A Component-based Simulation Model and its Implementation of a Switched Ethernet Network

Jörg Sommer[1] and Walter Franz[2]

[1] University of Stuttgart, Germany
Institute of Communication Networks and Computer Engineering (IKR)
Email: joerg.sommer@ikr.uni-stuttgart.de
[2] Daimler AG, Germany
Group Research and Advanced Engineering
Email: walter.franz@daimler.com

Abstract. Today, Ethernet is the most widely used local area networking (LAN) technology. Apart from LAN installations, Ethernet became also attractive for other application areas, ranging from telecommunications to industry and avionics. The range of application areas for Ethernet keeps growing. Since each application area has its unique requirements, the IEEE and interest groups have enhanced the Ethernet standards. In future, they will keep this evaluation going.
In this paper, we present a component-based simulation model of a switched Ethernet network to evaluate enhancements to the Ethernet standards. We decompose the network entities into a hierarchy of components interconnected in a plug-n-play manner. These components form the basis of the simulation program. Finally, we show the applicability of the simulation model and its implementation by using examples from the automotive domain.

1 Introduction

Today, Ethernet is the predominant local area network (LAN) technology and became the de facto standard network infrastructure. The flexibility and plug-and-play feature of Ethernet are its keys to success. Thanks to its low cost, availability of components, its bandwidth, backward-compatibility, and its further evolution that will satisfy future needs, Ethernet has become an attractive option in many application areas [1].

An example of Ethernet's popularity is the industrial domain where Ethernet partially replaces traditional fieldbuses. Today, Ethernet is even integrated into aircraft [2]. The motivation to use Ethernet in such environments is both to reduce complexity and costs by reusing commercial off-the-shelf (COTS) components, which increases the number of vendors and the range of products.

The range of application areas for Ethernet keeps growing. Due to the large production volume of the automotive industry, costs per piece play an important role. The manufacturers and suppliers evaluate if there are other low-cost network technologies available that can also fulfill requirements of an in-car network technology. Currently the automotive industry is investigating if Ethernet as a suitable in-car network technology, since commonly deployed in-car network technologies have become inflexible, complex, and costly [3–5] and are almost reaching their capacity limits.

During the last years, the market of home multimedia networks has been growing rapidly. The IEEE noticed that Ethernet has the potential of serving both data and multimedia networking requirements and thus is suited for an integrated digital home network. For this reason, the IEEE is specifying protocols and mechanisms supporting services for streaming multimedia applications with real-time requirements. They are discussing proposals for a stream reservation protocol [6] as well as forwarding and queuing enhancements for switches [7].

The reason for not adopting Ethernet one-to-one from one application area to another is that in each area the requirements are unique in its combination with quality of service (QoS), real-time behavior, data rate, reliability, usability, and price per unit. In the last years, the IEEE enhanced the IEEE 802 standards [8–11] to fulfill different requirements. Furthermore, different interest groups published a number of (commercial) domain-specific, Ethernet-based standards. In the next years, the evaluation of Ethernet will keep going, which is motivated by the unique requirements of the different application areas. Therefore, we can expect new Ethernet-(based) standards and enhancements. Examples are new traffic shapers and policers, e. g., to restrict unforeseen traffic to achieve timing guarantees and to minimize the resource usage, e. g., in in-car communication systems [5].

Since the performance evaluation of enhancements and proposals is often time-consuming and complex, we need a technique that enables to modify or to substitute only the relevant components of a network entity that are affected. Especially in high demanding time-to-market and cost-sensitive fields like automotive and avionics there is a strong need for a flexible method that enables evaluation of complex communication networks. Performance evaluation of networks through simulation is an excellent way to reduce the development time and costs. That way bottlenecks can be identified and it can be assured that needed QoS can be guaranteed in an early development stage.

In this paper, we present a component-based simulation model with a modular design that enables the separate handling and treatment of each component and that offers a high reusability of existing components. Examples of such components are buffers with a certain queuing discipline or switching fabrics. Therefore, we decompose network entities into a hierarchy of components. These components form the basis of the simulation program. The hierarchical decomposing alleviates the evaluation on different levels and therefore the handling of the network complexity. The result is an expandable, customizable, and configurable simulator for performance evaluation of a switched Ethernet network.

In the simulation program, all components are strictly encapsulated and communicate over a generic interface by exchanging messages. The interfaces, so-called ports, enable to interconnect components on each hierarchy level in a plug-and-play manner.

The rest of the paper is organized as follows: In section 2, we introduce to the Ethernet technology. Then, in section 3, we describe the components for modeling a switched Ethernet network. In the next section, we introduce the implementation of the simulation program. This includes the architecture and conceptual structure as well as the simulator's settings. In section 5, we demonstrate the applicability by using examples from the automotive domain. Finally, we give a conclusion and an outlook.

