A model-based systems engineering approach for the specification of test means for flight control computers

Hasti Jahanara

A Thesis

in the department

of

Mechanical, Industrial and Aerospace Engineering

Presented in Partial Fulfillment of the Requirements

for the Degree of Master of Applied Science at

Concordia University

Montreal, Quebec, Canada

April 2019

Concordia University

School of Graduate Studies

This is to certify that the thesis prepared

By: **Hasti Jahanara**

Entitled: **A Model-Based Systems Engineering Approach for the Specification of Test Means for Flight Control Computers**

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science (Mechanical Engineering)

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

______________________________ Chair
Dr. Ida Karimfazli

______________________________ External Examiner
Dr. Fereshteh Mafakheri

______________________________ Examiner
Dr. Catharine Marsden

______________________________ Supervisor
Dr. Susan Liscouët-Hanke

Approved by ______________________________
Martin D. Pugh, Chair
Department of Mechanical, Industrial and Aerospace Engineering

______________ 2019 ______________________________
Dr. Amir Asif, Dean
Gina Cody School of Engineering and Computer Science

Abstract

The aerospace industry is a competitive environment, in which aircraft system suppliers need to develop innovative, increasingly complex systems in ever-shorter time, serving various customer needs. Therefore, aircraft system suppliers need to improve all steps of their development process, from specification through testing.

This thesis proposes a model-based system engineering (MBSE) approach to improve the specification process. As part of a collaborative project with Thales Avionics Canada, this thesis focuses on specifically on the development of test means for flight control computers (FCC). There are two challenges regarding development of test means: (1) tight timeline from the specification to the entry into service and (2) complexity of the architecture. Implementing an MBSE approach to the development of test means is promising to reduce the development time and to increase the quality.

The ARCADIA/Capella MBSE framework is used to develop a generic, re-usable specification model for various flight control computer test means. To develop a generic model, various categories, types and component of FCC test means are analyzed. A variability management model is developed to manage efficiently common and optional feature and components. To do so, Pure::Variants, a product line engineering tool is used. The developed MBSE specification provides an overview of the test means entities, functions, interfaces and components.

In summary, this thesis establishes a more efficient approach for the FCC test means development, potentially decreasing development time and increasing the competitiveness of the system developer.

Contents

Abstract iii
Table of figure vi
Acronyms x
Chapter 1. Introduction 1
1.1. Background and motivation 1
1.2. Flight control systems 2
1.5. Organisation of the thesis 16
Chapter 2. The need for a model-based approach for test means development 17
2.1. Testing of flight control system and FCC 17
2.2. Analysis of current process in Thales 18
2.2.1. Challenges in current process 21
2.3. Potential benefits to model-based systems engineering 22
2.4. Objective of the thesis 22
Chapter 3. Methodology 24
3.1. Methodology overview 24
3.2. Analysis of test means types and components 25
3.2.1. Test Means categories and sub-categories 26
3.2.2. Description of test means components 31
3.3. Specification model development 36
3.3.1 Arcadia 37
3.3.2. Variability management 39
Chapter 4. Model implementation 42
4.1. Capella model implementation 42
4.1.1. Operational analysis level 42
4.1.2. System analysis level 47
4.1.3. Logical architecture level 60
4.1.4. Physical architecture level 75
4.2. Variants management model 83
4.3. Model-based specification process 87

4.4. Additional aspects for model-based specification ... 91
4.4.1. Linking requirements ... 91
4.4.2. Documentation generation ... 92
Chapter 5. Conclusion and future work ... 94
5.1 Summary and conclusion ... 94
5.2 Future work ... 95
References ... 97
Appendix A: Capella diagrams ... 101
A.1. Operational analysis ... 101
A.1.1. Operational capabilities [OCB] ... 101
A.1.2. Operational architecture [OAB] ... 102
A.2. System analysis ... 103
A.2.1. Missions and capabilities [MCB] ... 103
A.2.2. System architecture [SAB] ... 104
A.2.3. Functional dataflow [SDFB] ... 108
A.2.4. Functional breakdown [SFBD] ... 111
A.3. Logical architecture ... 112
A.3.1. Logical architecture [LAB] ... 113
A.3.2. Logical dataflow [LDFB] ... 116
A.3.3. Logical breakdown [LFBD] ... 120
A.4. Physical architecture ... 121
A.4.1. Physical architecture [PAB] ... 121
A.4.2. Physical dataflow [PDFB] ... 125
A.4.3. Physical functional breakdown [PFBD] ... 128
Appendix B: How to select a configuration in Pure::Variants ... 132
B.1. Pure Variants ... 132
B.2. Mappings view in Capella ... 134
Appendix C: How to use M2Doc ... 134

Table of Figure

Figure 1. Primary flight control axis and surfaces (adapted from [44]) 3
Figure 2. Mechanical flight control system [8]... 3
Figure 3. Hydro-mechanical flight control system (adapted from [45]).. 4
Figure 4. Fly-by-light architecture [46] 5
Figure 5. Basic architecture of A330 flight control system [15] 6
Figure 6. Architecture of a FCC of Airbus (adapted from [17])... 7
Figure 7. Schematic of flight control computer interaction with aircraft systems and components
.... 8
Figure 8. V-diagram of the aircraft development process 12
Figure 9. Relationship between the development V-cycle of the aircraft, the flight control computer and its test means 15
Figure 10. Life cycle of a typical FCC, adapted from [29]... 17
Figure 11. Current and improved test means development process 23
Figure 12. Overview of the methodology steps to develop a re-usable MBSE approach for test means specification 24
Figure 13. Overview of test means of flight control system... 26
Figure 14. Categories and sub-categories of test means 27
Figure 15. Architecture of different categories.. 29
Figure 16. Architecture of different sub-categories.. 29
Figure 17. Test means components... 32
Figure 18. [OCB] Operational capability diagram 44
Figure 19. [OAB] Operational architecture diagram 45
Figure 20. [MCB] Mission and capabilities diagram... 48
Figure 21. [SFBD] Functional breakdown of system analysis level 50
Figure 22. [SAB] General model diagram.. 51
Figure 23. [SAB] Simulating signal diagram 54
Figure 24. [SAB] Detecting response diagram 55
Figure 25. [SAB] Managing SUT diagram.. 56

Figure 26. [SDFB] Simulating signal diagram 58
Figure 27. [SFBD] System functional breakdown diagram 59
Figure 28. [LAB] General model diagram 63
Figure 29. [LAB] Simulating signal 66
Figure 30. [LAB] Detecting response diagram 67
Figure 31. [LAB] Managing SUT diagram 68
Figure 32. [LDFB] Scenario file vs. manual simulation 70
Figure 33. [LDFB] Detecting response signal 71
Figure 34. [LDFB] Detecting simulation signal 72
Figure 35. [LFBD] Functional breakdown of aircraft equipment simulation control 74
Figure 36. [LFBD] Functional breakdown of aircraft equipment simulation 75
Figure 37. An example of node and behavioral components 76
Figure 38. [PAB] General model diagram 78
Figure 39. [PAB] General model/ breakdown of the SUT installation component 79
Figure 40. [PAB] Simulating signal diagram 80
Figure 41. Functional breakdown of "simulate aircraft equipment status behavior" function 82
Figure 42. Feature model of test means 85
Figure 43. Model-based specification process 87
Figure 44. Variant editor in Capella 88
Figure 45. A sample of the diagram while variant mode is activated 89
Figure 46. Example of model-based specification process 90
Figure 47. Sample of requirements definition in Capella 92
Figure A. 1. Operational capability diagram [OCB] 102
Figure A. 2. Operational architecture diagram [OAB] 103
Figure A. 3. Mission and capabilities diagram [MCB] 104
Figure A. 4. General model diagram [SAB] 105
Figure A. 5. Simulating signal diagram [SAB] 106
Figure A. 6. Detecting response diagram [SAB] 107
Figure A. 7. Managing SUT diagram [SAB] 107

Figure A. 8. General model diagram [SDFB] 108
Figure A. 9. Simulating signal diagram [SDFB] 109
Figure A. 10. Detecting response diagram [SDFB] 110
Figure A. 11. Managing SUT [SDFB] 111
Figure A. 12. System functional breakdown diagram [SFBD] 112
Figure A. 13. General model diagram [LAB] 113
Figure A. 14. Simulating signal diagram [LAB] 114
Figure A. 15. Detecting response diagram [LAB] 115
Figure A. 16. Managing SUT diagram [LAB] 116
Figure A. 17. Simulating signal diagram [LDFB] 117
Figure A. 18. Detecting response diagram [LDFB] 118
Figure A. 19. Managing SUT diagram [LDFB] 119
Figure A. 20.Functional breakdown of aircraft equipment simulation control [LFBD] 120
Figure A. 21. Functional breakdown of aircraft equipment simulation [LFBD] 120
Figure A. 22. General model diagram [PAB] 122
Figure A. 23. Simulating signal diagram [PAB] 123
Figure A. 24. Detecting response diagram [PAB] 124
Figure A. 25. Managing SUT diagram [PAB] 125
Figure A. 26. Simulating signal diagram [PDFB] 126
Figure A. 27. Detecting response diagram [PDFB] 127
Figure A. 28. Control aircraft equipment status simulation breakdown diagram [PFBD] 128
Figure A. 29. Control aircraft equipment sensor simulation breakdown diagram [PFBD] 128
Figure A. 30. Control aircraft equipment status failure simulation breakdown diagram [PFBD] 129
Figure A. 31. Control aircraft equipment sensor failure simulation breakdown diagram [PFBD] 129
Figure A. 32. Aircraft equipment status simulation breakdown diagram [PFBD] 130
Figure A. 33. Aircraft equipment sensor simulation breakdown diagram [PFBD] 130
Figure A. 34. Aircraft equipment status failure simulation breakdown diagram [PFBD] 131

Figure A. 35. Figure A. 34. Aircraft equipment sensor failure simulation breakdown diagram [PFBD] 131
Figure B. 1 Perspectives in Pure Variants. 133
Figure C. 1. . M2Doc template editor 136
Figure C. 2. Generated configuration file view in Capella 136

Acronyms

ARCADIA	ARChitecture And Design Integrated Approach
CLAW	Control LAWs
CAS	Crew Alerting System
FBL	Fly-By-Light
FBW	Fly-By-Wire
FCC	Flight Control Computer
FCPS	Flight Control Primary Computers
FCS	Flight Control System
FCSC	Flight Control Secondary Computers
HIL	Hardware-In-The-Loop simulation
IRS	Inertial Reference System
IVV	Integration, Validation and Verification
LAB	Logical Architecture Blank
LDFB	Logical Data Flow Blank
LFBD	Logical Function Breakdown
MBSE	Model Based System Engineering
MCB	Missions Capabilities Blank
OAB	Operational Architecture Blank
OCB	Operational Capabilities Blank
PAB	Physical Architecture Blank
PDB	Power Distribution Box
PDFB	Physical Data Flow Blank
PFBD	Physical Function Breakdown
PLE	product line engineering
SAB	System Architecture Blank
SDFB	System Data Flow Blank
SFBD	System Function Breakdown
SOI	System of Interest
SOF	Safety of Flight

SPOST	System Power-On Self Tests
SUT	System Under Test
UUT	Unite Under Test
VDM	Variant Description Model

Chapter 1. Introduction

This chapter explains the background and motivation for the thesis, introduces the scope of the work performed and outlies the structure of the thesis.

1.1. Background and motivation

The aerospace industry is driven by technological innovation in a competitive environment. The key to success is to bring innovative products to the market in the shortest possible development time. Due to the complexity of the modern aircraft, there are challenges regarding the development process which often lead to delays in delivery to the customer [1].

Aircraft manufacturers and aircraft system suppliers experience these challenges. Therefore, they need to improve efficiency and effectiveness of the development process. Most of the challenges are inevitable but can be foreseen. For example, it is expected to face issues while developing a system that is being built for the first time. Many of recent aircraft programs, such as the Boeing B787 struggled with significant delays during the development. These delays are caused by several factors such as software issues, incorrect fastener installation, etc. [2]. All the potential obstacles must be considered while planning different stages of development to avoid delay. Moreover, there are precautions to take to avoid some of these difficulties. For instance, writing the specification of systems based on fully defined requirements in early stage of the design is a factor that could contribute to reduction in development time.

In this context, the role of the system suppliers and integrators is important. Before the system is integrated into the aircraft, it needs to be validated and verified at the supplier facility. Therefore, the system suppliers need to have efficient development processes at system, subsystem or component level. In addition, they also need to have efficient process to design the test means associated with their system, which are required for verification. The particular challenge for the system supplier is to adapt quickly their products to the aircraft manufacturer needs. Furthermore, suppliers work with different companies with various types of request and criteria, which adds to complexity of their work.

This thesis is part of collaborative project with Thales Avionics Canada. Thales Avionics, as a supplier to different aircraft manufacturers develops and tests flight control systems, in particular the flight control computers (FCCs), as well as the associated test means. The test means are used to validate the design of the FCC, verify its proper functioning and troubleshoot it while in service.

The objective of the thesis is to improve the process of the test means specification by establishing a model-based system engineering (MBSE) approach.

The following sections first introduce the context of flight control systems and provide background on their development process.

1.2. Flight control systems

Any flying vehicle needs to be controlled to perform appropriate maneuver. Aircraft use the flight control system (FCS) to control the motion around the roll, pitch and yaw axes. As shown in Figure 1 the flight control is performed by the deflection of flight control surfaces (such as aileron, elevator, rudder, etc.).

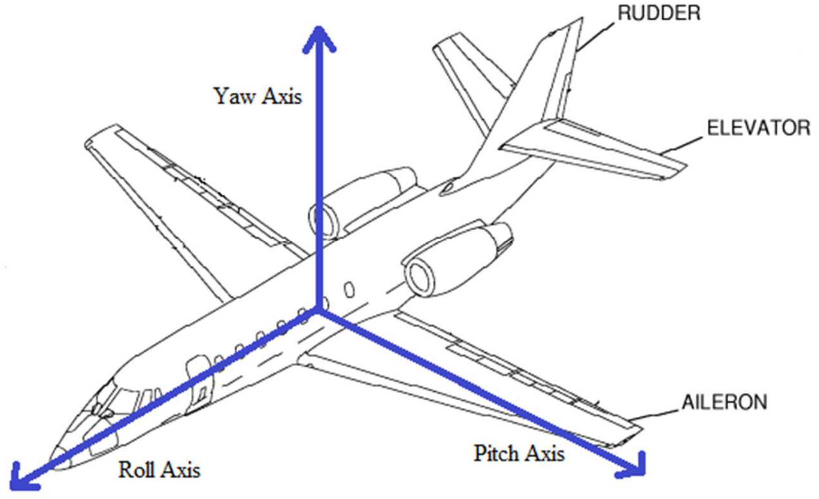


Figure 1. Primary flight control axis and surfaces (adapted from [44] *)*

The flight control system's function is to convey the pilot or autopilot's command to control surfaces. There are different types of flight control system technologies: Mechanical, Hydro-mechanical and electrical.

