A SysML Method with Network Dimensioning

Hugo Daigmorte
DISC department
ISAE-SUPAERO, Université de Toulouse
Toulouse, France
Hugo.Daigmorte@isae.fr

Pierre de Saqui-Sannes
DISC department
ISAE-SUPAERO, Université de Toulouse
Toulouse, France
Pierre.de-saqui-sannes@isae.fr

Rob Vingerhoeds
DISC department
ISAE-SUPAERO, Université de Toulouse
Toulouse, France
Rob.Vingerhoeds@isae.fr

***Abstract*—Acceptance of the Systems Modeling Language (SysML) among system engineers heavily depends on the method and tool associated with the language. This particularly applies to a family of systems where increasing data exchanges between equipments create high requirements for the networks. The paper therefore revisits the method associated with the free SysML tool TTool in order to take network dimensioning into account in the early steps of the life cycle of distributed systems. TTool is interfaced with WoPANets, a tool based on network calculus theory. An AFDX network serves as case study.**

***Index Terms*—System Modeling, SysML, Dimensioning.**

I. Introduction

The Object Management Group (OMG [1] and the International Council for Systems Engineering (INCOSE [2]) have jointly defined the Systems Modeling Language (SysML [1]) in the form of a notation, the syntax of which is standardized when the way of using it is not. Being a profile of the Unified Modeling Language (UML [3]), SysML accepts extension mechanisms such as stereotypes, and therefore customization for one specific application domain. This leaves some flexibility for developing a system modeling tool with an enhanced SysML syntax and a method accepted by practitioners of the application domain.

Over the past few years, proprietary and open-source SysML tools have been developed to edit diagrams and analyze these diagrams using simulation and formal verification techniques. This paper adds network dimensioning capabilities to TTool [4], a free SysML tool that enhances the expression power of SysML, gives the latter a formal semantics, and supports a method covering the requirement capture, analysis, and design steps in the life cycle of real-time systems. The design step is assisted by a simulator, a model checker and other formal verification tools that enable checking of SysML models against design errors.

In [5] de Saqui-Sannes *et al.* discussed the use of TTool and formal verification for checking a networked system against design errors. To demonstrate a protocol renders its expected service, the authors of [5] put the focus on the temporal ordering of messages in the network. Quantitative characteristics, such as throughput, were left for further study.

The current paper proposes to make these quantitative characteristics an important concern as early as possible in the design trajectory of networked systems. The paper therefore proposes to complement the requirement capture step of TTool's method with a dimensioning step that is focused on the network's links. The paper's contributions therefore address SysML, TTool and the method. First, SysML context diagrams are extended to depict a network architecture with parameters such as period and jitter. Second, TTool is interfaced with WoPANets [6], a tool for network dimensioning. The method associated with TTool is revisited accordingly.

The paper is organized as follows. Section II presents SysML and TTool. Section III presents WoPANets. Section IV-A integrates the use of a network dimensioning tool into the method associated with SysML and TTool. It extends SysML with context diagrams and explains how TTool and WoPANets are interfaced. Section V discusses a case study: an AFDX network. Section VI surveys related work. Section VII concludes the paper.

II. SysML and TTool

A. Method and Diagrams

The SysML standard [1] defines a diagrammatic notation that may be tuned to target an application domain and to cover one or several steps in a method accepted by practitioners of the domain. The SysML TTool named ”TTool” thus targets real-time systems.

The method associated with SysML and TTool covers the requirements capture, analysis and design steps of the traditional V life cycle.

- *Requirement capture* uses requirement diagrams to define stakeholder, user, and system requirements. Modeling Assumptions Diagrams list simplifications and other assumptions made at the time of creating the model.
- *Analysis* is use-case driven. Use-cases identify the main functions and services to be offered by the system. Sequence and activity diagrams document the uses cases in the form of scenarios and flow-charts, respectively. A preliminary architecture can be depicted by a context diagram.
- The *design* step architects the system in the form of a block diagram, and defines the inner workings of the blocks using state machine diagrams.

B. Context diagrams

Previous section sketched a method and identified the SysML diagrams supported by TTool. To extend the method with a dimensioning step (cf. Figure 1 and Section IV-A), this

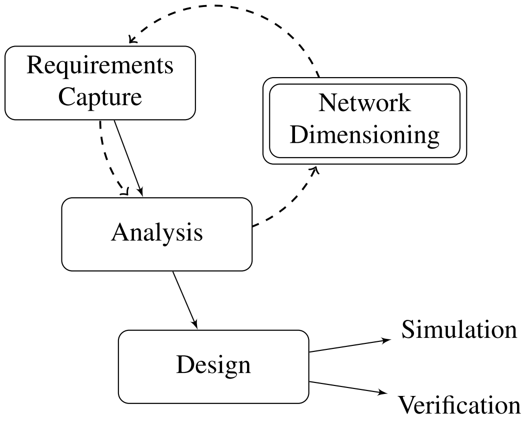


Fig. 1. Methodology

paper essentially addresses the context diagrams elaborated during the analysis phase.