2 The Ethernet LAN Technology

In 1972, Robert Metcalfe and his colleagues at the Xerox Palo Alto Research Center developed a LAN technology [12]. Later, they improved the throughput of the installed Aloha network by a collision detection algorithm [13]. This was the creation of Ethernet's media access control protocol carrier sense multiple access with collision detection (CSMA/CD). Together with Intel and DEC, Xerox first published a modified Ethernet version in 1980 that became the well-known DIX Ethernet II version. In 1985, the IEEE began to standardize the different versions of Ethernet. They avoided the brand name Ethernet and named the technology *802.3 CSMA/CD* [12] instead.

During the last decades, the number of stations in an Ethernet network increased. As an increasing number of stations reduce the network throughput due to collisions, the LAN topology changed. Today, switches connect stations directly. They operate in full duplex mode avoiding collisions and do not apply CSMA/CD in these networks. Interconnected switches form a mesh topology. Hence, Ethernet evolved to a so-called switched network with full duplex links.

Today, two types of Ethernet's frame format exist [1]: *Ethernet II* and *IEEE 802.3*. In both versions, the Ethernet frame starts with a preamble, which serves synchronization between sender and receiver. After the preamble, the start of frame delimiter (SFD) indicates the start of the frame. The destination (DA) as well as the source address (SA) is 6 Byte globally unique addresses administered by the IEEE.

The 2 Byte following the source address differentiate the Ethernet versions. The meaning of this type/length field is context dependent. It identifies either the type of the payload or the size of the MAC frame. In Ethernet II, this field defines the type of the protocol in the payload field. The Ethernet PHY determines the end of a frame on the wire and provides this information to the MAC layer. As the payload PDU indicates the data length anyhow, it is not necessary to provide the length of the MAC frame in the MAC frame itself. On the contrary, the IEEE promotes a MAC-layer independent LLC frame. Then, the frame provides the information on the length, as the knowledge on the frame length cannot be assumed for any technology, it guarantees an interworking independent of the underlying LAN technology. For compatibility reasons, today's devices distinguish both cases by the value of the type/length field.

The data field contains the payload of the frame. It needs a minimum size of 46 Byte required for collision detection by CSMA/CD. In case of fewer payload, the MAC driver adds padding data. The last field, the cyclic redundancy check (CRC), contains 4 Byte for error detection and correction.

For traffic engineering purpose and security reasons, the IEEE extended in 1998 the Ethernet frame in IEEE 802.1Q [10]. The extension enables the virtual separation of multiple LANs on the same physical infrastructure. The extension consists of an additional 4 Byte field followed by the Ethernet type field. Its first 2 Byte indicate the tag protocol identifier (TPID), while the second 2 Byte represent the tag control information (TCI) field. The TPID corresponds to the type/length field in Ethernet II and its value identifies an IEEE 802.1Q frame. The TCI field contains a VLAN identifier (12 bit), priority information (3 bit), and a canonical format indicator (1 bit). The priority information indicates the user priority class. IEEE 802.1Q defines eight different priority classes considered at the schedulers of interworking units.

3 Modeling

In this section, we describe the model and all the corresponding components that are necessary for performance evaluation of a switched Ethernet network. Each component consists of several components again. This leads to a hierarchical modeling approach with encapsulated components.

3.1 Full Duplex Link

A switched Ethernet network avoids collisions. Consequently, there is no need for CSMA/CD. Nevertheless, the connected station has to follow the inter-frame gap (IFG) procedure. This means when a link becomes idle, the station has to wait for a defined period, before transmitting its next frame. The IFG is a 96-bit time delay [11] to allow the network interfaces and other low layer components some recovery time after which it must be ready to accept a new frame. For example, for a 100 Mbps link the IFG is 0.96 μs.

The full duplex link model, shown in figure 1, consists of two identical parts: the transmission and the receiving link. This enables to transmit and receive frames simultaneously. Each link consists of two, non-preemptive servers. Typically, a server is used to model delays of all kinds. We use one server for representing the delay of a frame transmission and the other one for the IFG.

We map the IFG mechanism onto the link model. In case of switch position 1, the link is ready to transmit a frame. After transmitting a frame, the left server (with holding time T_{Tx}) takes a vacation and the switch changes to position 2 that represents the idle state. Upon return from vacation with a period of $T_{\mathrm{IFG}} = {}^{96\ \mathrm{bit}}/_{\mathrm{bit\ rate}}$, the link is ready to transmit a frame again. If a station wishes to transmit multiple frames, it must wait for the defined period IFG between each frame. The transmission time (T_{Tx}) of an individual frame is

$$T_{\mathrm{Tx}} = \frac{\text{frame length}}{\text{bit rate}} + \text{propagation delay}. \tag{1}$$

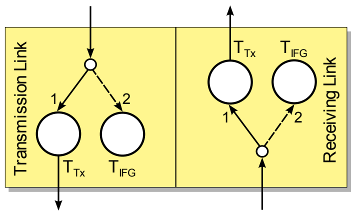

Fig. 1. Model of a full duplex link.

The *frame length* includes the preamble, the start of frame delimiter, and the cyclic redundancy check field. The propagation delay is the time for signal propagation through the physical cable. This delay depends on the medium type and the cable length.