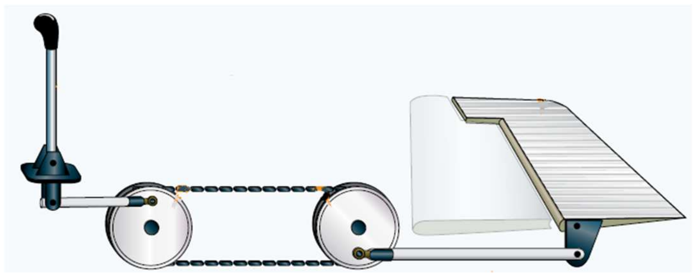


Figure 2. Mechanical flight control system [8]

Mechanical system has been in use since the invention of the aircraft. Figure 2 shows the implementation of a mechanical FCS. Pilot's orders are transmitted to the actuators through cables, rods, levers and cranks [3]. In mechanical system, pilot is supposed to overcome aerodynamic forces on the control surfaces manually which means that the forces to maneuver the airplane is limited by pilot's physical capabilities [4].

As aircraft became larger and faster, it became impossible to control aircraft by muscular strength. Hence, hydro-mechanical flight control systems, as shown in Figure 3 were invented which facilitated control of the aircraft using hydraulic power [4]. In hydro-mechanical system, the pulleys and rod are moved by assist of hydraulic system. Hydro-mechanical flight control systems require hydraulic systems (consisting of pumps, reservoir, pipes and valves) in addition to mechanical system (rods, pulleys and cables) which results in increment of weight of aircraft. It is also difficult to maintain a hydraulic system. Boeing 727 is an airplane with hydro-mechanical FCS [5]. Figure 3 shows the architecture of hydro-mechanical flight control system.

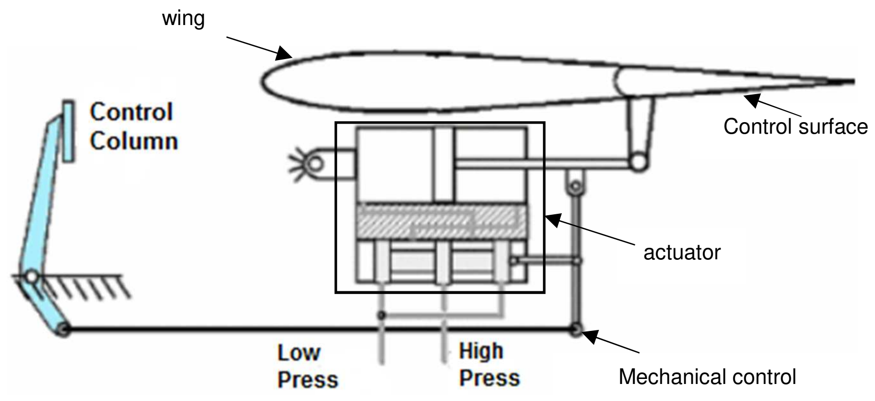


Figure 3. Hydro-mechanical flight control system (adapted from [45] *)*

Nowadays, most of the aircraft have fly-by-wire (FBW) control system. In a FBW system the pilot command is transmitted to the actuators and through electronic signals, enhanced by a Flight

Control Computer (FCC) [6]. However, mechanically controlled systems are still used for some surfaces in some aircraft such as Boeing 777 which has two mechanically controlled spoilers [7] or in small aircraft where the aerodynamic forces are not excessive [8] such as the Eclipse 500 [9]. There are several advantages to FBW systems, such as lighter weight, easier maintenance [10], replacing complicated mechanical linkage to signals [11] easier manufacturing and flexible to changes after initial design and production [12].

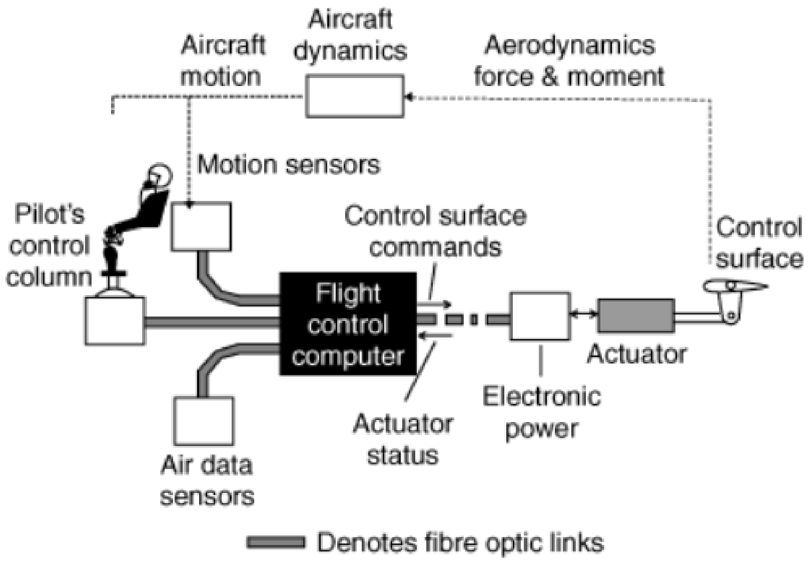


Figure 4. Fly-by-light architecture [46]

Earlier generation of FBW system are using electrical signals. However, the latest generation uses optic fibres and optical sensors. For example, the Gulfstream 650's FCS. This technology is known as Fly-by-Light (FBL). Figure 4 shows the architecture of a FBL system. Note that the only difference between FBW and FBL is regarding the type of signal that is being used. FBL systems are lighter and can transmit higher data rates over FBW. However, FBL is more expensive to develop and test comparing FBW system [5].

Within the flight control system, the flight control computers (FCCs) are critical components that implement the flight control laws. Typically, an aircraft with full fly-by-wire on all three axis of control will have dual or triple redundancy [13] such as Airbus 330/340 [14].

Airbus 330 has five FCCs: three flight control primary computers (FCPS) and two flight control secondary computers (FCSC) [15]. Each computer has two elements: the monitor element (MON) and the command element (COM)), each with different software and architecture [16]. The MON channel monitors the function of command channel. Both channels are active simultaneously [17]. For this aircraft, all surfaces are hydraulically actuated and electrically controlled, except for the rudder which is mechanically controlled [15]. The FCSC control the surfaces in standby mode [16]. For any given function, one computer is active and others are standby. As soon as the active computer interrupts its operation, one of the standby computers becomes active [17]. Figure 5 shows the basic architecture of Airbus A330 FCS.

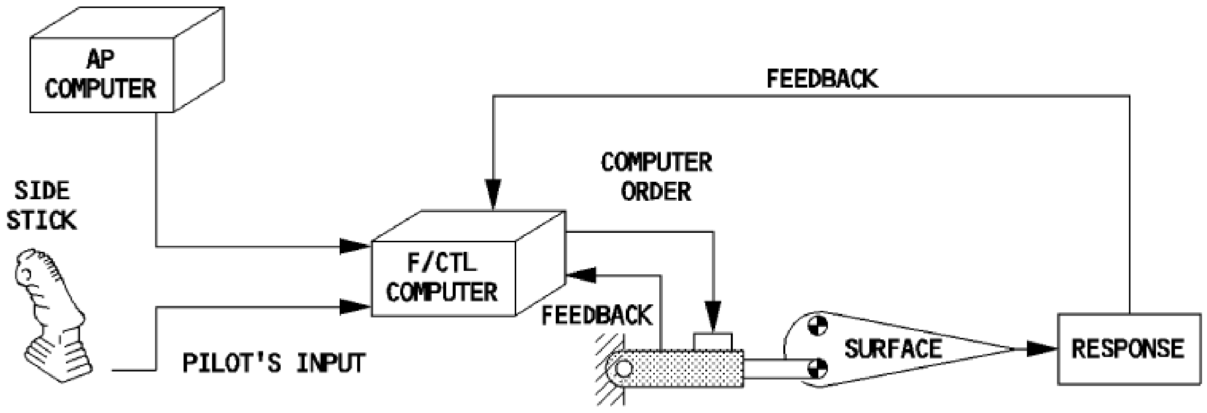


Figure 5. Basic architecture of A330 flight control system [15]

Figure 6 shows the architecture of FCC of Airbus, which has two channels (MON and COM). The difference between the results of MON and COM channels are compared considering a threshold. If the difference is above the threshold, a failure can be detected. The sign P in the Figure 6 represents the protection against over-voltage and under-voltage [17].

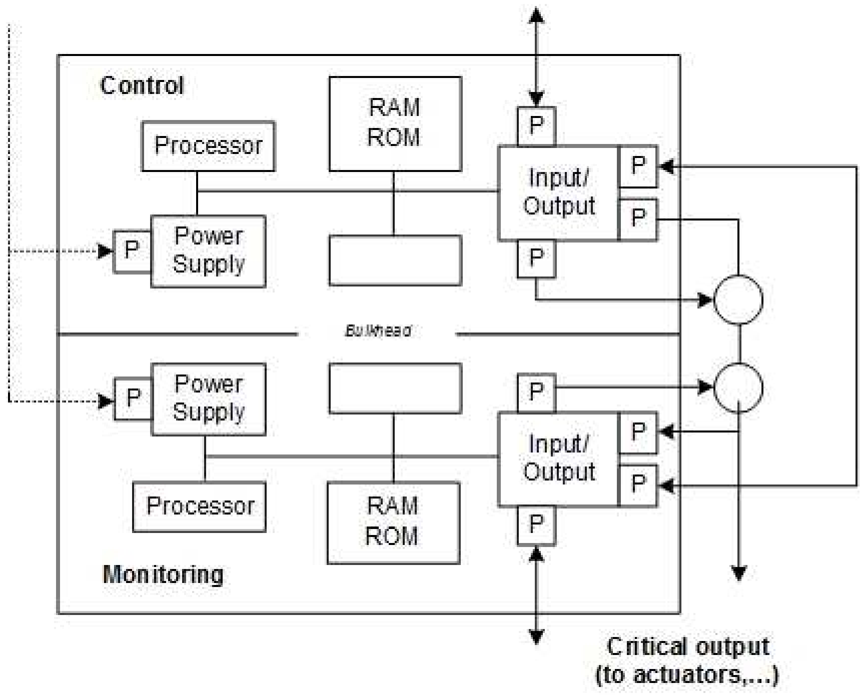


Figure 6. Architecture of a FCC of Airbus (adapted from [17]*)*

There are several types of flight control system architectures. For instance, Boeing 777 is the first Boeing commercial aircraft which employs FBW system [12]. It has three primary flight control computers. Each FCC has three lanes: command, standby and monitor [18]. Command lane outputs the flight control command. In case command lane fails, standby lane outputs the command to actuators. Standby and monitor lane perform the same calculation as the command lane [16].

Figure 7 shows a generalized view of the flight control system, focusing on the interactions between the flight control computers (FCC), actuators and surfaces. Depending on the design of the aircraft, FCC interacts with these various systems.

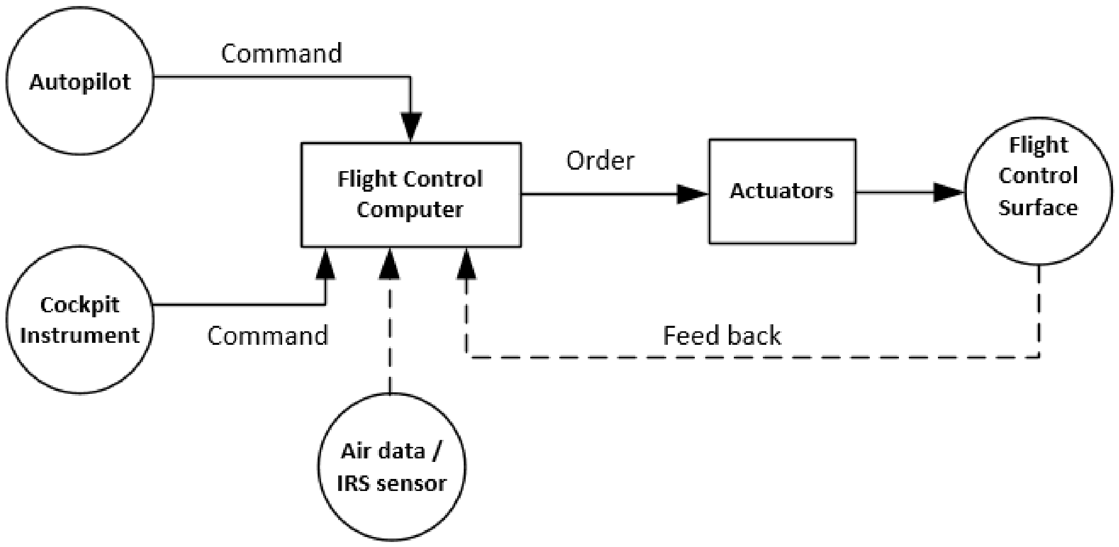


Figure 7. Schematic of flight control computer interaction with aircraft systems and components

The FCC receives commands either from the autopilot or from cockpit instrument (which is controlled by the pilot). It also receives inputs for sensors, such as air data information and current position of the aircraft. Based on these inputs, the FCC calculates the needed change of the position for the control surfaces and sends the command signal to actuators. Actuators change the position of the surfaces based on the FCC's command. Then, the FCC needs to detect the new position of the control surfaces through a feedback loop. There are two types of feedback. One the feedback from the surfaces and the other from the sensors (inertial reference system (IRS) and air data) which is the position of the aircraft. In a fly-by-wire system, there is no direct link between the pilot and control surfaces. Therefore, an artificial feel and feedback is used to avoid damage to the system [4]. Artificial feedback produces an artificial feel of movement of the control surfaces with the use of spring. The artificial feel is proportional to spring stiffness [16]. In most recent transport and military aircraft, actuators are controlled electrically but powered hydraulically [19].

Test means are developed to validate, verify and troubleshoot the FCCs before they are integrated into the aircraft, using hardware and software simulation. There are challenges regarding the development of the test means, especially for the FCCs. As the test means is required to verify the FCC, it should be fully developed before FCC's design and assembling ends. Moreover, the design of the test means is dependent on the design of the aircraft, mission of developing the test means (whether it is single-project, multi-project, etc. (section 3.2.1.)) and customer's need. This means that from one FCC to another, the design is different which makes it more complicated to identify the requirements and design. The flight control computers in an aircraft are safety critical components that interact with many different aircraft systems and components. Therefore, their development process, and particularly the validation, verification and integration is a complex task.

The next section discusses the background for the FCS and FCC and test means development process.

1.3. Test means as part of the aircraft development process

Development of complex, safety-critical aircraft systems, such as the flight control system, needs to follow a rigorous process to be able to obtain certification by the authorities. In particular, the

SAE ARP4754 “Guidelines For Development Of Civil Aircraft and Systems” [20] needs to be followed.

Generally, the life cycle process involves three main phases (if retirement is neglected):

1. Concept

This phase is the preliminary step in design. In the concept phase, the overall characteristics of the aircraft are determined such as the number and locations of engines, payload, range and aircraft size. Also, the flight control systems’ technology and high-level architecture will be evaluated and selected, without detailed specification.

