CyNC - a MATLAB/SimuLink Toolbox for Network Calculus

Henrik Schioler
Center for Embedded SW Systems/CISS
Institute for Electronic Systems
Aalborg University
Fr. Bajersvej 7G
9220, Aalborg, Denmark
henrik@control.aau.dk

Hans P. Schwefel
Section for Network Security
Aalborg University
Fr. Bajersvej 7G
9220, Aalborg, Denmark
hps@kom.aau.dk

Martin B. Hansen
Mathematical Institute
Aalborg University
Fr. Bajersvej 7G
9220, Aalborg, Denmark
mbh@math.aau.dk

ABSTRACT

This paper presents a graphical integrated modelling and performance-analysis tool based on deterministic network calculus (DNC) and implemented as an open source toolbox for the MATLAB/SimuLink environment. The paper introduces briefly the main concepts from network calculus and especially recent results for systems with cyclic dependencies, which appear in cases of cyclic data/work flow or counter directional resource and work flows. A number of network element types are supported including various arbitration/scheduling disciplines such as: Fixed Priorities, FIFO, TDMA, round robin/token passing and EDF along with packetization, flow control and flow convergence. These are all presented in the paper together with auxiliary tools like worst case backlog and delay calculations. Implementation details of general interest are presented along with illustrative examples demonstrating the virtues of the separate modelling elements and the overall tool framework. Discussion is provided concerning issues in system stability and the ability of DNC to provide usefull estimates of stability limits. Likewise current activities to support synchronous communication and flow control within the tool are presented.

Keywords

Network calculus, real time systems, modelling, performance analysis, tools

1. INTRODUCTION

Performance analysis for time sensitive systems appears in various shapes depending on the system context and the focus adopted by the analyst. Scheduling analysis [1], [2] provides analytic sufficient and exact criteria for the feasibility of fixed priority schedules including bounded blocking from lower priority tasks. [17] provides exact feasibility criteria for EDF (Earliest Deadline First) schedules based on task arrival bounds and relative job deadlines. Various results [3] exist for multi processor scheduling regarding both generation, feasibility and optimality of schedules. In this paper however, the paradigm adopted for multi processor system, is a highly versatile hardware and software platform comprising a number of hardware processors each equipped with suitable a operating system including a *plain vanilla* scheduler selected among a number of standard scheduling policies such as *Fixed Priorities*, *Round Robin*, *Time Division Multiple Access*, *EDF*, *Weighted Fair Queueing* etc. At a system design level or at the time of deployment of a newly installed 3 rd. party application, processing needs are identified and mapped to the platform as chains or multi chains of processing stages connected by job/event flows. Performance analysis should in this situation provide system wide results for local delay and backlog as well as upper delay bounds for entire processing chains/transactions.

While scheduling theory provides operational results for task dependence in terms of blocking/mutual exclusion, it so far failed to provide results for the most frequent pattern of Inter Process Communication (IPC) namely producer/consumer relations between tasks.

Other more general and precise methods for modelling and analysis of real time systems, such as Timed Automata [8], and Time Petri Nets [9] have for a long time been and are still the objects for active research. However they all seem to bear along inherent complexity difficulties, especially for systems dominated by asynchronous communication. This is mainly caused by the fact, that queue states themselves contribute to system state dimension, and also because asynchronicity allows subsystem states to evolve independently and in turn increase the size of the reachable state space.

Deterministic Network Calculus (DNC) was suggested by [5] in 1991 for real time analysis of QoS sensitive network applications. DNC occupies the gap between queueing theory on the one hand and scheduling theory on the other. The term *deterministic* was probably intended to distinguish the framework from the *probabilistic* queueing theory but may appear misleading, when the analysis is indeed non- deterministic, i.e. input data and results typically appear as intervals.

However, through the unification of queueing and scheduling in a non-deterministic framework, DNC offers analysis tools

Valuetools '07, October 23-25, 2007, Nantes, France

for a large variety of time-sensitive applications including distributed systems as well as embedded real time systems and combinations hereof. This was indeed recoqnized by [6] introducing the concept of *Real Time Calculus* (RTC) for DNC applied to real time systems in multiprocessor implementations.
RTC assumes a dedicated design approach to avoid the appearance of cycles of timed dependence in order for the RTC calculations to be sequentiable solvable. To overcome that limitation [10] suggests the formulation of a fix point problem and an associated iterative method of solution. On a mild basis of assumptions the solution found is proven to be optimal in the sense, that it provides lowest upper bounds as well as highest lower bounds.
This paper first introduces the main concepts of DNC and proceeds with presentation of RTC and the fix point problem presented in [10]. After that, the graphical modelling and analysis tool CyNC (Cyclic Network Calculus) is presented, where flavours of the various tool components are given along with relevant details of the implementation in the MATLAB/SimuLink framework. Herafter issues in system stability are discussed and the ability of DNC to provide estimates of stability limits for cyclic systems followed by a presentation of on-going work to extend CyNC with support for synchronous communication and flow control. Finally conclusions and directions for future research as well as ideas for improvement of the CyNC tool are given.

2. DETERMINISTIC NETWORK CALCULUS AND REAL TIME CALCULUS

The basic modelling elements in DNC and RTC are network/processing elements connected by work/resource constraints. Network elements are presented to work-flow and resource constraints as inputs. Depending on the specific characteristic of a particular element, it responds, as output, transformed work/resource contstraints. Output work flows are transformed to reflect the timed characteristics of the work flow forwarded to the network element next in line, whereas resource flow is transformed to model unused resources available to lower priority processing as illustrated in figure (1).

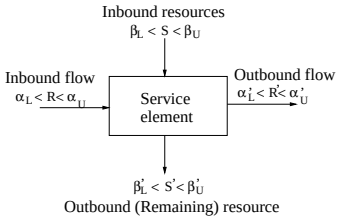

Figure 1: Propagating constraints through a service element.

