Model-driven Development and Simulation of Integrated Modular Avionics (IMA) Architectures

Björn Annighöfer

Institute of Aircraft Systems, University of Stuttgart; *bjoern.annighoefer@ils.uni-stuttgart.de*

SNE 28(2), 2018, 61 - 66, DOI: 10.11128/sne.28.tn.10414
Received: April 10, 2018 (Selected ASIM GMMS/STS 2018 Postconf. Publ.), Accepted: May 15, 2018
SNE - Simulation Notes Europe, ARGESIM Publisher Vienna, ISSN Print 2305-9974, Online 2306-0271, www.sne-journal.org

Abstract. Presented is a model-driven development method for avionics systems comprising of a domain-specific model, mathematical optimization, and an attached network simulation. For Integrated Modular Avionics (IMA) the degree of freedom in choosing the system architecture is so high that determining the optimum by hand is hardly possible for large aircraft. A domain-specific model was created with the Eclipse Modeling Framework (EMF) holding system requirements and architecture variants, such that it can automatically be validated and evaluated. Moreover, combinatorial optimization is used to determine optimal architectures by algorithm for single and multiple objectives. Optimization on civil aircraft and a space launcher revealed improvements of up to 30% in single design objectives. Moreover, the architecture model can automatically be converted in configuration stubs and an AFDX network simulation.

Introduction

Integrated Modular Avionics (IMA) are state-of-the-art for large civil and military aircraft. The concept of IMA is that computing, memory, and IO resources are shared between several safety-critical and non-critical system functions. System functions are, for instance, cabin pressure control and landing gear retraction. Those are loaded as segregated software partitions. The major portion of the avionics system's hardware is standardized. Computing and IO modules are configured in software to fulfil their purpose in multiple system functions.

IMA reduced the hardware, cost, weight, and space[1,2]. Two challenges arising are first design freedom and second the massive number of configuration parameters. Considering design, questions like, what is the lightest architecture, how many modules do I need, or what is the optimal IO distribution per module, can no more be answered optimally by hand. Considering configuration, current IMA systems require so many parameters that the process became inefficient and error-prone [3, 4, 5].

To improve the situation, a domain-specific model was developed for computer-aided design of avionics architectures. It holds the complete architectures, but in addition the system requirements, like functions, resource needs, and safety constraints. Architectures shall be automatically validated, evaluated and compared. It is generic in terms that no precise avionics technology or function architectures are predefined.

Requiring only a minimum mandatory information makes it applicable to the design phase. Nevertheless, it can be automatically converted to mathematical optimization problems. For instance, function assignment, routing, and module sizing. Moreover, the model is used to derive configuration stubs or simulations automatically. All is implemented in a seamless model-driven IMA design method depicted in Figure 1.

The remainder of this article is organized as follows. Section 1 introduces the domain-specific avionic architecture model. Section 2 explains a multi-objective optimization approach and Section 3 shows an example of an AFDX simulation derived from the model. The article ends with a conclusion and outlook.

1 Avionics Modeling

A standardized avionics system is a distributed computing platform, which provides computing resources and I/Os to aircraft system functions running as software.

