Modeling Real-Time Networks with MAST2

Michael González Harbour, J. Javier Gutiérrez, J. María Drake, Patricia López, and J. Carlos Palencia

Computers and Real-Time Group, Universidad de Cantabria, 39005-Santander, SPAIN
{mgh, gutierjj, drakej, lopezpa, palencij}@unican.es

Abstract[1]

Switched networks have an increasingly important role in real-time communications. The IEEE ethernet standards have defined prioritized traffic (802.1p) and other QoS mechanisms (802.1q). The Avionics Full-Duplex Switched Ethernet (AFDX) standard defines a hard real-time network based on switched ethernet. In the process of defining the new MAST2 model, the network elements have been enhanced to include switches and routers. This paper introduces the schedulability model that will enable an automatic schedulability analysis of an application using switched networks.

1. Introduction

MAST (Modeling and Analysis Suite for Real-Time Applications) [2][4] defines a model to describe the timing behavior of real-time systems designed to be analyzable via schedulability analysis techniques. MAST also provides an open-source set of tools to perform schedulability analysis or other timing analysis, with the goal of assessing whether the system will be able to meet its timing requirements, and, via sensitivity analysis, how far or close is the system from meeting its timing requirements.

The model defined in MAST is very similar to the model defined in the standardized UML profile for real-time embedded systems called MARTE [5]. A new enhanced model is currently being defined as a project called MAST2, trying to incorporate new modelling elements that can be found in real systems. It is expected that the ideas introduced in MAST2 will contribute to the future evolution of the MARTE standard.

Some of the new elements being defined in MAST2 are network switches and routers. Switched networks are being used increasingly to build real-time systems, as new network switches incorporate the real-time mechanisms being defined in standards such as IEEE 802.1p with prioritized traffic, 802.1q with various QoS mechanisms [6], or the Avionics Full-Duplex Switched Ethernet (AFDX) [1] that defines a hard real-time network based on switched ethernet.

1. This work has been funded in part by the Spanish Ministry of Science and Technology under grant number TIN2008-06766-C03-03 (RT-MODEL), and by the European Commission under project FP7/NoE/214373 (ArtistDesign).

This paper introduces the model elements required to add network switches and routers into the MAST model. These elements will allow an automatic schedulability analysis of applications using switched networks.

The paper is organized as follows. In Section 2 we present a general overview of the MAST2 model, and we focus on the network modelling elements in Section 3. The new elements introduced to model network switches are presented in Section 4, and similarly in Section 5 for network routers. Section 6 introduces the new modelling elements for AFDX networks and switches together with a simple example using these elements. Finally, Section 7 gives our conclusions.

2. Overview of the MAST2 Model

A real-time system is modelled in MAST2 using different independent views for describing the execution platform, the software modules and messages exchanged through the networks, the concurrent architecture, and the workload and flow of events for a particular configuration of the application. This independence among the various elements of the model is ideal for building a full model through the composition of partial models developed independently.

The execution platform view contains *Processing_Resources* such as *Processors* and *Networks*, together with their *Schedulers* and associated *Scheduling_Policy* elements. Each of these elements contains attributes that describe their timing behavior including overheads such as context switching, interrupt service, or system timers. *Processors* have a relative speed and *Networks* have a bandwidth expressed in bits per time unit. MAST2 also defines elements to describe clock synchronization mechanisms.

The operations view contains the elements that describe the usage of the processing resources, through *Operations*. *Code_Operations* model the execution of sequential code with a given execution time distribution, while *Message_Operations* represent data of a given size that is sent through a network. The *Code_Operation* abstract class is specialized with *Simple_Operation* and *Composite_Operation*, while *Message_Operation* is specialized with *Message* and *Composite_Message*.