2. Development

The development as shown in Figure 8, including design, integration and verification is the longest phase of life cycle process. It starts with definition of aircraft-level functions, interfaces and requirements. Then, it moves on to the definition of system-level and item-level functions and requirements and allocation of them to systems and components. Once the requirements are defined and the functions are allocated, hardware and software are designed, built and integrated. Integration starts with item by item integration until the system is completely assembled. During early stages of the programme, a delivery schedule is established. Items that take more time to build, are ordered ahead of time [16].

The verification process takes place along with integration. This facilitates detection of deficiencies and troubleshooting.

The validation, verification and integration process includes analysis, modelling and functional testing. Functional testing starts with ground tests and then the flight tests. Once the tests are done, the aircraft is certified to enter the market and be in operation [16].

3. Operation

Once the aircraft is in operation, it is working regularly and its performance is monitored. In case of any fault, it is reported to manufacturer for troubleshooting [16].

The aircraft development phases are illustrated through the so-called *V-diagram*, depicted in Figure 8. The left-hand leg represents analytical steps, while the synthesis steps are followed in the right side [4]. The process starts from left side and ends on the right side. The first step is to define the scope of the system, which is being designed and developed. Then, the requirements have to be defined and based on the requirements, specification and details of design are documented. The bottom of the diagram shows the building process, which is the step of developing the system. The right side of the diagram is where the system is integrated and verified based in the requirements defined in the left-side of the diagram. The progressive testing from subsystems to complete product is referred to as "integration" [21]. In each level of right-hand leg, system or subsystem is verified by the same level at left-side.

In the validation process, the question to answer is: "are we building the right product?" [22]. In other words, does the designing product meet the high level requirements and the customer's request?

However, in verification, the attempt is to find out whether "we are building the product right?" [22]. In other words, is the developed product compatible with the defined requirements? [22]

As shown in Figure 8, the process starts at high-level systems, breaks into subsystems and then components. As it progresses into the details, the requirements are evaluated based on the higher-level requirements (validation). However, the right-hand leg follows a bottom-up approach. In other words, it initiates the verification process with components and integrates them until the desired product is developed.

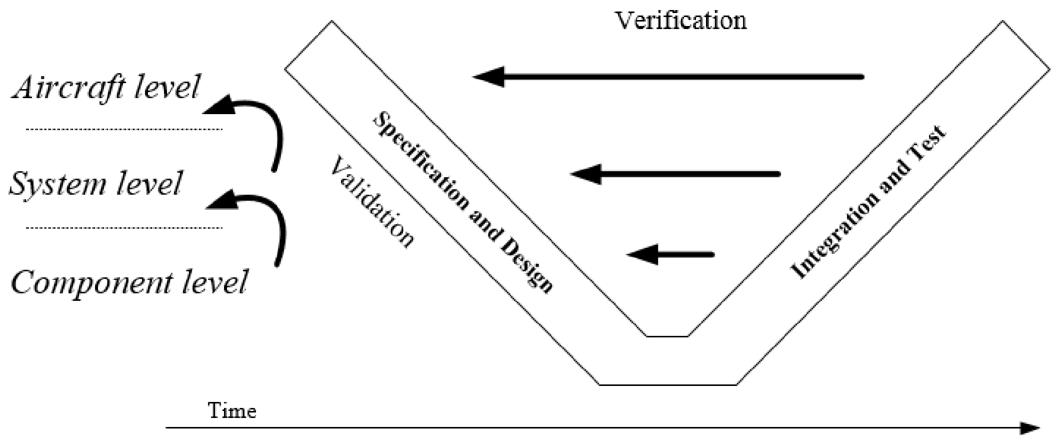


Figure 8. V-diagram of the aircraft development process

The testing has two objectives: to ensure that the system performs its intended function and to guarantee that the system does not perform unintended functions [20].

Testing of components and whole aircraft needs facilities and equipment. Facilities must be designed, built and calibrated before integration begins. The equipment must meet standards in terms of:

- Accuracy: The measurements must be more precise than the tolerance on the system input and response.
- Reliability: It must be highly reliable to minimize test discrepancies as a result of equipment error.
- Flexibility: Test equipment should be designed to serve several purposes. Thus the cost is minimized. However, the flexibility should not be at the expense of accuracy and reliability [23].

Tests are performed using a so-called test means, which can be considered as a system itself. Thus, it follows its own development process, including specification, building, validation and verification. The test means involves hardware equipment and software for simulation of physical or functional component [23]. The testing tools are verified based on the DO-178 [24] which is a guideline used by certification authorities.

The aircraft, systems and test means development processes can be approached as *systems engineering* processes. “Systems engineering is a methodical, multi-disciplinary approach for the design, realization, technical management, operations and retirement of a system” [25].Systems engineering is a logical sequence of activities and decisions that transforms an operational need into a description of system performance parameters and a preferred system configuration [26]. In a systems engineering approach, the test plan defines and specifies the parameters that should be tested. In addition, the test plan specifies how to diagnose discrepancies and how the test results should be analyzed [23].

In this thesis, a model-based systems engineering approach is investigated. “Model-Based Systems Engineering (MBSE) is the formalized application of modeling to support system requirements, design, analysis, verification and validation activities beginning in the conceptual design phase and continuing throughout development and later life cycle phases” [27]. In traditional systems engineering, the primary products are documents, while in MBSE the product is a model [23].

If a model (digital representation) of the to-be-built system is available, the individual views filtered through the perspective of the model have a better chance of being representative of reality. In other words, without the model, everyone’s view on the model can be different from others. The model can also optimize the validation process by providing a representation of desired outcome for customers and evaluating whether it is what the customer is looking for or not [28]. Moreover, identification of requirements and writing the specification is an iterative process. The document based specification process is difficult to manage in terms of changes and traceability. This leads often to increased development duration.

1.4. Problem statement

Figure 9 shows the relationship between the development cycles of the aircraft, the FCC and its test means, in form of the V-diagram. For the development of the FCC, the verification process (right-side of the diagram) is performed by use of the test means. Therefore, the test means has to be developed before the FCC is tested. However, the design and development of the test means has its own process. It means that the test means development process has the same procedure as the FCC and aircraft: definition of the requirements and specification, building, integration, validation and verification.

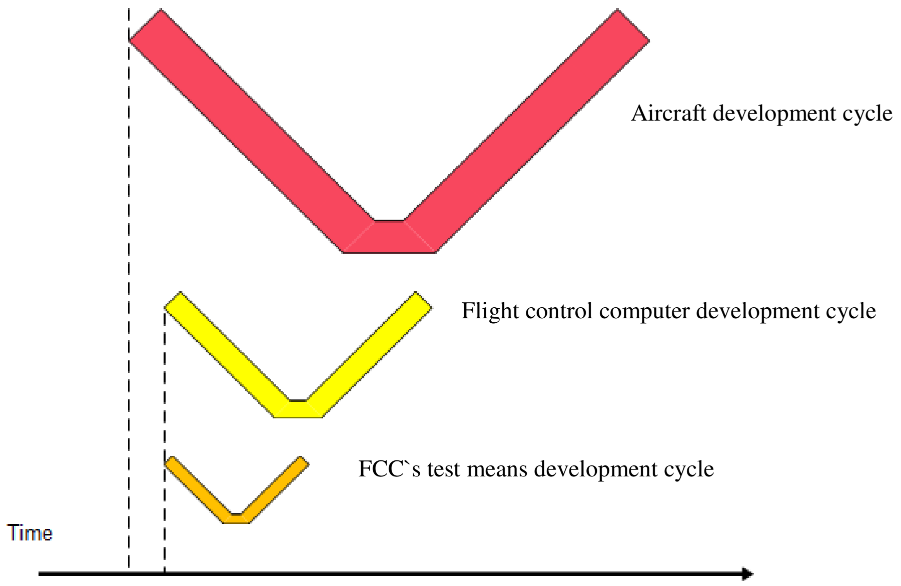

Aircraft development cycle
Flight control computer development cycle
FCC`s test means development cycle
Time

Figure 9. Relationship between the development V-cycle of the aircraft, the flight control computer and its test means

The test means development is completed before the FCC enters the integration and verification process. To achieve this, the test means specification needs to start at the same time as the FCC development. This is difficult, as the FCC requirements may not be mature yet.

Figure 9 clearly illustrates that the timely development of the test means is critical to the on-time development of the FCC, which is generally on the critical path in the aircraft development process. Failing to deliver the FCC on time could lead to delay in launching an aircraft and have economic impact on the system supplier and the aircraft manufacturer. It is therefore important to develop methodologies that increase the efficiency and effectiveness of the specification process of the test means.

This thesis aims to improve the test means development process by introducing a model-based system engineering approach.

1.5. Organisation of the thesis

The thesis is organized in following way: Chapter 2 discusses the problems and challenges regarding the current process of test means development. It also discusses how a model-based systems engineering approach, can help to address these challenges. Chapter 3 explains the taken approach and introduces the methodology for creating the specification models within the chosen MBSE framework and managing the variability. Chapter 4 represents the specification model implementation in different levels. In addition, it is presented how requirements are integrated into the specification model, how documents can be generated and how variable and common features are defined. Moreover, it provides an overview on the new, model-based process for the generation of the test-means specifications. Finally, the thesis wraps up by stating the conclusion and provides axes for future research.

Chapter 2. The need for a model-based approach for test means development

This section provides an overview on potential benefits associated with a Model-Based Systems Engineering (MBSE) approach and outlines the current process for test means development.

2.1. Testing of flight control system and FCC

This thesis is focused on FCC. However, the available literature is specific to the development process of the FCS. Thus, this section reviews the development process of the FCS. The typical development process of the FCS is defined in six steps as shown in Figure 10. The first step is where the mission-related requirements are defined. Then, system specifications and development specifications are written. System specifications documentation allocates the requirements to functions. Development specifications defines the performance, interfaces and technical requirements for items.

After that, the FCS is developed. The first version of FCS built based on the design is the prototype which is tested using the test means. At the implementation stage, different subsystems of FCS are integrated and verified. At the production stage, the developed FCSs are verified to ensure that they are working as good as the prototype. The final stage, fielded system, is post-production and maintenance service for FCSs in operation. [29]

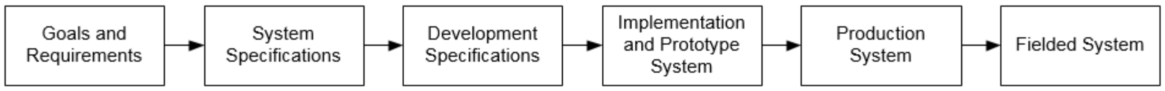


Figure 10. Life cycle of a typical FCC, adapted from [29]

There are different types of tests to verify a FCS. These tests are done in different stages of design for different purposes. The tests start at subcomponent level and then the complete system. Subcomponents of FCS are e.g. actuators, FCCs, wiring, etc. which are tested individually and then combined to ensure that the implemented system works well and subcomponents function efficiently. A large portion of time and effort in the verification process of the FCS is allocated to software tests. Since software testing is complex, software is broken into small modules. Each module is tested individually and then the integrated software is verified. One type of verification tests common for FCS is the so-called Hardware-In-The-Loop simulation (HIL). At its simplest level, it is used to verify that the software acts as designed in actual FCC hardware [30].

Another type of testing is the simulation with a so-called Iron Bird, which is higher level than HIL. In an Iron Bird, all subsystems of FCS (including actuators, electrical power system, etc.) are integrated and tested. Ground testing with the real aircraft is the tests to verify that the FCS works efficiently under different conditions and also to ensure that the structure of the aircraft does not disturb FCC's performance. The last test is the flight test where the aircraft is fully-developed and prepared to enter the market [30].

Thales develops the HIL simulator to verify the FCC and a type of test means used in aircraft ground testing which is the environmental qualification test means.

2.2. Analysis of current process in Thales

No literature is available on the detailed development steps of test means for flight control systems.

This section describes the current process for development of test means at Thales, in general terms. The current process comprises the following steps:

- Step 1: Request of development of the test means
- Step 2: Establishing the specification of the test means
- Step 3: Development of the test means, including trouble shooting
- Step 4: Internal delivery and entry into service
- Step 5: In-service life of test means

Step 1: This first step is initiated and driven by the customer (aircraft manufacturer). The customer specifies the required features, which should be included in the test means.

Step 2: The needed functions for the test means are defined according to the contract with customer, the aircraft design and the goal of developing the test means (whether it is an environmental qualification test means, a conventional one, etc.). Generally, there are two types of documents for specification of test means.

a. System Specification

This specification explains what we need the test means to have. In other word, it indicates functions that the test means must have. It also involves a description of the function of the FCCs.

b. Design Specification

This document gives a description of the elements of the test means and their requirements and functions. Most of the components are identical in different types of test means. The simulation is different from one project to another. Thus, the difference would be for components doing simulation.

The process of writing the specification takes around four weeks of full-time work. There would be two or three iterations to produce the final document. It is written by the test means team and will be reviewed by the systems team (who designs the flight control computer), the IVV (Integration, Validation and Verification) team and the quality team.

Step 4: The test means are not delivered to the customers who ordered the flight control computers. It will be used by the IVV team, which is a team in Thales performing the tests on the flight control computers.

Step 5: A test means should be in operation for as long as the aircraft, whose flight control systems is being tested, is in service. After the certification phase of FCC is finished and the FCC is in service, the test means is used to troubleshoot the FCC. It could be more than twenty years.

There are different phases of testing:

a. Tests before Safety of Flight (SOF): Before a software can be authorized for flight tests (SOF - Safety of Flight, also named Red Label), all system requirements concerning inputs and outputs management (especially what is interfacing with other systems), the Control Laws (CLAW), the System Power-On Self Tests (SPOST), the Crew Alerting System (CAS) must be tested. Some other tests are optional and dependent on the agreement with the customer (aircraft manufacturer), such as regression tests, robustness tests or maintenance tests.

b. Tests before aircraft certification: These tests are performed before entering into service (certification, also named Black Label). It is the completion of the optional tests (defined above) that are performed in the first phase and some additional performance tests.

c. Tests during the service life of the aircraft: In this phase, there are normally less tests to be performed, but having test means in operation is an obligation.

Currently, the test means design and development begins approximately six months before the first software of flight control computer is created. During the first two phases, another test rig is usually required because of the large amount of testing. For the rest of the aircraft life cycle, only one is needed. That is why it is essential that components of test means be re-usable.

2.2.1. Challenges in current process

There are challenges regarding development of FCC test means. Any of the challenges could result in increase of the development cost and delays. The following are some of the difficulties associated with test means building process:

- Since the design of a test means must be compatible with the design of the aircraft, from one test means to another, the architecture is diverse. Moreover, the mission of the test means and the customer's request are also factors that influence configurations of test means. Variety in types may make the identification of the required and proper architecture difficult to manage.
- As mentioned in previous section (1.4), the development of the FCC and its test means are done in parallel. For this reason, it is likely that the FCC's functions and requirements are not

completely defined and change through the development process. Once the test means is built, modifying the system or changing a function would be expensive and lengthy task.

