordingly. This can be easily done by generating random values for each of the random variables and summing them up. The fact that the delay distributions of domain models do not change can be used to make this procedure much more efficient, however. Since these delay distributions are discrete, their distribution functions can easily be convoluted into a single one, which reduces the task of simulating the domain delays to the generation of a single random value. The convolution can be performed efficiently by using the Fast Fourier Transform algorithm (Cooley and Tukey 1965). Convolving the link delay distributions is not efficient in normal scenarios as they change rather rapidly according to the load distribution. Having a random variable of a path's delay further allows to easily calculate moments like the mean delay or the path's jitter, which would be $\mathrm{Var}(\delta_P)$ if interpreted as delay variation.

## DOMAIN MODELS

Domain models represent network "clouds" in a simulation scenario where no congestion occurs. The partitioning of a topology into domains and inter-domain links can be freely configured but it must be chosen such as to satisfy this basic assumption as closely as possible. Network "clouds" of nodes under a common management (e.g. an ISP network) are good candidates, since with policing and shaping performed at the edge routers, congestion within the domain can be avoided. This is especially true for core networks, due to typical aggregation and traffic smoothing effects observed in backbone networks. The chosen abstraction allows that domain models only simulate the delay behavior of a network cloud and do not react to changes of network load. Domain models are black boxes; their interior structure is not explicitly modeled. The highest level of detail in a domain model is the distinction of paths through the domain. A model of a domain with $n$ edge nodes can thus contain $n(n-1)$ delay models, one for each ingress-egress node combination. Simpler cases with only one common delay model for all paths are useful if there is not enough information available about the network area. Using simple models can also significantly reduce the memory consumption of a simulation.

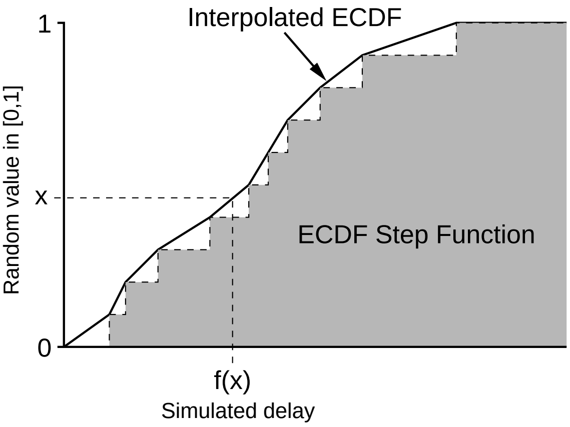

Figure 2: Generating random values using an interpolated ECDF

During preliminary evaluation we found that empirical cumulative distribution functions (ECDFs) are well suited to model the delay behavior of network domains. They can be easily built from a series of delay measurements taken from a real network. In the optimal case, one-way delays should be used, but as this requires clock synchronization of the measurement endpoints we can also approximate them by taking round-trip times divided by two. This requires a nearly symmetrical path, however. For the integration with packet-based simulation (see below) we need to be able to generate random values based on the ECDF. A basic approach is to store the observations in a table and then randomly selecting table entries using a uniform distribution. Given a sufficient sample size, this approach yields very good results if the basic assumption is not violated. The size of large tables can be reduced by using linear interpolation. The procedure can be seen in Figure **??**. We start by generating a random value $x$, uniformly distributed on $[0, 1]$, which designates a position in the sorted observation table (seen as a step function in the Figure). The two nearest observations are then interpolated to get a simulated delay value

$f(x)$.

It is important to note that ECDF models, while giving good reproductions of observed first and second-order moments in measurements, ignore any non-stationarity of the sample.

## INTER-DOMAIN LINK MODELS

Inter-domain link models cover the dynamic parts of network behavior, like the effect of queuing and overload, on delay and packet loss. Since they represent a single physical link between the interfaces of two nodes it is an obvious approach to model them as an analytical queues. We chose the simple M/M/1/K queue as a first approximation, that is, a queue with Poisson arrival and service processes, a single server (the physical link) and system capacity $K$. The arrival and services rates $\lambda$ and $\mu$ depend on the offered load on the link and the link's capacity, respectively. The system capacity $K$ can be set to a typical value (e.g. 128-packet buffers are rather common in routers).

