Model-based approach for IMA platform early exploration

Michaël Lafaye[a,b], David Faura[a], Marc Gatti[a], Laurent Pautet[b]

[b] TELECOM ParisTech
LTCI
46 rue Barrault
75634 Paris Cedex 13, France
{lafaye, pautet}@telecom-paristech.fr

[a] THALES Avionics
ACS/DTEA
18 avenue du Maréchal Juin
92366 Meudon-la-Forêt Cedex, France
{david.faura, marc-j.gatti}@fr.thalesgroup.com

Abstract

The avionics platform are now developed according to the Integrated Modular Avionics concept, allowing one processing module to host some applications in order to reduce weight, space and costs. This concept increases the development and certification complexity, while time to market tends to decrease. So new development processes are needed. Model-based approaches are now mature enough to design embedded critical systems and perform architecture exploration.

In this paper we propose a new modeling approach that aim to size an execution platform architecture (OS and hardware) according to the applications it has to process, and achieve early platform exploration.

Keywords : modeling, avionics systems, real-time, simulation, AADL, SystemC

1. Introduction

Avionics systems are critical real time systems composed of applications, real-time operating system and hardware modules. To reduce the environmental footprint in term of weight and space, and also the costs, the Integrated Modular Avionics (IMA) concept was developed in the 2000s. It defines integrated architectures, where one calculator hosts some applications. So the number of modules aboard is reduced. Following this evolution, suppliers developed network architectures in which modules are interconnected and communicate through a deterministic network. However, this evolution entails an increase of complexity in avionics platform design, verification and certification processes, while time to market tends to decrease. These developments require an early modeling of the system to validate and maximize the use of the future platform, while ensuring the critical level required by current standards in aviation (DO-178B, MILS-CC…).

To model IMA platform and perform early validation, **M**odel-**D**riven **E**ngineering approaches [1,2] are now suitable to describe system high-level functionalities. Many projects aim at modeling these platform with **A**rchitectural **D**escription **L**anguages as AADL [3] or MARTE [4], or with synchronous languages such Lustre or Signal [5]. However, they often focus on the applications description, model the hardware as connected blackbox components with a few properties, and perform static simulation. Moreover, there actually is no automated process for complete platform modeling and simulation.

We propose a new modelling approach that aims to design a complete avionics system (hardware, operating system and the applications), and perform dynamic simulation. It is a component-based approach, relying on two languages and taking advantages of both: AADL and systemC. AADL[8] is, as MARTE, an ADL particularly adapted for software architecture description [4,6], enabling the modeling of ARINC 653 embedded real-time system [7]. In view of the experience of partners who especially developed the ARINC 653 AADL annex and the Ocarina tool suite, we choose the AADL rather than MARTE. SystemC [9] is an IEEE standard widely used in industry for hardware platform description, and including a simulation kernel for architecture simulation and exploration.

In this paper we first present real-time system modeling works with ADL before introducing IMA concept, then we detail our method, and conclude with perspectives of our work.

2. State Of The Art

Some projects aimed at modeling real time critical systems with ADL. The SPICES project [3] and MARTES project [4] both tried to use a combination of an ADL (MARTE for MARTES, the AADL for Spices) and the systemC language. Their goal was to model at high level a real-time system, and generate by code translation a systemC platform. The idea of using

such a combination seems relevant, but the part of code translation from the ADL to systemC platform model is a harder operation, because there are lots of cases to manage. Moreover, the systemC hardware description does not allow enough accuracy in execution platform performances analysis for our approach.

Our approach is also partly based on the ARINC653 system modeling works with AADL, that led for the development of the AADL ARINC653 annex [7]. It enables the description of the spatial and temporal partitioning method, secure inter and intra partitions communications, and applications deployment on the execution platform. However, there is no behavior specification for the hardware part, and no dynamic simulation. Our approach looks at fulfill these lacks.

3. Integrated Modular Avionics Platform

An IMA platform, as illustrated in figure 1, is composed of avionics applications running on execution modules including embedded operating system and the underlying hardware. These modules communicate through a deterministic network, the AFDX (Avionics Full DupleX).

