WOPANets: A Tool for WOrst Case Performance Analysis of Embedded Networks

Ahlem Mifdaoui
Toulouse University
ISAE
1 place Emile Blouin
31056 Toulouse, France
Ahlem.Mifdaoui@isae.fr

Hamdi Ayed
Toulouse University
ISAE
1 place Emile Blouin
31056 Toulouse, France
Hamdi.Ayed@isae.fr

***Abstract*— WOPANets (WOrst case Performance Analysis of embedded Networks) is a design aided-decision tool for embedded networks. This tool offers an ergonomic interface to the designer to describe the network and the circulating traffic and embodies a static performance evaluation technique based on the Network Calculus theory combined to optimization analysis to support early system level design exploration for embedded networks.**

In this paper, we describe the features set of WOPANets tool and we provide analysis in the case of a realistic embedded avionic network to show how the variation of some system's parameters can improve the system performances to support the required constraints.

I. Introduction and Related Work

With the increasing complexity of embedded networks and the expansion of exchanged data quantity, making the right design decisions to guarantee the system's requirements becomes a difficult task for designer. Hence, it becomes one of the major challenges in the design process to analyze the system's characteristics till the first steps of the development cycle.

Simulations are commonly used for exploring the design space and validating the functional and timing behaviors of the embedded networks [1], [2]. However, these approaches are feasible for only small parts of the system and cannot cover the entire domain of the model applicability and specially rare events that represent worst-case functioning of the system. So, clearly, simulations cannot provide the deterministic guarantees required by critical embedded networks with hard certification requirements to respect like in civil and military avionics, automotive and satellites, where a failure might have a disastrous consequence on the system. With a formal specification language like StateCharts [3] or Specification and Description Language (SDL) [4], it is possible to verify the functional behavior of the network. However, for big networks with an important number of nodes, this approach leads to a space explosion problem inherent to the reachability analysis techniques implemented by the formal verification tools.

In order to overcome these limitations, our proposal consists in integrating an aided-decision tool till the first steps of the design process to choose the right system's parameters which respect the required constraints before investing too much time in detailed implementations. This tool embodies an analytical approach based on the Network Calculus formalism [5] to perform a system level worst case performance evaluation and includes optimization analysis to find the optimal design solution that guarantees the system's requirements. This approach would enhance the design process time and costs.

The Network Calculus has recently attracted a lot of attention as a deterministic framework for worst case analysis of delay and backlog bounds for communication networks as for example, Switched Ethernet [6], [7], wireless sensor networks [8], [9] or even System On chip [10]. Furthermore, there are available tools for network-calculus analysis based on Matlab like RTC [11] and CYNC [12] toolboxes or on java like DISCO toolbox [13]. However, these tools require a good knowledge of the Network Calculus formalism to be used.

The main contribution of our work is to offer a new tool for worst case performance and optimization analysis of embedded networks based on system level models and easy to use by any designer without any specific knowledge of the Network Calculus formalism.

In the next section, an overview of the Network Calculus formalism is provided. Afterward, the WOPANETS tool features and structure are detailed in section 3. Finally, the practical feasibility of our tool is illustrated within a realistic application in section 4. Section 5 concludes the paper.

II. Background

Network Calculus formalism [5] is based on *min-plus* algebra for designing and analyzing deterministic queuing systems where the compliance to some *regularity constraints* is enough to model the traffic. These constraints limit traffic burstiness in the network and are described by the so called *arrival curve* $\alpha(t)$, while the availability of the crossed node is described by a *service curve* $\beta(t)$. The knowledge of the arrival and service curves enables the computation of the delay bound that represents the worst case response time of a message, and the backlog bound that is the maximum queue length in the node. The delay bound D is the maximal horizontal distance between $\alpha(t)$ and $\beta(t)$ whereas the backlog bound B is the maximal vertical distance between them.