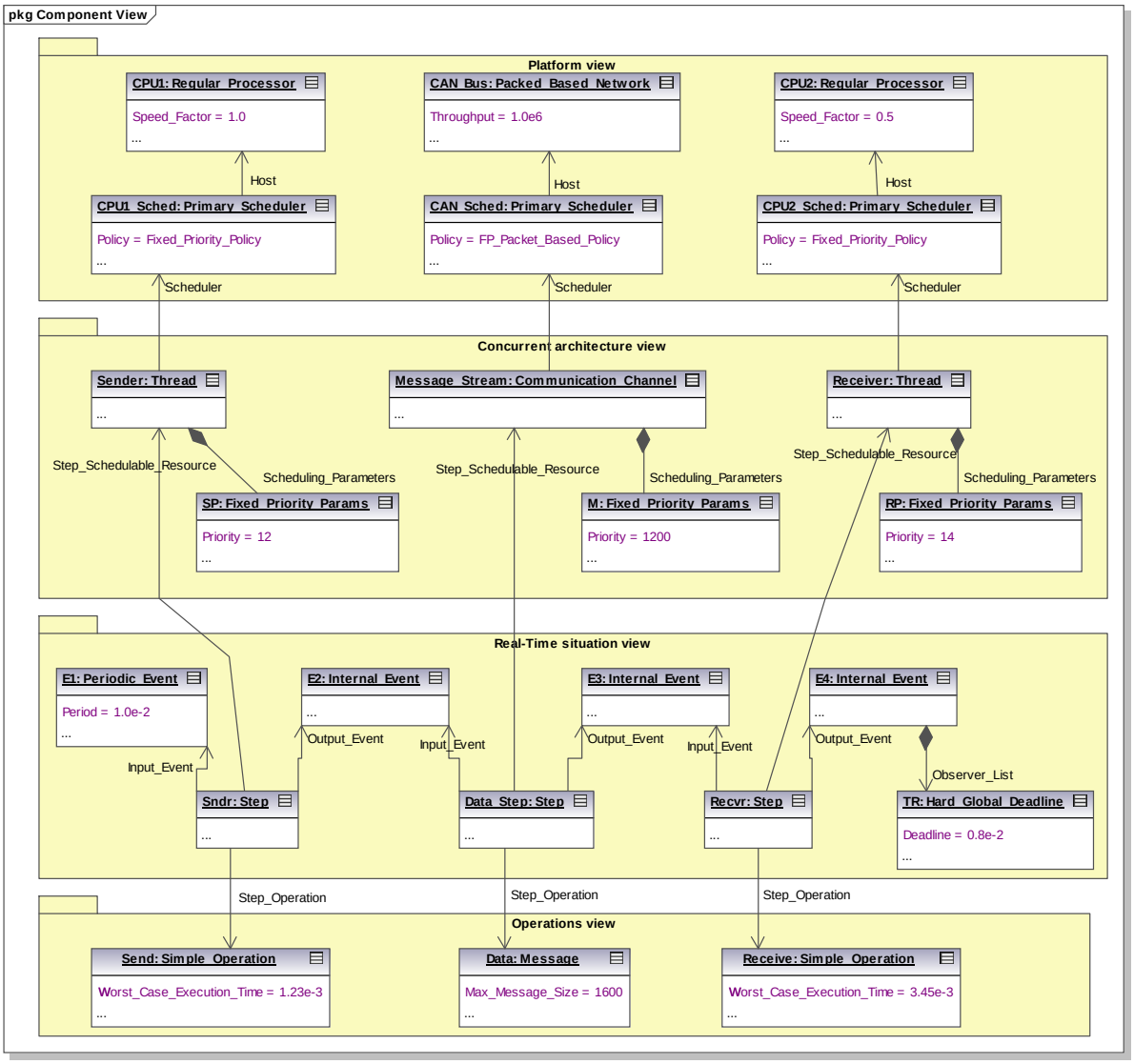

Figure 1. MAST2 elements of the simple distributed example

The concurrent architecture view contains the *Schedulable_Resources* that represent schedulable entities in a *Processing_Resource*. The *Thread* models a unit of concurrent execution in a processor (a task, single-threaded process, thread, or interrupt service routine) while the *Communication_Channel* models a message stream transmitted through a network with specific scheduling parameters. *Schedulable_Resources* have a reference to the *Scheduler* where they are scheduled, and the associated *Scheduling_Parameters* that are used by the scheduler to arbitrate access to the processing resource.

