MODEL-BASED CONTROL LOGICS DEVELOPMENT OF AIRCRAFT SYSTEMS

Viviam Lawrence Takase*, Humberto Hayashi Sano**
***EMBRAER S.A., Gavião Peixoto, Brazil, **EMBRAER S.A., São José dos Campos, Brazil**

Keywords: *Model-based development, Control logics, Modeling and simulation, Aircraft systems*

Abstract

The objective of this paper is to present an overview of the model-based process to develop aircraft systems control logics that has been applied in the latest Embraer aircraft developments. The process follows a Systems Engineering approach, enabling to perform continuous verification and validation since the early phases of the development. As a result, the aircraft systems present a higher level of maturity at the beginning of the test campaigns, reducing costs and development time.

1 Introduction

Modeling and simulation is used in aircraft systems development to assess system behavior in terms of performance and functionality before the real system exists, and afterwards as a cheaper alternative to ground and flight tests. As the complexity of the aircraft systems increases, modeling and simulation become essential in the development process.

Moreover, an increased amount of system analysis is required for systems where the aircraft manufacturer is responsible for the system. The same applies when the aircraft manufacturer is responsible for the control logics definition in the implementation level.

The main motivation to apply a model-based process in control logics development is to identify and correct defects in the earlier phases of the development. This brings many benefits, as the cost to correct defects is much higher in the later phases of the development [1].

The model-based process uses virtual test environments that enable execution of tests since the conception phase, enabling execution of extensive number of tests that would be more expensive and time consuming to perform in system rigs and test aircrafts. As a result, the aircraft systems present a higher level of maturity at the beginning of the test campaigns, reducing the number of tests that are conducted in expensive and disputed test environments.

Embraer has increasingly used modeling and simulation for systems development along the years, and the increasing level of complexity found in the latest aircraft developments has driven the need to apply a model-based process for control logics development.

1.1 Systems Development Process

Nowadays the development of aircraft systems follows well-established processes based on requirements and needs given by the airworthiness authorities, as well as internationally recognized standards such as the ARP 4754 [11] and *Systems Engineering* philosophy and principles [5].

A typical model of aircraft system life cycle as described in [8] is depicted in Figure 1.

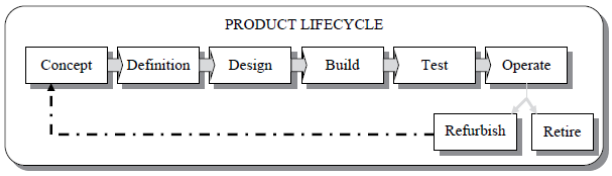

Fig. 1 – Aircraft system life cycle [8].

The cycle begins with the *Concept phase*, with the gathering of the user, customer and other

stakeholders needs. In this phase the primary role and functions of the required system are established, together with desired performance and market drivers. A conceptual system solution is created.

The next stage is the *Definition phase*, with the definition of a solution that is feasible to design and manufacture. System architectures and configurations are studied considering aspects such as safety, functional and operational needs, performance, physical and installation characteristics, interface requirements, qualification and certification requirements.

In the *Design phase* all definitions made in the previous phase are refined, with the chosen architecture being detailed to the point that it can be implemented and manufactured. The design of equipment and components is usually made by the suppliers. Test, qualification and certification processes are defined and agreed.

The *Build* and *Test* phases correspond to the latest steps of a system development process, ending with the certification of the aircraft or equipment.

Another way to represent the development life cycle is the *V-model* proposed by Forsberg and Mooz [3]. In this model the development activities evolve downwards the left leg of the *V*, following a hierarchical structure of specifications from stage to stage to more detailed levels. For control logics development, which is subject of this work, the bottom of the *V* is either the software code or the hardware.

In the same way, a step-wise integration and verification by testing evolves upwards the right leg of the *V*. In the model-based development it is possible to test the solution in distinct phases of the development, anticipating the physical tests, and the *V-model* takes the form shown in Figure 2.