A context diagram uses the syntax of SysML block diagrams to depict a preliminary architecture of the system. The architecture is preliminary because the blocks are not detailed in terms of attributes and methods. A context diagram thus depicts a tree structure linking blocks by containment relations (depicted by the SysML "black diamond" arrow). The parameters needed to achieve network dimensioning are included into comments, themselves linked to the relevant blocks depicting an equipment. Annotations are written in the language introduced by Section IV-B.

III. WoPANets and Network Calculus

A. WoPANets

WoPANets (Worst-case Performance Analysis of Embedded Networks [6]) answers an important critical embedded systems design challenge: verification of temporal and functional constraints in the worst-case. Besides ensuring the proper functioning of the system in its environment this allows to guarantee strict certification requirements, particularly for avionics and space systems.

The opportunity to make this worst-case performance analysis since the early design phases will allow designers to make important decisions concerning the systems parameters tuning and dimensioning.

To apply WoPANets, one first specifies the system in terms of application profiles and network architecture. The application profiles can then be defined relying on events traces or more commonly on traffic contracts. The latter define the main characteristics of the exchanged traffic flows using packet lengths, period or inter-arrival time, deadline, traffic class, jitter, burst size or source, just to mention a few. WoPANets implements a state-of-the-art Network Calculus timing analysis [6].

B. Network Calculus

Network Calculus is a mathematical theory introduced by Cruz in 1991 [7], [8]. It was extended and formalized using algebra *(min, +)* by Chang [9], by Le Boudec and Thiran [10], and by Bouillard, Boyer and Le Corronc [11].

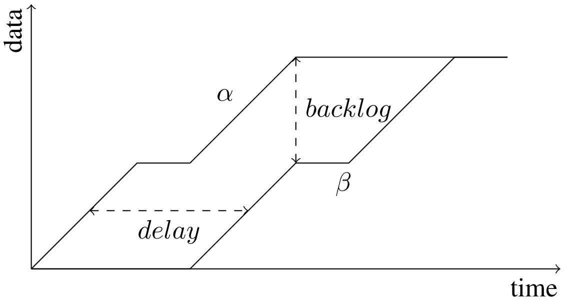


Fig. 2. Illustration of delay and backlog

Network Calculus was used to certify an industrial configuration of the network AFDX [12] and to study an extended AFDX network [13], and TTEthernet [14] .

Network Calculus derives deterministic upper bounds on network parameters such as the *backlog of network switches* and the *delay of messages*. In order to compute these bounds, Network Calculus models incoming flows with *an arrival curve* $\alpha(t)$ and availability of elements on traffic path with *a minimal service curve* $\beta(t)$. Traffic delays are bounded by the horizontal distance between α and β, whereas backlog are bounded by the vertical distance as illustrated in Figure 2.

IV. Contributions

A. New Method for SysML and TTool

Section II presented the three-step method associated with SysML and TTool. This section complements the requirements capture and analysis steps with a network dimensioning step (Figure 1).

The Network Dimensioning step has two roles.

- First, to check whether the requirements are correct or not, and to possibly initiate discussions with users, stakeholders and system designers if the requirements related to network dimensioning are not satisfied.
- Second, to refine the initial assumptions if the worst-case analysis revealed pessimism in the early assumptions.

For example, a common requirement in embedded networks is deadline. Every message must cross the network faster than its flow's deadline. The exact traversal time of one message is unknown. The classically associated assumption is to consider that this message's worst case traversal time (an upper bound to the exact traversal time) is equal to its deadline. Let's note D, the deadline, T the exact traversal time of the message, and $WCTT$ its worst case traversal time ($T \leq WCTT$). The requirement to check is: $T \leq D$. And the initial assumption is to consider that $WCTT = D$. With an early network dimensioning step for each message, it is possible to compute an upper bound of its exact traversal time. Let's note this bound B. Considering a message, if its bound is lower than its deadline ($B \leq D$) it means:

- First, the associated is satisfied: $T \leq B \leq D$

- Second, our initial assumption was pessimist. We can refine it: $WCTT = B$.

B. Annotated context diagrams

The network dimensioning step requires representing all network information on a diagram. In this paper, that information is located in annotating SysML block diagrams that serve as context diagrams.

An Annotated Context Diagram (Figure 3) is a graph with two types of nodes: Equipments and Switches. An equipment is a 2-uple (*name, service policy*) and a switch is a 4-uple (*name, scheduling policy, technological latency, switching technique*). Their attributes are specified in annotations. Nodes are connected by annotated links. A link is a 2-uple (*name, transmission capacity*). Flows are notes connected to an equipment. A flow is a 8-uple (*name, type, period, jitter, minimal packet size, maximal packet size, priority and routing*).