3.2 Station

In order to abstract higher communication layers and to focus on the relevant parts, in this case the data link layer, we choose the model shown in figure 2. We divide this model into two parts: a sender and a receiver part. On the sender-side, we have multiple traffic generators. Each generator represents a certain application on a higher layer and is individually configurable. The multiplexer *LLC*, which models the logical link control layer, combine the data packets that come from the higher-layer service users, into one single buffer. This buffer represents the transmission buffer (Tx buffer) on the network interface card (NIC). The transmission order depends on the queuing discipline. The loss rate depends on the buffer's size, the arrival rate of the multiplexed traffic, and the bit rate of the connected link.

The structure of the receiver part is similar to the sender part. On the data link layer, we have a single queue that represents the receiving buffer (Rx buffer). The server with the holding time T_{NIC} models the delay in the NIC. The delay depends on the processor's performance and the tasks on the data link layer (e. g., error detection). The demultiplexer *LLC* delivers the frames to the higher-layer communication ports. The sinks receive the corresponding frames. In case of a stateful application, a generator and sink might be combined in one entity. Here the generator responds depending on the internal state of the application and type of the incoming frame.

3.3 Switch

The main task of a switch is frame forwarding. Thereby, the destination MAC address determines the forwarding decision, i. e., outgoing port. A MAC address-learning algorithm enables the mapping of destination MAC address and corresponding port. On each received frame, the switch performs the following learning algorithm. The switch stores the source MAC address together with the

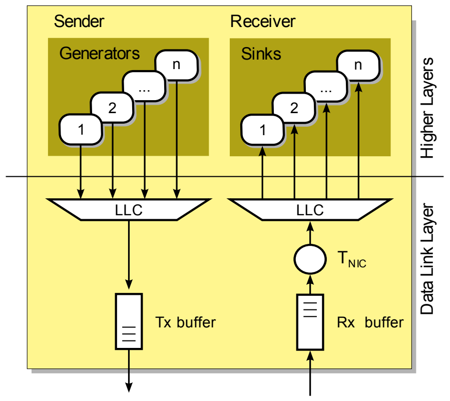

Fig. 2. The model of an Ethernet station.

receiving port identifier in a forwarding database (FDB). The forwarding decision of a frame bases on the FDB look-up for the DA. If the DA is already in the FDB, the switch forwards the frame to the corresponding port. If the database look-up fails, the switch forwards the frame to all ports except the receiving one, i. e., it floods the network.

There are two widely applied forwarding principles implemented in today's switches: *store-and-forward* and *cut-through*. The store-and-forward technique is the standardized one. In this case, the switch receives the frame completely and forwards it after frame processing. The storing enables to check the frame integrity and in case of a failure to drop the frame. The cut-through technique forwards the frame as soon as possible after the reception of parts of the Ethernet header. Thus, it is not able to check the frame integrity. On the other hand, the cut-through technique shows a smaller latency than the store-and-forward technique. However, both techniques are in general not able to guarantee any bounded latency, only statistical guarantees are feasible using prioritization.

The switch model shown in figure 3 follows the store-and-forward principle. The model consists of three parts: (1) Input stage, (2) forwarding stage, and (3) output stage. In case of a non-blocking switch, which is state-of-the-art, the size of each receiving buffer (Rx buffer) has to be equal to the maximum Ethernet frame size. After receiving a frame, the input logic unit reads the frame's SA and updates the FDB, if it does not contain the source address or if the incoming port has been changed. The output logic unit forwards the frame as described above. The switching fabric performs the above-mentioned tasks that need a certain time – the switching latency (SL). The SL is a fixed value that depends on the switch performance and is often provided by the switch vendor. In order to guarantee the non-blocking behavior, we use an infinite server. This server is like an infinite set of single servers connected in parallel. After forwarding the frame, it is buffered in the output stage.

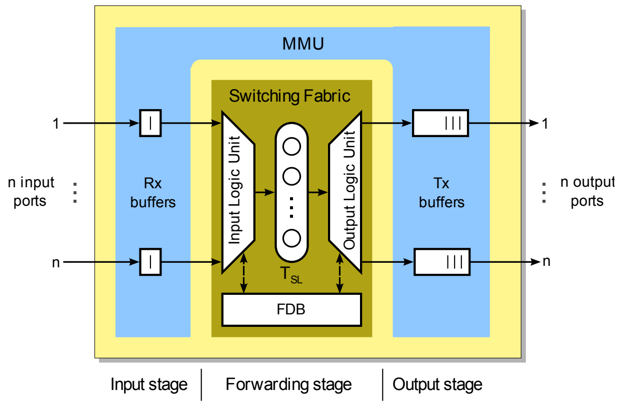

Fig. 3. The model of a switch.

The global memory management unit (MMU) handles memory requests to buffer frames. There are two widely applied memory architectures implemented in today's switches: *shared memory* and *multiple queues*. In the first architecture, all the receiving and transmission buffers share the total memory. A frame loss only occurs when the total memory is allocated. In the multiple queues architecture each buffer has its own dedicated memory (queue). Here, a frame loss occurs as soon as this dedicated queue is full.