2.1 Work Flow Constraints

Work flow constraints are in DNC given for an accumulated workflow $R(t)$ within $[0,t]$. We say that a flow complies to constraints α_L and α_U when for all $0 \leq s \leq t$

$$\alpha_L(t-s) \leq R(t) - R(s) \leq \alpha_U(t-s) \tag{1}$$

Upper constraints α_U are frequently given as staircase functions, i.e.

$$\alpha_U(t) = c\lceil \frac{t+\tau}{T} \rceil + B \text{ for } \ t > 0 \tag{2}$$

to model jittered periodic workflow or as affine functions used as an approximation from above, i.e.

$$\alpha_U(t) = B + rt \text{ for } \ t > 0 \tag{3}$$

or in turn combinations of linear and affine functions

$$\alpha_U(t) = \min\{B + rt, pt\} \text{ for } \ t > 0 \tag{4}$$

in which case an INTSERV T-Spec is obtained [4], where r acts as the sustainable rate, p the peak rate, and B the burst parameter. Generally we define sustainable flow rates by $\lim_{t\to\infty} \alpha_U(t)/t$.
Other examples exist, but the class of non-negative subadditive funtions passing through $(0,0)$, seems to be an adequate class from which to select upper flow constraints [4].
Lower constraints may also be given as staircase functions, i.e.

$$\alpha_L(t) = c\lfloor \frac{t-\tau}{T} \rfloor + B \tag{5}$$

to model jittered periodic workflow or as an approximation from below the so called *rate-delay* functions, i.e.

$$\alpha_L(t) = [rt - B]^+ = [r(t-D)]^+ \ \text{ for } \ rD = B \tag{6}$$

where D is the delay parameter, and $[]^+$ denotes positive parts.
Altogether, workflow constraints are given as pairs of upper and lower constraints (α_L, α_U) which are transformed by network elements according to various rules.

2.2 Resource constraints

Resource constraints are associated to network elements and are used to model the way such elements transform timed characteristics of inbound work flow to its outbound equivalent. Basically two different variants of service/resource constraints (curves) exist in literature: *abstract* and *strict* service curves, where the latter implies the former.
Abstract service curves are defined algebraically, referring to in- and out-bound accumulated work flow, R and R'. That is, a network element is said to conform to resource constraints β_L and β_U iff. [4]

$$R'(t) \geq (R \otimes \beta_L)(t) = \inf_{0\leq s\leq t}\{\beta_L(t-s) + R(s)\} \tag{7}$$
$$R'(t) \leq (R \otimes \beta_U)(t) = \inf_{0\leq s\leq t}\{\beta_U(t-s) + R(s)\}$$

where $\otimes$ denotes the so called *inf-plus* convolution. A less general but stronger definition, also given in [4] applies the concept of *strict service curves*. If s and $t \geq s$ belong to the same busy period and

$$\beta_L(t-s) \leq R'(t) - R'(s) \leq \beta_U(t-s) \tag{8}$$

we say that β_L and β_U are strict service curves. It is readily shown that if contraints β_L and β_U fulfill definition (8) then also (7) is fulfilled. Whereas definition (8) is the stronger and gives tighter results, (7) is more general, and is applicable to f.ex. propagation delays, which are not covered by (8). For strict service curves we generally define sustainable service rates by $\lim_{t\to\infty} \beta_L(t)/t$.

As examples of abstract and strict service curves linear function serve to provide lower and upper constraints of raw processing resources, whereas rate-delay and affine function are well suited as lower and upper constraints for the service received by lower priority tasks in a fixed priority scheduling hierarchy, whereas stair case functions are suitable for modelling systems applying time-cyclic resource sharing, e.g. TDMA schedulers.

2.3 Transformation of workload and resource constraints

When a workflow is processed in some processing element, it is in DNC/RTC assumed, that the element in return outputs a workflow which is then forwarded to the next processing element in line or crosses the boundary of the system under analysis. The basic assumption is that for each identifiable unit of inbound workflow an associated unit of outbound work is produced or in other words the processing element maintains the *work conserving* characteristics of an ideal network element. Attemps to relax that assumption are presented in [16] through the notion of *workload correlations*. In this paper we assume that any workload modelling beyond the basic assumption is achieved through proper scaling of workload constraints. Likewise is convergence of flows modelled by adding constraints for individual flows.
Without scaling [6] presents the following workload transformation rules for strict service curves

$$\alpha'_L(t) = \inf_{0\le u\le t}\{\alpha_L(u) + \beta_L(t-u)\} \tag{9}$$
$$\alpha'_U(t)$$
$$= \inf_{0\le u\le t}\{\sup_{v\ge 0}\{\alpha_U(u+v) - \beta_L(v)\} + \beta_U(t-u), \beta_U(t)\}$$

to be interpreted such that, when inbound workflow conforms to constraints α_L, α_U and the processing element provides strict service conforming to β_L, β_U, then outbound workflow conforms to constraints α'_L, α'_U.
Likewise are transformation rules for service contraints presented in [6], i.e.

$$\beta'_L(t) = \sup_{0\le u\le t}\{\beta_L(u) - \alpha_U(u)\} \tag{10}$$

$$\beta'_U(t) = \sup_{0\le u\le t}\{\beta_U(u) - \alpha_L(u)\} \tag{11}$$

so that a work flow, processed with a priority immediately below, receives service constrained by β'_L, β'_U. For abstract service curves less tight transformation rules are given in [4]. Altogether arranging inbound workflow constraints in vectors α_L and, α_U and outbound workflow constraints in vectors α'_L and α'_U respectively transformation rules (9) may be compactly formulated as

$$(\alpha'_L, \alpha'_U) = \Phi(\alpha_L, \alpha_U) \tag{12}$$

and similarly for resource constraints transformed through (11)

$$(\beta'_L, \beta'_U) = \Psi(\beta_L, \beta_U) \tag{13}$$

Workflow scaling and convergence as well as the introduction of external workflows to the system under analysis is comprised in the following affine operation

$$(\alpha_U, \alpha_L) = \mathbf{A}(\alpha'_U, \alpha'_L) + (Ua, La) \tag{14}$$

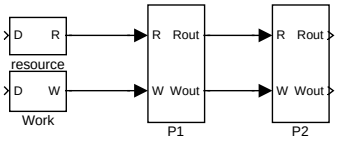

Figure 2: Example with two processing elements in a fixed priority hierarchy.

where $\mathbf{A}$ is a matrix of positive scaling factors and (Ua, La) comprise lower and upper external flow constraints.
When scaling is applied to workflow constraints no scaling is needed for the propagation of processing resources between elements. Therefore resource propagation from higher to lower priority tasks may take place through affine operation, i.e.

$$(\beta_U, \beta_L) = \mathbf{B}(\beta'_U, \beta'_L) + (Ub, Lb) \tag{15}$$

where $\mathbf{B}$ is an acyclic incidence matrix associated to the priority chain and (Ub, Lb) account for external resource constraints.