- It might be confusing for the user of the test means (IVV team) to fully understand the architecture of the test means. As they (IVV team) write the test plan and evaluate the FCC based on the standards using the test means, it is important that they understand the tasks that the test means is capable of performing.

2.3. Potential benefits to model-based systems engineering

Based on the current problems of developing the test means, it is expected that a model-based system engineering (MBSE) approach would be a solution and facilitate the progress. By applying an MBSE approach, the implementation of the system (test means) is represented in graphical form in terms of the components and functions. This model would be simple to grasp.

Different MBSE frameworks and tools exist today. The most common ones are UML [31], SysML [32]. Due to the specific needs of the aerospace systems engineering process, Thales developed its own methodology and tool, which is now available in the open source as ARCADIA (methodology) and Capella (tool) [33]. This thesis will use ARCADIA / Capella as MBSE framework. More details about the underlying methodology a given in section 3.3.1.

2.4. Objective of the thesis

The purpose of this thesis is to improve the efficiency and effectiveness of the test means development process. The aim is to reduce the development time of the test means, and to be able to start the test means design earlier.

Figure 11 illustrates how the development time of the test means would be different once it is improved. In improved process, the left wing of the V diagram is slightly more steep which indicates less time spent on requirements identification and specifications writing. In addition, it is aimed to start earlier, even before the FCC's development process starts.

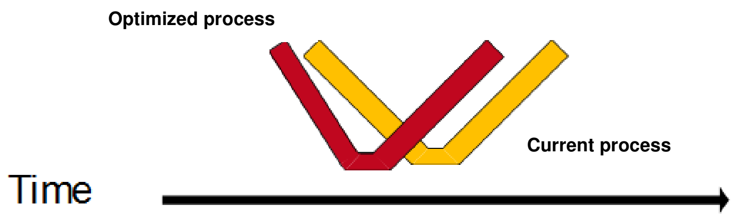


Figure 11. Current and improved test means development process

The question is how a generic and re-usable model of the test means architecture would result in shorter development process. Basically, the generic specification model could address the challenges mentioned earlier which will affect the total development time.

- The generic model which includes all the components and functions that may or may not be in one specific test means, gives a wide picture of the whole system. Thus, implementation of the test means is already modeled when the requirements are being defined.
- The generic model assists to recognize all the potential features that could exist in a specific type of test means in early stage of design. When the developers are missing a feature, the model helps them correct the specifications. Moreover, if they are in doubt of what feature will be needed, they would consider applying all the possible features. It will be a more efficient approach compared to selecting one of them and ending up reforming entire system.

The next chapter describes the methodology of developing the generic and re-usable model.

Chapter 3. Methodology

This chapter provides an overview on the methodology employed to develop generic MBSE approach for test means specification. The different types and associated components of test means are analyzed and categorized. Finally, an overview of the ARCADIA methodology is given. The implementation into the Capella tool is discussed in Chapter 4.

3.1. Methodology overview

To achieve the objective of establishing a re-usable specification framework for test means using model-based systems engineering, an analysis of the different test means is performed. The generic model needs to include all potential functions and the associated the components that could potentially exist in a test means architecture. Figure 12 shows an overview of the pursued methodology. The process starts with test means categorization and continues with analysis of components. The next step is to create a generic model in Capella and finally managing the generic model using variability management.

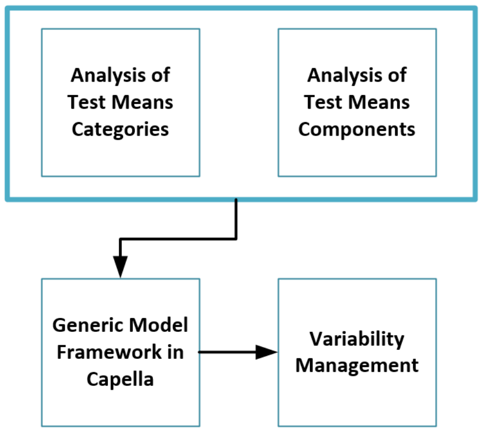

Analysis of Test Means Categories
Analysis of Test Means Components
Generic Model Framework in Capella
Variability Management

Figure 12. Overview of the methodology steps to develop a re-usable MBSE approach for test means specification

The next section explains the first part of the methodology, which is the analysis of the test means categories and components.

3.2. Analysis of test means types and components

In this section the diverse categories and components of the test means are explained. To understand the architecture and types of test means it is essential to clarify the purpose and the associated functions of different test means types. By analyzing eight different test means developed by Thales such as the Cessna M700, the CRJ 700/900/1000 and by discussing with the engineers responsible to develop the test means, the following three key tasks are identified:

1. Simulating of aircraft equipment: When the FCC is inside the aircraft, it is interacting with various systems (dependent on the design of the aircraft). The test means simulates all these systems.

2. Detecting the response of the FCC under test: To test and troubleshoot the FCC, the operator (the person who runs the test) needs to detect the response of the FCC. The response is the command to actuators. Based on the response, the FCC is evaluated.

3. Managing FCC's software: Another task of test means is to manage the software installed in FCC. The test means is able to read the signals that are being exchanged inside different channels of the FCC, to modify the variables, and to calibrate it once there is a change in hardware or software.

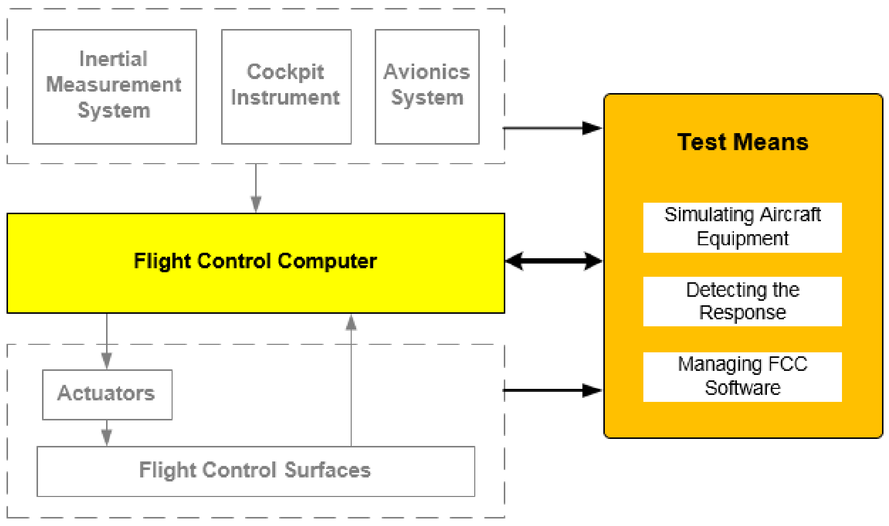


Figure 13. Overview of test means of flight control system

Figure 13 shows the test means and its capabilities. The test means substitutes all the systems that are interacting with the FCC and connects directly to the FCC.

3.2.1. Test Means categories and sub-categories

There are different types of test means for FCCs. To classify them, the terms categories and sub-categories are introduced.

Figure 14 shows a breakdown of categories and sub-categories of FCC's test means. Three categories are proposed and each of them has three sub-categories. The coming sections explain categories and sub-categories in details.

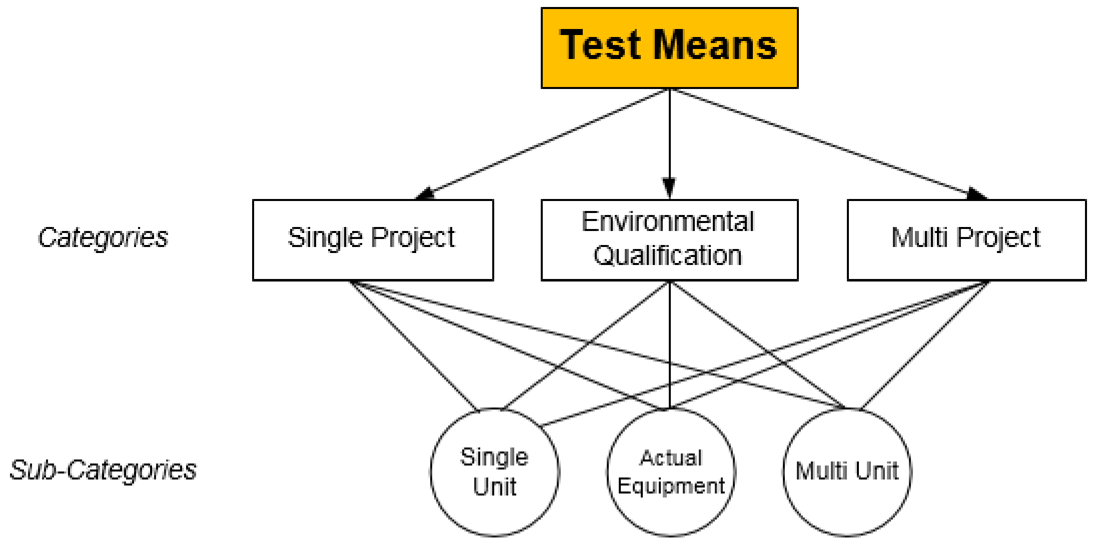


Figure 14. Categories and sub-categories of test means

Any type of test means is classified within only one of the categories. In other words, it is either single-project, multi-project or environmental qualification. The test means can have multiple sub-categories. However, single-unit and multi-unit sub-categories cannot exist together. Thus, the sub-categories in test means are either single-unit, multi-unit, single-unit and actual equipment or multi-unit and actual equipment.

Test Means Categories:

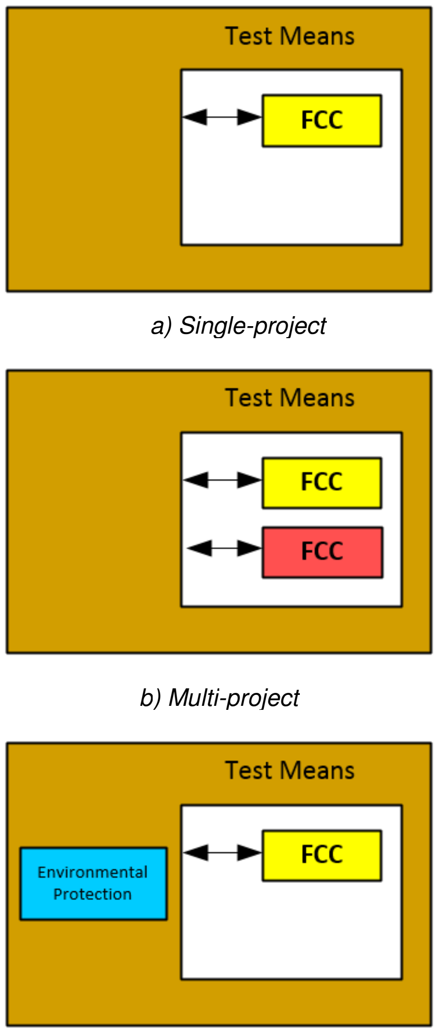
The term "Category" is referring to the differences that result of the different purpose of building the test means. There are three categories of test means:

1. Single-project test means: A single-project test means is a type of test means that is built for one specific FCC. The FCC of different projects may comprise different controllers or they could have the same controllers but with different connection and input/output signals. Therefore, one cannot replace the other. In other words, one can use the single project FCC only for one specific FCC. The purpose of having this category is to do tests of one type of aircraft in a certain location (not transportable). There can be different sub-categories to this category, depending on the type of aircraft.

2. Multi-project test means: This kind of test means is built to host different type of FCCs. It is adaptable to different aircraft development projects. Since many tests need to be performed in a short time frame during the FCC certification phase, the validation and verification team might need to work simultaneously with two units of the same test means. As it is not possible to manage that with a single test means, the multi-project test means type is used. Once the certification phase is over, the multi-project test means can be adapted to other project.

3. Environment qualification test means: This type of test means is the transportable type, which is to perform environment qualification test on the aircraft. Environment qualification tests are tests against external conditions. For example perturbations such as lightning or electromagnetic fields, or other as specified in the DO-160 [34]. It needs to be transported to different locations to be integrated with other equipment of the aircraft and at a specific laboratory where environment condition can be reproduced.

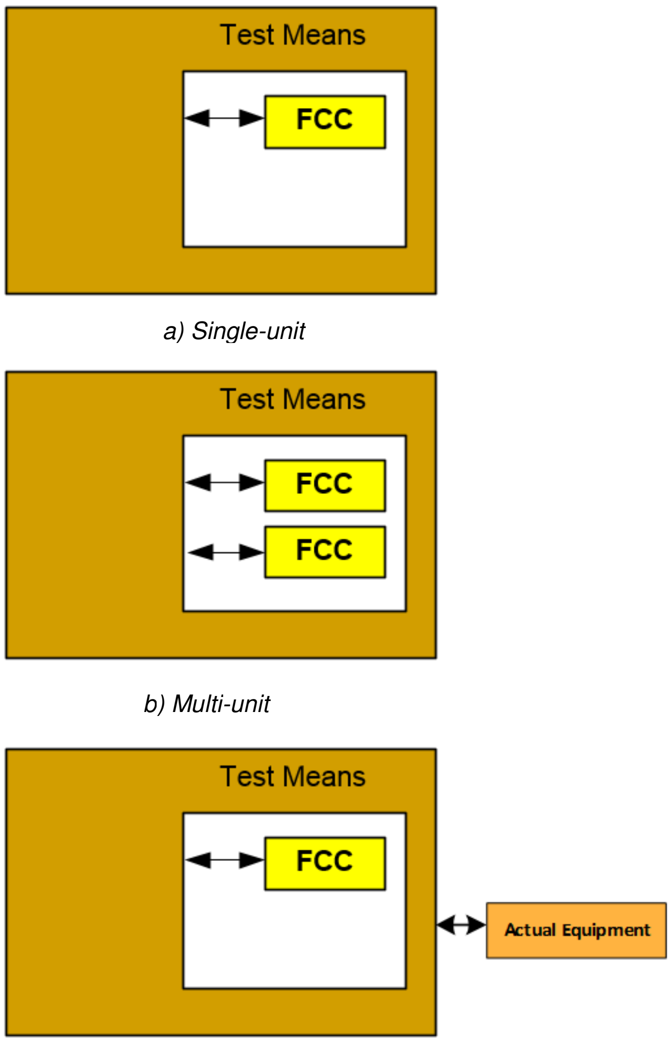
Figure 15 compares and illustrates schematically the architecture of different categories of test means. Figure 15.a shows the test means that is built for one specific project, b shows multi-project test means and c depicts the environmental qualification test means, which has environmental protection.



a) Single-project

b) Multi-project

c) Environmental qualification



a) Single-unit

b) Multi-unit

c) Actual equipment

Figure 15. Architecture of different categories

Figure 16. Architecture of different sub-categories

Test Means sub-categories:

Sub-categories are variants of test means features, which can exist in each category. These features are dependent on the customer's need and the design of the aircraft under development.

1. Single-Unit Test Means: In single unit test means, there is only one FCC being tested. Usually fly-by-wire test means have multiple FCCs. However, in some cases, only one FCC is tested. For example, if there is a new technology introduced to FCC, firstly it can be tested as a single unit under test. Figure 16.a represents single-unit test means.