The real-time situation view represents a particular mode of operation of a real-time system. It is modelled as a set of concurrent *End_To_End_Flows* (previously called transactions) that compete for the resources offered by the platform. Each end-to-end flow is activated from one or more *Workload_Events*, and contains a sequence of *Steps* that are executed in the system. Each *Step* represents the execution of an *Operation* (*Code_Operation* for a processor or *Message_Operation* for a network) by a *Schedulable_Resource*. When a *Step* finishes its execution it generates an *Internal_Event* that may, in turn, activate other *Steps*. Special *Event_Handlers* exist in the model to

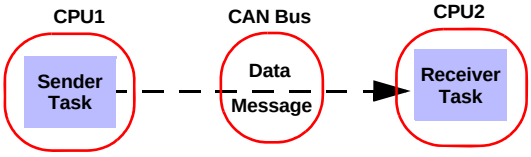

Figure 2. Example with a simple distributed application

handle events in special ways. *Internal_Events* may have *Observer*s associated with them , and rep resenting timing requirements or other requirements that must be observed.

We will use the example with a simple distributed application depicted in Figure 2 t o show the MAST2 elements related to the networks. The application has a periodic task sending a message through a CAN bus to a second task executing in a different processor, and its model is shown in Figure 1. In the real-time situation view we can see th e single end-to-end flow consisting of three steps joined through the corresponding events, with the first event defining the periodicity, and the last one with a hard end-to-end deadline.

3. Networks

A *Network* is a ki nd of *Processing_Resource* that models a communication system specialized in the transmission of messages among processors or net work switches.

Figure 3 show s the class diag ram of t he networks in MAST2. The main element th at represents a real-time network is the *Packet_Based_Network*, which models a network that uses some kind of real-time protocol based on non-preemptible packets for sending mes sages. This element can represent a networ k that supports priorities in its standard protocol, such as th e CAN bus, and other networks with no priorities such as the point-to-point ethernet links tha t are used to connect CPUs to/from network switches.

Other priority-based networks may need more complex models. For instance, the *RTEP_Network* represents a network that uses the RTEP protocol [3], which is a token-passing protocol with prioritized messages, that uses a two-phase mechanism to send each information packet: a priority arbitration phase and the transmission phase.

Network switches were n ot part of the original MAST model, neither are defined in the MA RTE standard. We will now introduce modelling elements to represent the network switches and the cont ention inside them. To support the A FDX networks we wi ll need to introduce a special *AFDX_Switch* and the *AFDX_Link*, as a new kind of network. Both will be described in Section 6.

4. Network Switches

A network switch is a communications subsystem that is capable of establishing simu ltaneous connections between a number of input networks and output networks. These connections are establ ished on a per-message basis. In a store and forward switch each message arriving at an input port is stored in the switch's internal memory. As t he message carries information on its dest ination port, the switch forwards this message to the output port, or to the specified output ports in case of a multicas t or broadcast message. The switch is capable of managing multiple such connections simultaneously, through replicated hardware. This ability allows a full usage of the available network bandwidth, depending on the traffic source and destination addresses.