3. SYSTEM ANALYSIS

The primary objective for DNC system analysis is to establish lower- and upper- resource- and workflow-bounds α_U, α_L and β_U, β_L respectively, found as a joint solution to (12), (13), (14) and (15).
When the matrix sum $\mathbf{A} + \mathbf{B}$ is itself acyclic the solution can be found through a finite number of iterations of relations (12), (13), (14) and (15) corresponding to the application of network element transforms, scalings and convergence along the workflow and resource propagation graph. On the other hand when $\mathbf{A} + \mathbf{B}$ is cyclic, the solution may take an infinite number of iterations from an initial point, which has to be selected carefully as shown in [10].
The initial point is closely related to the concept of sustainable workflow- and resource-rates. When $\mathbf{A}$ is stable, i.e. its eigenvalues have less than unity magnitude, a finite solution exists for sustainable workflow rates. Issues in system stability are discussed further in section 7.

3.1 Example

Consider two network elements P_1 and P_2 sharing common processing resources, and where P_1 has priority over P_2 as shown in figure (2) (as the system diagram would appear in the CyNC tool to be presented later). Likewise an external flow is fed into P_1, which processes and forwards workflow to P_2. In this case

$$\mathbf{B} = \mathbf{A} = \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} \tag{16}$$

$$\tag{17}$$

and in turn

$$\mathbf{A} + \mathbf{B} = \begin{bmatrix} 0 & 0 \\ 2 & 0 \end{bmatrix}$$

which is obviously acyclic. However if workflow direction in figure (2) is reversed as shown in figure (3) we obtain

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \tag{18}$$

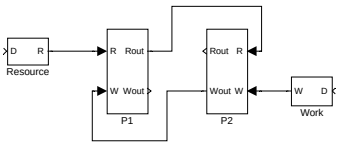

Figure 3: Example with two processing elements in a fixed priority hierarchy with reversed work flow direction.

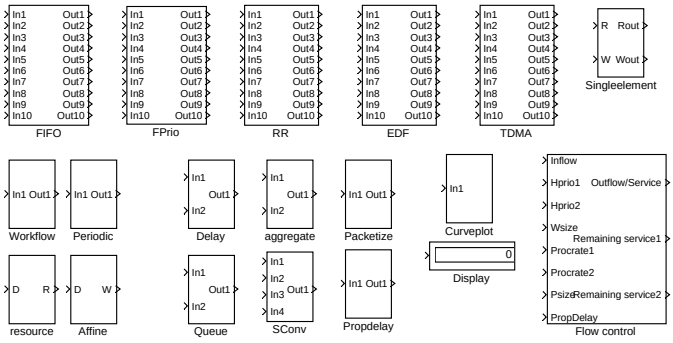

Figure 4: SimuLink library with DNC modelling objects currently available in CyNC.

and

$$\mathbf{A}+\mathbf{B} \quad = \quad \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{19}$$

which is definitely cyclic. Counter directional work- and resource-flows are probably the most frequent cause of cyclic dependencies, but also cyclic workflow propagation and flow control are possible causes.

3.2 CyNC

CyNC, which is short for *Cyclic Network Calculus*, refers to an integrated tool for modelling and timed analysis of distributed embedded systems. The tool is presently implemented as a library in the Matlab/SimuLink environment, i.e. a collection of modelling blocks each representing some DNC object. The present collection of DNC objects implemented so far are depicted in figure (4). The available modelling objects are conceptually divided into generators, network elements, auxiliary tools and display tools.

4. GENERATORS

Generators fall into two groups; workflow- and resource-generators. These are used to specify external workflow and resource constraints.

The basic workflow generator is the *Workflow* block, which receives numerical data as inputs and produces a pair of lower and upper workflow constraints as output. Numerical data format closely relates to the basic constraint representation in CyNC, which is presently time discrete, but extensible to other finite representations. Each constraint is represented by a vector a of non negative reals, so that $a = (a_0, a_2, .., a_n)$ represents a constraint $\alpha(t) = a_t$ for $t < n-1$ and $\alpha(t) = a_n \cdot (t-n-1) + a_{n-1}$ for $t \geq n-1$. Thus a_n generally comprises sustainable service rate. As an example $a = (5, 0.5)$ represents an affine function crossing

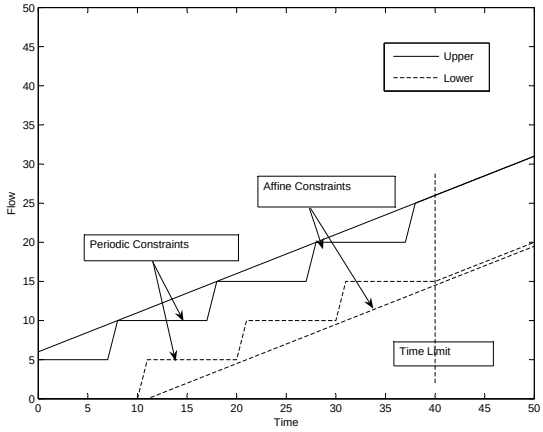

Figure 5: Periodic and corresponding affine approximations.

$(0, 5)$ and with a slope 0.5. A finite continuous time piecewise affine representation of constraints is included in plans for future work and would decrease complexities of representations as well as computations.

Two other workflow generators exist; affine and periodic. User input for affine work flow is given as (r, B) representing slope and burst parameter respectively for the upper constraint. For the lower constraint, user data (r, B) represent a corresponding rate-delay function $[rt - B]^+$.

Constraints representing a jittered periodic source are specified through:

(period=T, packet size=P, tolerance=τ and time limit=L), yielding lower and upper constraints

$$\begin{aligned} \alpha_L(t) &= P[\lfloor \frac{t-\tau}{T} \rfloor]^+ \text{ for } t-\tau < \lceil \frac{L}{T} \rceil T \\ &= P\lfloor \frac{L}{T} \rfloor \\ &+ P/T(t-\tau-\lceil \frac{L}{T} \rceil T) \text{ for } t-\tau \geq \lceil \frac{L}{T} \rceil T \end{aligned} \tag{20}$$

$$\begin{aligned} \alpha_U(t) &= P\lceil \frac{t+\tau}{T} \rceil \text{ for } t+\tau < \lceil \frac{L}{T} \rceil T \\ &= P\lceil \frac{L}{T} \rceil \\ &+ P/T(t+\tau-\lceil \frac{L}{T} \rceil T) \text{ for } t+\tau \geq \lceil \frac{L}{T} \rceil T \end{aligned} \tag{21}$$

This means that for time instants lower than L constraints are indeed staircase functions, whereas for larger times conservative affine approximations are used. This allows to map periodic constraints directly to the basic constraint representation in CyNC. L is presently specified by the user, which represents a possible weakness, since it seems hard to predict the impact of the affine approximation on the tightness of results. An example of periodic and corresponding affine constraints is shown graphically in figure (5).