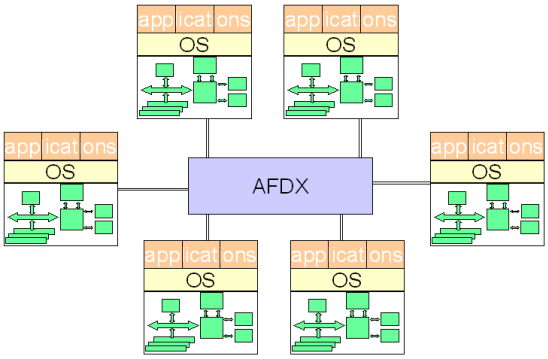

Figure 1. IMA platform

A processing module hosts some applications at different criticality levels, it is then necessary to respect safety constraints. That's why OS ARINC 653 standard was defined, which specifies space and time partitioning. To ensure space partitioning and prevent failure propagation, each application is enclosed in one or some partitions, isolating it from each other. Each partition is bound to a part of memory, so it only access its own memory area (figure 2). The standard also defined secure intra and inter-partitions communications. To ensure time partitioning, each partition has its own execution time window, during which the application has access to all dedicated resources (processing, memory, dedicated I/O, etc.).

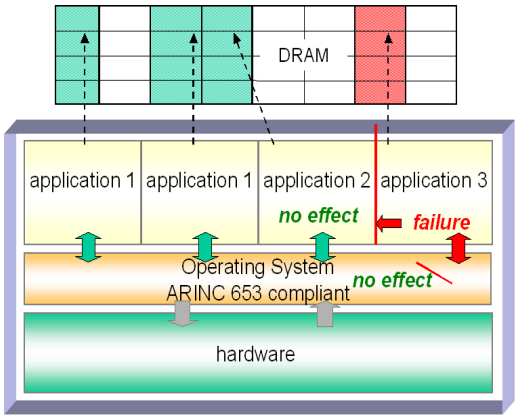

Figure 2. ARINC 653 spatial partitioning

4. IMA Platform Modeling

4.1. Overview

Model-Driven Engineering approaches are now mature enough to serve as a basis for building embedded systems and perform early validation. They are especially suitable for modeling high-level functional architecture (description of the functionalities offered by the system) and logical architecture (description of how the system is structured into logical components cooperating by communications) [10]. But at platform architecture level, these approaches describe both hardware and software as static blackbox elements with some properties.

Some projects [3,4,5] aim at building more accurate platform models, but they mainly focus on the software behavior, and model hardware as one or a few blackbox components without behavior information. They after perform static analysis on the model. So there is no method to retrieve dynamic performances from the hardware in order to validate it according to the applications requirements.

Our method develops a new approach for avionics platform modelling, that refines architecture description at platform decomposition level, specially the hardware. In order to refine this later, our approach models and sizes the execution platform according to applications requirements. As we can see in figure 3, this approach consists of different tasks:

- system high level modeling;
- applications characteristics extraction;
- platform generation according to requirements;
- platform simulation and performances analyses.