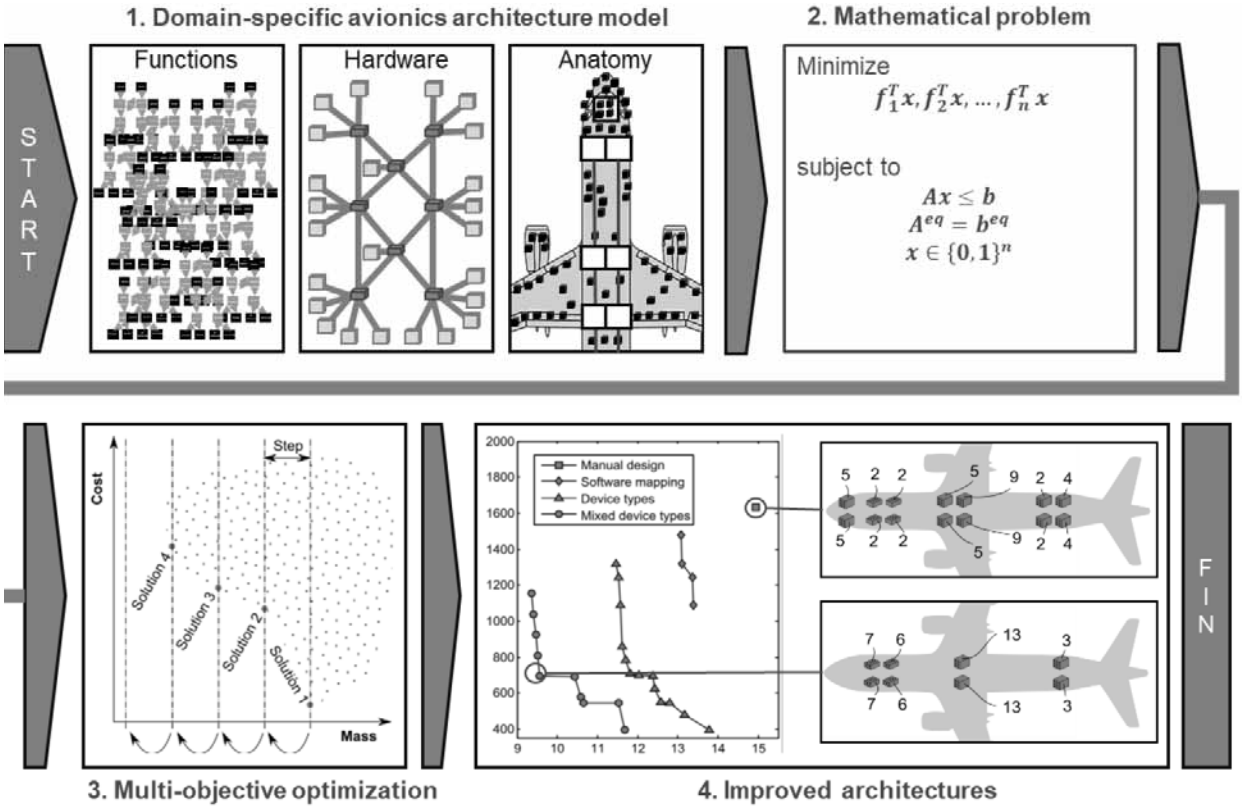

Figure 1. Workflow of the model-driven IMA architecture design method.

Since the number of different module types is kept as small as possible in order to reduce development costs, there is usually a large number of equal hardware modules, which could technically host the same functions. Whether a certain module should host a function or not, often depends on the position, the distance to required sensors and actuators, and safety consideration. Overall, there is a high degree of freedom, e.g., the dimension of modules, the installation locations, the assignment of functions, the network topology, and the routing of signals.

A domain-specific model was developed especially for the purpose of avionics system design. Therefore, it generically captures the capabilities and resources of hardware, without requiring a certain module type or technology. Moreover, it generically covers the software as atomic building blocks, so called tasks, signals, and their resource requirements. In addition, segregation, symmetry, location, and power constraints can be attached to single tasks or task groups.

The first instance of an avionics architecture model [6] was used in several air and space research programs. At the end of 2017, a second generation of the domain-specific model has been finalized [7], which is more, considering bus systems and hardware. It has been made available as Open Source as the **Open Avionics Architecture Model (OAAM)** [8].

OAAM is designed in nine almost independent layers. This matches the concurrent development process of IMA systems, which is distributed over multiple parties. The nine layers are Library, Scenarios, Systems, Functions, Hardware, Anatomy, Capabilities, Restrictions, Mapping.

The four main layers are Function, Hardware, Anatomy, and Mapping.

The **Functions layer** holds all tasks to be assigned to the avionics system and the signals that must be routed. In addition, it includes timing and safety constraints.

The **Hardware layer** allows modelling device instances and interconnection topologies without physical dimensions.

Within the **Anatomy layer**, the installation locations and cable routes of the aircraft including the length and positions are modelled. It is a graph-like representation of a simplified 3-dimensional construction plan.

In the **Mappings layer** assignment objects can be created, which assign task to devices, devices to installation locations, as well as signals and cables. The basic constraint determining if an assignment is valid or not, is a linear **resource provisioning and resource consumption model**. Each task or device K_i requires a set of certain resources r^{K_i} in a certain amount $r_j^{K_i} \in \mathbb{R}_+$, i.e.

$$r^{K_i} = (r_1^{K_i}, \ldots, r_n^{K_i}) \tag{1}$$

A resource is an abstract unit of what has to be provided to the task or device and is consumable, e.g. computational power, memory or space in installation location. Devices and locations provide resources. An assignment to D_j is valid as long as the available resources r^{D_j} are not exceeded, i.e.

$$\sum_{K_i \subseteq K} r^{K_i} \leq r^{D_j}. \tag{2}$$

This must hold for all assignments of the architecture. In addition, constraints on devise, locations, power sources, areas, symmetries, and co-locationing control what are valid mappings. Multiple mapping variants of the same elements can be created to represent different architecture variants. For each variant the validity and design objectives are individually be calculated.