5. NETWORK ELEMENTS

CyNC presently supports 6 different network element types; *single element* as well as a number of compound blocks FP, FIFO, RR, TDMA and EDF, where the single element acts

as described in figure (1) and equations (9-11). It provides 2 input ports; R for resource specification and W for workflow specification as well as 2 output ports for output workflow and remaining processing resources.

5.1 Fixed Priorities

The FP block comprises af ladder of 10 single elements in a common fixed priority hierarchy useful for modelling tasks sharing common processing resources under FP scheduling.

5.2 FIFO

For FIFO scheduling [4] defines for any $\theta > 0$

$$\begin{aligned} \beta^i_\theta(t) &= [\beta(t) - \alpha^i(t-\theta)]^+ \cdot I_{t>\theta} \\ \alpha'_i(t) &= \inf_{\theta>0} \sup_{v\geq 0}(\alpha_i(t+v) - \beta^i_\theta(v)) \end{aligned} \quad (22)$$

where α_i and α'_i are input and output upper flow constraints for subflow i, β is the overall service curve for the aggregate service and α^i constraints the aggregate flow not including i. To benefit from equation (22) minimization has to take place over all positive θ, which is generally unbounded. In CyNC an upper delay D is computed from aggregate flow and service constraints β and α. Thus δ_D is a service curve for all individual subflows leading to a modified version of (22)

$$\alpha'_i(t) = min\{\inf_{0<\theta<D} \sup_{v\geq 0}(\alpha_i(t+v) - \beta^i_\theta(v)), \alpha_i(t+D)\}$$

which improves (22) and gives a bounded minimization.

5.3 Round Robin

Round robin or token passing is supported by the RR block, which, based on inbound constraints, token passing overhead O and packet length P, solves a set of inequalitites as described in [13]. It is shown in [13], that a *critical instant* $t_0 = 0$ appears for some node i in the token ring, when all other nodes consume maximum resources $(\alpha^j)'_U(t-t_0)$ within each interval $[t_0, t]$. At t_0 the token is passed on to node $(i+1)_N$, where N denotes the number of nodes in the ring. Initiating a node i busy period at a critical instant yields the latest possible service intants $\{t_1, t_2, ..\}$ for packets transmitted from node i. Thus

$$\beta^i_L(t) = P\sum_{n=1}^{\infty}(t \geq t_n) \quad (23)$$

defines a lover service constraint for node i. Likewise does

$$\beta^i_U(t) = P\sum_{n=1}^{\infty}(t \geq (n-1)(P+NO)) \quad (24)$$

define upper service for node i, reflecting the situation where all other nodes are idle. From (9) we find an upper output flow constraint $(\alpha^j)'_U(t-t_0)$, which bounds the resource consumption from node i in intervals $[t, t_0]$. Altogether equations (23), (24) and (9) define a system of functional inequalities for which a unique least conservative solution is provided with the CyNC *RR*-block.

5.4 TDMA

The TDMA block provides for each inbound flow an unjittered periodic service, with a period NT and packet size P, where N denotes number of inbound flows.

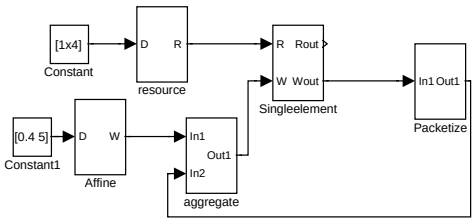

Figure 6: Example of the use of packetization and aggregation in CyNC.

5.5 EDF

For each inbound flow i to the EDF block a time invariant relative deadline d_i is specified. Initially feasibility of the EDF schedule is checked through

$$\sum_i \alpha^i_U(t-d_i) \leq \beta_L(t) \quad \forall t \geq 0 \quad (25)$$

which is known to be sufficient for no deadlines to be missed [17]. If (25) is fulfilled no workunit is delayed more than the relative deadline d_i associated the particular workflow. Hence a lower abstract service curve is defined from this upper delay bound, i.e.

$$\begin{aligned} \beta^i_L(t) = \delta_{d_i}(t) &= 0 \quad \text{for } t \leq d_i \\ &= \infty \quad \text{for } t > d_i \end{aligned}$$

6. AUXILIARY TOOLS

A number of auxiliary tools are provided as SimuLink blocks in CyNC: packetization, aggregation, convolution of service curves, maximum delay computation and maximum backlog computation.

6.1 Packetization and Aggregation

The packetization block is provided to allow easy and flexible modelling of packetizing effects as well as workload scaling. A common constant M is multiplied to lower and upper constraints after which another common constant A is subtracted and added respectively to lower and upper constraints. For affine constraints $\alpha_L(t) = [rt - B]^+$ and $\alpha_U(t) = rt + B$ resulting constraints after packetization are $\alpha'_L(t) = [M(rt-B) - A]^+$ and $\alpha'_U(t) = M(rt+B) + A$. For pure workflow scaling $A = 0$. However modelling a situation where smaller units of size P are collected/packetized into larger units of size P' may be modelled through $M = 1, A = P' - P$.

Aggregation adds a number of incoming constraints to model flow convergence. An example illustrating the use of packetization, where $M = 0.5, A = 0$, and aggregation is given in figure (6) resulting in a flow matrix $A = [0.5]$, which is indeed cyclic. In this case we may find the sustainable work flow rate α iteratively from an initial value 0, which is shown in figure (7) For an external flow rate 0.4 the output flow rate α solves $\alpha = 0.5\alpha + 0.4$, i.e. $\alpha = 0.8$, corresponding to the convergence point in figure (7).

6.2 Convolution of Service Curves

The CyNC block *SConv* provides *inf-plus* convolution of service curves. This facility is provided to deduce overall service curves for tandem configurations of service elements, i.e. where network elements $1, .., n$ are placed in line so that

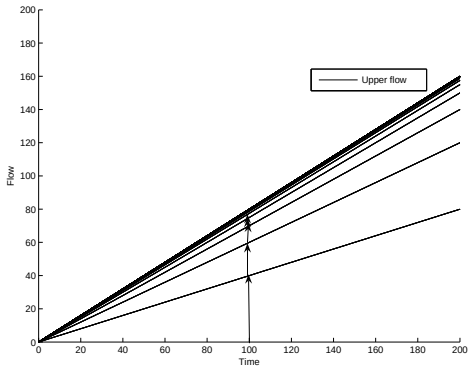

Figure 7: Convergence of sustainable flow rates.