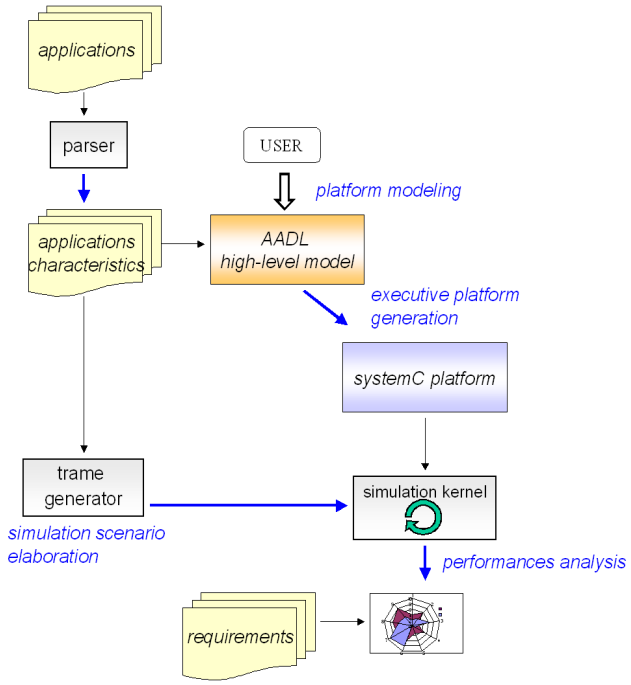

Figure 3. Modeling Approach

4.2. Applications modeling

An important point at this step is that we have not necessarily access to the applications code, so we can not directly generate the applications characteristics files (Figure 4). Consequently, we define the main applications characteristics we need in order to generate stimuli to simulate the platform. These characteristics are for example parallel code percentage, I/O access, processes duration and deadline…which can be obtained by code profiling or extracted from the applications configuration files.

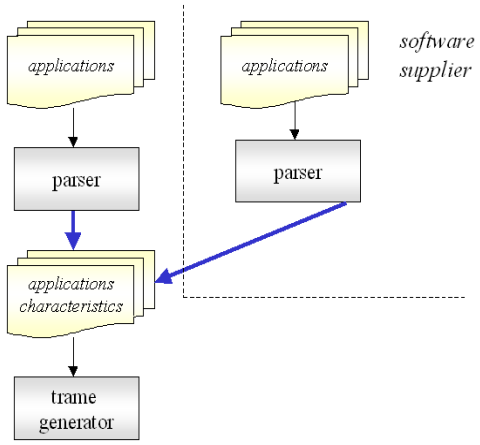

Figure 4. Application modeling.

These characteristics are first used to model the applicative part of the platform with the AADL. Partitions are modeled with the AADL processes, and ordered according to a given pattern called Major Frame, as in figure 5, where P1 represents the partition 1 and so on. The AADL threads represent the ARINC 653 processes of the applications. They are configured (deadline, priority etc.), bound to an AADL processor and a part of the main memory. Previous works defined rules to model ARINC653 system with the AADL [7].

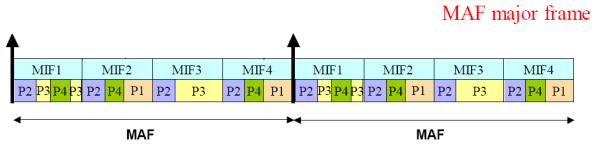

Figure 5. Example of Major Frame

These applications characteristics files is our input. We defined a set of macro-instruction (for example dram_read_command) which target the different modules of the platform. Thus we translate the different characteristics into these systemC macro-instruction. These instructions will be injected into the simulation kernel when simulating the future platform.

4.3. High level system modeling with AADL

Contrary to the applications, described with AADL threads, the operating system is defined by some properties dispatched in the different hardware components. For example, scheduling policy is set in the processor module, partition security level is defined in the virtual processor, etc. To model an ARINC 653 operating system, we use the AADL 653 annex, and the method described in this article [7].

Each hardware component is modeled with the AADL corresponding component, or with the device element. Some more complex components, like network, are modeled as a sub-system containing other components. We first model hardware component as a pseudo blackbox element, where behavior is not defined. We set interface information like ports, and a few properties (memory size, bus transfer latency etc.). In order to refine those hardware properties, we defined or completed some AADL property sets, by introducing behavior and specific properties like cache hit rate for cache component, or refresh time for DRAM component.

The user models the avionics system with the AADL, and choose which viewpoint(s) will be set when analyzing the platform. Viewpoints can be for example timing performance, power consumption etc., and enable the platform investigation under different angles. We then parse the AADL model to retrieve properties, connections and deployment information, that are configuration information for the next step.

4.4. Platform integration by generation

We developed a database of configurable systemC components. They are automata which can be configured with properties and viewpoints. For example, on the figure 5, the automata of the DRAM was configured according to the timing viewpoint, and then with timing properties (tRefresh = refresh latency etc.).

In order to generate the refined executive platform, we retrieve from the AADL model the different hardware properties and connections, configure the corresponding systemC element, and connect them.