An attribute is described using an annotations: equipment, switch, link, or flow. Figure 4 depicts the meta-model of annotated Context Diagrams.

The syntax used in annotations is defined by the following Backus-Naur form:

```
⟨Annotation⟩ ::= ⟨equipment⟩ | ⟨switch⟩ | ⟨link⟩ | ⟨flow⟩
⟨equipment⟩ ::= Equipment ⟨name⟩ ⟨serv policy⟩
⟨switch⟩     ::= Switch⟨name⟩ ⟨serv policy⟩ ⟨tech latency⟩
                 ⟨switch tech⟩
⟨link⟩       ::= Link⟨name⟩ ⟨capacity⟩
⟨flow⟩       ::= Flow⟨name⟩ ⟨type⟩ ⟨period⟩ ⟨jitter⟩
                 ⟨min size⟩ ⟨max size⟩ ⟨priority⟩ ⟨routing⟩
⟨name⟩       ::= name"="⟨string⟩
⟨serv policy⟩ ::= service-policy"="
                 (FIRST_IN_FIRST_OUT
               | STATIC_PRIORITY)
⟨tech latency⟩ ::= tech-latency"=" ⟨integer⟩ ⟨time unit⟩
⟨time unit⟩  ::= µs | ms | s
⟨switch tech⟩ ::= switching-technique"="
                 (STORE_AND_FORWARD
               | CUT_THROUGH | WORM_HOLE)
⟨capacity⟩   ::= transmission-capacity"="
                 ⟨integer⟩ ⟨speed unit⟩
⟨speed unit⟩ ::= bps | Mbps | Gbps
⟨type⟩       ::= type"="(PERIODIC | SPORADIC)
⟨period⟩     ::= period"=" ⟨integer⟩ ⟨time unit⟩
⟨jitter⟩     ::= jitter"=" ⟨integer⟩ ⟨time unit⟩
⟨min size⟩   ::= min-payload"=" ⟨integer⟩ bytes
⟨max size⟩   ::= max-payload"=" ⟨integer⟩ bytes
⟨priority⟩   ::= priority"="(Low | High)
⟨routing⟩    ::= ⟨routing⟩ ⟨target⟩ | ⟨target⟩
⟨target⟩     ::= Routing":" ⟨string⟩ "=" ⟨path⟩
⟨path⟩       ::= ⟨path⟩ "/" ⟨string⟩ | ⟨string⟩
```

C. Interfacing TTool and WoPANets

TTool allows you to create annotated context diagrams and save them as an xml file. A parser has been developed using Python. It takes as input an annotated context diagram edited with TTool, and converts the XML file of the model into the input format of WoPANets (Figure 5).

It is then possible to make a worst case performance analysis with WoPANets. The output of WoPANets allows us to check the requirements and to update the assumptions of the initial model.

V. Case Study

We will now apply the method described above to a case study. Section V-A describes the considered network. Section V-B explains how this case study can be modeled with a context diagram. Finally, section V-C shows how the worst-case analysis of WoPANets can help refining the initial assumptions.

A. Network description

The network to model [15], [16] is part of an aircraft's Flight Management System that controls the display of navigation information on the flight displays. In order to realize this function, several calculators and sensors are interconnected through an AFDX network. AFDX [17] is an aeronautical embedded network based on switched Ethernet technology. Each data flow is modeled with a *virtual link*, namely a unidirectional path from the source equipment to all the destination equipments. The virtual link is characterized by its BAG (minimum distance between two consecutive messages) and its maximal message size.

On Figure 6, the AFDX network consists of seven modules (from M1 to M7) that host the avionics functions. These modules are interconnected by five switches (from S1 to S5). And finally the information supplied by the sensors arrive on the network via two remote data concentrators RDC1 and RDC2. Switches ports and module ports operate at 100 Mbps.

The different elements of the network must exchange information. This implies having multiple data flows through the network. In this case study there are twelve periodic flows (twelve virtual links, from VL1 to VL12) and so their BAG is also their period.

To each of these flows is associated a deadline constraint. In this case study, the deadline of each flow is equal to its period. Therefore the associated assumption consists in assuming the worst traversal time of each flow equal to its period. Table I summarized all the constraints and assumptions associated with the twelve flows. Moreover assumptions can be used in order to build a network abstraction diagram (see Figure 7). This diagram does not represent the entire network, flows being only characterized by their destinations an by their worst case traversal time. This type of diagram is used to model the rest of the system with the application part, so the detailed characteristics of the network are no longer needed.