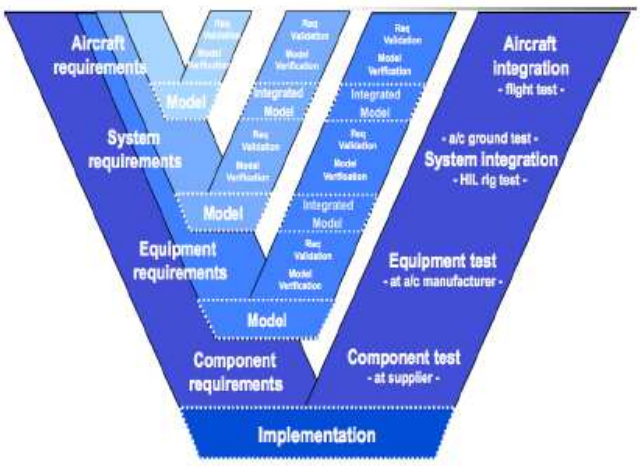

Fig. 2 – *V-model* of a model-based development process [4].

2 Model-Based Development Process

In this section, the model-based process which is applied from preliminary to detailed system development phases will be presented, describing the characteristics of the models, simulations and the main results that are obtained for each phase.

Depending on the complexity of the system, it may be necessary to divide it into subsystems, for which the same process is applied. Various levels of integration are then created to test the control logics in the component, subsystem or system level, depending on the test needs.

2.1 Preliminary Development Phases

The initial stages of the development consist in defining the physical and the logical system architectures. Design drivers for defining system architectures are performance, efficiency, safety, reliability, operational requirements and requirements from other systems in terms of energy and communication.

Physical architecture is a representation that shows the breakdown of the system into physical components (such as actuators, pumps, generators, valves) and the physical connections between them. Figure 3 shows an example of fuel system physical architecture for the first generation of Embraer E-Jets [6].

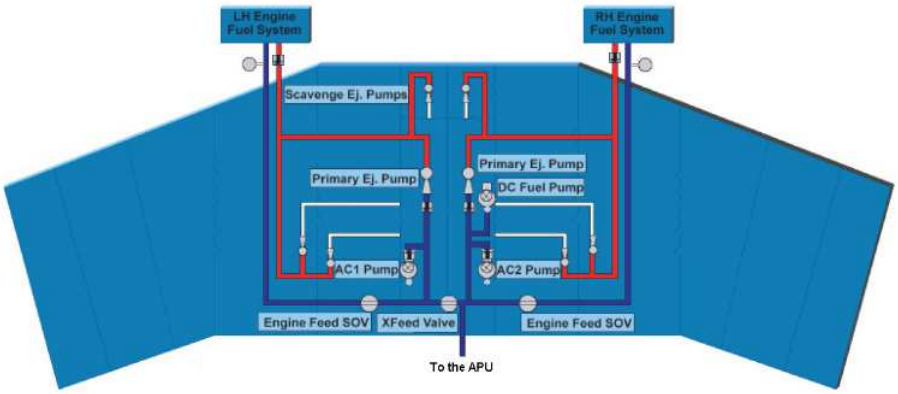

Fig. 3 - Embraer E170/E190 fuel system architecture (adapted from [6]).

The *logical architecture* describes the system in terms of its functions and the logical relationship between them.

Simulation models representing the physical architecture integrated with the first logics that define the functional behavior are built at these preliminary phases. The process is iterative, in the sense that the models are used to perform analyses that help to define the system architectures.

The physical part of the system, commonly denominated as system plant, is modeled with component models that represent physical behaviors such as energy and mass conservation. The plant interacts with the control logics through actuation and feedback signals (such as voltage, pressure, current, temperature) which are also represented in this model.

In the preliminary phases the functional behavior of the system is modeled as simple logics, not considering details such as signal redundancies, consolidations, etc. These aspects are modeled in later phases with the definition of the computational architecture of the system. The way how these logics are organized in the model may facilitate reuse in the later phases. For example, grouping logics in a set that controls or monitors a physical component, and creating another set that contains the logics associated with the system operating modes. Both inputs and outputs of these logics represent flux of information in a high level of abstraction.

Embraer's process for model-based development of control logics is based on *MATLAB/Simulink* [7]. Models developed in other modeling tools can be integrated into the *MATLAB/Simulink* environment by code generation. Commonly used *Simulink* toolboxes for modeling control logics are *Stateflow* and *SimPowerSystems* [7].