Recent work (Klemm et al. 2003) suggests that the arrival process would be better modeled as a Batch Markovian Arrival Process (BMAP). Also, sophisticated techniques like traffic-based decomposition (Heindl and Telek 2002) or the decomposition approach in Sadre et al. 1999 could be used, especially in the above section about multi-domain models. These techniques also consider the effect of correlations in network traffic, which the M/M/1/K queue clearly ignores. However, we argue that, while these approaches use traffic models that describe traffic behavior over long periods of time, our approach only considers load produced by the traffic sources at the instant when the system is inspected. Correlations are thus only ignored on the small time-scale. The system's behavior in the long run is not modeled but rather simulated and hence also includes the effects of correlations.

In order to model the behavior of the inter-domain link we have to find the probability $p_i$ of the system to be in state $i$, where state $K$ means the queue is full, and state 0 means the system is empty and does not send. The M/M/1/K queue is a birth and death process as shown in Figure **??**. For a birth and death process of this kind the probabilities $p_i$ are given by

$$p_i = \begin{cases} \frac{1-\lambda/\mu}{1-(\lambda/\mu)^{K+1}}\,, & i = 0 \\ (\lambda/\mu)^i p_0\,, & i > 0 \end{cases} \tag{1}$$

if $\lambda \neq \mu$, and

$$p_0 = p_1 = \ldots = p_K = \frac{1}{K+1} \tag{2}$$

if $\lambda = \mu$. As states above, $p_K$ is the probability of the system being full. Therefore, $p_K$ is also the loss ratio of the link. The functional representation of the inter-domain link used in the section about multi-domain models above can thus be written as $L(\lambda) = (1 - p_K)\lambda$, with $p_K$ calculated according to formulas **??** and **??**. From the probabilities $p_i$ we can further construct a discrete density function of the link's delay distribution. The number of bytes that are in the system when another byte arrives is proportional to the time this byte has to wait before it is sent to the link. $\delta_{pr}$ is the propagation delay on the link, which depends on physical properties of the link, e.g. its length. The discrete delay distribution looks like this

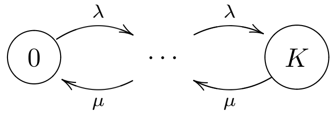

Figure 3: Birth and Death Process

$$\begin{pmatrix} p_0 & \cdots & p_{K-1} & p_K \\ \delta_{pr} + \frac{1}{\mu} & \cdots & \delta_{pr} + \frac{K}{\mu} & \infty \end{pmatrix} \tag{3}$$

The infinite delay in the case of a full queue indicates that this packet is effectively lost.

## HYBRID SIMULATION

Creating a hybrid simulation of packet-based and analytical models makes it possible to combine large-scale, coarse-grained topologies with fine-grained models for points of special interest in the scenario. Especially simulations of multi-site corporate VPNs may benefit from this: The intranet components can be modeled using the packet-based approach, while the inter-site connections over the public internet benefit from the efficiency boost of analytical simulation.

We propose to enhance traditional packet-based simulators by enabling their nodes to contain analytical multi-domain models. In this way a simulator node can stand for and behave like a whole network cloud (typically a multi-domain model). Figure **??** shows an example for this. When a simulated packet reaches an enhanced node, it triggers an inspection of the underlying multi-domain model to determine how much the packet should be delayed and whether it should be forwarded at all. Both decisions are based on the cumulative forwarding probability and delay distribution calculations described in the section about multi-domain models. This approach necessitates a new load generator in the multi-domain models: the bandwidth estimator (BE). It converts packet reception events to a bandwidth estimate for every routing path between an ingress and an egress node of the multi-domain model. A good way to estimate bandwidths from packet events is to use a sliding time window algorithm. The number of bytes received in the time window $\Delta t$ is added up and divided by $\Delta t$. While packets generated in the event-driven simulator influence the analytical models inside enhanced nodes, loads generated by the application traffic models of a multi-domain model do not create additional packets outside of the enhanced node. Our approach only allows packets to go through enhanced nodes, not to be created by them. The reason for that is the higher level of abstraction used in multi-domain models.