This formalism gives an upper bound for the output flow $\alpha^*(t)$, initially constrained by $\alpha(t)$ and crossing a system with a service curve $\beta(t)$, using min plus deconvolution $\oslash$ where:

$$\alpha^*(t) = sup_{s\geq 0}(\alpha(t+s) - \beta(s)) = (\alpha \oslash \beta)(t) \quad (1)$$

Another important result given in the Network Calculus formalism is the concatenation theorem that is as follow:

Assume a flow with arrival curve $\alpha(t)$ traverses systems S1 and S2 in sequence where S1 offers service curve $\beta 1(t)$ and S2 offers $\beta 2(t)$. Then, the concatenation of these two systems offers the following single service curve $\beta(t)$ to the traversing flow:

$$\beta(t) = (\beta 1 \otimes \beta 2)(t) = inf_{0\leq s\leq t}(\beta 1(t-s) + \beta 2(s)) \quad (2)$$

There is also another known result concerning the blind multiplexing:

Assume flows 1 and 2 with arrival curves $\alpha 1(t)$ and $\alpha 2(t)$ traverse system S which offers a strict service curve $\beta(t)$. Then, the minimal service curve offered to flow 1 is:

$$\beta 1(t) = \max(0, \beta(t) - \alpha 2(t)(t)) \quad (3)$$

However, this result has to be used carefully because the strict service curve assumption is essential and it is not verified in the general case except when the crossed node has a constant rate service or a FIFO multiplexing service. Further explanations could be found in [5].

III. WOPANets Features and Structure

A. Features Set

The WOPANets tool can handle the following parameters:

- Traffic types: periodic traffic and aperiodic traffic with jitter or not.
- Different communication types: unicast, multicast and broadcast.
- Technology types: many technologies are supported by WOPANets like Ethernet, AFDX and SpaceWire.
- Different scheduling policies like First Come First Served (FCFS), Static Priority (SP), Weighed Fair Queuing (WFQ), Round robin (RR); and many control mechanisms like TDMA, Master/Slave and Token Ring.
- Different performance metrics like: end to end delays, used memory, network load and loss rate.
- Different optimization constraints (temporal and hardware), different variable parameters (discrete and continuous) and mono objective criteria.

B. WOPANets Structure

WOPANets tool consists of three main modules as described in the figure 1: the Graphical User Interface, the Network Calculus Analyzer and the Optimization Analyzer.

1) Graphical User Interface (GUI): The Graphical User Interface allows the user to define:

- the network topology using different types of nodes like End System, Switch or Router with different variable parameters like transmission capacity, memory, policy or control mechanism. An example of topology is shown in figure 2(a);

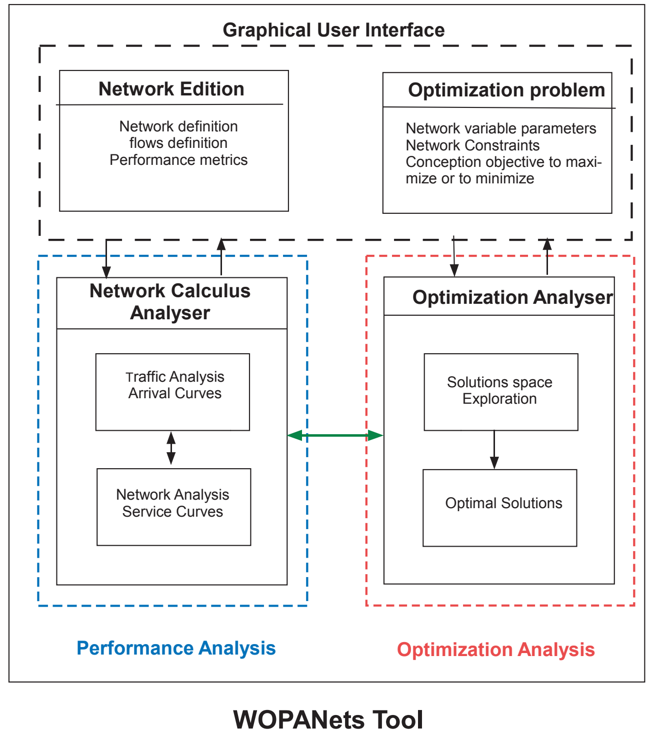


Fig. 1. WOPANETS Structure

- the circulating flows of different types (periodic or aperiodic with jitter or not) where each flow has different variable characteristics like length, period, deadline, burst, jitter, priority, source, destination. The flows could be created one by one or loaded from an input XML file. An example is shown in figure 2(b).