4 Implementation

In this section, we introduce the implementation of the simulation program. This includes the architecture and the conceptual structure as well as a discussion of the advantages and the flexibility of the component-based approach. Furthermore, we specify the simulator's settings.

4.1 Architecture and Conceptual Structure

The simulation program bases upon the IKR Simulation Library (IKR SimLib) [14]. The IKR SimLib is an object-oriented class library for discrete event-driven simulation [15] of complex systems in the area of communications engineering. The IKR SimLib is publicly available under the GNU Lesser General Public License (LGPL). To the best of our knowledge, the IKR SimLib and consequently our Ethernet simulator is one of the first network simulators implemented in Java. The IKR SimLib does not utilize any other libraries, beside *Java's Base Libraries*, which are part of the *Java Platform, Standard Edition* (Java SE) [16]. The Java Base Libraries like the *lang* and *util* libraries provide all fundamental data structures, functions, and a rich set of application programming interfaces (APIs).

The IKR SimLib includes all components that are necessary to control and execute an event-driven simulation, e. g., a global calendar and events. While processing an event, it is possible to post new events, which are entered into the calendar. After processing of an event is finished, the next event in the calendar is processed. The simulation control handles the initialization, e. g., when to stop the transient phase and begin with the actual performance evaluation phase and finally when to stop the simulation batches. The control also signals the according changes to all objects needing this information. The library supports also a flexible and modular I/O concept. Part of this concept is a file parser to read parameters, e. g., for simulation control, as well as to define and parametrize the simulation model. The IKR SimLib computes statistical data, e. g., confidence interval, during the simulation. A complex post-processing step is unnecessary like it is often the case in other simulation tools.

In comparison to other simulation tools, the IKR SimLib, and thus the build on simulation program, supports stochastic processes and on the flight statistical evaluation. The latter is supported by many different statistics, e. g., a sample, counter, conditional mean, and correlation statistic. One distinguishing feature from many other simulation tools is the provisioning of metrics dealing with the statistical significance, i.e., the student t-test calculated a confidence interval [15]. Since the IKR SimLib has proved its applicability for performance evaluation in a multitude of communication areas, e. g., for IP, mobile, in-vehicle, and P2P networks, the statistical results have been validated with measurements and analytical results over years [17]. In addition, we successfully validated by exhaustive tests the new, problem-specific components that were implemented for the simulation program.

We decompose each network entity, namely link, station, and switch, into a hierarchy of components. This leads to a hierarchical model. Examples of such components are buffers with a certain queuing discipline or switching fabrics. This component-based approach offers a flexible substitution of components and thus enables a simple comparison of different implementations, e. g., different shapers and policers. It enables the separate handling and treatment of each component. The hierarchical decomposing of a component alleviates also the handling of the network complexity.

In the simulation program, all components are strictly encapsulated and communicate with each other by exchanging messages. This message exchange occurs by using so-called ports, which define a generic external interface of a component. The port concept enables to interconnect components in a plug-n-play manner. Furthermore, a port offers to connect a number of filters and meters. Filters inspect and may change messages based on certain rules. In contrast to this, meters primarily update statistics, with values derived from the messages, e. g., the length or time of arrival.

For illustrating the concepts of the simulation program, figure 4 depicts the sender part of the station's model. The port concept and a message transfer protocol enable that the simulation messages are passed from component to

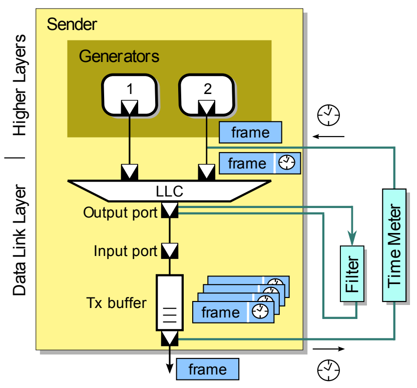

Fig. 4. The port concept and the message-based approach using the example of the sender part of the station model.

component. In the simulation program, an Ethernet frame is representing by a message containing all the information needed.

After a component registers a new message at the output port, this port notifies the corresponding input port. For example, in figure 4 each time when the LLC multiplexer has a new message, its output port informs the input port of the transmission buffer. Then, the receiving component decides if the message will be accepted. Thanks to the flexible port concept, the integration of filters and meters into the model is easy. They read and evaluate the flow of messages at various points within the model. In figure 4, the integrated *TimeMeter* measures the transfer time in the station. For this purpose, the time meter adds a time stamp to the message when this is passing the output port of the traffic generator. When the frame is passing the output port of the station, which is the same as the output port of the transmission buffer, the time meter reads and removes the time stamp. In this figure, we have also a filter that observes the messages that are passing the output port of the LLC multiplexer. For example, this filter might record a trace of messages of a defined traffic class.