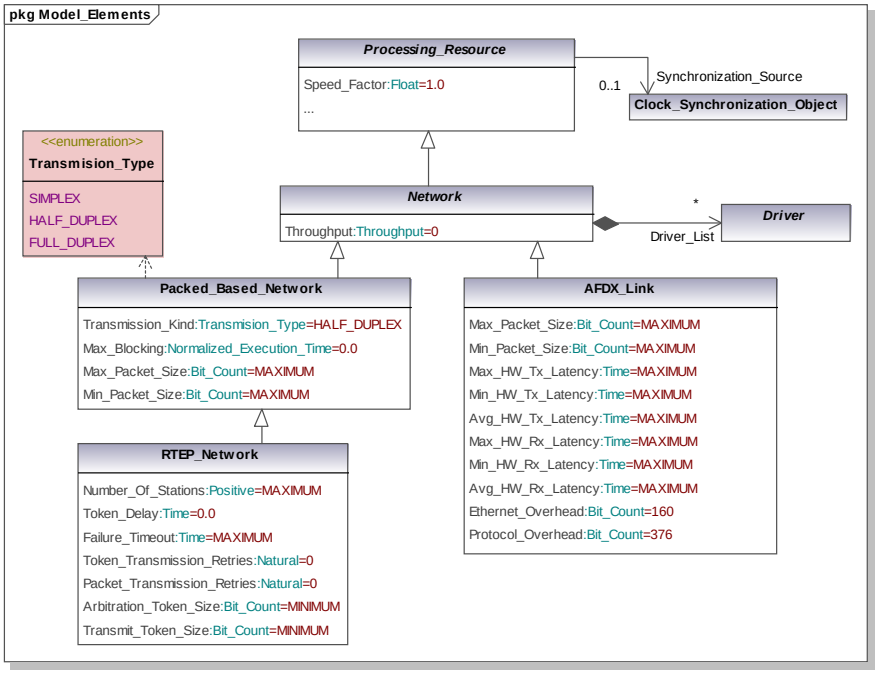

Figure 3. Network modelling elements

Contention may occur in the switch when several messages need to be forwarded to the same output port concurrently. The switch usually manages a queue of messages addressed at a specific output port. Some switches manage this queue in FIFO order, and others may use information contained in the message to prioritize the queue.

New modelling elements are introduced in MAST2 to model the timing effects introduced by network switches, and in particular the worst-case behavior of the contention in the output ports.

The *Network_Switch* (Figure 4) is an abstract *Processing_Resource* that models a communication system specialized in the transfer of messages among networks. It is capable of delivering messages arriving at an input port to one or more output ports.

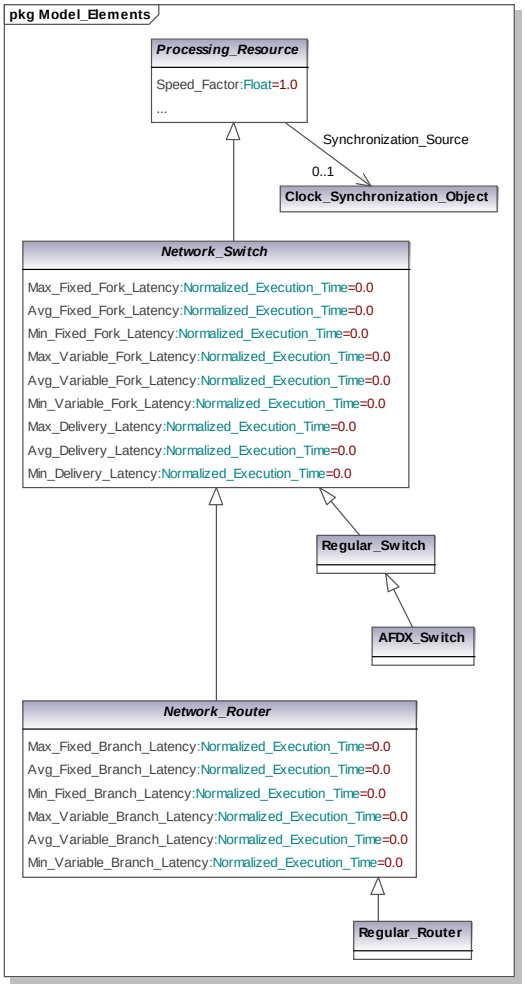

Figure 4. Network switches and routers

The delivery of a message inside a switch is specified through new special-purpose *Message_Event_Handlers*, described in Figure 5. The *Message_Delivery* element represents a simple port-to-port delivery, while the *Message_Fork* handler enables modelling multicast messages. The destination port or ports are implied in the message stream.