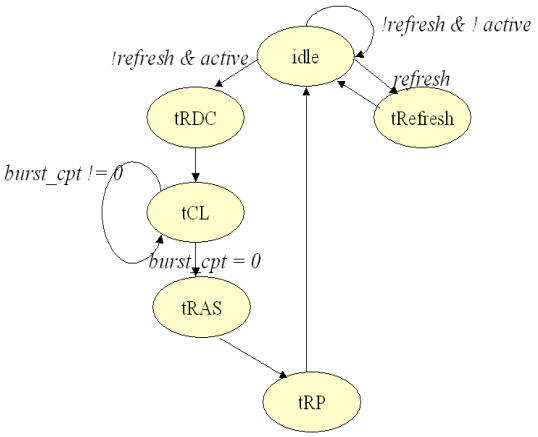

Figure 6. Dram automata example with timing annotations

5. Platform Simulation and Results

Currently, this is a work in progress. However, we have already encouraging results. The AADL part of the process has been specified and is under development, while systemC main hardware have been developed (behaviour, main properties and communication interface), as the first blocks of a future model of an ARINC653 OS (scheduler, timer, switch partition and context). In order to test and refine these elements, we also developed a minimum hardware platform and a basic scenario that simulates it in order to test and refine the elements behaviour and the platform communications.

To test our future platform, we defined a simulation and performances analysis method: to see if the hardware platform built is compliant with the application requirements, we will perform a simulation using systemC kernel, which will take the generated platform, the viewpoint(s) set, and will apply the simulation(s) scenario(s). The user will be able to analyze the platform performances by examining performances graphs and simulation traces. If the execution platform is not compliant with the applications requirements, we then will investigate and try to refine or modify one or some components.

6. Conclusion

Current early platform validation methods center on software modeling, regarding the hardware as blackbox components which can't be dynamically simulated. We have proposed a new early validation approach, that aims to model a complete avionics platform, software and hardware. Our method should automatically generates hardware and simulation scenario. It should also enables a dynamic simulation of the platform, and analyzes its performances according different viewpoints (timing, power consumption or safety). It takes advantages from the AADL, particularly adapted for software architecture modeling, and from systemC, industrial standard for hardware architecture description.

To validate the accuracy of our modelling methodology, we will first model electronic evaluation boards. We will afterwards model a complete IMA platform to compare the model performances with the experimental results. Otherwise, we will connect with existing model-driven engineering methods and improve the platform development process.

References

[1] J.A. Estefan. "Survey of model-based systems engineering (MBSE) methodologies". Technical report, *INCOSE MBSE Focus Group*, may 2007

[2] Bernhard Schätz, Manfred Broy, Franz Huber, Jan Philipps, Wolfgang Prenninger, Alexander Pretschner, Bernhard Rumpe, "Model-Based Software and Systems Development – a white paper", 2004

[3] Support for Predictable Integration of mission Critical Embedded Systems project (SPICES), 2009 http://www.spices-itea.org

[4] Model-Based Approach for Real-Time Embedded Systems development project (MARTES), 2007. http://www.martes-itea.org/

[5] C. Brunette, R. Delamare, A. Gamatié, T. Gautier, J-P. Talpin, "A Modeling Paradigm for Integrated Modular Avionics Design", IRISA report, 2005.

[6] P. Dissaux, F. Signhoff, "the AADL as a Pivot Language for Analyzing Performances of Real Time Architectures", *4th European Congress ERTS Embedded Real Time Software*, 2008.

[7] J. Delange, L. Pautet, A. Plantec, M. Kerboeuf, F. Singhoff, F. Kordon, "Validate, Simulate and Implement ARINC653 systems using the AADL", *CM SIGAda Ada Letters*, 2009.

[8] AADL Portal at Telecom Paristech : http://aadl.telecom-paristech.fr/

[9] Open SystemC Initiative. IEEE 1666: SystemC Language Reference Manual, 2005. www.systemc.org.

[10] L. Cai and D. Gajski, "Transaction Level Modeling: An Overview", *Center for Embedded Computer Systems, University of California*, 2003