4.2 Settings

Each component allows a number of settings. In table 1, we list these settings offered by the simulation program. Thanks to the modular I/O concept of the IKR SimLib, that enables the setting via a parameter file. For reading such a parameter file, the IKR SimLib comes up with its own parser.

The default value of a link bit rate is 100 Mbps. The IEEE 802.3 standard [11] defines an IFG of 96-bit times. This is the default value in the simulation library. It is self-evident that we may also change this default value to evaluate the performance, e.g, of the Ethernet frame burst mode. In most cases, the propagation delay is insignificant and thus the default value is zero. As already mentioned, the propagation delay depends on the medium type and cable length.

Table 1. Settings of the simulation program.

Component	(Sub-)Component	Parameter	Default value	Valid values / range
Link	–	bit rate [Mbps]	100	$1 \ldots \infty$
	–	IFG [bit times]	96	$1 \ldots \infty$
	–	propagation delay [μs]	0	$0 \ldots \infty$
Station	Generators	number of generators	0	$0 \ldots \infty$
	Generator	arrival process	non	set of distributions and source models
	Tx buffer	size [byte]	∞	$1 \ldots \infty$
	Tx buffer	queuing discipline	FIFO	FIFO, strict priority
	Rx buffer	size [byte]	∞	$1 \ldots \infty$
	Rx buffer	queuing discipline	FIFO	FIFO, strict priority
	NIC (receiver part)	T_{NIC} [μs]	non	$0 \ldots \infty$
	Sinks	number of sinks	1	$1 \ldots \infty$
Switch	MMU	architecture	non	shared memory, multiple queues
	Tx buffer	size [byte]	∞	$1 \ldots \infty$
	Tx buffer	queuing discipline	FIFO	FIFO, strict priority, token bucket
	Input and output logic unit	forwarding algorithm	standard	standard, ring
	Forwarding unit	T_{SL} [μs]	non	$0 \ldots \infty$

We consider a heterogeneous, i. e., only fast Ethernet links with 100 Mbps, switched, in-car Ethernet network. Thus, it is more comfortable and well structured to define this parameter globally on system level. However, it is also possible to set up these parameters for each link individually with minor extensions. We define the topology in such a way that we assign in the parameter file each link a start and an end node.

In contrast to a link, we have to configure each station individually in the parameter file. This includes the number of generators, the arrival process for each generator, the size, and the queuing discipline for the transmission buffer and receiving buffer, the holding time T_{NIC} of the NIC's receiving part, and the number of sinks. The IKR SimLib offers a set of distributions (e. g., negative exponential, binomial, Erlang-k, and Weibull) and source models (e. g., On-Off, Video Source, Markov Modulated Deterministic Process).

The queuing discipline can be FIFO or strict priority. In the latter, the queue selects the Ethernet frame with the highest priority. For this purpose, the queue reads the priority information in the TCI field. The IEEE 802.1Q standard defines that the traffic class with number 7 has the highest priority, the traffic class with 0 (default) the lowest [10]. The strict priority queuing discipline enables the evaluation of application assignment to different traffic classes.

The performance of the NIC has an impact on the needed receiving buffer size. For example, if we want to guarantee no frame loss, the NIC has to be powerful enough.

In section 3.3, we introduced a switch model that follows the store-and-forward principle. The simulation program offers a set of settings to configure such a switch entity. For the global memory management unit, which handles memory requests to buffer frames, two principle architectures, which we also introduced in section 3.3, are possible: *shared memory* and *multiple queues*. In case of multiple queues, each transmission buffer has the same size. The queuing discipline of the transmission buffers can be FIFO, strict priority or token bucket. The token bucket algorithm is similar to the credit-based shaper algorithm, which is under discussion as a queuing discipline an audio/video bridging (AVB) compliant switches [7].

The input and output logic support two kinds of topology: A *normal* full duplex topology and a unidirectional ring. The idea of the unidirectional ring comes from the automotive domain [4,5]. In a car, the installation space is limited and the weight plays an important role. In a unidirectional ring, the cabling effort is lower than in Ethernet networks with full duplex links. We integrate a two port switch into each station and use one full duplex port to connect the station and the other one to connect the switch with its successor (Rx-port) and its predecessor (Tx-port).

The current IEEE 802.3 [11] standard does not define this solution and thus we cannot implement it with COTS Ethernet switches. In case of unicast messages, the ring topology works with COTS Ethernet switches. In case of multicast messages, we have to modify the forwarding process that is defined in the IEEE 802.1D [9] standard. This standard defines that a switch forwards a broadcast message to each port except the receiving one. According to this, in a unidirectional ring the switch would forward an incoming broadcast message only to the connected station and not to its successor. On the other hand, if the switch forwards an incoming multicast message to each port, it floods the network. Thus, we have to mark the broadcast messages to identify them after they passed the same switch a second time. Another solution is to expand the Ethernet frame format by an extra time-to-live (TTL) field. Both proposals are not compatible with the IEEE standard [9]. In the simulation program, the output logic unit for a unidirectional ring tags the frame with a unique switch identifier, if necessary. The input logic unit reads this tag and decides if the frame has to be forwarded or not. This is an excellent example of an ongoing discussion about a domain-specific proposal motivated by the requirement of low packaging and weight.

