A UML Framework for the Dimensioning and Formal Verification of Embedded Systems

Ludovic Apvrille
Institut Telecom, Telecom ParisTech, CNRS LTCI
2229, routes des Crêtes, B.P. 193
F-06904 Sophia-antipolis Cedex
Email: ludovic.apvrille@telecom-paristech.fr

Pierre de Saqui-Sannes, Ahlem Mifdaoui
CNRS ; LAAS ; 7 avenue du colonel Roche,
F-31077 Toulouse, France
Université de Toulouse ; UPS, INSA, INP, ISAE ; LAAS ;
F-31077 Toulouse, France
Email: {pdss, ahlem.mifdaoui}@isae.fr

***Abstract*— The TURTLE UML profile and the open source toolkit TTool define a formal modeling and verification framework for communicating embedded systems design. The paper extends TURTLE and TTool with network calculus techniques. Dimensioning diagrams enable system dimensioning prior to usual object-oriented design and facilitates formal verification of design models. The paper uses a video-conferencing system to illustrate the proposed approach.**

I. Introduction

The unified Modeling Language, or UML for short [1], defines a set of diagrams that particularly support object-oriented design of systems and objects deployment over implementation nodes. The TURTLE profile [2] customizes the OMG-based UML and defines a method for distributed real-time system design [3]. So far, the TURTLE method addressed deployment issues after the design step was complete. That approach was common to a broad variety of modeling languages applied in the distributed system area. The paper transforms the TURTLE method to reduce design and formal verification efforts. Dimensioning issues are addressed before object-oriented design starts. UML-TURTLE deployment diagrams are revisited into dimensioning diagrams. Their semantics relies on Network Calculus theory [4]. We use dimensioning diagrams made up of traffic sources and switches to compute worst-case latencies that may be reused at design step (see Figure 1 and example in Section IV).

II. Network Calculus

The Network Calculus (NC) theory introduced by Cruz [5] and improved by Le Boudec [4] enables worst-case dimensioning of critical systems. NC particularly applies when traffic source parameters (e.g. periodicity, bandwidth) and network equipment parameters (e.g. queuing approach, interface bandwidth) are known: this theory is meant to derive deterministic maximal waiting bounds for each traffic source. NC relies on *traffic irregularity* whereas queuing theory is based on *stochastic processes*.

Maximal transmission delays of traffic sources and queue sizes of network switches depend on traffic arrival policy described with an arrival curve named α [4], and the availability of nodes on traffic path described with an availability curve named β [4]. Traffic delays are bounded by the horizontal distance

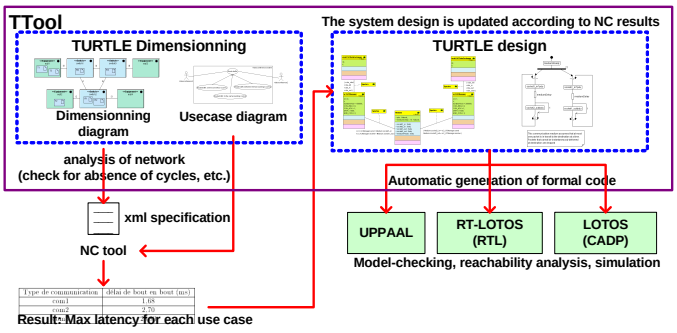

Fig. 1. NC-based methodology

between α and β, whereas queuing bounds are given with the vertical distance. The computation of those two bounds is simplified when both curves are affine.

III. Dimensioning Distributed Systems

To compute the maximum latency encountered by traffics, the following information is requested: equipments connected to the network (mobiles phones, web servers, etc.), traffics emitted on network equipments (video stream, http stream, etc.), switches of the network (Ethernet switch, IP router, etc.), physical interconnections between switches, and at last, use cases identifying all possible scenarios of destinations for traffic sources.

The above information is contained in extended UML deployment diagrams that we name *dimensioning diagrams*:

- Equipments are modeled with UML nodes and are stereotyped as $<< Equipment >>$.
- Traffics are modeled as $<< Traffic >>$ artifacts deployed over $<< Equipment >>$. A traffic is further characterized with a type (periodic, aperiodic), a deadline, its minimal and maximal packet size, and its class of traffic (i.e. its priority in switches).
- Routers and switches are modeled with UML nodes stereotyped $<< Switch >>$. They are further characterized with a scheduling policy (Static Priority, First Come First Served), with their capacity in Mbs, and with a set of entry / exit interfaces.
- Routing information is modeled using $<< Routing >>$

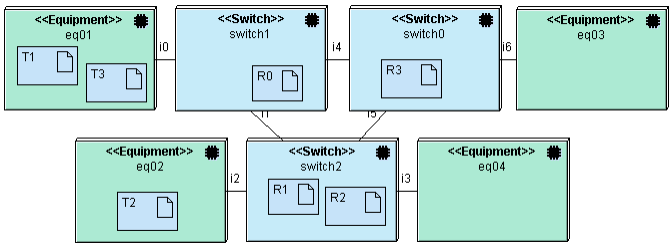

Fig. 2. Example of a dimensioning diagram: 4 equipments, 3 switches and 3 traffic sources

artifacts meant to be deployed over $<< Switch >>$ nodes. A route is a 3-uple (*entry interface*, *traffic*, *exit interface*).