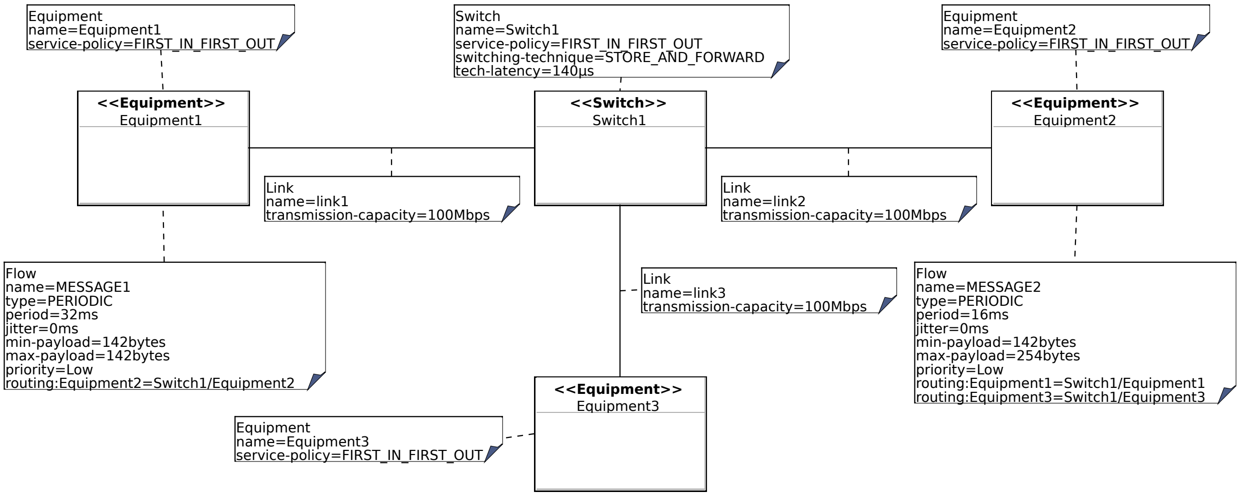


Fig. 3. Context diagram example.

B. Integration of the case study into TTool

The characteristics of the network being known, we can create a SysML context diagram (section IV-B) using the diagram editor of TTool.

Each switch is modeled by one node stereotyped by <<Switch>>. The module and remote data concentrators are both modeled by nodes stereotyped by <<Equipment>>.

Each node is annotated with a comment that contains all information relative to the relevant switch or module.

Last, all the flows are modeled via notes connected with the sender of each flow.

A simplified (without annotation) context diagram of the case study is represented by Figure 6. Part of the complete diagram is reproduced on Figure 8: it particularly contains the characteristics of two flows (VL3 and VL4). Finally the complete diagram is represented by Figure 9.

C. Worst case analysis of WoPANets

The worst case analysis of WoPANets computes a traversal time upper bound for each flow (B see Section IV-A). In this case study, the network is lightly loaded. Further, all deadlines are met: all upper bounds returned by WoPANets are lower than the previously expected maximum deadlines ($B \leq D$). Moreover, it is possible for us to update the assumptions, by using these upper bounds in our estimate of the worst case traversal time ($WCTT = B$). Table II shows what are the new assumptions for each flow. These news values allow to update the network abstraction diagram Figure 10.