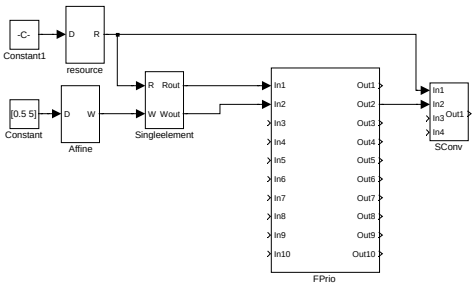

Figure 8: Convolution of service curves in tandem configuration.

the output of i is used as input for $i+1$. If lower and upper service curves β_L^i and β_U^i are associated to element i we have for the entire tandem [4]

$$\begin{aligned} \beta_L &= \beta_L^1 \otimes \beta_L^2 \otimes .. \otimes \beta_L^n \\ \beta_U &= \beta_U^1 \otimes \beta_U^2 \otimes .. \otimes \beta_U^n \end{aligned} \tag{26}$$

For single service elements service curves may be taken from the resource input, whereas for compound elements each workflow output includes information about the service constraints associated to that particular workflow. For the example shown in figure (8) a single element appear in a tandem with a compound FP element. Individual service curves are convolved to compute the overall service for the entire tandem.

6.3 Delay and Backlog Computations

For a pair of flow and service constraints α_U and β_L associated to a common service element [4] provides bounds for so called *virtual* delay D and backlog Q, i.e.

$$R(t) - R'(t) \leq Q = \sup_u \{\alpha_U(u) - \beta_L(u)\} \tag{27}$$

Similarly [4] provides tight bounds for the maximum delay D, i.e. $R'(t) \geq R(t-D)$, where

$$D = \sup_u \{\inf\{\tau | \alpha_U(u) \leq \beta_L(u+\tau)\}\} \tag{28}$$

CyNC provides, as auxiliary tools, blocks for delay and backlog computations according to (27) and (28). Both take corresponding resource and flow constraints as inputs and delivers delay and backlog bounds respectively. To view results delay and backlog blocks need to be connected to SimuLink display blocks, which consequently are included

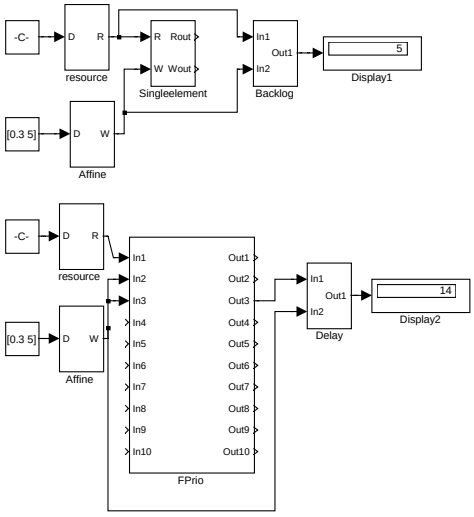

Figure 9: Delay and Backlog computations for single and compound service element.

in the CyNC toolbox. The use of delay and backlog blocks is shown in figure (9) also illustrating the use for both single and compound network elements.

7. ISSUES IN SYSTEM STABILITY

For networks of work conserving processing elements some notion of stability should exist, capturing the ability of a system to confine its operation to a limited/safe subset within prescribed boundaries or to return to safe operation after occasional vacations outside boundaries.

System state for processing networks is largely comprised in queueing backlogs, i.e. safe operation translates into the absence of buffer overflow and packet discard. DNC analysis of processing networks specifically provides answers to this problem. That is, for external stimuli complying to specified constraints, it is verified that internal backlogs stay below limited upper bound.

When upper service constraints are finite, a finite solution to internal workflows exist even for systems with insufficient sustainable service rates. This is caused by the fact that egress flows invariantly complies to upper service constraints (see (9)). However in order to maintain backlogs and queueing delays finite, sustainable services rates need to be sufficient. In that case sustainable inbound and egress workflow rates are identical, i.e. on the average inflows equal outflows. We shall in the following only consider solutions for sufficient sustainable service rates.

For acyclic networks, limited solutions to the DNC analysis exist whenever sustainable processing resources are sufficient in every element. More explicitely when lower average processing rates exceed upper average workflow rates. Thus when it is at all possible to maintain limited backlogs, a limited DNC solution exists. In this way DNC analysis exactly captures the rate stability limits for acyclic networks.

However for cyclic networks the existence of limited solutions is a more complicated. Consider the DNC/CyNC model in figure (10). We assume a processor of constant rate= 2, shared non-preemptively between three tasks processing the same workflow in priority order $3, 1, 2$. That is, the external inbound workflow is initially processed with low

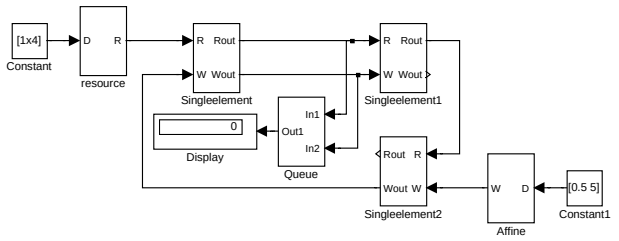

Figure 10: Cyclic DNC model exhibiting counter-priority work flow.

priority and in turn with priorities 1 and 2 respectively. External workflow is periodic with period 10 and a constant packet/unit size 5. Thus an average workflow rate of 0.5 is submitted to the system and since each workflow unit is processed in three consecutive stages each consuming 5 processing time units, the overall processing time consumption per time unit adds up to $1.5 < 2$.
Inspecting the detailled operation of the system reveals that the n th. workunit is processed nonpreempted in stages $3, 1$ and 2 respectively during $t_n + [0, 7.5]$, where t_n denotes the arrival time of the n th. unit. This gives and idle time 2.5 until the arrival of the next unit at $t_n + 10$. The existence of positive idle time indicates robustness to occasional processor vacations, where workunits are backlogged. Indeed detailled analysis reveals that the system in finite time returns to a state with only one unit in the system at a time, after backlog of arbitrary size.