Technically OAAM is realized with the ECORE **metamodeling** language of the Eclipse Modelling Framework (EMF) [9], which allows to define formal UMLlike meta-models and automatically derive the implementation, persistence layer, and edit tools. Moreover, extensions exist for the verification and evaluation of EMF derived domain-specific models. OAAM models are edited, validated, and evaluated within a specialized Eclipse instance. In addition, an interface to MATLAB was developed.

2 Avionics Architecture Optimization

The IMA systems of current aircraft have more than 4000 tasks and peripheral as well as more than 50 devices and thousands of possible installation locations and cable routings. The pure number of elements prevents that the design engineer is able to derive the optimal dimensioning, installation, and software assignment by hand. Even if he would do, he would not be able to prove it. Moreover, the optimality of an avionics system is not a unique property.

There are multiple design objectives desired in the design process. Simple examples are cost and mass, which shall both be minimal. Objectives that are more complex are installation cost or maintenance effort. There is usually not a single architecture optimizing all objectives, but objectives are partially contradictional, such that the best tradeoff is desired.

In order to automate and prove the optimality of recurring design tasks, a link between the architecture model and combinatorial optimization was developed. Overall, **eight generic optimization routines** for avionics architecture were developed as depicted in Figure 2. Optimization routines are classified in 1-level , 2-level, and 3-level assignments depending on how many different element types are assigned in parallel.

The most basic optimization routine is **function assignment**, which assigns a set of functions to a given set of devices. The optimization objectives are, for instance, the device weight (i.e. use a few devices as possible) or the wire weight (i.e. put tasks as close as possible to the related sensors and actuators). In addition, all defined constraints for the tasks must hold.

A more complex routine is **device type optimization**, which starts from an empty topology and selects the optimal number, installation, and dimensioning of devices, while assigning tasks.

3-level assignment is the highest level of automation. It derives devices and network, as well as task assignment and signal routing in a single step. It has the highest degree of freedom and it was shown in [10] that in general this leads to the highest optimization potential. However, also the complexity of the optimization problem rises, such that this is only feasible to calculate single systems with no more than 50 tasks. 1-level and 2-level assignments are feasible for scenarios with 4000 and more tasks. Calculation times are between several hours and four weeks.

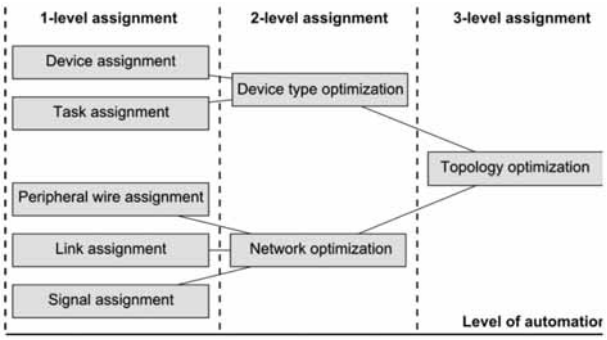

Figure 2. Eight generic optimization routines for avionics

architecture optimization.

Technically, all optimization routines are implemented as multi-objective binary programs (BP). A multi-objective BP searches for a binary solution vector x such that

$$f_1^T x, f_2^T x, \dots, f_n^T x \tag{3}$$

are minimized subject to

$$\begin{aligned} Ax &\leq b \\ A^{eq} &= b^{eq} \\ x &\in \{0,1\}^n. \end{aligned} \tag{4}$$

x encodes - depending on the optimization tine - either task assignments, signal routes, or hardware topologies. The cost vectors $f_1, \dots, f_n$ allow for linear objectives. With the addition of auxiliary variables, also non-linearities can be included. Linear inequalities A and equalities A^{eq} constrain the resource consumption, enforce segregation, and ensure a unique mapping of every object. For more information on the optimization problem formulations please refer to [11, 12, 13, 14, 15].