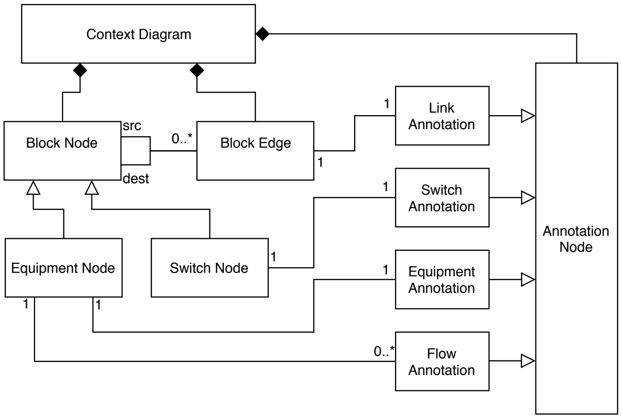


Fig. 4. Annotated Context Diagram meta-model

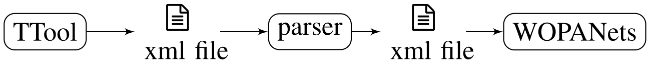


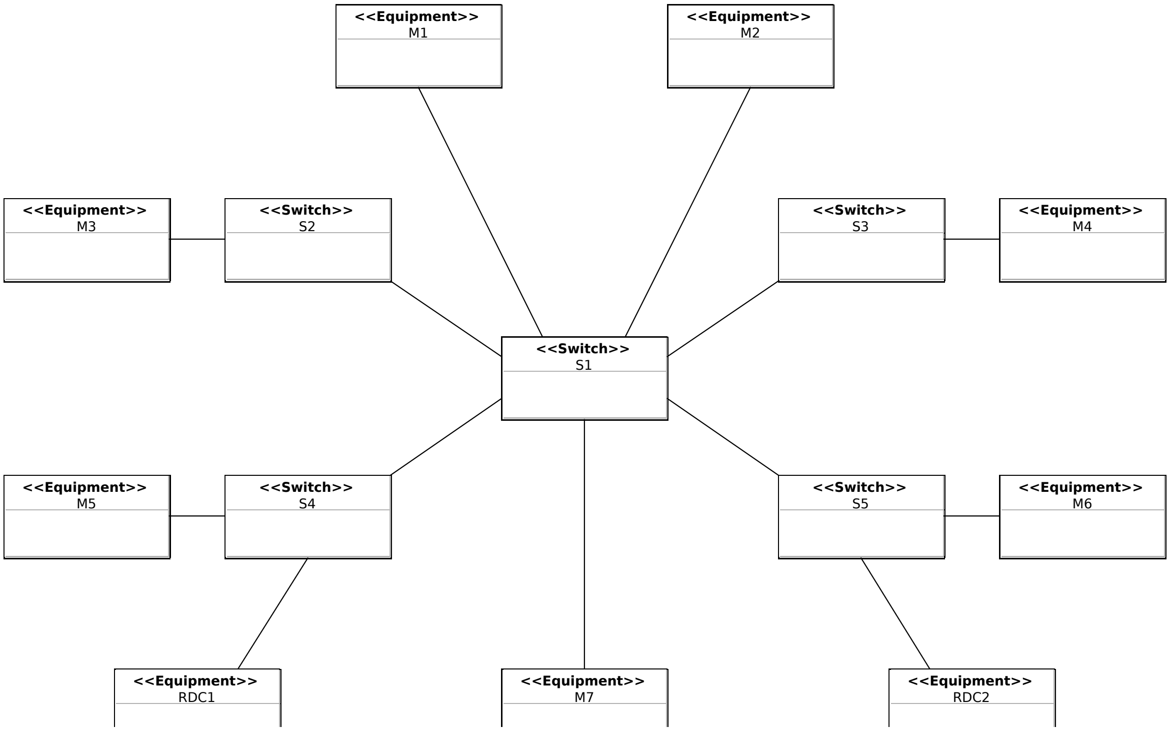
Fig. 5. Interfacing TTool and WoPANets

TABLE I
INITIAL REQUIREMENTS AND ASSUMPTIONS

Name:destination	Deadline (requirement)	Worst Case Traversal Time (assumption)
VL1:M3	32ms	32ms
VL1:M4	32ms	32ms
VL2:M3	32ms	32ms
VL2:M4	32ms	32ms
VL3:M1	8ms	8ms
VL4:M7	16ms	16ms
VL5:M1	8ms	8ms
VL6:M7	16ms	16ms
VL7:M3	64ms	64ms
VL8:M4	64ms	64ms
VL9:M5	32ms	32ms
VL10:M6	32ms	32ms
VL11:M3	32ms	32ms
VL11:M4	32ms	32ms
VL12:M3	32ms	32ms
VL12:M4	32ms	32ms

VI. RELATED WORK

Little work has been published on including a network dimensioning activity using network calculus in the early stages of the life cycle of distributed systems. In [18], Apvrille



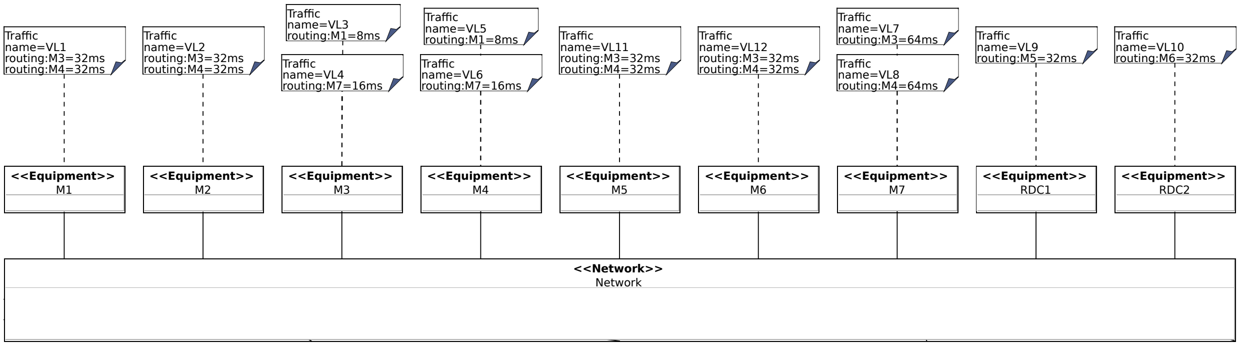


Fig. 7. Initial network abstraction diagram

et al. proposed a UML-class notation as a front-end for an early version of WoPANets ; however this approach was restricted to software modeling while ours fits in with a systems approach. Considering UML diagrams, they have been used in [19], [20] punctually and for documentation purposes only. [20] integrates network calculus to Matlab/Simulink for schedulability purposes. However, [20] does not address distributed systems nor formal verification and moreover the choice of Matlab/Simulink restricts the work on the design phase. In [15], the idea of associating the network calculus to modeling is proposed for AADL models and considering scheduling issues.