The *Message_Delivery* may contain scheduling parameters to define the priority used to resolve the contention at the output port queue in the switch.

The attributes of the *Network_Switch* allow calculating the overheads associated with the message delivery. Max/Min/Average overheads are defined for the delivery of a message to a single destination port, and for the delivery to multiple destinations through a *Message_Fork* handler. This latter overhead has a fixed part and a variable part that depends on the number of output ports.

A *Regular_Switch* is a concrete and simple switch in which the output queues are assumed to work according to the policy associated with the scheduling parameters of the message delivery elements allocated to the switch. For instance, priority-based in case of fixed-priority parameters, or FIFO when no scheduling parameters are specified. The switch is assumed to have routing information preconfigured, so there is no need for additional network traffic originated by the switch.

MAST2 does not require defining the network topology, since it is implicitly defined in the end-to-end flows.

5. Network Routers

A *Network_Router* is an abstract specialization of the *Network_Switch* that models a communication system specialized in routing messages among networks. In addition to the capabilities of a switch, the router is able to route a message through a destination port that may be dynamically obtained.

Routers support a new *Message_Branch* event handler, which is capable of delivering a message to one output port chosen form of a set of possible outputs. The overhead model for the *Message_Branch* handlers is similar to the model for message fork. It contains a fixed latency and a variable latency that depends on the number of possible output ports.

The class *Regular_Router* is a concrete and simple router in which the output queues are assumed to work according to the policy associated with the scheduling parameters of the allocated message delivery elements.

6. Modelling AFDX Networks

AFDX (Avionics Full Duplex Switched Ethernet) is a communications network defined in the ARINC-644, Part

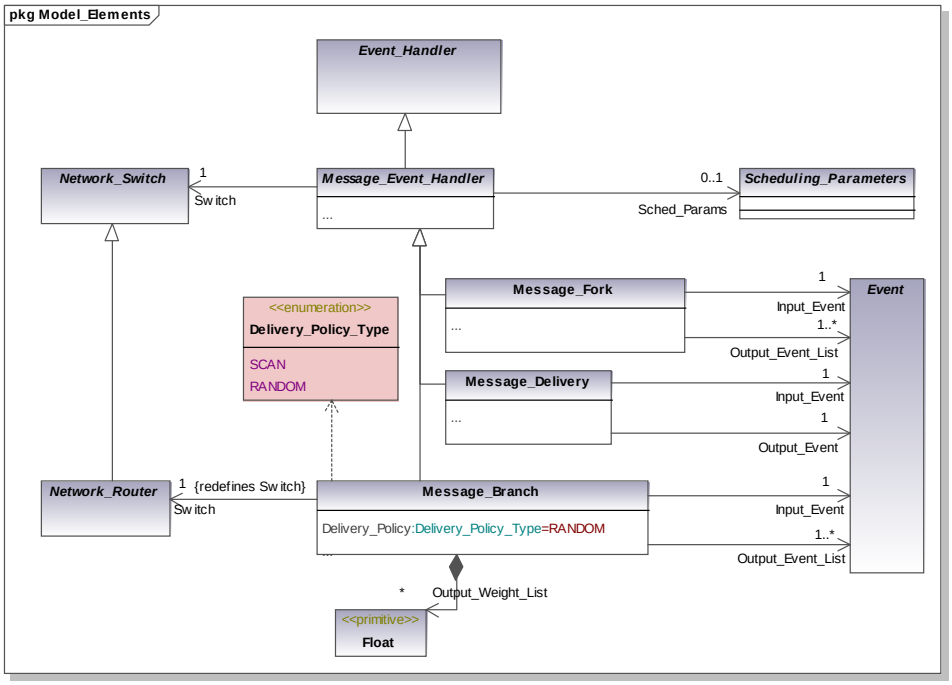

Figure 5. Message event handlers

7 standard [1] and based on the use of point to point full duplex ethernet links and special purpose switches. The routing of messages is preconfigured so that there is no delay in the discovery of routing addresses through network protocols that could interfere with the transmission of application messages.