2. Multi-Unit Test Means: In this kind of test means, there are several FCC units under test e.g. two FCCs and one Back-up Flight Control Units (BFCU). Since there are more units under test, the signals to detect multiply. Moreover, the exchanged signals between units has to be detected as well. Thus, this kind of test means needs larger test rig. Figure 16.b represents multi-unit test means.

3. Actual Equipment Test Means: The actual equipment that can be used for FCC's test means are actuators and cockpit instruments. In test means with actual equipment, the particular physical equipment is used instead of a software simulation. Therefore, beside the test rig, where the harnesses, power supply and simulation computers are installed, actual actuators, which are powered hydraulically or electrically, and actual cockpit instruments need to be integrated. In addition, the aerodynamic load acting on flight control surfaces is mimicked by a physical object pushing against surfaces representing the flight control surfaces. Figure 16 c) represents an actual equipment test means. In this figure, the actual equipment is shown as a separate system interacting with the test means. In the model, the actual equipment is considered separated from the test means.

However, if there is actual equipment, the test facilities need to be larger for the installation and it needs to track signals (input and output) of the actual equipment. Thus, it affects the configuration of test means.

Figure 16 illustrates and compares different sub-categories of test means. Figure 16. a) shows one single project test means, b) depicts multi-unit test means and c) shows the actual equipment test means.

3.2.2. Description of test means components

There are different components in test means performing various tasks. In this section, these components are identified, described and categorized. As a result of analyzing various test means implementations, Figure 17 presents a generic view of test means components. The components are categorized as hardware, software, common (blue) and variant (orange). In each test means, there are three computers: one for equipment simulation, one for data acquisition and one for software management. In the following, first the various software components are described, then the hardware components. This high-level description made to provide an understanding of the components purpose and function, which will be translated into a specification model in chapter 4.

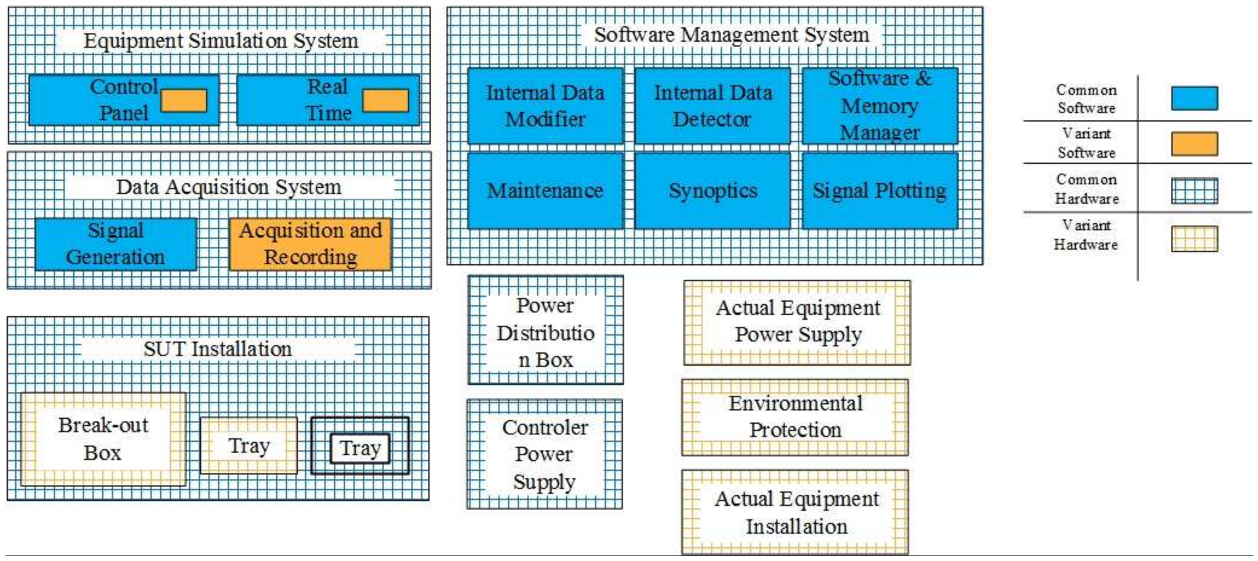


Figure 17. Test means components

3.2.2.1. Overview of software components

There are three groups of software applications used to preform testing and troubleshooting: environment simulation system, data acquisition system and software management system. The following explains each group and their functions.

Environment Simulation Computer

This system performs the simulation of the environment that the FCC is interacting with in real aircraft. There are two kinds of input to the software. The first input is the so-called “scenario file” which consists of a list of simulation signals, their values and their sequence. The second input is given by operator, for example, to change the value of signals during a simulation activity. The simulation control panel is classified into two categories: (a) system panel and (b) sensor panel. The system panel is used to change the state of the aircraft. The sensor panel is used to modify data being measured by sensors in order to simulate failures. For example, for air data, there is a parameter which represents the state of the airplane such as altitude, speed, etc. There is also sensor measuring this parameter and generating another parameter. These two parameters must have same value since they are representing the same thing. These values can be modified separately. Since the state can affect several sensors, altering the value of one state, would result in having all signals detected by sensors changed. As the test involves different scenarios, it is required to have value of only one sensor changed, thus these parameters have been separated. This configuration would enable the operator to simulate the condition of the sensors failing or put incorrect value to verify/troubleshoot the system.

In the test means, there are two software allocated to the environment simulation system. One is the control panel and the other is to perform real-time simulation (section 4.1.3/ [LAB] simulating signal). As shown in Figure 17, both control panel and real time definition includes variability. This variability is related to the type of equipment to be simulated which is dependent on the design of the aircraft and the scope of the test means.

Data Acquisition Computer

Data acquisition system has two software as well. The signal generation, which receives data from simulation computer and generates the signals which are compatible to FCC and also detects the response of the FCC.

The other computer, which is a variable, is acquisition and recording tool that detects the response of the FCC, records them and displays them to operator for test or troubleshooting report.

Software Management Computer

The system management system is one computer with several applications. The word “software” refers to the software that is installed to the FCC. The management system is the part of the test means which is controlling the FCC by installing new software, modifying the variables, detecting internal signals, etc. The application of software management systems are:

I. Internal Data Monitoring: This software monitors internal signals. Internal signals are data being exchanged between different unit under tests (UUT) or different channels or lines of a UUT. UUT refers to one FCC while system under test (SUT) refers to one or multiple FCCs.

II. Internal Data Modification: This software modifies the internal variables. In the software uploaded on the FCC, there are some variables in formulas to calculate the actuator movement. The output of the FCC goes as a signal to actuators to move it. This tool modifies these variables.

III. Plotting acquired signal: It is used to plot signals, which have been detected. It could be simulation and response signals acquired by data acquisition system or could be internal signal detected by internal data monitoring.

IV. Maintenance: This application simulates the maintenance system of the aircraft, which does the maintenance of FCC. The operator must change the mode of FCC to maintenance and run this application. Maintenance needs to be done when there is a new version of FCC or simulated components of aircraft has been replaced.

V. Synoptic: This tool simulates the synoptic information sent by the FCC. It shows the status of cockpit instrument and control surfaces.

VI. Software Management: Changing the software of FCC and getting the version of uploaded software is done using this application

3.2.2.2. Overview of hardware components

Hardware parts of the test means are shown in Figure 17 in grid boxes. Main components are:

Power Distribution Box (PDB)

Power distribution box is a device that divides the power coming from the power supply to different components of test means.

Power supply

There are two types of power supply. Electrical one which is used to power the test means and actual equipment power supply which consists of two types. Hydraulic power supply which is to power actuators and electrical one is to supply power to actual cockpit instrument and actuators.

Actual equipment installation

The actual instrument itself is considered a separate system interacting with test means (entity). However, its installation to the test means is part of the architecture and it needs to be part of the specification.

Environmental protection

Environmental testing is performed to verify the behavior of the FCC in real condition of operations. The FCC is monitored while being exposed for instance to vibration, extreme temperature, high-intensity radiated field (HIRF) and lightning. In the case of HIRF and lightning, the injected signals are propagating along the harnesses, and since the test means must be connected to the FCC to monitor it, they are exposed to same signals. The FCC is protected (the environmental testing objective is to verify it) but the test means are not. Therefore, protection needs to be added in the form of diodes between the signal and the ground. When very high voltages occur, the diodes lead them to the ground and prevent them from reaching the test means.

Break-out Box

The break-out box is a hardware with ports mounted on it for electrical jumpers to read signals or to inject failure to them without the use of software.

Tray

The tray is a device which exists on the actual aircraft as well. The FCC is placed inside tray. The harnesses are usually connected to the FCC through tray.

Now that the types, components and implementation of test means are identified, the next step would be to develop the generic model. The next section discusses the methodology of generating the generic model and managing variabilities.

3.3. Specification model development

The model was developed following the steps of the ARCADIA methodology. Using Capella tool, which implements the ARCADIA methodology, a generic model is developed, including all components and functions that could exist in the architecture. Hence, it is essential for the user to be able to manage the variants. The methodology regrading the variability management is explained later in this section.

3.3.1 Arcadia

This section provides an overview on the ARCADIA (ARChitecture And Design Integrated Approach) methodology which Capella is based on. It also explains how variable functions and components are categorized.

ARCADIA methodology was defined by Thales and has been used since 2011 [35]. There are four main levels on ARCADIA methodology:

- Operational analysis:

This first level identifies what the users of system must accomplish [35]. At this level, only the users (which are called *entities* in ARCADIA methodology) are considered. Functions and capabilities that the users need to perform are also specified. The system itself remains abstract [33].

- System analysis:

The purpose of this level is to specify what the system has to accomplish for the users. To do so, the functions and interfaces of the system need to be identified, considering the system itself as a black box [35]. Moreover, mission and capability of the system are defined in this level. The *mission* is the purpose of the system, while *capability* refers to the tasks that the system needs to perform, with respect to the users.

- Logical architecture:

The logical architecture level aims to identify how the system will work to fulfill the expectations, identified at operational and system level [35]. At the logical level, the specification of the system starts, by identifying the logical components and their relations. The term “logical component” refers to notional breakdown of the system into the components [36]. In other words, it is not specified whether it is a hardware or software and whether it is designed by the system developer or is purchased form a supplier.

- Physical architecture:

Physical architecture level is the last level where the model illustrates “how the system will be developed and built” [35]. It defines the final architecture of the system [36]. In other words, it is the representation of the system based on blocks symbolizing functions and components. Functions and components are modeled in detail. Type of components (hardware, software, computers, etc.) is specified and functions are allocated to subsystems and components.

Note that the final model in Capella must be consistent which means that each component and function at any level must be referenced to function or component at higher level. It also needs to be referenced by lower-level functions and components. Which means, any function or component that is introduced in any level must exist in lower levels and it is not possible to create a function or component, which is not represented at upper level in generic form or as a part of another function/component.

The methodology suggests a top-down approach, but it does not necessarily need to be. For example, if the system already exists, the purpose of applying model-based system engineering is to study the architecture. Therefore, the focus would be on logical and physical architecture.

There are various types of diagrams available in Capella for representing different aspects of the system. Depending on the purpose and detail of the model, any of them can be used. For this thesis, only certain types of diagrams have been employed because the model of the test means is being developed for the first time and it needs to be focused on the functions, components and the implementation of the test means. Chapter 4 provides the details of the specification model in Capella.

3.3.2. Variability management

Nowadays, the size and complexity of systems are increasing. Moreover, the customers are demanding products that specifically address their segment. This could result in constant increase of cost of development due to increase of complexity and customer-specific adaptations [37]. As a result, companies deploy the so-called Product Line Engineering (PLE) technic. "*PLE is an approach that aims at exploiting reuse potential between products developed in an organization by identifying the commonalities between the products and systematizing the variabilities*" [38]. In PLE, "*variabilities have to be identified, modeled, stored, resolved, instantiated and changed*". Thus, they have to be managed through the lifecycle of PLE [38].

All test means share same core functions. However, there are differences in components, sub-functions and implementations. The variability management is applied to manage the product-specific features. This section explains the variability management methodology exercised in this thesis.

The created model in Capella is a generic model, which means it includes all the potential features that could exist in the system. However, for one specific test means only some of those features are required which means the rest features must be filtered. In this thesis, the word "feature" indicates functions or components which are defined in the Capella model. Features could be mandatory or optional. Mandatory features are those which are always in the system. However, the optional ones are those which only exist as needed.

In the developed model, variability exist at different levels and are of various types (function, component, etc.). For example, "protect signal" is a function for environmental qualification type of test means which exists in the system analysis level. However, different types of simulation function (based on the type of signals) are defined in the physical level which are variant as well. Therefore, variables need to be defined in different levels.

For defining all the variants of the different modelling levels in Capella, a list is created. This list includes both mandatory and variable features. Also, the list identifies the feature in different levels. For example, the feature "power supply" has two sub-features: electrical and actual equipment power supply. The electrical power supply (controller power supply) is for the test means, therefore it is mandatory and must be in all types of test means. However, actual equipment power supply only exists in actual equipment type of test means, thus it is a variable feature. The actual equipment power supply could be electrical or hydraulical. The purpose of categorizing the features into different level was to show the hierarchical relationship between them and to simplify managing it.

After defining the list of features, it must be integrated to the model. For this purpose, they are allocated to the components and functions in the model. The variant management software used for this project is Pure::Variants [39] and it is linked to Capella with use of an add-on called Pure:variants connector to Capella [40]. Section 4.2 illustrates variability management model.

The next chapter illustrates various diagrams of Capella in different levels and discusses the requirement management and document generation from the model.

Chapter 4. Model implementation

This chapter illustrates the implementation of the developed generic specification model in Capella. Moreover, it discusses how the requirements are integrated into the model and variabilities and commonalities are managed using additional software for variability management.

4.1. Capella model implementation

As introduced in Section 3.3.1., Capella is a software used for Model Based System Engineering (MBSE) based on ARCADIA methodology, having four modelling levels: operational analysis, system analysis, logical architecture and physical architecture. Lower the level, more detailed the function and components.

This section explains how the specification model for the test means is structured and implemented using various diagrams in each modelling level of Capella. Note that in this section the word “system/ system of interest (SOI)” refers to the test means. Moreover, “simulation” refers to software simulation.

A complete lest of developed diagrams can be found in Appendix A.

4.1.1. Operational analysis level

As explained in section 3.3.1., the purpose of operational analysis level is to identify what the future users of the system need to accomplish. In other words, it discusses what the users must execute for the system. In this level, the other systems (called “entities” in Capella) which are interacting with SOI (the system we are modelling), the capabilities of SOI and the activities of entities are defined. Here the SOI is the test means.

To model the test means at operational level, the following diagrams are selected to represent the operational analysis:

- [OCB] Operational capability diagram
- [OAB] Operational architecture diagram

For the purpose of modelling the test means, these types of diagrams provide the best platform to illustrate the architecture of test means. Capabilities and functions of the entities (while interacting with SOI) and their exchanges are defined using these diagrams.