The infinite server of the switching fabric guarantees a non-blocking switch architecture. We assume a constant switching latency for each frame. Thanks to the component-based approach, it is simple to substitute this component by another one, e. g., a single server for modeling a switch with a back-plane archi-

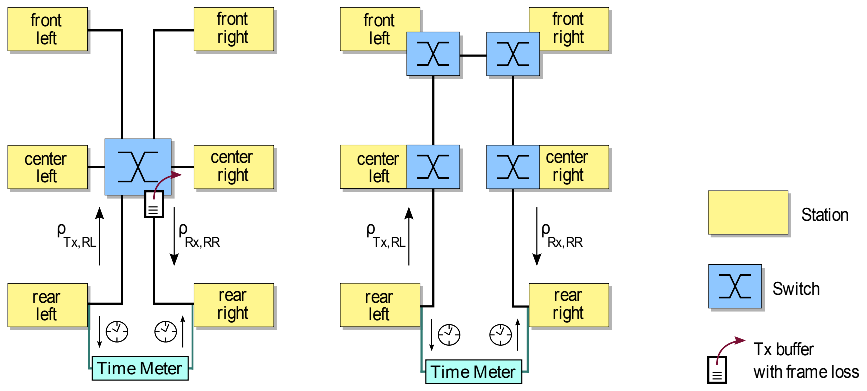

Fig. 5. Two proposals for an in-car Ethernet topology: Star (left side) and daisy-chain topology (right side).

tecture. We are able to model different switch architectures by using different servers.

5 Examples

The simulation model and its implementation have already proved its applicability for performance evaluation in a research project in the domain of in-car communication networks. Therefore, we demonstrate in this section the applicability of the simulation model and its implementation using two examples from the automotive domain. We investigate a star topology and a daisy-chain topology depicted in figure 5 and evaluate their QoS guarantee in section 5.1. Furthermore, we discuss the results of both examples in a short manner. Moreover, the examples should demonstrate the nature of the simulation results and the benefits of the simulator.

Each network in figure 5 consists of 6 stations, namely front left (FL), front right (FR), center left (CL), center right (CR), rear left (RL), and rear right (RR). We deploy five, equal generators, one in each station except in station RR, and one sink in station RR. Each generator (source) sends tagged Ethernet frames with maximum length according to a general distribution with a certain mean inter arrival time μ and given coefficient of variation CoV. The general distribution used generates frames by linking two phases (phase model). The greater the value of CoV, the higher the burstiness of the traffic.

With the mean inter arrival time μ and the length of the Ethernet frames sent *len*, we can calculate the traffic load ρ of each station as

$$\rho = \frac{\frac{\text{len}}{\mu}}{\text{bit rate}}. \tag{2}$$

As an example, we investigate the end-to-end latency and the impact of priority queuing disciplines in the switches' queues in dependency of different link utilization. The automotive industry operates in a highly competitive market where the pressure on any costs is very high. Consequently, we have to dimension the resources needed in order to achieve minimal costs [18]. In section 5.2, we analyze, as an example, the frame loss probability in dependency of the buffers' sizes. In both examples, all the stations generate the same traffic load and send frames to the same destination, namely station RR. Therefore, we can calculate the utilization of RR's receiving link as

$$\rho_{\mathrm{Rx,RR}} = 5 \cdot \frac{\frac{\mathrm{len}\,+\,\mathrm{len}_{\mathrm{Pre}}}{\mu}}{\text{bit rate}} \tag{3}$$

where $\mathrm{len}_{\mathrm{Pre}}$ is the length of the preamble. Furthermore, the station CL sends tagged Ethernet frames with the highest priority (7), the other ones with a middle priority (4). Besides, we assume a propagation delay of zero bit times.

5.1 End-to-end Latency

In case of the daisy-chain topology, we assume three port switches that are embedded into the stations. The advantages are the low cabling effort (less cable bundles). On the other side, this topology requires a larger number of ports in comparison to the star topology, which leads to higher costs and the end-to-end latency might be higher due to the larger number of intermediate switches. For the latter reason, we have to evaluate if this topology can fulfill the QoS requirements in terms of end-to-end latency.

The minimum end-to-end $T_{\mathrm{e2e,min}}$ latency of a frame is

$$T_{\mathrm{e2e,min}} = \sum_{i=1}^{n} T_{\mathrm{Tx}} + \sum_{i=1}^{n-1} T_{\mathrm{SL}}, \tag{4}$$

where n is the number of switches on the path from the source to the destination. The minimum end-to-end latency is only reached in the optimal case of empty transmission buffers in each switch and each previous frame was sent more than 96-bit times before. Otherwise, we have to sum up the waiting time in the switches' transmission buffers and the residual IFG times, too.