Information from the domain-specific model are automatically converted to the mathematical optimization problems. The conversion and the invocation of combinatorial optimization are implemented in MATLAB. Commercial-of-the-shelf solvers [16] for Mixed Integer Linear Programming (MILP) are integrated to efficiently solve BPs. The solution is converted back in meaningful model information. In case of multiple-contradicting objectives, a custom iterative multi-objective solver calculates the so-called Pareto optimum [17], i.e. the set of best possible trade-off solutions. See an example of a Pareto optimum in Figure 1 on the lower right.

Several of the optimization routines were used to derive an optimal avionics system for the ARIANE 5 space launcher [18, 19] and for deriving some general scaling laws for avionics architecture design [20]. In all application, up to 30% improvements in single objectives between the manually derived architectures and the optimized architectures were found.

3 AFDX Simulation

The avionics architecture model presented above is static. For design, validation, and evaluation static properties and capacities are assumed that have sufficient safety margins such that the architecture should also be valid during operation. For instance, during architecture design the CPU is modelled as a static resource, i.e. the percentage of CPU load consumed by each task vs. 100% available CPU. For sure, however, within the real system, most tasks are periodic and must be scheduled and the schedule must be valid in terms of the individual deadlines. This can be considered by assuming, for instance, a maximum load of 70% during design. Nevertheless, real schedules have to be determined and their correctness has to be proven.

Another example is the network. During design and optimization, bandwidth is assumed as a single consumable resource. Each signal has a bandwidth portion it consumes. However, in the real systems, network messages have to be scheduled and maximum transfer delays have to be proven. Typically, this happens with network calcus or network simulations.

A common network for IMA systems is **Avionics Full Duplex Switched Ethernet (AFDX)**. AFDX is an asynchronous switched network based on Ethernet. AFDX messages address virtual links (VL) instead of target devices.

VLs are preconfigured and static communication routes in the network. Each VL is assigned a maximum bandwidth (Bandwidth Allocation Gap - BAG). If too large or too many messages are sent, those are dropped by the switches. An AFDX network and the corresponding message routing is assumed to be valid if for each VL a valid BAG can be found and if under worst case conditions the maximum delay and jitter requirements for each signal are met. It is common to validate this in simulations.

Worst case conditions, delay, and jitter measurements cannot be made in the OAAM model. However, a simulation framework for AFDX was developed within MATLAB/SIMULINK. The simulation framework provides basic components for switches, as well as the AFDX send and receive interfaces of avionics devices. In addition, it is able to run on a rapid prototyping system, which is able to output the virtual communication on a real AFDX interface.

All components of the simulation framework can be used to create virtual AFDX networks by hand and validating the timing of all messages up to an accuracy of 1 ms. More common is, however, to derive the AFDX simulation model automatically from the OAAM static architecture, which – if a mapping was calculated – includes all necessary information. Figure 3 shows an example of an Airbus A380-like AFDX network completely derived from an architecture model. It was possible to run the simulation in realtime on a rapid prototyping system.

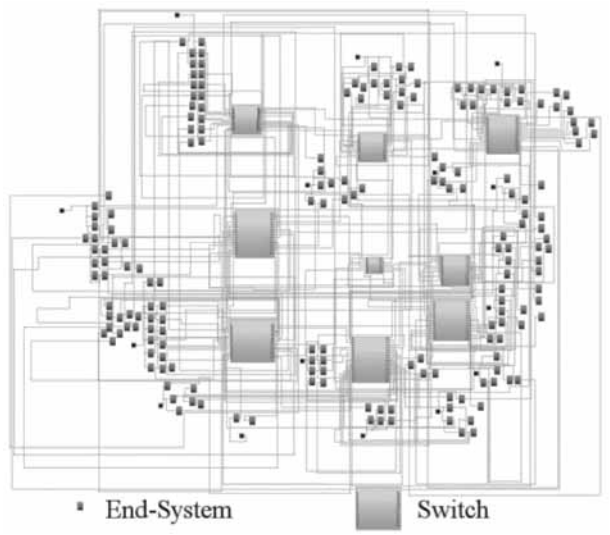

Figure 3. An A380-like AFDX system simulated in MATLAB/SIMULINK.

The detailed description of the AFDX simulation and the results can be found in [21].

4 Conclusion

Current avionics systems are generic resource-sharing distributed computing systems. The most common representatives are Integrated Modular Avionics (IMA). A domain-specific model for avionics architectures and system function requirements enables computer-aided design of avionics systems including automated validation, evaluation, and optimization. Eclipse EMF and MILP solvers are used as technologies. Optimization of real aircraft and the ARIANE 5 space launcher revealed improvements of up to 30% in single objectives as weight, compared to the manual design. The applicability of the model during system development is extended by the possibility to derive dynamic simulations, which was shown for a MATLAB/SIMULINK AFDX simulation.