The AFDX network provides traffic regulation at the sending end via *Virtual Links*, defined as conceptual communication objects to establish a logical unidirectional connection from one source end system to one or more destination end systems having a dedicated maximum bandwidth. Each virtual link (VL) is characterized by two parameters used for traffic regulation: the largest Ethernet frame (*Lmax*) in bytes, and the Bandwidth Allocation Gap (*BAG*), a power of the value 2 in the range [1,128]. The *BAG* represents the minimum interval in milliseconds between Ethernet frames transmitted on the VL.

Each virtual link has a FIFO queue for all the fragmented packets to be transmitted through it. This queue introduces contention that needs to be modelled and analyzed to determine real-time schedulability.

The AFDX network is modelled in MAST2 through two new elements: The *AFDX_Link* network, and the *AFDX_Switch*.

An *AFDX_Link* (see Figure 3) represents a link between a processor and/or a switch, allowing full duplex communications. When a message originated in a processor traverses this link, it uses the new *AFDX_Policy* scheduling policy. The *Schedulable_Resources* that may be used on this link need to specify as *Scheduling_Parameters* an object of the *AFDX_Virtual_Link* class, which describes the corresponding *Lmax* and *BAG* parameters as it is shown in Figure 6.

An *AFDX_Switch* is a concrete and simple switch working according to the AFDX specification. The output queues in the switch have two priorities. The priority may be assigned in the model by specifying it in the scheduling parameters of the *Message_Delivery* operation executed inside the switch.

Figure 7 shows a partial view of the example described in Figure 2 and Figure 1, focussing only on the network part, and assuming that instead of a CAN Bus we are using