Then, the user can launch the performance analysis and display the selected metric, as for example the histogram of the maximal end to end delay bounds or the maximal backlog bound in each node or the loss rate in the network due to temporal and memory constraints violation as shown in figure 3.

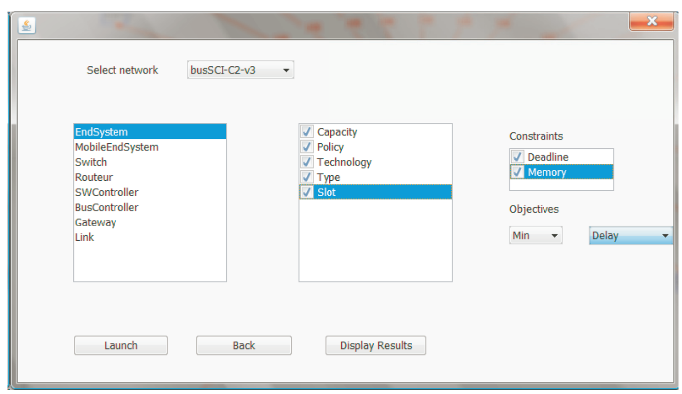
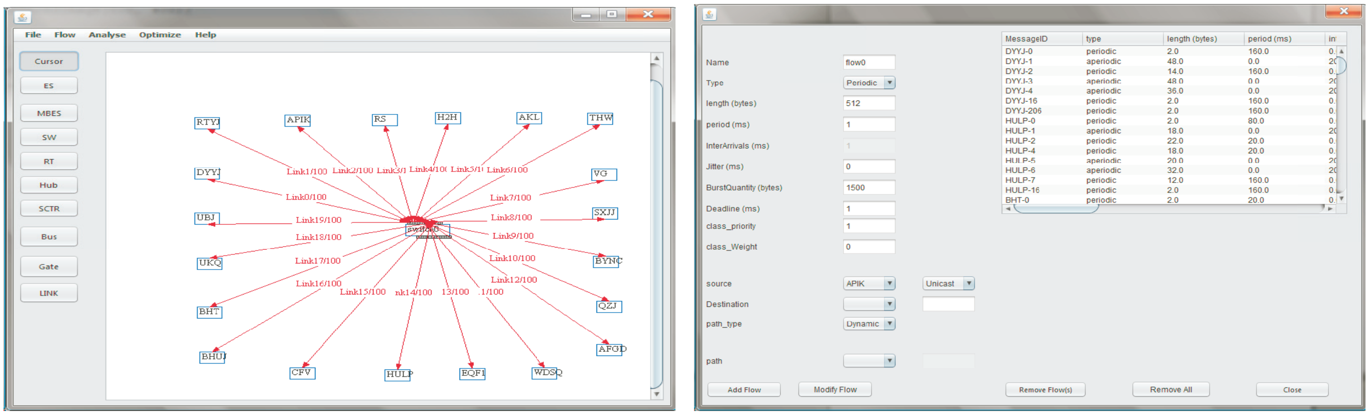