Network calculus and WoPANets are not the only approach used to compute deterministic performances of embedded network. For example, MAST [21], an open model for the description of event-driven real-time systems, allows to compute a worst-case response time schedulability analysis using a set of open source tools. Also, CPA (compositional performance analysis) is an approach to formal performance analysis of large embedded systems commercialized as SymTA/S [22]. All of these approaches have been compared in [23].

VII. Conclusions

The paper revisits the method associated with SysML and TTool to include a network dimensioning activity in the early stages of the life cycle of distributed systems. TTool is interfaced with WoPANets and thus builds upon the network calculus theory to dimension networks. The proposed approach is illustrated using an AFDX network.

In [18] Apvrille *et al.* proposed a UML-class notation as a front-end for an early version of WoPANets. In the

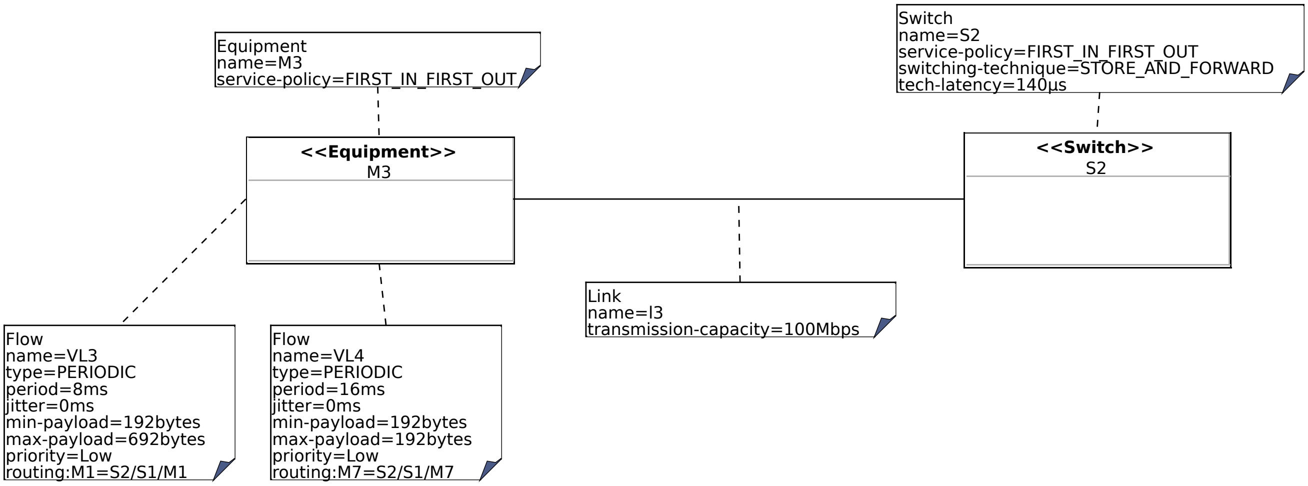


Fig. 8. Part of the complete context diagram of the case study

TABLE II
UPDATED REQUIREMENTS AND ASSUMPTIONS

Name:destination	Deadline (requirement)	Worst Case Traversal Time (assumption)
VL1:M3	32ms	434μs
VL1:M4	32ms	434μs
VL2:M3	32ms	434μs
VL2:M4	32ms	434μs
VL3:M1	8ms	483μs
VL4:M7	16ms	504μs
VL5:M1	8ms	483μs
VL6:M7	16ms	504μs
VL7:M3	64ms	552μs
VL8:M4	64ms	552μs
VL9:M5	32ms	172μs
VL10:M6	32ms	172μs
VL11:M3	32ms	594μs
VL11:M4	32ms	594μs
VL12:M3	32ms	594μs
VL12:M4	32ms	594μs

current paper, the idea of dimensioning networks is no longer restricted to software modeling and our proposal fits in with a systems approach. Further, each block in the class diagram can be cloned in the requirement or modeling assumption diagram in order to concretely link dimensioning to requirements and assumptions, and contribute to achieve requirement traceability, which is an important issue in system engineering.

Finally, the annotated context diagram that has been discussed is specifically addressing AFDX networks. The approach is being adapted to other types of networks; for example, the CAN bus.

ACKNOWLEDGMENT

Acknowledgements are due to Prof. Ahlem Mifdaoui and Prof. Ludovic Apvrille for their valuable inputs on WoPANet and TTool, respectively.

REFERENCES

[1] "Object management group." https://www.omg.org/.

[2] "International council for systems engineering." https://www.incose.org/.

[3] https://www.omg.org/spec/UML/2.5.

[4] https://ttool.telecom-paristech.fr/.

[5] P. de Saqui-Sannes, R. A. Vingerhoeds, and L. Apvrille, "Early checking of sysml models applied to protocols," *12th International Conference on Modeling, Optimisation and Simulation (Mosim 2018)*, 2018.