Several tools that support this process have been developed in-house: libraries of component models for reuse, fault modeling tools, tools for integration of models, and tools for large-scale testing and simulation.

An example of control logics in the preliminary phase is shown in Figure 4. Figure 5 shows a system model integrating logics and plant also in a preliminary phase.

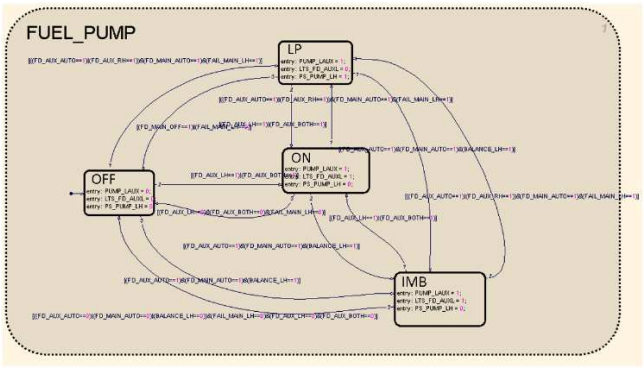

Fig. 4 – Example of fuel pump control logics in the preliminary phase.

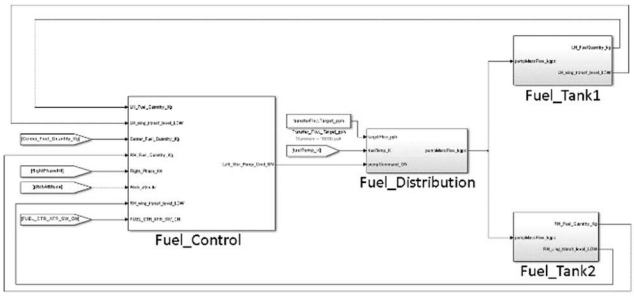

Fig.5 – Example of fuel system model integrating plant and preliminary control logics.

Once the system preliminary model has been developed, simulations are performed to verify that the simulation is reflecting the operating modes of the system, normal and abnormal scenarios, and to check if the high-level system requirements (HLR) are being met through the analysis of results.

In a first moment, execution of tests manually is more efficient than by automation, especially with the use of models representing the human-machine interfaces (HMI) and its symbologies. For this, HMI models representing panels and the pictorial system are integrated with the system model, thus facilitating user interaction during the simulation and interpretation of the results.

Embraer uses the commercial tool *VAPS XT* [10] to create symbology models, allowing the generation of graphical interfaces that are very representative of the HMI interfaces of the aircraft. Models developed with *VAPS XT* can be integrated with *Simulink* models through code generation.

Another result obtained by using representative models of the HMI at the early development phases is the validation of the HMI definitions with the customer, by actually interacting with the system. Figure 6 shows an example of fuel system model panel, and Figure 7 shows a fuel system model synoptic, both representations of the system described in [6].

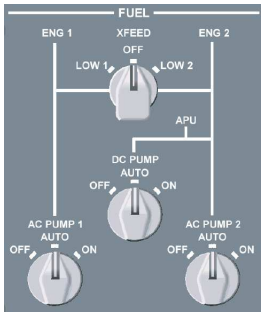

Fig. 6 – Embraer E170/E190 fuel system model panel (adapted from [6]).

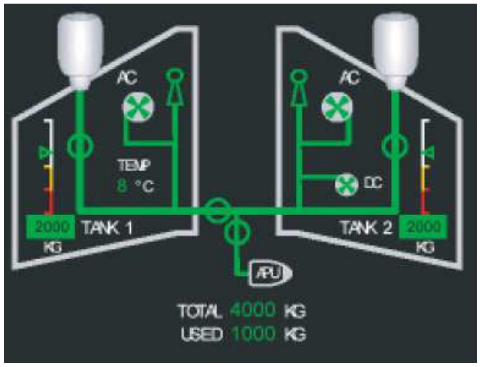

Fig. 7 - Embraer E170/E190 fuel system model synoptic (adapted from [6]).