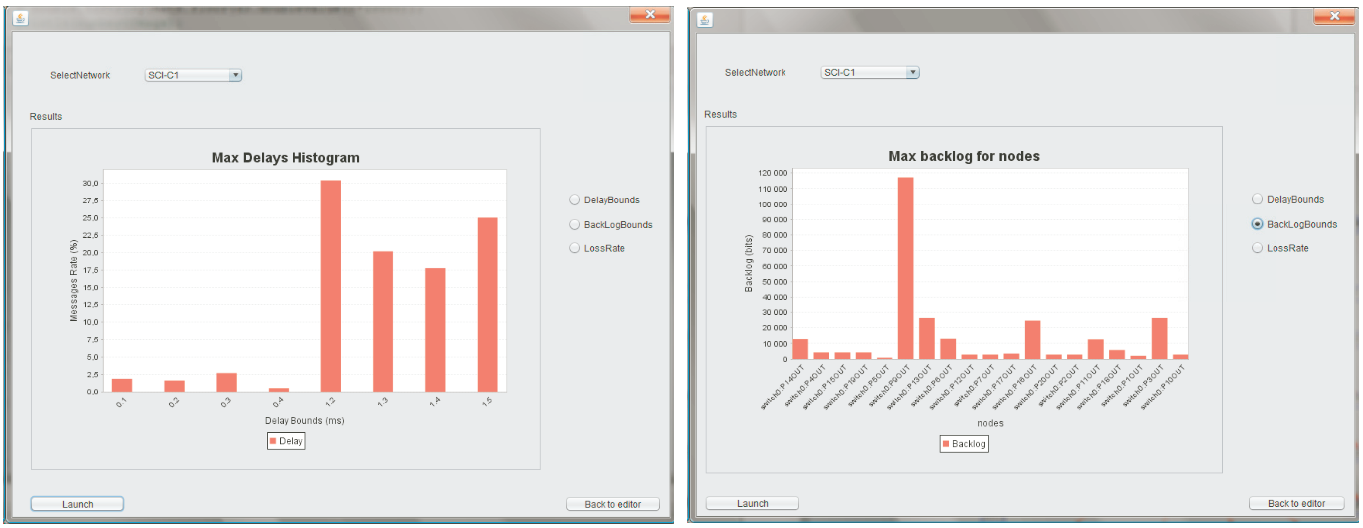
Fig. 4. GUI screen shot: Optimization Problem Definition

Furthermore, the interface allows the user to define the optimization problem with the different variable parameters



(a) Example of an input technology (b) Flows Definition

Fig. 2. GUI screen shot: Network Edition



(a) Maximal Delay Bounds Histogram (b) Backlog Bounds

Fig. 3. GUI screen shot: Performance Analysis Display

of the system, the different constraints to integrate and the design objective to minimize or to maximize. Then, the user can launch the optimization analysis and display the obtained optimal solutions as shown in figure 4.

2) Performance Analyzer: First, the performance analyzer defines the arrival curve of each flow according to its characteristics and the service curve of each node in the network according to its policy or its control mechanism.

Afterward, in order to calculate the metric selected by the user which could be either the maximal end to end delay bound, the maximal backlog bound or the maximal loss rate bound, there are two possible performance analysis algorithms: Propagation Analysis algorithm and PBOO (Pay Burst Only Once) algorithm. The former is the easiest one where the flows are analyzed as a whole in each crossed node and the calculus are propagated from one node to another using the min-plus deconvolution (1); whereas the latter gives less pessimistic bounds for each individual flow using the concatenation theorem (2) and the blinding multiplexing property (3). However, the PBOO algorithm could not be used in the general case due to the blinding multiplexing restrictions as explained in section II.

The propagation analysis algorithm is described in algorithm 1. First, the sets of received streams at each terminal are identified (line 5). Then, for each stream in the identified set, it determines its initial arrival curve (line 9), its associated path (line 10) and the service curves offered by crossed components along that path according to their processing mechanism (line 11). Afterward, the delay bound calculation is propagated from one crossed component to another by resolving the burstiness constraint evolution of each stream. Knowing the arrival curve and service curves, the submitted delay and backlog bounds are calculated for each stream (lines 13-14) and then its output arrival curve (line 15). This latter curve will be the input arrival curve for the next network component and so on until the last component. Now, since submitted delay bounds are known for