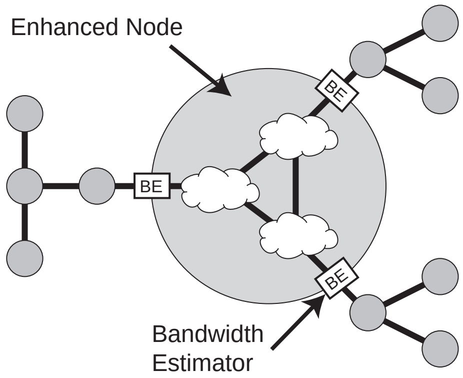

Figure 4: Enhanced node in a packet-based simulator

## EVALUATION

We implemented the concept of hybrid simulation in the `ns2` simulator by extending the simulator with a mechanism that makes it possible to overload the behavior of the simulator nodes with arbitrary loadable modules. The analytical models presented in this paper were implemented as such a module. All parameters of the models can be configured using XML files. During preliminary evaluation we tested the behavior of the implemented inter-domain link and domain models.

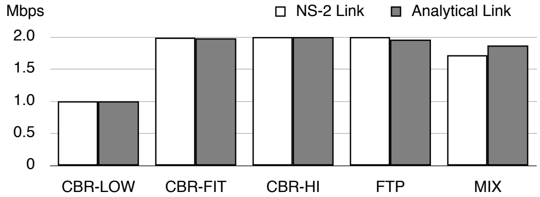

Figure 5: Comparison of `ns2` and analytical link: transfer rates

We compared the inter-domain link model with a standard `ns2` link. For this we used a scenario with three consecutive links, of which the middle one was the 2 Mbps bottleneck and studied the behavior of this link under five kinds of traffic load: 1 Mbps, 2 Mbps and 4 Mbps CBR traffic, FTP traffic (5 sources), and a mix of FTP traffic (3 sources) and 1 Mbps CBR traffic. Figure **??** shows a comparison of the transfer rates achieved with the `ns2` link and the analytical link model. While in the CBR and FTP cases the performance is good, the mix of CBR and FTP (i.e. TCP) seems to be more problematic: With both link types the transfer rate decreases but not by equal amounts. We believe this is due to the stochastic nature of dropping in the analytical model, similar to the behavior of random early detection (RED) queues (Floyd and Jacobson 1993), which are known to enable higher transfer rates with TCP than traditional drop-tail queues do.

As a preliminary evaluation of the domain model, the delay characteristics between the network of the University of Bern and the ETH Zürich have been measured. In a first step the delay between two hosts in the networks was measured. Both networks are connected by the Swiss scientific network SWITCH, and the distance between the measurement hosts was nine hops. Based on the measurements an empirical distribution was computed and used to configure the domain model. For the simulation the simple `ns2` network in Figure **??** with three nodes was set up. While the two outer nodes act as source and sink, the central node has the domain model attached. For both measurements and simulation we used probe packets with a rate of one packet per second.

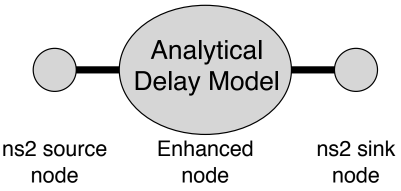

Figure 6: `ns2` setup to simulate the delay of a single ISP

Figure **??** shows a comparison between the measured delays and the delays in the simulation. Both graphs show almost exactly the same delay behavior for the measurement and the simulation.

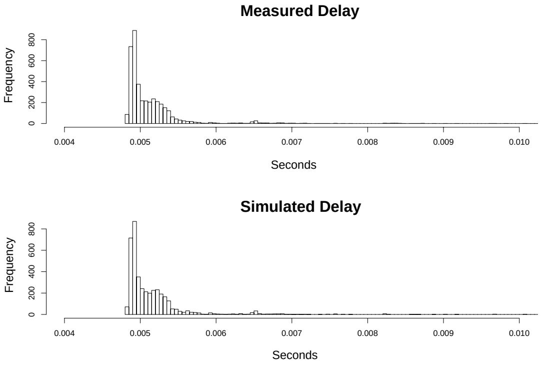

Figure 7: Delay histograms from measurements (upper graph) and simulation (lower graph)

## CONCLUSION

In this paper we have presented a scalable approach to simulating large-scale inter-domain networks. This scalability is achieved by partitioning the simulation scenario into congested bottleneck links and the congestion-free areas in between, and by creating analytical models for both (inter-domain link models and domain models, respectively). These models are configured by measuring the characteristics of a live network and can then predict delay and dropping behavior of this network. We have further presented a concept to combine these high-level models with traditional packet-based simulators, which we implemented in the `ns2` simulator. Some preliminary evaluation was also done for the basic models, comparing an inter-domain link model with a link model of the `ns2` simulator, and comparing the measured delay between two real network nodes to the simulated delay of a correspondingly configured domain model in `ns2` .