Once the scenarios have been manually tested and the results are correct, test cases are created so that they can be automatically reproduced to verify that the system meets the requirements. For this, it is necessary that the test cases have the values of both the inputs and the expected results. In addition, creating a link between these test cases and the system requirements in a systematic way helps in the verification and validation activities of the system development.

In summary, the use of modeling and simulation in the preliminary phases brings the following results:

- Deeper understanding of the system;
- Better knowledge of the system necessities related to intra-system information (physical and logical architecture) and inter-system information (system and other systems/aircraft);
- Definition of a system architecture that has been tested against the high-level requirements;
- Creation of a set of test cases that will evolve during the system development.

Figure 8 shows the developed artifacts (requirements, models and test cases) and the relationship between them in the process.

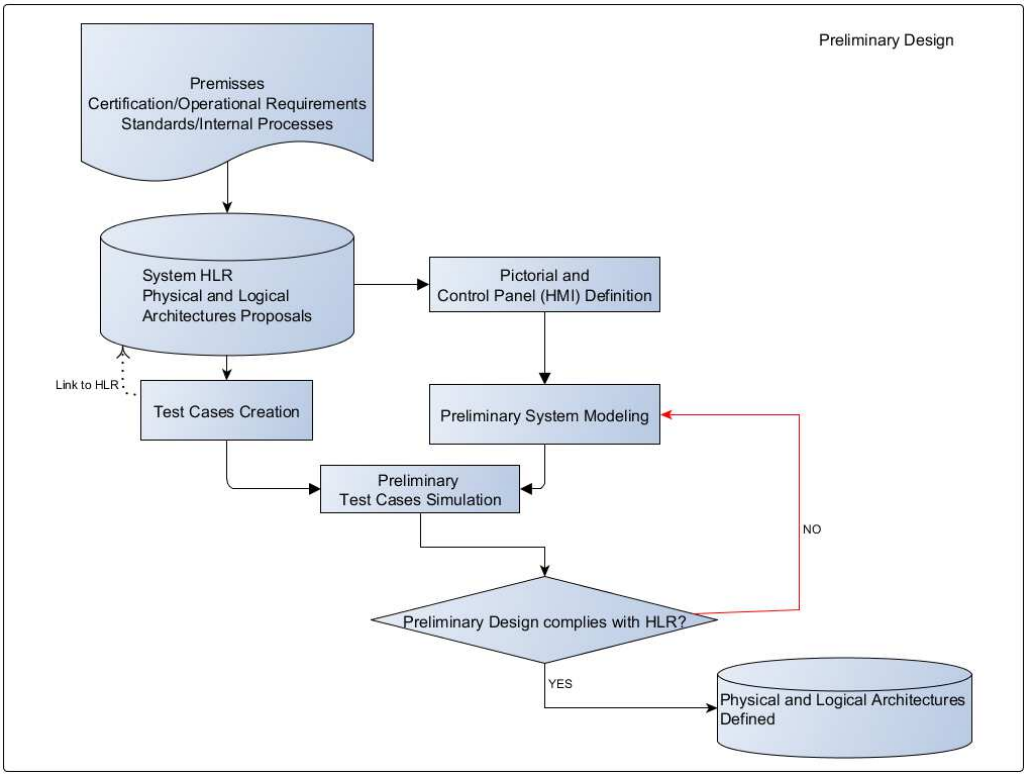

Fig. 8 – Flowchart of the model-based process in the preliminary phases.

2.2 Detailed Development Phases

It is in the detailed development phases that the computational architecture is defined, with definition of types and number of line replaceable units (LRUs), types and number of buses for communication, electrical circuits, and for each LRU number of channels, types and number of interfaces (analog, digital, discrete, etc.). Figure 9 illustrates the computational architecture for the example case of [6].

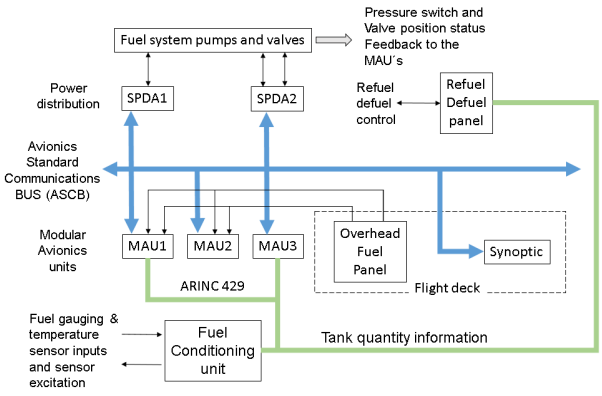

Fig. 9 – Computational architecture of the Embraer E170/E190 fuel system (adapted from [6]).