Algorithm 1 Propagation Analysis Algorithm

1: $T \leftarrow \{T_1, T_2 ... T_{n_{terminals}}\}$
2: $S \leftarrow \{s_1, s_2 ... s_{n_{streams}}\}$
3: $EED_{DEST} \leftarrow$ HashMap <Terminal, List <double>>
4: $Backlogs \leftarrow$ HashMap <Terminal, double>
5: **for** $i = 1$ to $n_{terminals}$ **do**
6: $\quad R \leftarrow$ Vector-rcv-streams(T_i, S)
7: $\quad EDD_{streams} \leftarrow$ List $(R.length)$
8: $\quad$ **for** $j = 1$ to $R.length$ **do**
9: $\quad\quad \alpha \leftarrow$ Initial-arrival-curve$(R(j))$
10: $\quad\quad$ Path $\leftarrow$ Vector-crossed-components$(R(j))$
11: $\quad\quad \beta \leftarrow$ Vector-service-curves(Path)
12: $\quad\quad$ **for** $k = 1$ to Path.length **do**
13: $\quad\quad\quad D \leftarrow$ Delay-calculus $(\alpha, \beta(k))$
14: $\quad\quad\quad B \leftarrow$ Backlog-calculus $(\alpha, \beta(k))$
15: $\quad\quad\quad \alpha \leftarrow$ Deconvolution $(\alpha, \beta(k))$
16: $\quad\quad\quad EED_{streams}(j) \leftarrow EED_{streams}(j) + D$
17: $\quad\quad$ **end for**
18: $\quad$ **end for**
19: $\quad EED_{DEST}(i) \leftarrow < T_i, EED_{streams} >$
20: $\quad Backlogs(i) \leftarrow < T_i, B >$
21: **end for**

each stream and in each point of the network, a maximal end-to-end delay bound can be determined for each stream along its path (line 16).

3) Optimization Analyzer: The Optimization Analyzer integrates the different parameters of the optimization problem fixed by the user. Then, it modelizes the optimization problem variables, constraints and objective to minimize or maximize. Given the variables and constraints types, the analyzer chooses the accurate algorithm for the solutions space exploration. We consider two algorithms: the SIMPLEX and the genetic algorithm. Candidate architectures are generated using a library of components and each candidate is evaluated based on a set of metrics that may guide future architecture generation. Optimal solutions are then described to the user within the Graphical User Interface. Currently, the optimization features are under implementation and will not be addressed for the case study.

IV. Using WOPANets

A. Case Study

The aim of this case study is to find a network architecture based on Switched Ethernet to replace the current network in a modern French military aircraft that respects the required temporal constraints. We will show through this case study how the variation of some parameters can improve the network performances and enforce the network to fulfill the system requirements.

The Network consists of seven switches as shown in figure 5. The traffic is circulating between about eighty subsystems. The different categories of the Real-time traffic are described in tables I and II. So, one can see that for periodic messages, the largest period is about 160 ms and the most common value is 20 ms; and for aperiodic messages, there are different response time bounds and the most urgent one is about 3 ms.

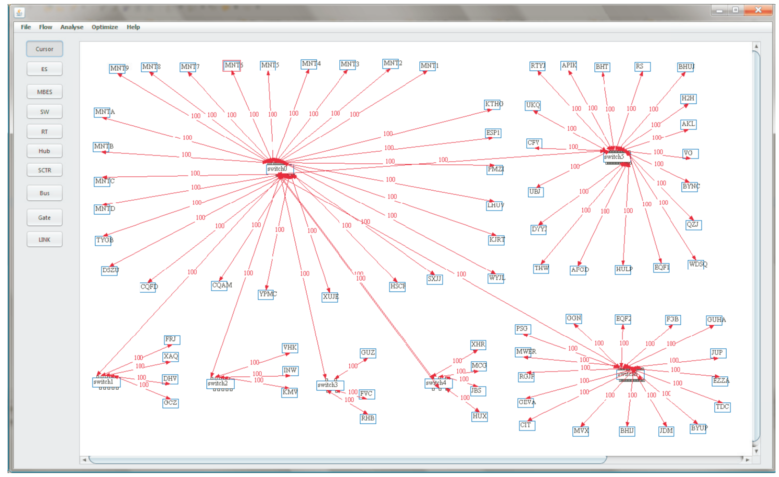


Fig. 5. The Input Topology of the Case Study

TABLE I
Periodic Traffic Description

Period (ms)	Number of flows	Data payload (bytes)
20	698	92
40	60	92
80	56	92
160	630	1492