[6] A. Mifdaoui and H. Ayed, "WOPANets: a tool for WOrst case Performance Analysis of embedded Networks," in *2010 15th IEEE International Workshop on Computer Aided Modeling, Analysis and Design of Communication Links and Networks (CAMAD)*, pp. 91–95, IEEE, 2010.

[7] R. L. Cruz, "A calculus for network delay. I. Network elements in isolation," *IEEE Transactions on information theory*, vol. 37, no. 1, pp. 114–131, 1991.

[8] R. L. Cruz, "A calculus for network delay. II. Network analysis," *IEEE Transactions on information theory*, vol. 37, no. 1, pp. 132–141, 1991.

[9] C.-S. Chang, *Performance Guarantees in Communication Networks*. Springer Science & Business Media, 2000.

[10] J.-Y. Le Boudec and P. Thiran, *Network calculus: a theory of deterministic queuing systems for the internet*, vol. 2050. Springer Science & Business Media, 2001.

[11] A. Bouillard, M. Boyer, and E. Le Corronc, *Deterministic Network Calculus: From Theory to Practical Implementation*. John Wiley & Sons, 2018.

[12] J. Grieu, *Analysis and evaluation of Ethernet switching techniques for avionic systems interconnection (in French)*. PhD thesis, Institut National Polytechnique de Toulouse, Toulouse, France, 2004.

[13] F. F. Finzi Anaïs, Mifdaoui Ahlem and E. Lochin, "Network calculus-based timing analysis of afdx networks with strict priority and tsn/bls shapers," in *13th International Symposium on Industrial Embedded Systems (SIES 2018)*, (Graz, AT), pp. 1–10, August 2018.

[14] M. Boyer, H. Daigmorte, N. Navet, and J. Migge, "Performance impact of the interactions between time-triggered and rate-constrained transmissions in TTEthernet," in *8th European Congress on Embedded Real Time Software and Systems*, 2016.

[15] G. Brau, J. Hugues, and N. Navet, "Refinement of AADL models using early-stage analysis methods," in *Proceedings of the 4th Analytic Virtual Integration of Cyber-Physical Systems Workshop; December 3; Vancouver; Canada*, pp. 29–32, Linköping University Electronic Press, 2013.

[16] M. Lauer, *A global method for real-time requirement verification: application to Integrated Modular Avionics (in French)*. PhD thesis, Institut National Polytechnique de Toulouse, Toulouse, France, 2012.

[17] "ARINC. 2005. Aircraft data network Part 7: Avionic full duplex switched Ethernet (AFDX) network. ARINCSpecification 664P7."

[18] L. Apvrille, A. Mifdaoui, and P. de Saqui-Sannes, "Real-time distributed systems dimensioning and validation: The turtle method," *Studia Informatica Universalis*, vol. 3, no. 8, pp. 47–69, 2010.

[19] E. Wandeler, L. Thiele, M. Verhoef, and P. Lieverse, "System architecture evaluation using modular performance analysis: a case study,"