The preliminary control logics defined earlier in the process are at this point allocated in the corresponding hardware, and afterwards they are further detailed. Figure 10 shows the allocation of one function of the E-Jets fuel system example (Command AC Pump 1) in the computational architecture.

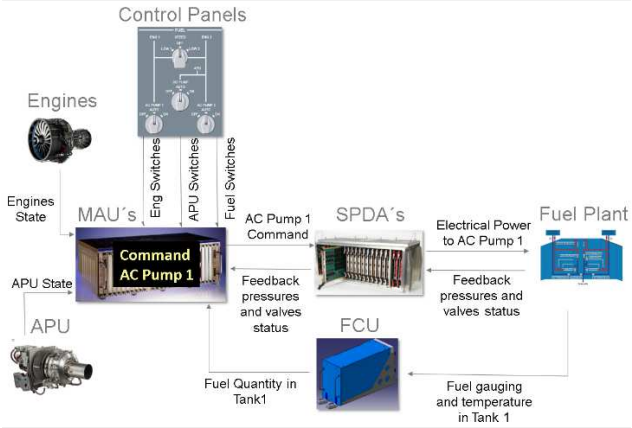

Fig. 10 – Example of allocation of one fuel system function in the computational architecture.

The detailing of the logics involves changes in the basic elements that comprise the logics due to the allocation itself (e.g. split of equations), changes due to the type of implementation (e.g. logics implemented as hardware or software) and consolidation of inputs/outputs.

In cases where the logics are implemented as hardware, for example as relay logics, a physical model in the electrical domain representing the relays is created using components from the electrical library that has been built in-house by Embraer.

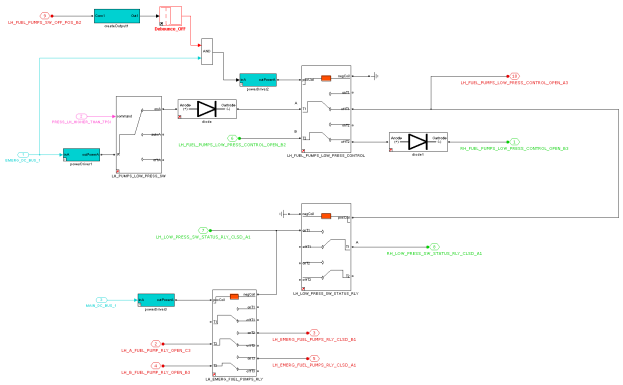

Fig. 11 – Example model of a relay logics.

With such a model additional tests can be performed, such as relay failure analysis to verify that the function that is lost in the system is not in disagreement with the safety requirements. These relay logics models can even serve as background documentation (principle diagrams) for electrical designers to develop the electrical diagrams of the system.

When control logics are implemented as software, the modeling domain is restricted to the discrete domain, and the simulation steps are counted in cycles instead of time. There is also a limitation on the types of blocks that are used in the logics constructs, respecting the modeling domain and characteristics of a code that is executed in cycles. There is also a more rigorous treatment of the data types in these models. Figure 12 shows an example from [2] of a function that is modeled in the continuous domain, which is a characteristic of the preliminary models, and one that is modeled in the discrete domain to represent an implementation in software.

Normally the consolidation of inputs/outputs is modeled separately from the logics that implements the system functions. This organization of the model allows the complete reuse of the logics that have been developed in the preliminary phases. An example model of a control software illustrating this structure is shown in Figure 13.