The activities defined are only between entities. Test means is introduced to the model at the next level, system level. Functions of test means could be a part of the activities, which have been defined in this level or be separated.

a. [OCB] Operational capability diagram

Figure 18 shows the operational capability diagram which is used to specify entities, actors and operational capabilities.

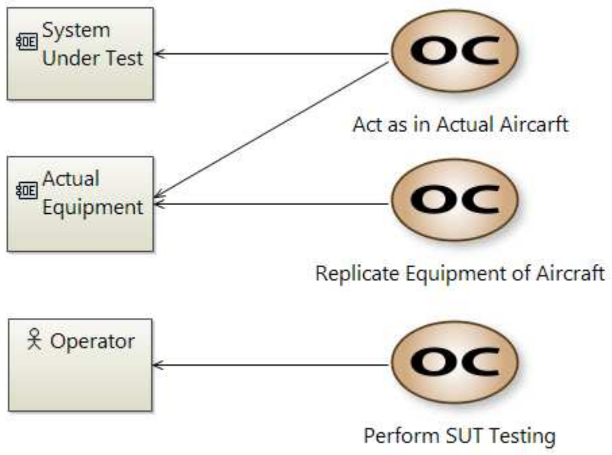


Figure 18. [OCB] Operational capability diagram

Operational capability represents the task that the entities are capable of performing for the system. Entities are the systems, groups or organizations interacting with SOI. Actors are the people who are using the SOI. The entities and actor of the test means are:

In Figure 18, system under test is one of the entities. Depending on the test means, the system under test (SUT) is different**.** SUT could be a combination of FCCs or a single one. What the SUT does for the system is to response to the input signal sent by the system. Since the system is replicating the aircraft environmental behavior, the SUT is acting as it is in an actual aircraft. Thus its capability is to “act as in actual aircraft”.

The actual equipment is basically the actuators and cockpit instrument which mimics the aircraft’s instrument like actuators or cockpit instrument. Actual equipment’s installation is a part of test means (section 3.2.2.) whereas the equipment itself is an entity. They could be parts of SUT along with FCCs and be tested. If so, their capabilities would be the same as SUT’s. However, if they are applied to reproduce the aircraft’s environment, their capability would to “replicate equipment of aircraft". Whether the actual equipment is a part of test facilities or it is being tested itself, the design of the test means is the same.

The operator is the person who operates the test means, gives the inputs and monitors the outputs, detects the problem, etc. The operator is an actor of the test means.

b. [OAB] Operational architecture diagram

Operational architecture diagram shows the operational activities and the interactions which are performed by the entities and actors. Operational activities are allocated to entities and actor. Figure 19 illustrates the activities that each entity and actor is expected to perform while interacting with the SOI (test means).

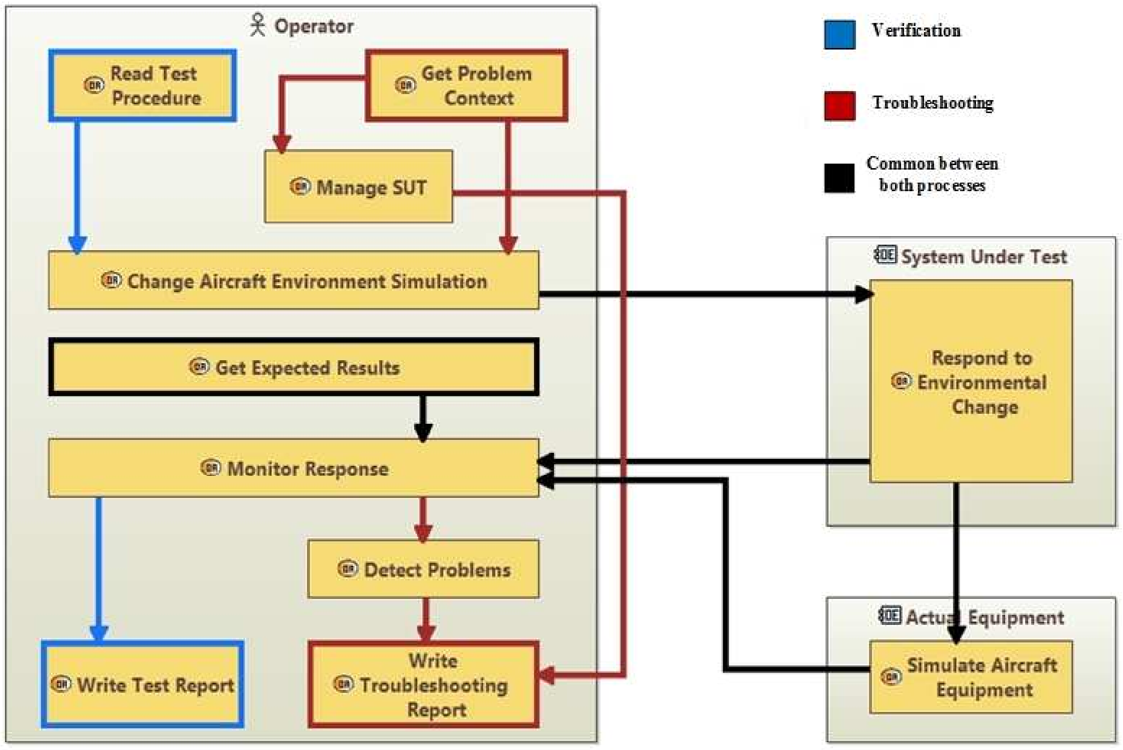


Figure 19. [OAB] Operational architecture diagram

Operational processes are series of activities and interactions that are performed consecutively to fulfill a purpose. There are two operational processes shown in Figure 19.

- Verification: Once a new FCC is programmed, it must be tested to assure that it works correctly and meets the requirements. In Figure 19, the operational process is represented by blue lines. The process starts by the operator who reads the test procedure, sets the simulation input, then the FCC responds to the inputs and the operator monitors the response and compare them with the expected results. Finally, the results are to be documented. If there is actual equipment, it has to respond to the changes of simulation and will be monitored by the operator.

- Troubleshooting: in case FCC has a problem while it is in operation, the test means is used to identify the origin of the fault. The exchanged signals would be detected and the problem will be caught. The process starts with operator running the test. He has the expected results that should be sent by the FCC (output of FCC). If the response of the test means does not match the expected result, there is a problem with software. To detect the problem, operator must read the internal signal of FCCs software using SUT management system. Once the problem is detected, it will be reported to the system team to be solved. The troubleshooting process is represented by the red lines in Figure 19.

These two operational processes are the objectives of developing test means. The black lines show those operational activities that are common between both processes. Basically, for test means design, there is no difference between these operational activities. But the activities of actor (operator) are different for each objective. Objectives in operational level are introduced as missions in system analysis level which will be explained in the next section.

4.1.2. System analysis level

In System Analysis level, the tasks of the system while interacting with entities are identified. System analysis mostly focuses on the functional exchanges of the system and the entities. There are four types of diagrams used in this level:

- [MCB] Mission and capabilities diagram
- [SAB] System architecture diagram
- [SDFB] System dataflow diagram
- [SFBD] System functional breakdown diagram

a. [MCB] Mission and capabilities diagram

To specify the mission of a system, this question needs to be answered: What customer pays for when purchasing the system? By this definition, missions of the test means would be troubleshooting and verification of the SUT. As defined in section 4.1.1., this could be for new software or hardware of the SUT.

How does the system realize the mission? Answer to this question will specify the capability. Capability can also be defined as this: “It is the ability of the system to provide a service that supports the achievement of high-level operational goals.” [36]

Figure 20 shows the mission and capabilities diagram of test means. Test means has three capabilities: “simulating signal”, “detecting response” and “managing the SUT”. Simulating signal is the task of generating the signal input and sending it to SUT. Detecting response refers to the act of receiving the output signals from SUT and running the simulation based on that. Lastly, managing the SUT consists of several tasks like modifying and detecting the variables of the software, maintenance, etc. (section 3.2.2.1/ software management computer)

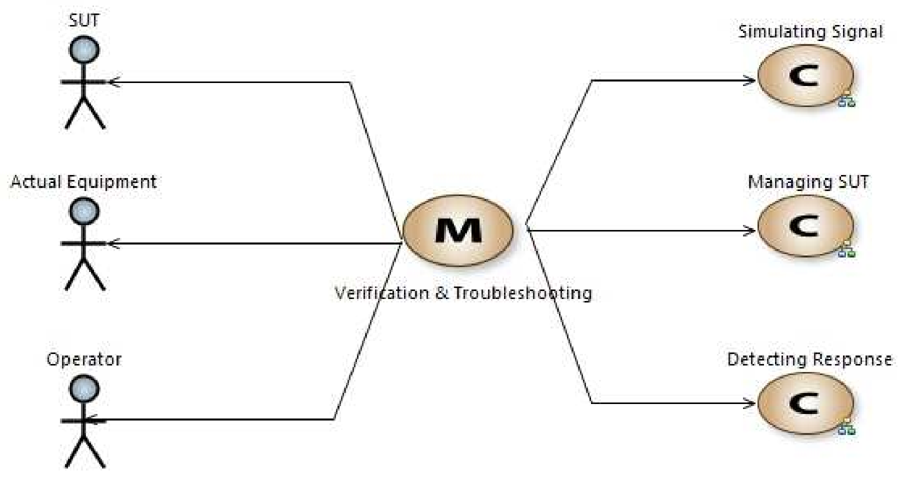


Figure 20. [MCB] Mission and capabilities diagram

b. [SAB] System architecture diagram

In the system architecture diagram, the functions of the system of interest (test means) and actors are defined. Note that, according to Capella definition, from system level to physical architecture level, the word “actor” refers to both “actor” and “entity” defined in operational analysis. To keep the model consistent, each actor defined in system level is referenced to its counterpart in operational level.

To generate well-structured models, for each capability, there are separate diagrams (of same type) modeled in each level. For example, in system level, there are four diagrams of system architecture type [SAB]. Three of them are specifically for the capabilities mentioned in Figure 20. The generic one which is an overview diagram, represents of all these capabilities together, but in high-level functions.

1. [SAB] General model
2. [SAB] Simulating signal
3. [SAB] Detecting response
4. [SAB] Managing SUT

For example, in general model [SAB] (Figure 22), the high-level function "Simulate Aircraft Environment" consists of four sub-functions: "Simulate aircraft equipment", "Simulate invalidation", "Simulate internal signal failure" and "simulate power supply". These sub-functions are modelled in the [SAB] simulating signal (Figure 23).

Figure 21, is used to specify breakdown relations between functions. In general model, the function "simulate aircraft environment" is defined as a representative of all sub-functions. However, in detailed model, sub-functions are employed.

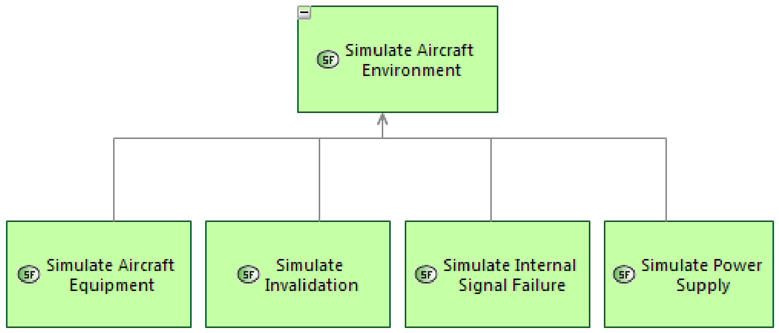

Simulate Aircraft Environment
Simulate Aircraft Equipment
Simulate Invalidation
Simulate Internal Signal Failure
Simulate Power Supply

Figure 21. [SFBD] Functional breakdown of system analysis level

[SAB] General model

The general model provides overview of all the functions related to all capabilities together in one diagram. As the diagram, which contains all sub-functions, makes the model large and hard to perceive, high-level functions have been introduced in a general model.

An example of the breakdown of a high-level function ("Simulate aircraft equipment") into detail is shown in functional breakdown diagram [SFBD], Figure 21.

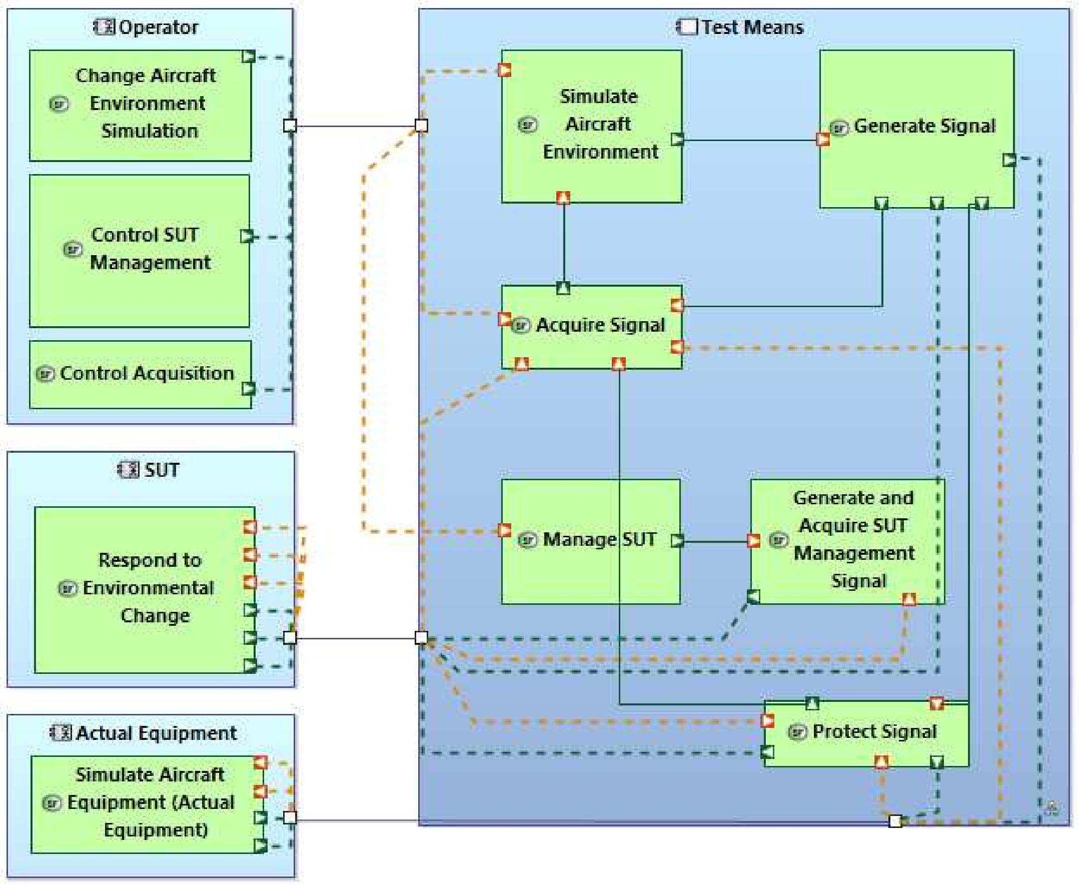

Operator
Change Aircraft Environment Simulation
Control SUT Management
Control Acquisition
SUT
Respond to Environmental Change
Actual Equipment
Simulate Aircraft Equipment (Actual Equipment)
Test Means
Simulate Aircraft Environment
Generate Signal
Acquire Signal
Manage SUT
Generate and Acquire SUT Management Signal
Protect Signal

Figure 22. [SAB] General model diagram

Figure 22 shows the model of the system architecture type of diagram. According to this diagram, test means has six main functions, which will be in more detail in Figure 23 , Figure 24 and Figure 25.