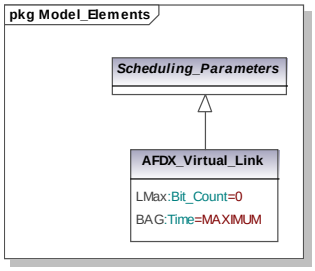

Figure 6. Virtual Link Scheduling Parameters

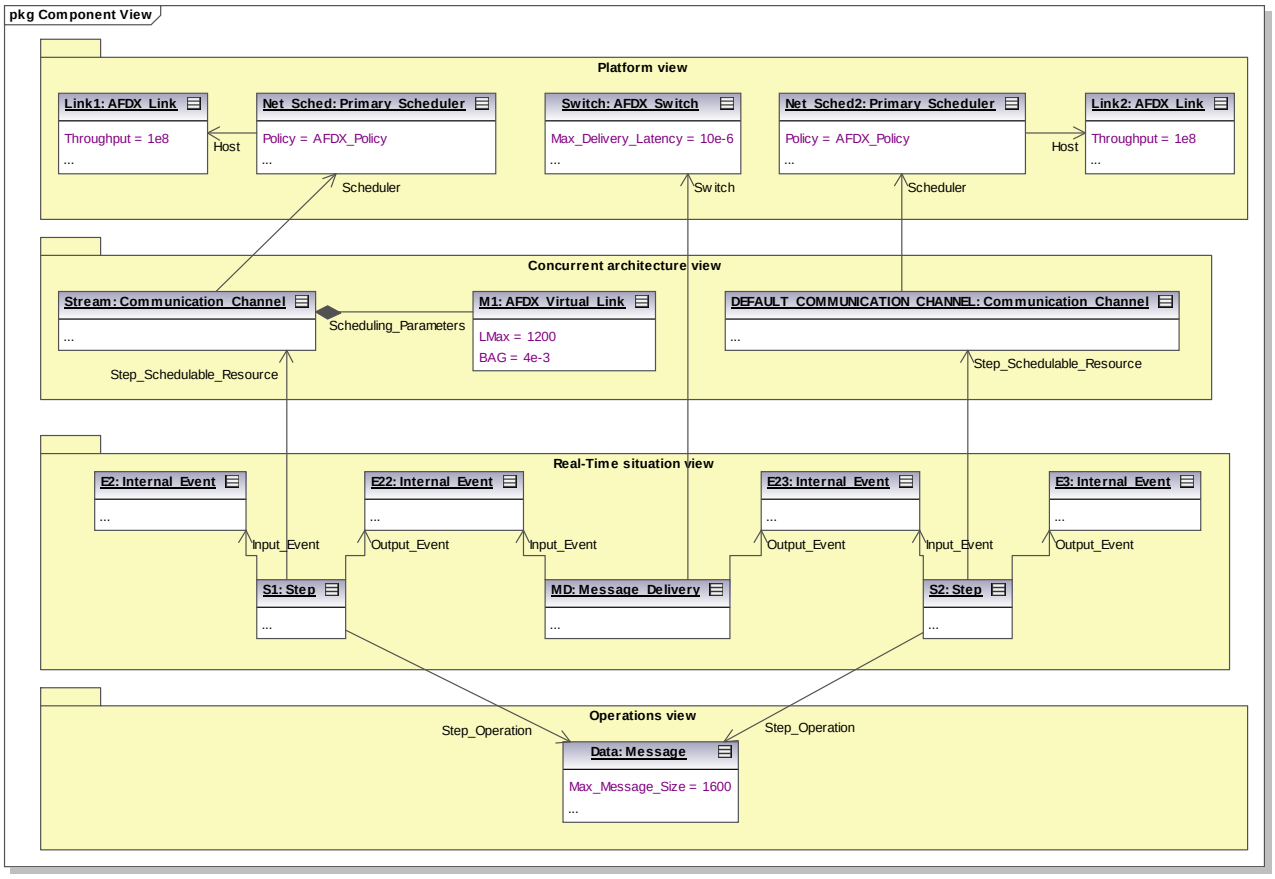

Figure 7. Partial view of an end-to-end flow using an AFDX switch

an AFDX network with a switch and two AFDX links communicating each CPU with the switch. No priority is specified.

7. Conclusions

In this paper we propose new modelling elements to support network switches and routers and AFDX real-time networks in the timing models used to assess the schedulability of real-time distributed applications. These elements, together with their associated analysis techniques are being implemented in MAST2, and will be proposed for a future version of the MARTE UML profile for real-time embedded systems.

References

[1] Airlines Electronic Engineering Committee, Aeronautical Radio INC., "ARINC Specification 664P7: Aircraft Data Network, Part 7 - Avionics Full Duplex Switched Ethernet (AFDX) Network," June 27, 2005.

[2] M. González Harbour, J.J. Gutiérrez García, J.C. Palencia Gutiérrez, and J.M. Drake Moyano. "MAST: Modeling and Analysis Suite for Real Time Applications". Proceedings of 13th Euromicro Conference on Real-Time Systems, Delft, The Netherlands, IEEE Computer Society Press, pp. 125-134, June 2001.

[3] J.M. Martínez, and M. González Harbour. "RT-EP: A Fixed-Priority Real Time Communication Protocol over Standard Ethernet". Proc. of the 10th International Conference on Reliable Software Technologies - Ada-Europe 2005, York (UK), in LNCS, Vol. 3555, Springer (2005).

[4] MAST: Modelling and Analysis Suite for Real-Time Systems. Home page: `http://mast.unican.es`

[5] Object Management Group. "A UML Profile for MARTE: Modeling and Analysis of Real-Time Embedded systems". MARTE specification version 1.0 (formal/2009-11-02).

[6] Xu, Lei. "Industry Network QoS Approach Based on New Ethernet Standards". Proceedings of 2009 4th International Conference on Computer Science & Education.