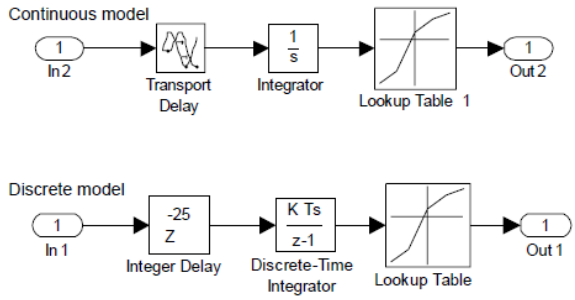

Fig. 12 – Example of control logics construction in preliminary and detailed phases [2].

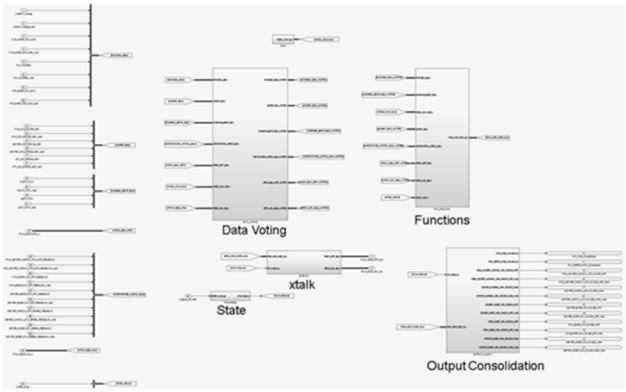

Fig. 13 – Example showing the structure of a software model.

In this model, information of validity associated with each input signal is added. These validities indicate to the software application if the information consumed is indeed valid.

Once the allocation of the logics into the controllers has been done including the validity of the signals, additional tests are created to verify the robustness of the allocation in terms of safety, that is, to ensure that the proposed solution complies with the functions availability requirements. Some of the scenarios exercised in the tests are:

- Loss of communication buses or failure to update the values to be transmitted in digital communication: one way of representing these events in the model is invalidating the signals through the information validity signals.

- Loss of controller channels: all outputs of the failed channel are invalidated. In the case of discrete and analog signals, it is assumed a default value.

- Loss of controller electrical power supply: tests are made considering loss of the electrical buses, encompassing simple and combined failure scenarios. In this case, the controller

model behaves reflecting the values that the analog and discrete outputs assume with the fault. For digital signals, the validity signal is set to FALSE.

Figure 14 shows a complete system model in the detailed phases, integrating the plant model, the software model, the relay logics and HMI models.

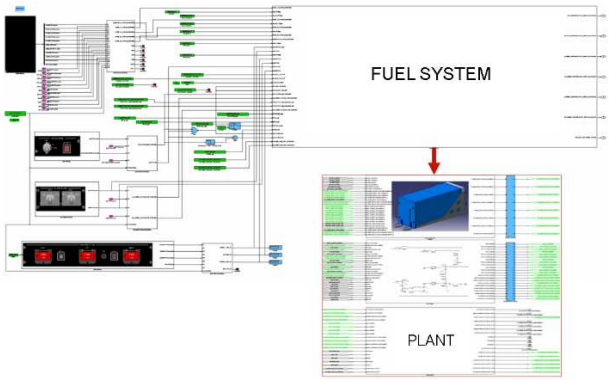

Fig. 14 - Complete system model in the detailed phases.

Once the detailed models are built, tests developed in the preliminary phases are reapplied using the detailed model, to verify if during the process of detailing the control system, errors have not been introduced and compromised the correct behavior of the system. These tests must be adapted, because they were previously based only on information, and now they must consider redundancies and validity of the signals.

The main deliverables of the model-based process in the detailed development phases are:

- Deeper understanding of the system;
- Validation of the system interface requirements;
- Definition of a computational system architecture that has been tested against the system high-level requirements;
- Definition of the system low-level requirements allocated to software and hardware, that have been tested against the system high-level requirements;
- Evolution of the set of test cases that has been first created at the preliminary phases.

Figure 15 shows the flowchart of the complete model-based process with the corresponding artifacts that are generated.