TABLE II
Aperiodic Traffic Description

Response time (ms)	Number of flows	Data payload (bytes)
3	106	14
20	420	92
160	215	92
infinity	360	1492

B. Obtained Results

For the considered network, the objective of the designer is to have zero loss rate.

First, we start by testing a network architecture with 100Mbps as transmission capacity, FCFS as scheduling policy and switches having a memory of 32Mbytes. The end-to-end delay bounds for each stream are computed and the obtained end-to-end delay bounds are larger than 3 ms which means that the deadline constraint for the aperiodic messages is violated. Moreover, the maximal loss rate bound is about 1.6% due to temporal constraints violation. This means that the switches memories are enough to avoid queue overflows but the selected architecture does not fulfill the designer objective.

Hence, we keep the transmission capacity of 100Mbps and the switches memories of 32Mbytes and we choose SP as scheduling policy to guarantee the priority handling. We define four priorities: the highest priority 0 for aperiodic traffic with response time 3ms, the medium priorities for periodic traffic and aperiodic traffic with response time less than 160ms, and the lowest priority 3 for non real time messages with infinite response time. The obtained end to end delay bounds are inherently reduced for urgent aperiodic traffic and satisfy the associated deadline constraint. However, the loss rate is about 1.1% which still is not acceptable by the designer.

Then, we increase the transmission capacity to 1Gbps and we keep a simple scheduling policy FCFS and switches memories of 32MBytes. The obtained maximal loss rate bound is equal to zero. The obtained histogram of the maximal end to end delay bounds is described in figure 6 and as it can be noticed 93% of messages have delay bounds less than 1ms. Hence, this architecture could be considered as a satisfying solution which respects the required temporal constraints.

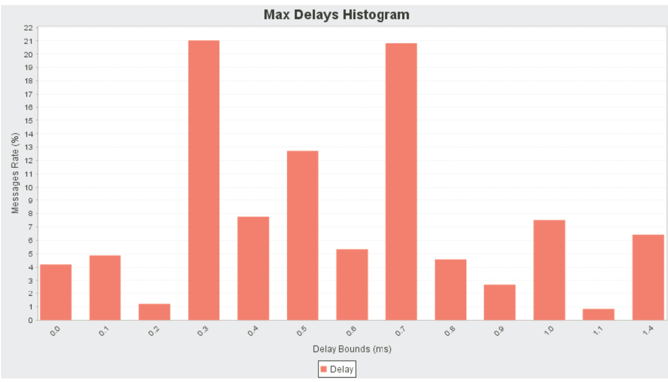


Fig. 6. Maximal Delay Bounds Histogram (1Gbps)

The tool run time for this case study was about 3 seconds and as shown in figure 7, the tool run time is less than 15 seconds for different network configurations with a number of hops that varies from 1 to 5 and a number of flows that varies from 200 to 6000.

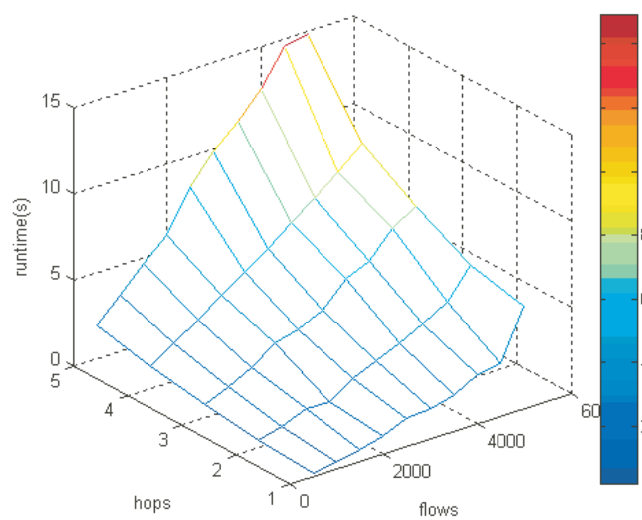


Fig. 7. Tool Run time as a function of the number of hops and flows

Hence, we have shown through this case study the ability of WOPANets tool to help the designer to find an admissible solution that respects the fixed objective in a very short time.