7.1 Stability of Single Processor Systems

Consider a priority hierachy of tasks sharing a single work conserving processor of speed ρ. Assume from the beginning ($t = 0$) of each system busy period that the work forwarded by some task τ_j to another τ_i is bounded by $a_{ij}R_j(t)$, where $R_j(t)$ denotes the accumulated work fed to τ_j and a_{ij} is a non-negative real. The system example above falls within this category. Letting $R = [R_1, .., R_N]^T$, we have

$$R(t) = AR(t) + E(t) \tag{29}$$

where $E(t)$ is an $N \times 1$ vector denoting the accumulated external work fed to tasks $\tau_1, .., \tau_N$, and A is the matrix $\{a_{ij}\}$. A is assumed stable, i.e. to have all eigenvalues of modulus strictly less than one. Thus $I - A$ has a unique positive inverse and

$$R(t) = [I - A]^{-1}E(t) \tag{30}$$

yields a positive solution for R. Assuming an affine upper bound for E, i.e.

$$E(t) \leq E \cdot t + B \tag{31}$$

an equivalent bound for R may be found

$$R(t) \leq [I - A]^{-1}E \cdot t + [I - A]^{-1}B \tag{32}$$

If $\rho \geq |[I - A]^{-1}E|$, where $|\cdot|$ denotes induced 1-norm, we have an upper backlog bound $|[I - A]^{-1}B|$ and a bounded busy period duration of $|[I - A]^{-1}B|/(\rho - |[I - A]^{-1}E|)$.
Thus any such system, where the produced work rate $|[I - A]^{-1}E|$ does not exceed the processor speed, is stable.

For the above example we have

$$A = \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \tag{33}$$

whereas

$$\begin{aligned} E &= [0\ 0\ 0.5]^T \\ B &= [0\ 0\ 5]^T \\ \rho &= 2 \end{aligned} \tag{34}$$

Thus $|[I - A]^{-1}E| = 1.5 < \rho$ and the system is stable. Thus the entire system backlog is bounded by $|[I - A]^{-1}B| = 15$, and any busy period is shorter than $15/(2 - 1.5) = 30$.
The bounds so obtained are however overall system bounds and are typically not sufficiently precise for separate tasks, as the above example shows.
Results for uni processor systems do not extend generally to multiprocessor systems. An example of stability results for multiprocessor systems are given in [12], where it is shown that any work conserving schedule stabilizes a ring of separate network elements, when spatial reuse is assumed.

7.2 Affine DNC

A more detailled parametric stablity result may be obtained from approximating all work flow constraints by affine upper bounds and all service contraints by lower rate-delay bounds as presented in [4]. In this case a single task may be considered mapping inbound rates, burst parameters and delays into their outbound counterparts. In this case work and service rates map as follows:

$$\begin{aligned} r' &= r \\ \rho' &= \rho - r \end{aligned} \tag{35}$$

whereas burst and delay parameters follow

$$\begin{aligned} B' &= B + rD \\ D' &= 1/(\rho - r)B + \rho/(\rho - r)D \end{aligned} \tag{36}$$

When equations (35) and (36) are propagated to an entire system of network elements, an overall parametric stability criterion is obtained. That is, if there is a unique positive solution, the system is stable and finite burst, delay and backlogs are guaranteed.
Applying affine DNC equations to the above example for processor speeds $\rho \leq 2$ however yields infinite/negative solutions, which indicates a methodological problem. That is, the inability of the method to capture exact stability bounds. The same problem is expected to carry over to the refined cyclic calculus serving as the basis for CyNC. Thus an emergent need for tighter bounds, improved assistance for identifying critical loops or upper bounds for over-approximations is identified.
Indeed CyNC guarantees stability for the above example system only for processor speeds $\rho > 2$. Conversely affine DNC provides a sufficient criterion for stability and in turn the existence of a finite solution in CyNC, since affine DNC applies and provides conservative approximations to all quantities used in CyNC.

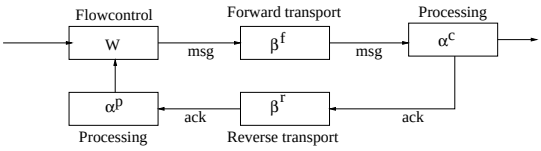

Figure 11: Abstract service model for sliding windows flow control.

8. SYNCHRONOUS COMMUNICATION

One may view the above example with counter priority work flow as an example of flow control, i.e. resource is passed to the original transmitter only when the transmitted work packet has received service in the entire chain, ensuring only one packet resident in the system at a time. Flow control as well as bounded buffer action with *blocking* writes are both examples of synchronous communication, which has not received much attention in DNC so far.

In [4] a *sliding window* flow control protocol is analyzed and lower service contraints are deduced. Consider the situation depicted in figure (11) where forward flow is offered service constraints β^f, modeling message transport from transmitter to receiver. The flow of acknowledge messages is offered processing service at the receiver side α^C, reverse transport service β_L^b and processing service α^P before acknowledging to the flow control protocol at the transmitter side. Altogether lower and upper open-loop constraints are obtained

$$\begin{aligned} \beta_L &= \beta_L^f \otimes \alpha_L^C \otimes \beta_L^b \otimes \alpha_L^P \\ \beta_U &= \beta_U^f \otimes \alpha_U^C \otimes \beta_U^b \otimes \alpha_U^P \end{aligned} \tag{37}$$

From open-loop constraints β_L, β_U lower and upper closed-loop constraints $\beta_L^{cl}, \beta_U^{cl}$ are obtained in [11] and [4] respectively:

$$\begin{aligned} \beta_L^{cl} &= \overline{\beta_L + W} \otimes \beta_L^f \\ \beta_U^{cl} &= \overline{\beta_U + W} \otimes \beta_U^f \end{aligned} \tag{38}$$

where W is window size and $\bar{\cdot}$ denotes *subadditive closure* (see [4] for details).

Service constraints $\beta_L^{cl}, \beta_U^{cl}$ seem appropriate when open-loop service is dominated by propagation delay. However when open loop service is dominated by processing/scheduling delay it is shown in [11] that the lower constraint β_L^{cl} is arbitrarly conservative even w.r.t its sustainable rate, i.e. $\lim_{t\to\infty} \beta_L^{cl}(t)/t$. An improved sustainable rate r is obtained in [11] from detailled analysis of a flow control loop including both propagation- and processing- delays

$$r = \frac{W(1 - \alpha_C - \alpha_P)}{2D + WK(1/r_C + 1/r_P)} \tag{39}$$

where K is packet processing time on a processor of normalized speed one, r_C, r_P are processor speed in transmitter and receiver ends respectively, α_C, α_P are normalized sustainable rates of higher priority activity in transmitter and receiver and D is one way propagation delay. In (39) r is given in $1/s$ units, i.e. packet rate. For the same case (38) gives approximately

$$\frac{W}{2D + C_C + C_P} \tag{40}$$

for high values of r_C, r_P and $\alpha_C, \alpha_P << 1$. C_C, C_P denote burst parameters associated to affine higher priority activity constraints in transmitter and receiver.