- At last, links between nodes make it possible to model exit and entry interfaces of equipments and switches. At equipment level, interfaces may be characterized with a multiplicity parameters so as to model large numbers of similar equipments in a system (e.g. thousands of mobile phones).

The dimensioning diagram described above has been implemented in TTool [6] (see Figure 2). From a dimensioning diagram, TTool generates a list of all possible traffic paths that constitute inputs for applying network calculus techniques.

IV. Case Study

We consider a student residence composed of two four-floor buildings with 28 rooms on each floor. The network topology is as follows: two full-duplex Ethernet switches are installed on each building floor to connect rooms to the network. Also, a 100 Mbs connects the two switches (each switch has sixteen 100Mbs interfaces). The eight building switches are themselves connected to a building switch with eight 100 Mbs ports and one Gbs port used to interconnect the two buildings. Each video-conferencing application running in the system generates a full-duplex 120 kbs audio-video stream (Packets of 1500 Bytes are periodically sent).

Three use-cases are evaluated:

1) A communication between two students located on the same floor (uc1).
2) A communication between two students located in the same building (uc2).
3) A communication between two students located in two different buildings (uc3).

We first modeled the system described above with a TURTLE dimensioning diagrams. Then we calculated the maximal bounds for each use case. For uc1 (resp. uc2, uc3), the maximal end-to-end delay is 1.68 ms (resp. 2.70 ms, 28.80 ms).

From those results, we were able to model the video-conference system as a set of three classes: two video-conference classes plus a communication medium modeling the whole network architecture. That medium is simply modeled as an intermediate system that adds a 28.80 ms delay to all packets transmitted between the two video-conferencing applications. The extended TURTLE model enabled proving important safety properties. In particular, it is impossible to exchange data before a connection is properly set up. Also, formal proof indicates that the system does not lose any packet.

V. Related Work

The idea of linking a modeling language (UML), a toolkit (TTool) and a network calculus-based analysis is a new contribution. Indeed, [7] and [8] uses UML diagrams for documentation purpose only. [8] further integrates network calculus to Matlab/Simulink for schedulability analysis purpose, but does not address distributed systems nor formal verification. SCADE [9] introduces the computation of bounds for establishing worst-case scenarios but SCADE is not focused on distributed systems issues.

VI. Conclusions and Future Work

The paper proposes an innovative approach for computing worst-case network transmission delays, the latter being reused at design step for easing the formal verification process. Network-calculus techniques, associated with UML dimensioning diagrams, and included in TURTLE [2], are used to compute those delays. To our knowledge, this is the first time a UML/SysML profile relies on Network Calculus.

Today, TTool [6] can edit those diagrams, and can generate an *xml* representation of the network, that may be taken as input by a network calculus toolkit developed at ISAE.

Future work shall now focus on the reuse of those techniques for schedulability analysis purpose.

References

[1] O. M. Group, "UML 2.0 Superstructure Specification," in *http://www.omg.org/docs/ptc/03-08-02.pdf*, Geneva, 2003.

[2] L. Apvrille, C. Lohr, J.-P. Courtiat, and P. de Saqui-Sannes, "TURTLE: A Real-Time UML Profile Supported by a Formal Validation Toolkit," in *IEEE transactions on Software Engineering*, vol. 30, no. 7, July 2004, pp. 473–487.

[3] L. Apvrille, P. de Saqui-Sannes, R. Pacalet, and A. Apvrille, "Un environnement UML pour la conception de systmes distribus ," *Annales des Tlcommunications*, vol. 61:11/12, pp. 1347–1368, Novembre 2006.

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

[5] R.Cruz, "A calculus for network delay, part 1 : network elements in isolation," *IEEE transactions on information theory*, vol. 37, January 1991.

[6] LabSoc, "TTool: The TURTLE Toolkit," in *http://labsoc.comelec.enst.fr/turtle/ttool.html*.

[7] E. Wandeler, L. Thiele, M. Verhoef, and P. Lieverse, "System architecture evaluation using modular performance analysis: a case study," *Int. J. Softw. Tools Technol. Transf.*, vol. 8, no. 6, pp. 649–667, 2006.

[8] H. Schioler, H. P. Schwefel, and M. B. Hansen, "Cync: a matlab/simulink toolbox for network calculus," in *ValueTools '07: Proceedings of the 2nd international conference on Performance evaluation methodologies and tools*. ICST, Brussels, Belgium, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007, pp. 1–10.

[9] T. le Sergent, R. Heckmann, and D. Kastner, "Methodology and benefist of Timing Verification for Safety-Critical Embedded Software," in *DO-178 White paper*, http://www.esterel-technologies.com/DO-178B/request/whitepaper, 2008.