V. Conclusion and Future Work

The WOPANets is presented in this paper as an easy design aided-decision tool that allows the designer to choose the good parameters of the system that respect the required constraints before investing too much time in detailed implementations, as shown through the case study.

This tool takes into account different parameters to lead the performance analysis: (i) different types of traffic like periodic and aperiodic with jitter or not; (ii) multiple classes of priorities and communications (unicast, multicast and broadcast); (iii) different types of nodes with different characteristics like technology, transmission capacity, memory, control mechanism or scheduling policy; (iv) different performance evaluation metrics like end to end delays, used memory and loss rate.

As future work, some features would be finalized and tested through WOPANets like the optimization analyzer and the integration of more technologies like CAN, TTCAN, FlexRay, wireless sensor networks and System On Chip.

References

[1] R. Henia, A. Hamann, M. Jersak, R. Racu, K. Richter, and R. Ernst, " System level performance analysis - the SymTA/S approach ." Proceedings of IEEE Computers and Digital Techniques, 2005.

[2] S. Schliecker, J. Rox, M. Negrean, K. Richter, M. Jersak, and R. Ernst, " System Level Performance Analysis for Real-Time Automotive Multicore and Network Architectures ," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 28, 2009.

[3] J. Whittle and J. Schumann, " Generating statechart designs from scenarios." Proceedings of the 22nd international conference on Software engineering, 2000.

[4] F. Belina, D. Hogrefe, and A. Sarma, " SDL with applications from protocol specification ," *Prentice-Hall Bcs Practitioner Series archive*, vol. 20-13-785890-6, 1991.

[5] J. Leboudec and P. Thiran, *Network Calculus*. Springer Verlag LNCS volume 2050, 2001.

[6] T. Skeie, S. Johannessen, and O. Holmeide, " Timeliness of real-time IP communication in switched industrial Ethernet networks ," *IEEE Transactions on Industrial Informatics*, vol. 2, 2006.

[7] A. Mifdaoui, F. Frances, and C. Fraboul, " Performance Analysis of a Master/Slave Switched Ethernet for Military Embedded Applications ," *IEEE Transactions on Industrial Informatics (In press)*, 2010.

[8] J. Schmitt, F. Zdarsky, and L. Thiele, "A Comprehensive Worst-Case Calculus for Wireless Sensor Networks with In-Network Processing ." Proceedings of Real-Time Systems Symposium, 2007.

[9] P. Jurcik, R. Severino, A. Koubaa, M. Alves, and E. Tovar, "Real-Time Communications Over Cluster-Tree Sensor Networks with Mobile Sink Behaviour ." Proceedings of Real-Time and Embedded Computing Systems and Applications conference, 2008.

[10] S. chakraborty, S. Kuenzli, L. Thiele, and A. Herkersdorf, " Performance evaluation of network processor architecture: combining simulation with analytical estimation," *Computer Networks*, vol. 42, 2003.

[11] "RTCtoolbox home page." [Online]. Available: http://www.mpa.ethz.ch/Rtctoolbox/Overview

[12] "CYNC home page." [Online]. Available: http://www.control.aau.dk/~henrik/CyNC/manual/CyNC

[13] J. B. Schmitt and F. A. Zdarsky., " The DISCO Network Calculator - A Toolbox for Worst Case Analysis." In Proceedings of the First International Conference on Performance Evaluation Methodologies and Tools, 2006.