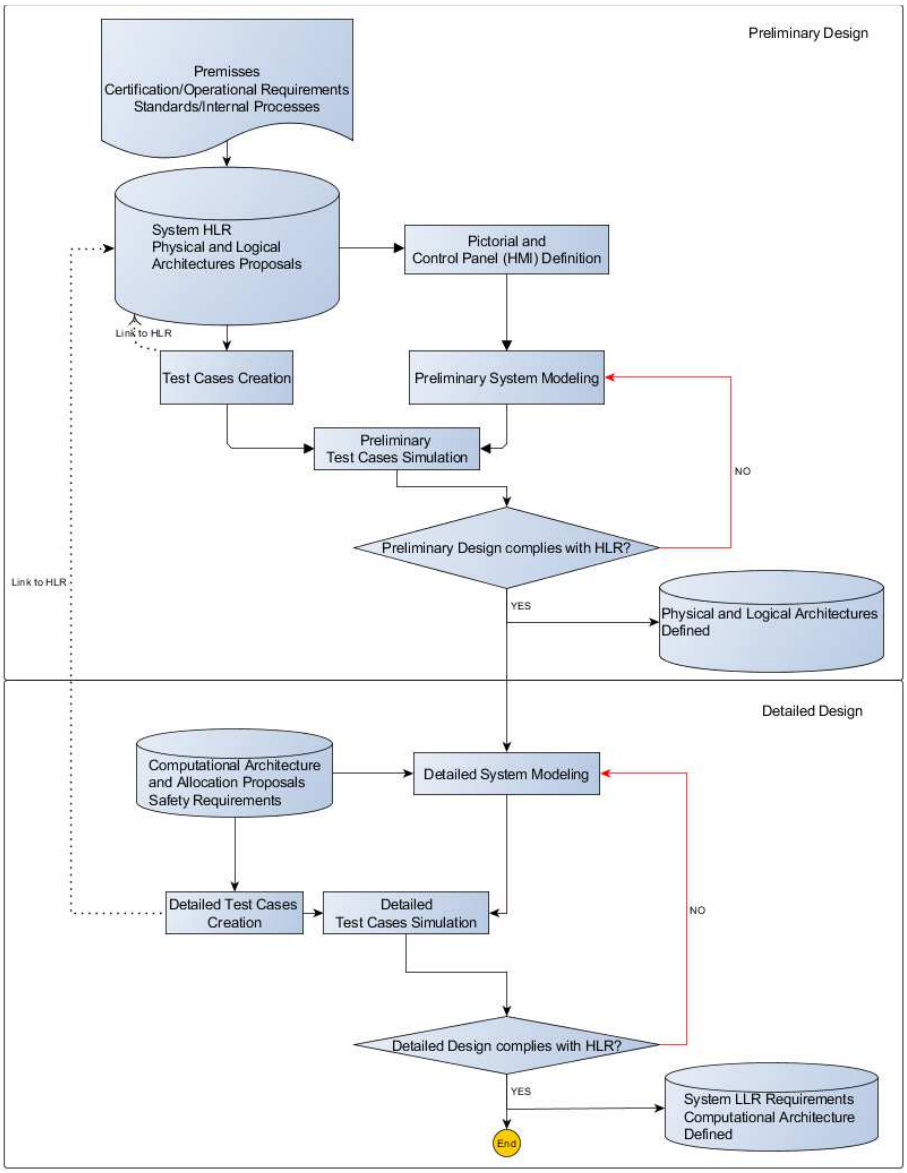

Fig. 15 – Flowchart of the complete model-based process.

2.3 Physical Test Phases

In more advanced phases of the system development, more specifically during the physical test campaigns, the models can be reused in the test environments (rigs) replacing real parts, to aid in fault detection tests, power up sequence tests, and other test campaign activities.

Another application in advanced phases is the reuse of the detailed tests to be executed in the systems rigs. For this, an adaptation is needed to map the model signals to the actual interfaces of the system in the rig.

3 Examples of Application

An example of application is seen in references [9] and [12], that show the results achieved in the development of the Legacy 500 flight control system.

The main goal of this project was to deliver low-level requirements for the supplier who would develop the flight controls system (FCS) software.

A detailed model of the FCS including the aircraft dynamics has been created, comprising more than a million blocks and dozens of

components, many with more than 700 inputs and 500 outputs.

Simulations were executed intensively, with workstations running more than 1500 test cases continuously, aiming to achieve full test coverage as needed to develop a Level-A software.