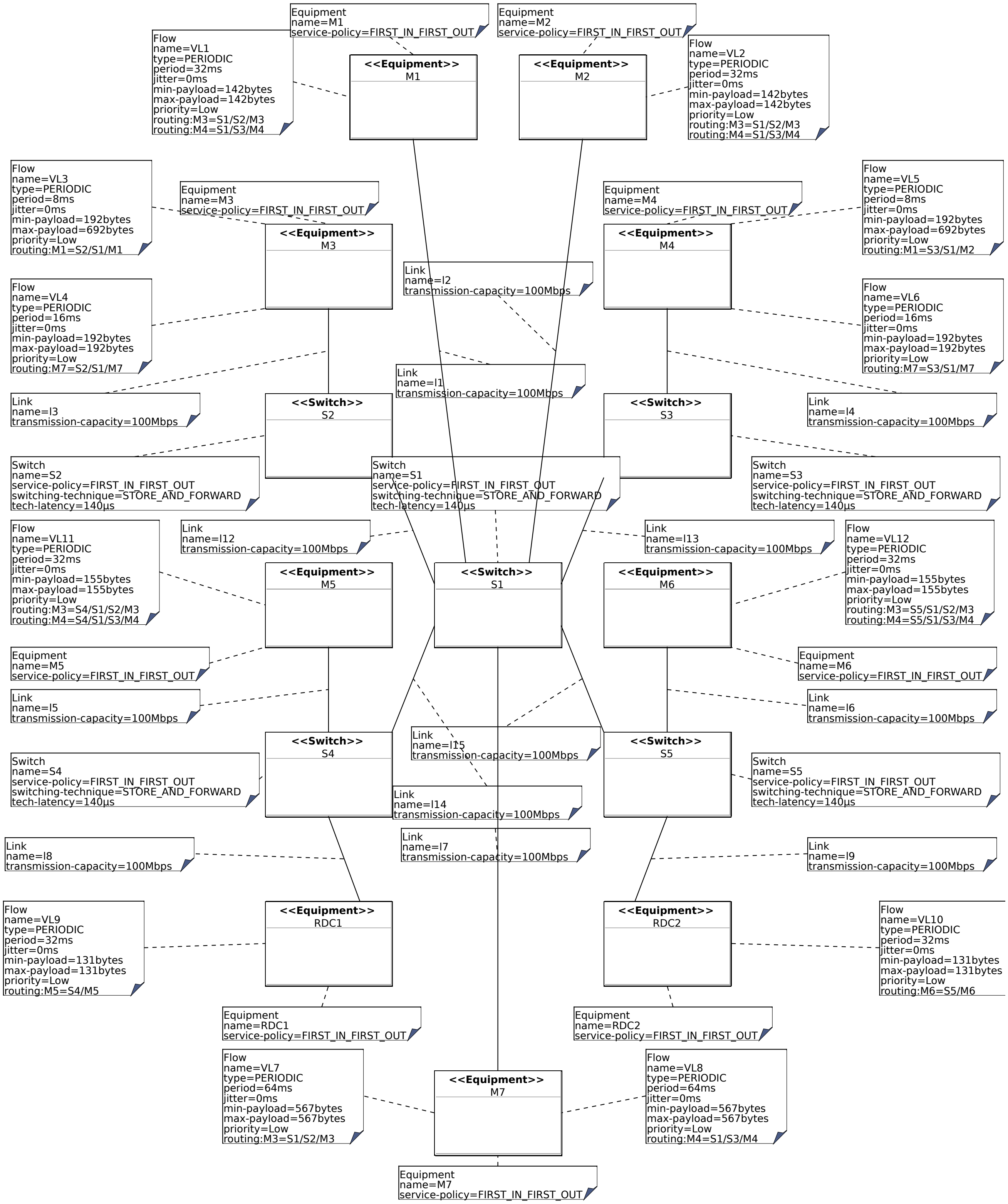


Fig. 9. Context diagram of the case study

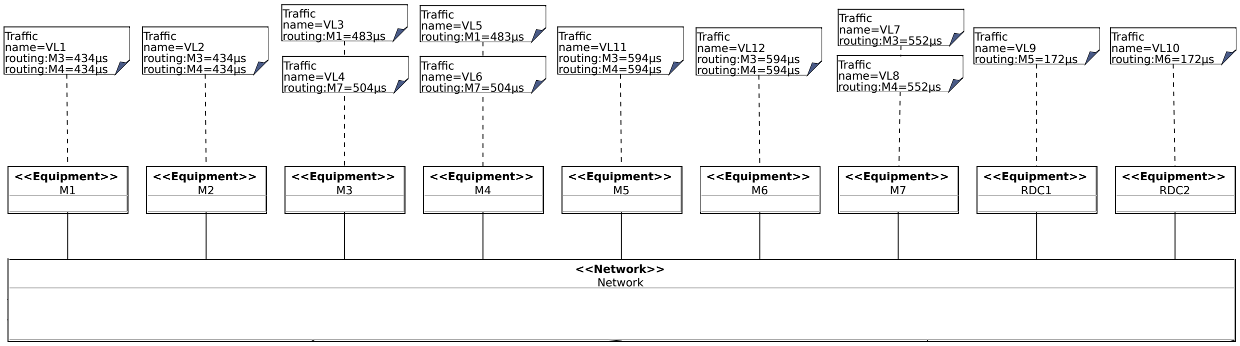


Fig. 10. Updated network abstraction diagram

International Journal on Software Tools for Technology Transfer, vol. 8, no. 6, pp. 649–667, 2006.

[20] H. Schioler, H. P. Schwefel, and M. B. Hansen, "CyNC - a MATLAB/SimuLink Toolbox for Network Calculus," in *Proceedings of the 2nd international conference on Performance evaluation methodologies and tools*, p. 60, ICST (Institute for Computer Sciences, Social-Informatics and , 2007.

[21] M. G. Harbour, J. G. García, J. P. Gutiérrez, and J. D. Moyano, "MAST: Modeling and analysis suite for real time applications," in *Proceedings 13th Euromicro Conference on Real-Time Systems*, pp. 125–134, IEEE, 2001.

[22] R. Henia, A. Hamann, M. Jersak, R. Racu, K. Richter, and R. Ernst, "System level performance analysis–the SymTA/S approach," *IEE Proceedings-Computers and Digital Techniques*, vol. 152, no. 2, pp. 148–166, 2005.

[23] S. Perathoner, E. Wandeler, L. Thiele, A. Hamann, S. Schliecker, R. Henia, R. Racu, R. Ernst, and M. G. Harbour, "Influence of different abstractions on the performance analysis of distributed hard real-time systems," *Design Automation for Embedded Systems*, vol. 13, no. 1-2, pp. 27–49, 2009.