In system level, dark blue blocks represent system of interest while the light blue blocks are the actors. Moreover, functions are shown with green color. The dotted orange lines represent input while green ones show the output of the system (SOI or actors). The solid line between test means and actors represent the “component exchange” between the system. Furthermore, the solid line between functions represent the “functional exchanges” of the functions of the test means.

There are six high-level functions for test means: simulation of aircraft environment, signal acquisition, signal generation, SUT management, SUT management signal generation and protection. Test means receives the command from operator, sends the simulation data to signal generation software. Then, it acquires the SUT's response and sends it back to simulation part (the function "Simulate aircraft equipment") to complete simulation loop. Operator also manages software of SUT by sending command to SUT management system. This system receives the command and sends the management data to signal generation part and then to SUT. The feedback on software management from SUT is also recorded by the SUT management system. The functions defined for the operator are in more detail comparing to what is specified in Figure 19.

[SAB] Simulating signal

Figure 23 shows the functions regarding the capability "Simulating Signal". To start the process, there should be a command from the operator. Typically, there are four types of simulation.

a. The Power Supply Simulation is a simulation of the actual power in the aircraft. It can be electrical and hydraulic power supply. If there are actuators (as actual equipment) in test means, the hydraulic power supply is employed. Otherwise, only electrical power supply is used for test means and actual equipment.

b. The Aircraft Equipment Simulation is the simulation of the equipment existing in the aircraft, interacting with the SUT such as avionics system.

c. The Internal Signal Failure simulation is used to verify the FCC. This function disconnects the connection between two FCCs.

d. The last one is Invalidation Simulation which will create the condition for the SUT to declare itself invalid, for instance, stop the communication between the command and monitor of an FCC channel. The simulation is in the form of data. It needs to be converted to signal. This is done by the function “generate signal”.

If the type of test means is Environmental Qualification, there should be a protection from the lightening, temperature, etc. Protection is a variable feature, which may or may not exist. Another variable feature is “Actual Equipment” which represents actuators and cockpit instrument. This variable feature is an actor.

According to Figure 23, operator changes the input of the simulation (basically starts the simulation or modify the input), then test means simulates the environment by generating data and converts the data to signals which is readable for SUT and sends it to the SUT and actual equipment (if exists). Based on the type of the test means, there might be protection (environmental qualification test means). Thus, there are two functional exchanges coming from “generate signal”. One goes to the protect signal and one directly to the SUT and actual equipment. Only one of these exchanges exist at a time.

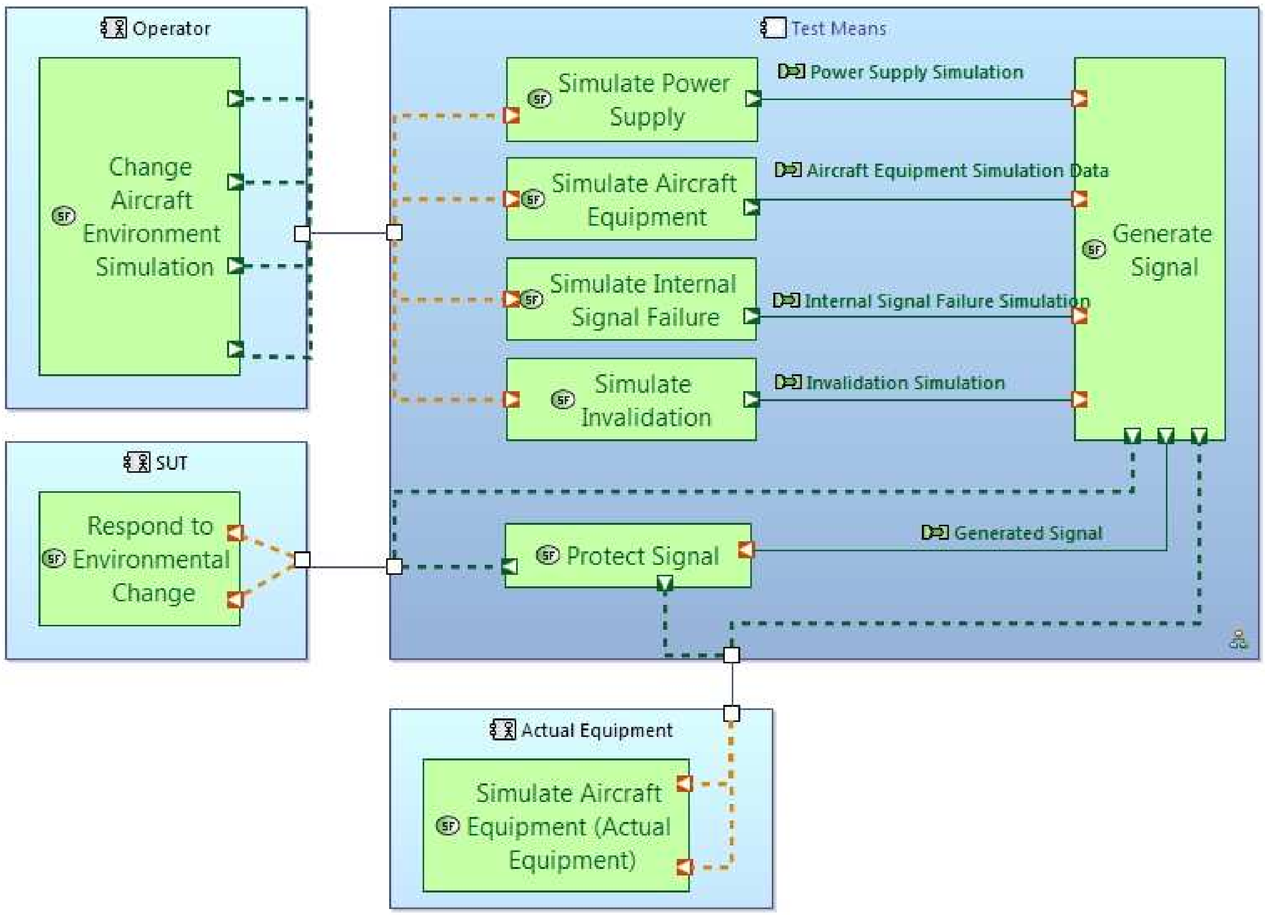

Operator
Change Aircraft Environment Simulation
Test Means
Simulate Power Supply
Power Supply Simulation
Simulate Aircraft Equipment
Aircraft Equipment Simulation Data
Simulate Internal Signal Failure
Internal Signal Failure Simulation
Simulate Invalidation
Invalidation Simulation
Generate Signal
SUT
Respond to Environmental Change
Protect Signal
Generated Signal
Actual Equipment
Simulate Aircraft Equipment (Actual Equipment)

Figure 23. [SAB] Simulating signal diagram

[SAB] Detecting response

The other capability is the "detecting response" which is represented in Figure 24. There are two types of signals detected by the test means and displayed to operator: 1) The simulation signal generated by the test means. 2) The response of SUT. The response signals are read by the operator to check the performance of the SUT. Plus, these signals are detected to keep the simulation loop of the environment according to the SUT response. Therefore, a functional exchange exists between the function “Acquire Signal” and the function “Simulate Aircraft Equipment”. In other words, the response signal is sent to simulation system for the next loop of simulation.

As shown Figure 24 the function “acquire signal” receives signals from SUT and actual equipment and also from the function “generate signal” which is the simulation signal. In an environmental qualification test means, there is no detection of response signal. That is why the “protection”, which is a feature of environmental qualification test means is not included in here. However, detection of simulation signal exists in environmental qualification test means.

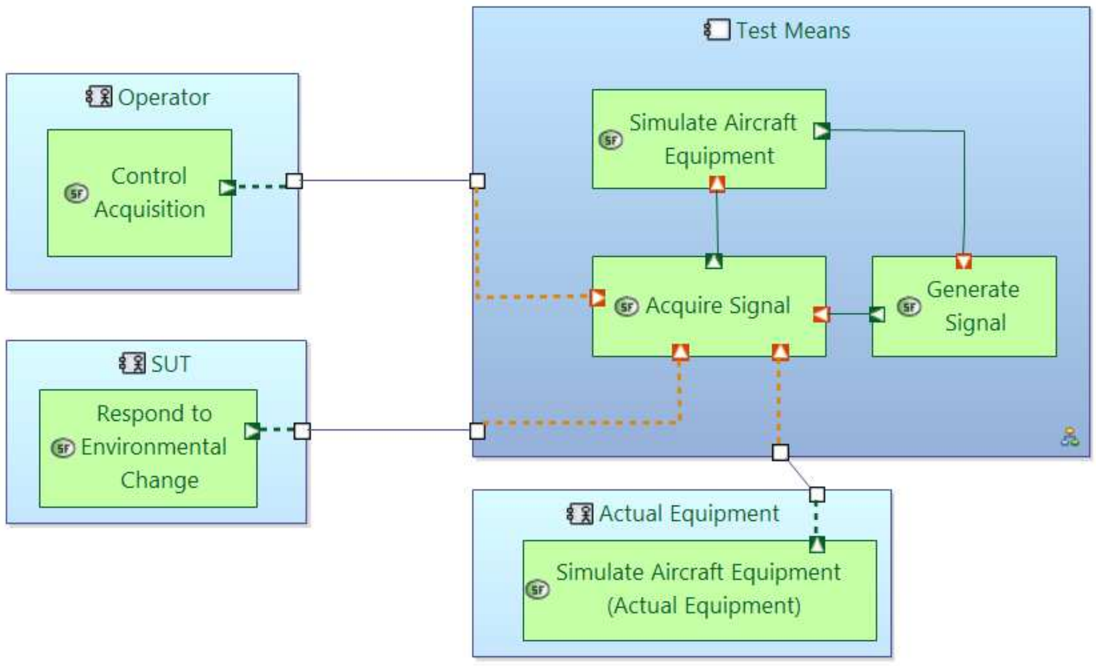

Test Means
Operator
Control Acquisition
Simulate Aircraft Equipment
Acquire Signal
Generate Signal
SUT
Respond to Environmental Change
Actual Equipment
Simulate Aircraft Equipment (Actual Equipment)

Figure 24. [SAB] Detecting response diagram

[SAB] Managing SUT

The capability of managing the SUT software is for modifying the software variable, changing the software version, reading its variable, etc. which is shown in Figure 25. There are six functions for management of the SUT. Each function is fulfilled by a separate software. Operator controls the management through the software control panel. The management data is sent to signal generation and then the SUT. The response is acquired by the system as well.

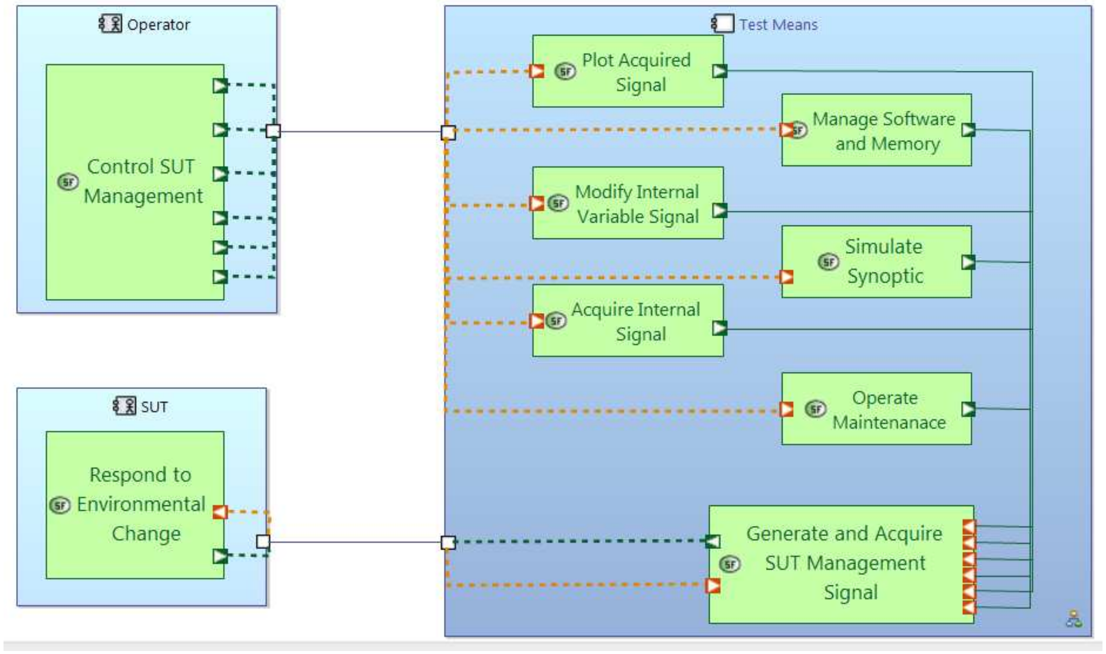

Operator
Control SUT Management
SUT
Respond to Environmental Change
Test Means
Plot Acquired Signal
Manage Software and Memory
Modify Internal Variable Signal
Simulate Synoptic
Acquire Internal Signal
Operate Maintenanace
Generate and Acquire SUT Management Signal

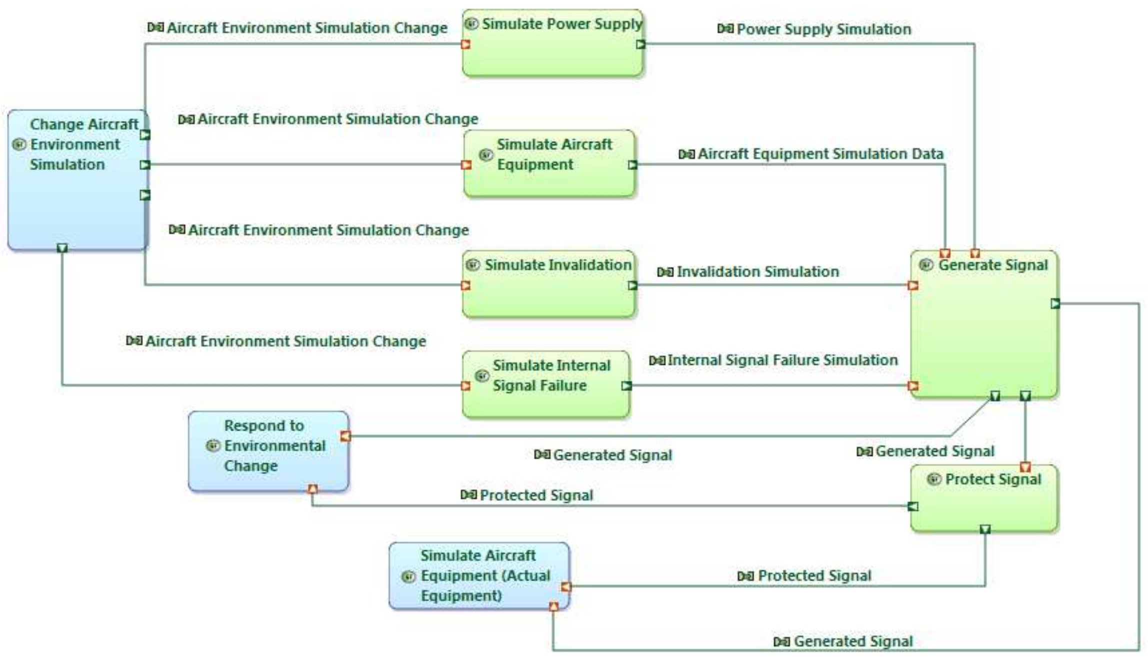
Figure 25. [SAB] Managing SUT diagram

c. [SDFB] Functional dataflow diagram

The SDFB type of diagram shows functions and functional exchanges. In system architecture type of diagram, functional exchanges between test means and other entities are not shown. The dataflow between functions are important to understand as they represent input and output of functions. Note that green blocks represent functions of test means while blue blocks are for functions of entities. Same as system architecture type of diagram, there are four diagrams of this type in the generated specification model.