Figure 6 shows the end-to-end latency with different topologies, queuing disciplines, and CoV. We do not analyze the results in detail, since the objective of this section is to demonstrate an exemplary application of the simulation program. However, we see that the strict priority queuing discipline improves the mean end-to-end latency of frames with the highest priority. As expected, the end-to-end latency in the daisy-chain topology is longer due to the larger number of intermediate switches. Furthermore, a greater value of CoV leads to a higher burstiness, which leads again to a larger number of frames in the queues and to a longer end-to-end latency.

It is also possible to plot a complete probability density function (PDF), e. g., for analyzing the jitter, not only the mean and 95 quantile values.

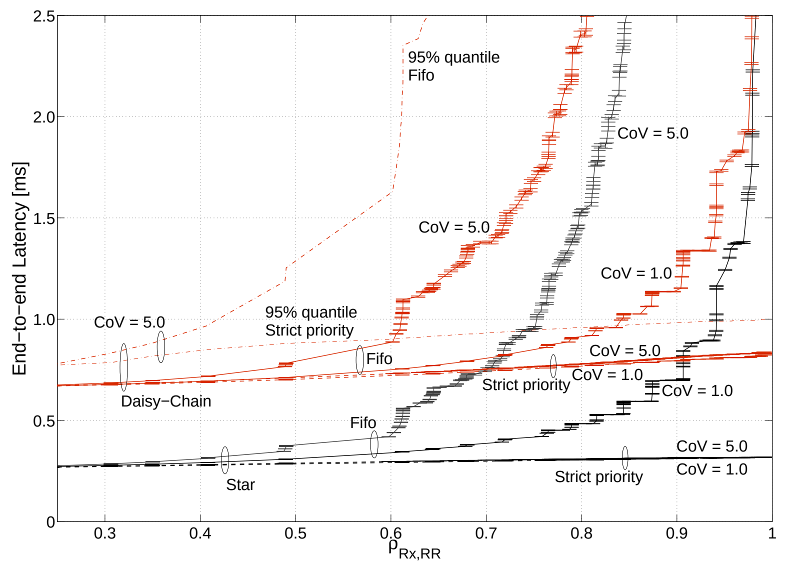

Fig. 6. The end-to-end latency of frames with RR as source and RL as destination station with different topologies, queuing disciplines, and coefficients of variation.

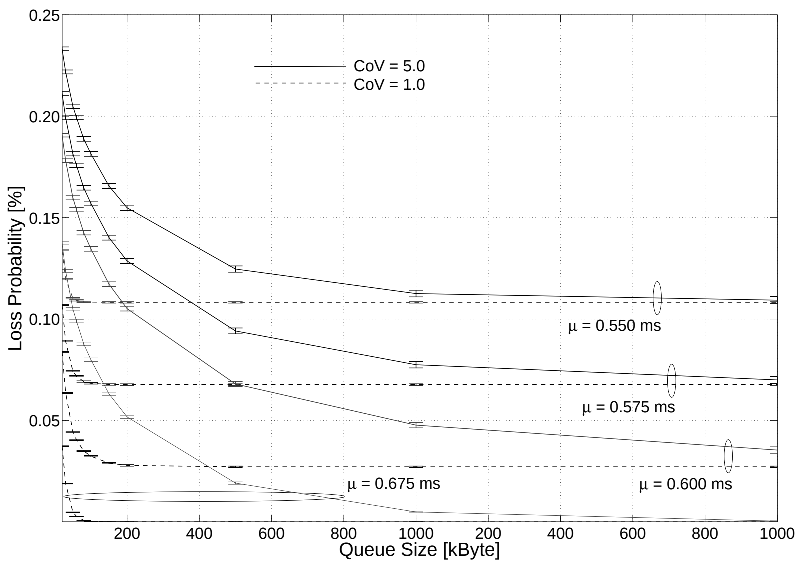

Fig. 7. The loss probability of frames in the switch's transmission buffers that goes in the direction to the station RR with different queue sizes and coefficient of variation.

5.2 Resource Dimensioning

In this example, we will analyze the frame loss probability in dependency of the buffers' sizes of the switches. The delay in the transmission buffers of the switch will become long in case of an overloaded network. Among other things, the buffer's size affects the loss probability and the mean waiting time of frames. In in-car networks with real-time applications, it might be preferred to drop a frame instead of queuing and wasting resources while its information gets obsolete. This leads to trade-off between loss probability and mean waiting time. The challenge is to find a proper network dimensioning.

We investigate different buffers sizes with 10 kByte up to 1 MByte. While we have in the daisy-chain topology no frame losses in any switch, we depict in figure 7 the frame loss probability of the switch's transmission that goes in the direction to the station RR. As expected, with increasing link utilization and a decreasing buffer size the frame loss probability increases. In the daisy-chain topology in any transmission buffer frames coming from maximal two links interfere with each other. On the contrary, in the star topology, in the transmission buffer considered frames coming from five links interfere with each other. Normally, the next step would be to investigate the total memory needed in the whole Ethernet network.