## ACKNOWLEDGEMENTS

This work was funded by the Swiss Federal Office for Education and Science (BBW, contract number 01.0551) and was done in the context of the InterMON project (European Union project IST-2001-34123), which is part of the European Union's 5th Framework Programme.

## REFERENCES

Ahn, J. S. and P. B. Danzig. 1996. "Packet network simulation: speedup and accuracy versus timing granularity." *IEEE/ACM Transactions on Networking* 4, No. 5, 743–757.

Ammar, M. H., G. F. Riley and R. M. Fujimoto. 1999. "A generic framework for parallelization of network simulations." In *MASCOTS'99* (College Park, MD).

Chandy, K. M. and J. Misra. 1981. "Asynchronous distributed simulation via a sequence of parallel computations." *Communications of the ACM* 11, No. 24, 198–205.

Cooley, J. W. and J. W. Tukey. 1965. "An algorithm for machine calculation of complex Fourier series." *Math. Comp* 19, 297–301.

Floyd, S. and V. Jacobson. 1993. "Random early detection gateways for congestion avoidance." *IEEE/ACM Transactions on Networking*.

Guo, Y., W. Gong, and D. Towsley. 2000. "Time-stepped hybrid simulation (TSHS) for large scale networks." In *Proceedings of IEEE Infocom*.

Heindl, A. and M. Telek. 2002. "Output models of MAP/PH/1(/K) queues for an efficient network decomposition." *Performance Evaluation* 49, 321–339.

Klemm, A., C. Lindemann, and M. Lohmann. 2003. "Modeling IP traffic using the Batch Markovian Arrival Process." *Performance Evaluation* 52, 149–173.

Liu, B., Y. Guo, J. Kurose, D. Towsley, and W. Gong. 1999. "Fluid simulation of large scale networks: Issues and tradeoffs." In *PDPTA'99* (Las Vegas, NV), 2136–2142.

Liu, B., D. R. Figueirido, Y. Guo, J. Kurose, and D. Towsley. 2001. "A study of networks simulation efficiency: Fluid simulation vs. packet-level simulation." In *Proceedings of IEEE Infocom.*

Sadre, R., B. Haverkort, and A. Ost. 1999. "An efficient and accurate decomposition method for open finite and infinite-buffer queueing networks." In *Proceedings of the Third International Workshop on Numerical Solution of Markov Chains*, 1–20.

Yan, A. and W. B. Gong. 1999. "Fluid simulation for high speed networks with flow-based routing." *IEEE Transactions on Information Theory* 1588–1599.

## AUTHOR BIOGRAPHIES

**MATTHIAS SCHEIDEGGER** got his masters degree in computer science in 2001 at the University of Bern, Switzerland. His research topics were IP telephony and differentiated services. Since August 2001 he is a Ph.D. student with the Computer Networks and Distributed Systems research group at the Institute of Computer Science and Applied Mathematics of the University of Bern. Current research interests include measurement based admission control, QoS on peer-to-peer networks, and network modeling and simulation.

**FLORIAN BAUMGARTNER** got his masters degree in information science and physics in 1997 at the University of Regensburg in Germany. Until 1998 he worked at the European Networking Center in Heidelberg in the area of network optimization, before starting his Ph.D. at the Institute of Computer Science and Applied Mathematics in Bern (Switzerland). His main research topics were quality of service, active networks and network management. After finishing his Ph.D. thesis in February 2002, he worked at Purdue University, Indiana, USA in the area of network security and mobile ad hoc networks. Since 2003 Florian Baumgartner works as research assistant the the Institute of Computer Science and Applied Mathematics in the area of wireless networks, network modelling and simulation.

**TORSTEN BRAUN** got his diploma and Ph.D. degrees from the University of Karlsruhe, Germany, in 1990 and 1993, respectively. From 1994 to 1995 he was a guest scientist with INRIA Sophia Antipolis. From 1995 to 1997 he worked at the IBM European Networking Center, Heidelberg, Germany, at the end as a project leader and senior consultant. Since 1998 he has been a full professor of computer science at the Institute of Computer Science and Applied Mathematics (University of Bern, Switzerland), heading the Computer Networks and Distributed Systems research group. He is a member of the foundation council of SWITCH, the Swiss national research network.