Compared to the traditional development approach, the development time has been shortened by at least six months, due to the increased efficiency of the model-based process, and the higher maturity of the requirements delivered to the supplier. More requirements have been produced (twice as many when compared to the traditional approach) with fewer issues per requirement (50 times less), as shown in Figure 16.

Process	Number of Requirements	New Requirements	Number of Issues
Model-Based	x	82.5%	0.5%
Traditional	0.5x	6.9%	20.8%

Fig. 16 – Demonstration of results for the Legacy 500 flight controls system [12].

Results from this work also showed that for the model-based process the longest delay in the development due to requirements issues was one day instead of two weeks for a traditional process.

Another example of application in which significant benefits of using the model-based process have been demonstrated was an electrical system case. As a result of the modeling and simulation work, the number of discrete and analog interfaces of the controllers was corrected to twice the initial estimated number of interfaces. This correction was done before the requirements baseline was issued, avoiding costs due to requirements changes.

3 Conclusions

This process has been applied in the latest Embraer aircraft developments in diverse ways, considering the particularities of each system development. The most important factor that tailors this process to a given system is the way of working with the supplier dictated by the contract, concerning the level of responsibility of each of the parts in the system, equipment or software development.

System complexity is also a key factor that raises the need to use this process. Organizational and cultural aspects are a challenge when implementing a model-based process that requires more engineering resources at the preliminary and detailed development phases.

Results have been demonstrated with higher level of maturity of the systems that applied this process and smoother physical test campaigns.

It is fundamental that the model development is driven by the use cases and analyses that have to be performed in the systems development. The level of detail of the models should evolve according to the system development phases, avoiding unnecessary work and resources.

Besides the models, test cases are important artifacts in the process. With this methodology there is a guarantee that continuous verification and validation is made since the earliest phases of the development.

References

[1] Andersson H. *Aircraft systems modeling - model based systems engineering in avionics design and aircraft simulation*. Master Thesis - Linköping University. Linköping, Sweden, 2009.

[2] Andersson H, Weitman A and Ölvander J. Simulink as a core tool in the development of next generation Gripen. *Proceedings of Nordic MATLAB User Conference 2008*, Stockholm, Sweden, 2008.

[3] Forsberg K and Mooz H. The relationship of Systems Engineering to the project cycle. *Engineering Management Journal*, v. 4, n. 3, p. 36- 43, 1992.

[4] Hernandes G D, Bezerra J de M, Hirata C M, Starr R R. Towards a workflow to support the integration of aircraft systems' models. *2013 IEEE/AIAA 32nd Digital Avionics Systems Conference (DASC)*, East Syracuse, USA, 2013.

[5] INCOSE, *Systems Engineering vision 2020*, Document No. INCOSE-TP-2004-004-02, Version 2.03, September 2007.

[6] Langton R, Clark C, Hewitt M, Richards L. *Aircraft fuel systems*. 1st edition, John Wiley, 2009.

[7] Mathworks, *Simulink – simulation and model-based design*. [Online] https://www.mathworks.com/products/simulink.html [Accessed 27/06/2018].

[8] Moir I and Seabridge A. *Aircraft systems: mechanical, electrical, and avionics subsystems integration.* 3rd edition, John Wiley & Sons, Ltd., 2008.

[9] Opencadd and Mathworks. *User story Embraer Legacy 500 - Embraer speeds requirements engineering and prototyping of Legacy 500 flight control system.* [Online] https://www.opencadd.com.br/user-story/embraer/index.html [Accessed 27/06/2018].

[10] Presagis, *VAPS XT.* [Online] https://www.presagis.com/en/product/vaps-xt/ [Accessed 27/06/2018].

[11] SAE Aerospace Recommended Practice, *Development of civil aircraft and systems*, SAE Standard ARP4754A, Rev. Dec. 2010.

[12] Souto R F. desenvolvimento de sistemas aeronáuticos utilizando testes e simulações. *Proceedings from 11º Simpósio SAE Brasil de Testes e Simulações na Indústria da Mobilidade.* São Paulo, Brazil, 2013.

9 Contact Author Email Address

mailto: viviam.lawrence@embraer.com.br
mailto: humberto.sano@embraer.com.br

Copyright Statement