Whereas (39) is insensitive to burstiness of higher priority scheduling delay in transmitter and reciever (38) yields positive sustainable closed-loop rates for all $\alpha_C, \alpha_P < 1$. Thus the two service constraints should be combined to obtain an overall improved estimate.

8.1 Flow control in CyNC

Incorporating flowcontrol and synchronous communication into CyNC is a current activity. The results from [4] and [11] are comprised in a single CyNC block named *Flow control*, which is fed with information about input flow constraints, processing rates, higher priority flow constraints, propagation delay, window size and packet size. In return closed loop service constraints and output flow constraints are produced by combining the above results to give the tightest overall bound.

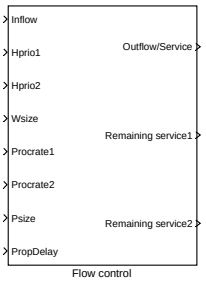

Figure 12: Flow control block in CyNC.

9. CYNC/DNC IN A DEVELOPMENT CHAIN

Professional development chains for embedded systems come in a multitude of variations more or less inspired by one or more design paradigms from litterature, e.g. SASD/RT [18], CODARTS/COMET [19], ROOM [20] e.t.c. In order for a verification methodology like DNC/CyNC to receive recognition in professional development environments, it is of major importance that a seamless integration with defacto standards for methodologies, tools and notation is ensured.

9.1 CODARTS and DNC

One of the earliest methods to accomodate for the real time aspects of embedded systems development is CODARTS (Concurrent Object Based Analysis and Design of Real Time Systems), where analysis and design phases end up producing a Task Architecture Diagram (TAD) as depicted in figure (13) A TAD includes task symbols (parallelograms) to support concurrency. A task encapsulates an asynchronously exectuting entity and in this respect bears much similarity to the network element of DNC. Communication between tasks take place either through synchronous (tight coupling) or asynchronous (loose coupling) communication channels of which the latter allows for prioritization. DNC supports directly asynchronous communication as a part of its foundation, whereas the attempts to support synchronous communication made in [4] and the further refinements in [11] are currently being implemented in CyNC. As such, a TAD is mapped seamlessly to a DNC/CyNC diagram as illustrated

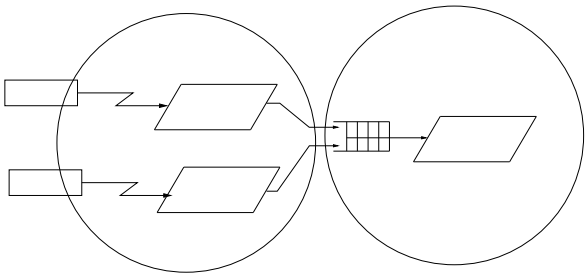

Figure 13: Task Architecture Diagram

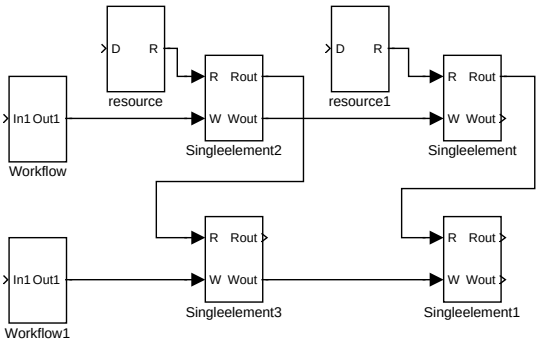

Figure 14: Task Architecture mapped to DNC/CyNC specfication

in figure (14).

Early subsystem decomposition follows the deployment structure of distributed systems, i.e. independently executing processors are entirely enclosed in unique subsystems associated to a specific TAD. This allows for precise mapping to external processing resources in CyNC, since each subsystem enclosing an independent processor defines a unique priority chain in DNC. No means for specifying the quantitative characteristics of communication media seem to be offered in CODARTS and therefore such specifications need to be added subsequently in the DNC domain.

CODARTS advocates for the use of textual Task Behavioural Specifications (TBS) for task interfaces, structure, timing characteristics, relative priority and sequencing logic. Timing characteristics include periods for periodic tasks and various timing statistics for aperiodic tasks as well as execution time estimates (WCET). Thus TBS timing characteristics map to external periodic flow constraints as presented in the CyNC tool above, whereas relative priority may govern the definition of the DNC priority hierachy.

TBS are written informally and thus allowing for any desired extensions such as relative deadlines or extended timing characteristics such as DNC flow constraints. However the lack of formality prevents TBS in its current form from taking part in an automated verification process based on scheduling theory or in turn DNC.

9.2 UML profiles

Formal approaches to quantitative specifications of embedded systems exist, based on Unified Modeling Language (UML,[21]) such as the UML profiles for *Schedulability, Performance and Time* (SPT) [14] and *Modelling and Analysis of Real-Time and Embedded Systems* (MARTE), [15]. All of the mentioned profiles offer structured/formal frameworks

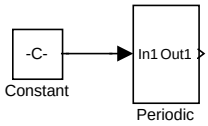

Figure 15: CyNC model for specifying periodic flow constraint.

for specifying quantitative properties. SPT in particular models resources as servers with *offered QoS* and processing tasks as clients with specified *required QoS*. This allows for the annotation of sequence diagrams with e.g. time information such as periods for periodic tasks, execution times and relative deadlines for clients as well as speed and delay characterisics for servers. While the above mentioned UML profiles offer a broader scope of quantitative specifiers such as memory and power consumption, it must be noted that none of them encompass the broader notions of timed behaviour covered by DNC.