6 Conclusion and Outlook

Motivated by the growing range of application areas for Ethernet as well as the permanent evolution of the Ethernet standards, we presented a component-based simulation model for a switched Ethernet network. This modular designed model enables to evaluate enhancements like a novel traffic scheduler in a simple manner by modifying or substituting the affected components. These components formed the basis of the simulation program. At next, we described the settings of the simulation program. Finally, we demonstrated the applicability using examples from the automotive domain.

Part of our ongoing work is to evaluate the performance of novel schedulers and shapers for resource-efficient in-car communication that support services for applications with real-time requirements. As today's Ethernet show meshed topologies, loops may occur in the network, which may cause infinite cycling of messages and network flooding in case of broadcast messages. As native Ethernet is not capable to avoid loops, Radia Perlman developed the spanning tree protocol (STP) [9]. The drawbacks of STP are its high convergence time and that it reacts only in case if the physical topology changes. It does not consider system's states like link loads. Currently, we are working on a novel dynamic forwarding algorithm, which eliminates the drawbacks of STP and improves the QoS in terms of end-to-end latency. For this purpose, we monitor the link loads and reconfigure the logical topology dynamically in case of a highly loaded link. This algorithm is also applicable for an in-car bidirectional Ethernet ring. The advantages of this topology are the redundancy and its reliability. In the near future, we will present this algorithm and its improvements.

References

1. Sommer, J., Gunreben, S., Feller, F., Köhn, M., Mifdaoui, A., Saß, D., Scharf, J.: Ethernet – a survey on its fields of application. IEEE Communications Surveys & Tutorials **12**(2) (2010)
2. ARINC 664: Aircraft Data Network, Part 7: Deterministic Networks. (2003)
3. Hillebrand, J., Rahmani, M., Bogenberger, R., Steinbach, E.: Coexistence of Time-Triggered and Event-Triggered Traffic in Switched Full-Duplex Ethernet Networks. In: Proceedings of the IEEE Second International Symposium on Industrial Embedded Systems – SIES'2007. (2007) 217–224
4. Rahmani, M., Steffen, R., Tappayuthpijarn, K., Steinbach, E., Giordano, G.: Performance Analysis of Different Network Topologies for In-vehicle Audio and Video Communication. In: 4th. International Telecommunication Networking Workshop on QoS in Multiservice IP Networks. (February 2008) 179–184
5. Rahmani, M., Tappayuthpijarn, K., Krebs, B., Steinbach, E., Bogenberger, R.: Traffic Shaping for Resource-Efficient In-Vehicle Communication. IEEE Transactions on Industrial Informatics (2009) Accepted for future publication.
6. IEEE Computer Society: P802.1Qat/D1.3 Draft Standard for Local and Metropolitan Area Networks–Virtual Bridged Local Area Networks–Amendment: Stream Reservation Protocol (SRP). Technical report, IEEE (May 2008) This is an unapproved IEEE Standards Draft.
7. IEEE Computer Society: P802.1Qav/D3.1 Draft Standard for Local and Metropolitan Area Networks–Virtual Bridged Local Area Networks–Amendment: Forwarding and Queuing Enhancements for Time-Sensitive Streams. Technical report, IEEE (October 2008) This is an unapproved IEEE Standards Draft.
8. IEEE Computer Society: IEEE Standard for Local and Metropolitan Area Networks: Overview and Architecture (2002)
9. IEEE Computer Society: 802.1D: IEEE Standard for Local and Metropolitan Area Networks–Media Access Control (MAC) Bridges (2004)
10. IEEE Computer Society: 802.1Q: IEEE Standard for Local and Metropolitan Area Networks–Virtual Bridged Local Area Networks (2005)
11. IEEE Computer Society: 802.3: IEEE Standard for Local and Metropolitan Area Networks–Carrier sense multiple access with collision detection (CSMA/CD) access method and physical layer specifications (2005)
12. Spurgeon, C.E.: Ethernet: The Definitive Guide. Volume 1. O'Reilly Media (February 2000)
13. Metcalfe, R.M., Boggs, D.R.: Ethernet: Distributed Packet Switching for Local Computer Networks. Technical report, XEROX Palo Alto Research Center, 3333 Coyote Hill Road, Palo Alto, California, USA (1975)
14. Institute of Communication Networks and Computer Engineering: IKR Simulation and Emulation Library (2009) http://www.ikr.uni-stuttgart.de/IKRSimLib.
15. Law, A.M., Kelton, W.D.: Simulation Modeling and Analysis. McGraw-Hill Inc. (December 1990)
16. Sun Microsystems, I.: Java SE overview – at a Glance (2009)
17. Institute of Communication Networks and Computer Engineering: IKR Simulation Library – Usage in projects (2009) http://www.ikr.uni-stuttgart.de/Content/IKRSimLib/Usage/.
18. Sommer, J., Blind, R.: Optimized Resource Dimensioning in an embedded CAN-CAN Gateway. In: Proceedings of the IEEE Second International Symposium on Industrial Embedded Systems – SIES'2007. (July 2007) 55–62