In future works, it is intended to extend the capabilities of the model-driven avionics architecture tool chain by additional optimization routines. Moreover, it is worked on a self-configuring avionics system, which utilizes OAAM as its online knowledge base.

References

[1] Ramsey JW. “Integrated modular avionics: Less is more - approaches to IMA will save weight, improve reliability of A380 and B787 avionics,” Avionics magazine, 2007. [Online]. Available: http://www.aviationtoday.com/av/categories/commercial/8420.html

[2] Wilkinson C. “IMA aircraft improvements,” Aerospace and Electronic Systems Magazine, IEEE, vol. 20, no. 9, pp. 11–17, September 2005.

[3] Butz H. “Open integrated modular avionic (IMA): State of the art and future development road map at airbus deutschland,” in Proceedings of the 1st International Workshop on Aircraft System Technologies, March 2007, pp. 211–222.

[4] Watkins C, Walter R. “Transitioning from federated avionics architectures to integrated modular avionics,” in Digital Avionics Systems Conference, 2007. DASC ’07. IEEE/AIAA 26th, oct. 2007, pp. 2.A.1–1 –2.A.1–10.

[5] Wilson A, Preyssler T. “Incremental certification and integrated modular avionics,” in 27th Digital Avionics Systems Conference, 2008.

[6] Annighöfer B , Kleemann E, Thielecke F. “Model-based development of integrated modular avionics architectures on aircraft-level,” in Deutscher Luft- und Raumfahrtkongress, Bremen 27. - 29. Sept. 2011, no. 1395. Bremen: Deutsche Gesellschaft für Luft- und Raumfahrt, September 2011.

[7] Annighöfer B, Riedlinger M, Marquardt O. “How to tell configuration-free integrated modular avionics what to do?!” in 36th Digital Avionics System Conference, Saint Petersburg, FL , USA, September 2017.

[8] www.oaam.de

[9] www.eclipse.org/modeling/emf/

[10] Annighöfer B, Thielecke F. “A systems architecting framework for optimal distributed integrated modular avionics architectures,” CEAS Aeronautical Journal, pp. 1–12, 2015.

[11] Annighöfer B. “Model-based architecting and optimization of distributed integrated modular avionics,” Dissertation, Hamburg University of Technology, March 2015. ISBN: 978-3-8440-3420-2

[12] Annighöfer B, Kleemann E, Thielecke F. “Automated selection, sizing, and mapping of integrated modular avionics modules,” in 32st Digital Avionics System Conference, Syracuse, NY, USA, October 2013.

[13] Annighöfer B, Thielecke F. “Supporting the design of distributed integrated modular avionics systems with binary programming,” in Deutscher Luft- und Raumfahrtkongress, Berlin 10. - 12. Sept. 2012. Berlin: Deutsche Gesellschaft für Luft- und Raumfahrt, September 2012.

[14] Annighoefer B, Thielecke F. “Multi-objective mapping optimization for distributed modular integrated avionics”, in 31st Digital Avionics System Conference, Williamsburg, VA, USA, October 2012.

[15] IBM CPLEX or GUROBI

[16] Annighoefer B, Reif C, Thielecke F. “Network topology optimization for distributed integrated modular avionics”, in 33rd Digital Avionics System Conference, Colorado Springs, CO, USA, October 2014.

[17] Ehrgott M. Multicritia Optimization, 2, Ed. Springer Berlin Heidelberg, 2005.

[18] Annighoefer B, Nil C, Sebald J, Thielecke F. “Structured and symmetric ima architecture optimization: Use case Ariane launcher,” in 34th DASC, 2015.

[19] Annighöfer B, Çelen Nil, Sebald J, Thielecke F. “Ariane-5-based studies on optimal integrated modular avionics architectures for future launchers,” in 6th EUCASS, 2015.

[20] Annighöfer B, Posternak V, Thielecke F. “Empirical investigations on avionics scaling laws,” in 35th Digital Avionics System Conference, Sacramento,CA , USA, September 2016.

[21] Annighöfer B, Ihle H, Thielecke F. “An easy-to-use real-time AFDX simulation framework,” in 35th Digital Avionics System Conference, Sacramento,CA , USA, September 2016.