9.3 Interfacing to CyNC

Supporting seamless automated transfer of design specifications to a verification tool like CyNC requires well defined source and target formats, i.e. from UML-SPT to some textual/linguistic specification format for DNC models. In [22] a specification language for network models is presented along with a DNC inference engine in the TRAFFIC (Typed Representation and Analysis of Flows For Interoperability Checks) framework. The popularity of eXtensible Markup Language (XML), on the other hand, indeed advocates this as a basis for specifying DNC models as in UPPAAL [8]. However an immediate linguistic interface is offered by the MATLAB/SimuLink model specification language. This is illustrated by the simplistic CyNC model in figure (15) showing the specification of an external periodic flow constraint and the corresponding subset of SimuLink model specification below

```
System {
    Name     "Simplistic"
    Block {
      BlockType       Constant
      Name       "Constant"
      Value       "[5 10 2 100]"
    }
    Block {
      Name       "Periodic"
      Ports       [1, 1]
      SourceBlock       "CyNC/Periodic"
    }
    Line {
      SrcBlock       "Constant"
      SrcPort       1
      DstBlock       "Periodic"
      DstPort       1
    }
  }
```

: SimuLink/CyNC model specification

10. CONCLUSIONS

A novel tool - CyNC for real time performance analysis of embedded and networked systems is presented. The tool is

based on the Deterministic Network Calculus (DNC) theoretical framework and is implemented as a library/toolbox in MATLAB/SimuLink. Following a brief description of the various elements of DNC the corresponding CyNC components are presented along with guidelines and examples for their use. CyNC includes modeling blocks for DNC characterization of external flow generators and processing resources as well as a variety of service elements like FIFO, Fixed Priorities, Round Robin, TDMA etc. as well as a number of auxiliary tools for computation of queueing delays and backlogs, packetization and convolution of service constraints.

CyNC is distinguished by its ability to provide solutions for systems comprising cyclic dependency, which appears in case of feedback flow, opposite flow and priority directions as well as flow control. The paper provides an example for opposite flow and priority directions revealing an inherent problem with estimation of stability bounds. For flow control, CyNC provides a modeling block combining two different analytical results to provide an overall improved bound.

Guidelines and examples are given to bring about an idea of how to use the tool for analysis of embedded/distributed real time systems. A discussion is provided concerning how to integrate CyNC in a professional development chain for embedded systems. Seamless integration is mainly ensured by interfacing properly to existing design documentation such as Task Architecure Diagrams, Sequence Diagrams and Deployment diagrams.

Directions for future research include an implementation of CyNC outside the MATLAB/SimuLink environment as well as interfacing CyNC to design documentation from existing design tools. Stability analysis and improved stability bounds point out an equally important direction for future research.

11. REFERENCES

[1] C. L. Liu and J. W. Layland, Scheduling Algorithms for Multiprogramming in a Hard Real Time Environment., *JACM*, pp.46-61,1973.

[2] J. P. Lehoczky, L. Sha and Y. Ding, *The Rate Monotonic Scheduling Algorithm: Exact Characterization and Average Case Behaviour*, Proceedings of the 10th IEEE Real-Time Systems Symposium, pp. 166-171, 1989.

[3] M. Bertogna, Michele Cirinei and Guiseppe Lipari, *Improved Schedulability Analysis of EDF on Multiprocessor Platforms* Proceedings of the 17th IEEE Euromicro Conference on Real-Time Systems, pp. 209-218, 2005, ISBN 0-7695-2400-1.

[4] J. L. Boudec an P. Thiran *Network Calculus - A Theory of Deterministic Queuing Systems for the Internet.*,Springer-Verlag, LNCS 2050, 2004.

[5] R. L. Cruz, *A Calculus for Network Delay Part I: Network Elements in Isolation*, IEEE trans. on Information, 1991.

[6] Samarjit Chakraborty, Simon Kunzli, Lothar Thiele, Andreas Herkersdorf, Patricia Sagmeister, *Performance Evaluation of Network Processor Architectures: Combining Simulation with Analytical Estimation*, Computer Networks, Vol. 41, No. 5, pages 641–665, 2003.

[7] R. Alur and D.L. Dill, *A Theory of Timed Automata*, Theoretical Computer Science, pp. 183–235, 1994.

[8] K. G. Larsen, W. P. Pettersson and W. Li, *UPPAAL in a Nutshell*, Int. Journal on Software Tools for Technology Transfer, Vol. 1, 1997.

[9] B. Berthomieu and M. Diaz, *Modeling and Verification of Time Dependent Systems Using Time Petri Nets.*, IEEE Transactions on Software Engineering, 17(3):259-273, 1991.

[10] H. Schioler, J. Dalsgaard, K. G. Larsen and J. Jessen, *CyNC - a method for Real Time Analysis of Systems with Cyclic Data Flows*, 13 th. RTS Conference on Embedded Systems, Paris, 5-7 april, 2005.

[11] H. Schioler, J. Jessen, Jens Dalsgaard Nielsen, K. G. Larsen, *Introducing Synchronisation in Deterministic Network Models*, Computer Applications in Industry and Engineering (CAINE-2006), 19th Int'l. Conference. ISCA, 2006.

[12] L. Tassiulas and L. Georgiadis, *Any work-conserving policy stabilizers the ring with spatial re-use*, IEEE/SCM Transactions on Networking, Vol. 4, pp. 205-208, 1996.

[13] H. Schioler, J. Dalsgaard, N. Jorgensen, N. Nielsen, *Traffic Analysis for Real-Time Communication Networks on board Ships*, Control Applications in Marine Systems (CAMS98), IFAC, 1998.

[14] Object Management Group, *UML Profile for Schedulability, Performance, and Time, Version 1.1.*, 2005. OMG document: formal/05-01-02.

[15] Object Management Group, *UML Profile for Modeling and Analysis of Real-Time and Embedded systems (MARTE)*, RFP. 2005. OMG document: realtime/05-02-06.

[16] E. Wandeler and Lothar Thiele, *Workload correlations in multi-processor hard real-time systems*, Journal of Computer and System Sciences, Vol. 73, No. 2, pages 207-224, March, 2007.

[17] S. K. Baruah, R. Howell and L.Rosier, *Algorithms and complexity concerning the preemptive scheduling of periodic real-time tasks on one processor*, Real-Time Systems,2:301-324, 1990.

[18] P. Ward and S. Mellor, *Structured Development for Real-Time Systems, Volume 1-3, Yourdon Press, New York, 1985-86.*

[19] H. Gomaa, Software design methods for concurrent and realtime systems*, Menlo Park, CA: Addison-Wesley, 1993.*

[20] B. Selic, G. Gullekson and P. T.Ward, Real-time object-oriented modeling*, New York, NY: John Wiley & Sons, 1994.*

[21] G. Booch, J. Rumbaugh and I Jacobson, The Unified Modeling Language User Guide*, Addison Wesley, 1998, ISBN 0-201-57168-4.*

[22] A. Bestavros, A. D. Bradley, A. J. Kfoury, and I. Matta, Safe compositional specification of networking systems.*,ACM Computer Communications Review, 34(3):21.33, 2004.*