1. [SDFB] General model
2. [SDFB] Simulating signal
3. [SDFB] Detecting response
4. [SDFB] Managing SUT

Figure 26 shows the dataflow diagram of simulating signal capability. It is the same process as Figure 23 with same functions except that the functional exchanges are displayed as well. The blue functions represent actor's functions, which are allocated to actors in Figure 23 while the green block are functions of the test means. The process starts with operator running the simulation. The test means sends the simulation data to signal generation part and then to the SUT and actual equipment. If there is protection in the architecture, the generated signals would pass through it before entering SUT.

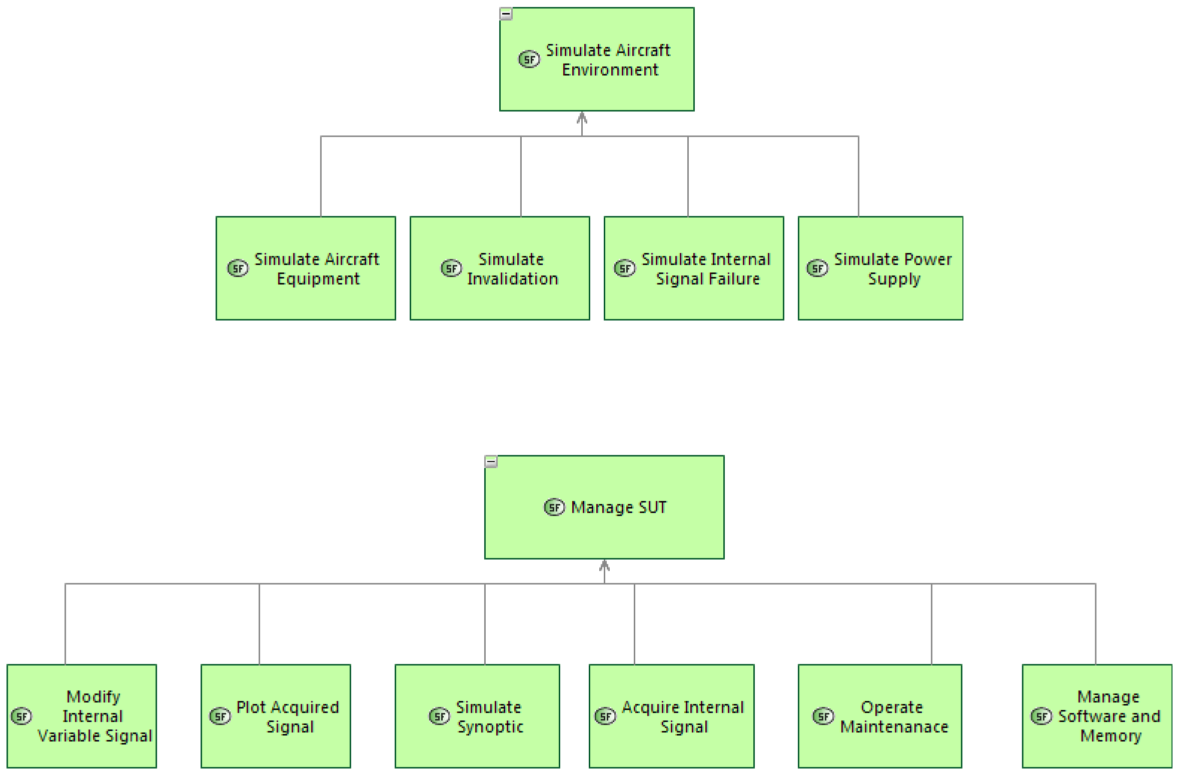


Change Aircraft Environment Simulation
Aircraft Environment Simulation Change
Simulate Power Supply
Power Supply Simulation
Aircraft Environment Simulation Change
Simulate Aircraft Equipment
Aircraft Equipment Simulation Data
Aircraft Environment Simulation Change
Simulate Invalidation
Invalidation Simulation
Generate Signal
Aircraft Environment Simulation Change
Simulate Internal Signal Failure
Internal Signal Failure Simulation
Respond to Environmental Change
Generated Signal
Generated Signal
Protected Signal
Protect Signal
Simulate Aircraft Equipment (Actual Equipment)
Protected Signal
Generated Signal

Figure 26. [SDFB] Simulating signal diagram

d. [SFBD] System functional breakdown diagram

Functional breakdown diagram represents the hierarchy of functions. It is helpful to represent how lower level functions are related to a high-level function. In the system level, the two functions. “Simulate Aircraft Environment” and “Manage SUT” are used in generic model and their subsets are in detailed model. Figure 27 shows functional breakdown of system analysis. Part of this diagram is shown in Figure 21.



Simulate Aircraft Environment
Simulate Aircraft Equipment
Simulate Invalidation
Simulate Internal Signal Failure
Simulate Power Supply
Manage SUT
Modify Internal Variable Signal
Plot Acquired Signal
Simulate Synoptic
Acquire Internal Signal
Operate Maintenanace
Manage Software and Memory

Figure 27. [SFBD] System functional breakdown diagram

There are more diagrams modelled in system level which are shown in appendix A. In system level, the high-level functions and functional exchanges have been defined. Moreover, mission and capabilities of the test means are identified. The next section discusses the logical level.

4.1.3. Logical architecture level

In the logical level, the components of the system and their implementations are modeled and functions are allocated to them. These components are logical components, meaning that the type and the quantity of them are not discussed.

In ARCADIA methodology, logical level is where the internal functions are defined. The term "internal function" refers to the functions that are allocated to different components of the system of interest (SOI). The functions in this level are specified in more detail. There are three main types of diagrams that are used in this level.

- [LAB] Logical architecture diagram
- [LDFB] Logical dataflow diagram
- [LFBD] Logical functional breakdown diagram

The logical architecture [LAB] diagram represents the allocation of the functions to components. The dataflow model, [LDFB], represents the functions and functional exchanges of different components.

There are several equipment of the aircraft to be simulated. Depending on the mission, only some of them exist. Hence, the functions of these simulations are separated into more detailed functions so the variability is managed easier. The breakdown is shown in functional breakdown diagram [LFBD]. Following sections explain each type in detail.

a. [LAB] Logical architecture diagram

Logical Architecture is a diagram consisting of the components and allocated functions. Same as the system level, modelling all functions and components in one diagram is not feasible. Thus, there are four different architecture diagrams. These diagrams are similar to those in system level, except that components are introduced and functions are more specific:

1. [LAB] General model
2. [LAB] Simulating signal
3. [LAB] Detecting response
4. [LAB] Managing SUT

[LAB] General model

The general model involves all the main functions of the system in logical level. The purpose of this diagram is to have an overview of the logical components and their functions at once. Figure 28 represents all the logical component of the test means and their functions. Within, there are the main three computers (simulation, acquisition and SUT management), which are explained in section 3.2.2. Inside each computer, there are blocks, which represent the various software installed. The components modelled in Figure 28 are representing the components introduced in Figure 17. Only the power supply is different: here in the logical level, it is defined as one component. It will be broken into two components at physical level.

In general model, all the components are represented. However, type of the components is not identified. The dark blue blocks represent test means and its components. The light blue blocks represent actors and green ones are functions.

As the model is consistent, the functions and components introduced at high level must exist in lower level whether themselves or their derivatives. Moreover, any functions or components that are in the model must be part of the upper level function/ components. Thus, all the functions defined here are traced back to system level's functions. For example, the red-framed blocks represent the functions that are referenced to the function ”simulate power supply” in Figure 23. In other words, one function in system level is represented by four functions in logical level. However, it is possible that a function is not broken in several ones as the levels proceed. Because the specified function in upper level is elaborative and generic. For instance, all the functions of SUT management system stayed as represented in system level (Figure 25.).
The defined components are based on the architecture of test means in Thales. For test means developed by other companies, the logical components might differ slightly. The presented model can be considered a generic representation of FCC test means.

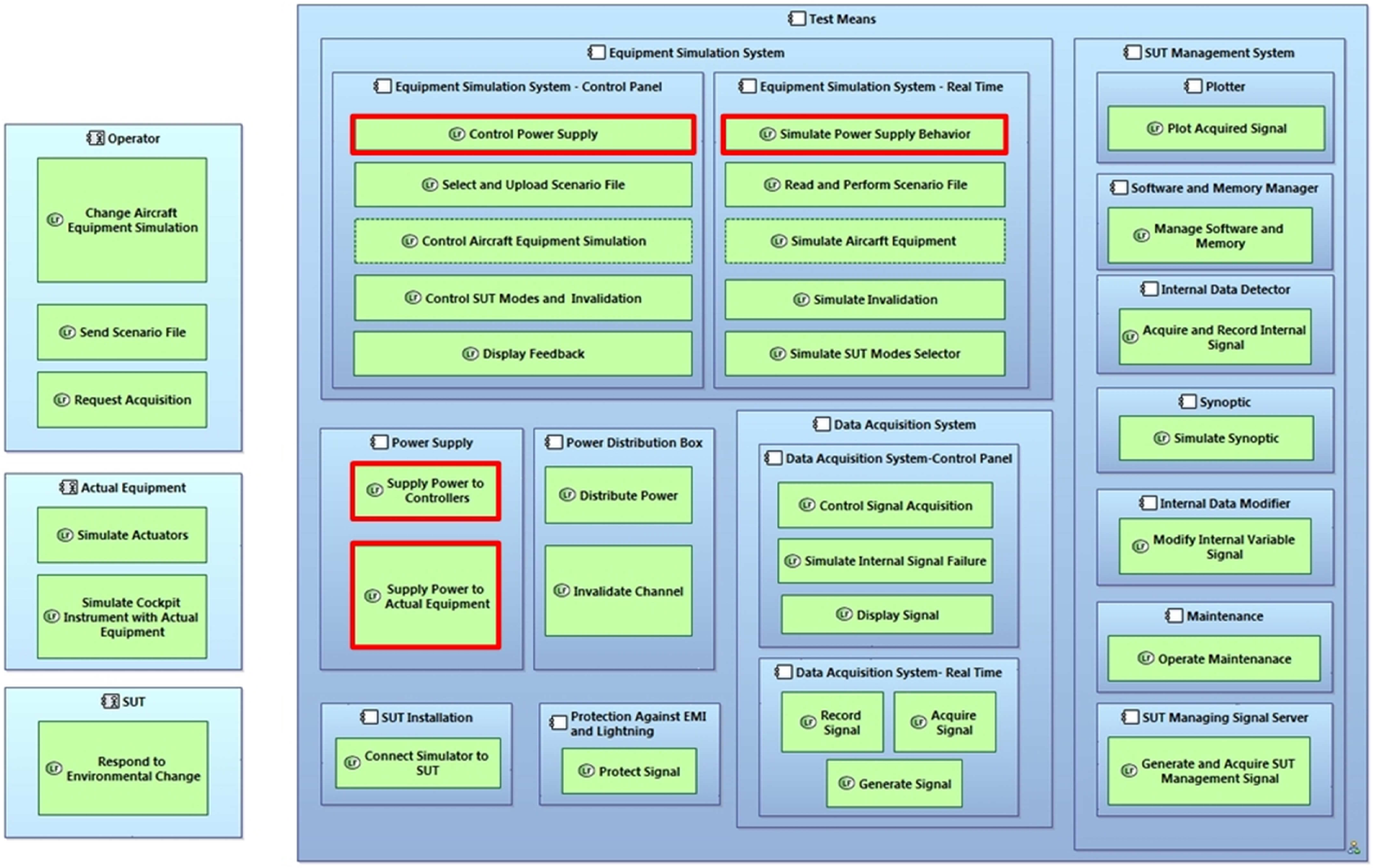

Operator
Change Aircraft Equipment Simulation
Send Scenario File
Request Acquisition
Actual Equipment
Simulate Actuators
Simulate Cockpit Instrument with Actual Equipment
SUT
Respond to Environmental Change
Test Means
Equipment Simulation System
Equipment Simulation System - Control Panel
Control Power Supply
Select and Upload Scenario File
Control Aircraft Equipment Simulation
Control SUT Modes and Invalidation
Display Feedback
Equipment Simulation System - Real Time
Simulate Power Supply Behavior
Read and Perform Scenario File
Simulate Aircarft Equipment
Simulate Invalidation
Simulate SUT Modes Selector
Power Supply
Supply Power to Controllers
Supply Power to Actual Equipment
Power Distribution Box
Distribute Power
Invalidate Channel
Data Acquisition System
Data Acquisition System-Control Panel
Control Signal Acquisition
Simulate Internal Signal Failure
Display Signal
Data Acquisition System- Real Time
Record Signal
Acquire Signal
Generate Signal
SUT Installation
Connect Simulator to SUT
Protection Against EMI and Lightning
Protect Signal
SUT Management System
Plotter
Plot Acquired Signal
Software and Memory Manager
Manage Software and Memory
Internal Data Detector
Acquire and Record Internal Signal
Synoptic
Simulate Synoptic
Internal Data Modifier
Modify Internal Variable Signal
Maintenance
Operate Maintenanace
SUT Managing Signal Server
Generate and Acquire SUT Management Signal

Figure 28. [LAB] General model diagram

[LAB] Simulating signal

Figure 29 repesents the architecture of "simulating signal" capability in logical level. In logical level, the simulation function is broken into control panel and real-time. The Equipment Simulation System consists of two parts: the interface, which enables user to manage the simulation and the real-time part, where the simulation data are generated.

There are two ways to provide inputs for simulation:

- Using a scenario file: The user prepares the file containing the scenario. The file is loaded to the simulation system and executed. During the execution, the signals from scenario file cannot be modified.
- Providing manual input through user interface: the signals, which do not exist in scenario file are modifiable through control panel while the test is running. For instance, if the operator wants to simulate aircraft turning at different altitude, the simulation signals of turning are in scenario files, however, the altitude would be a variable controlled by the operator through the control panel.

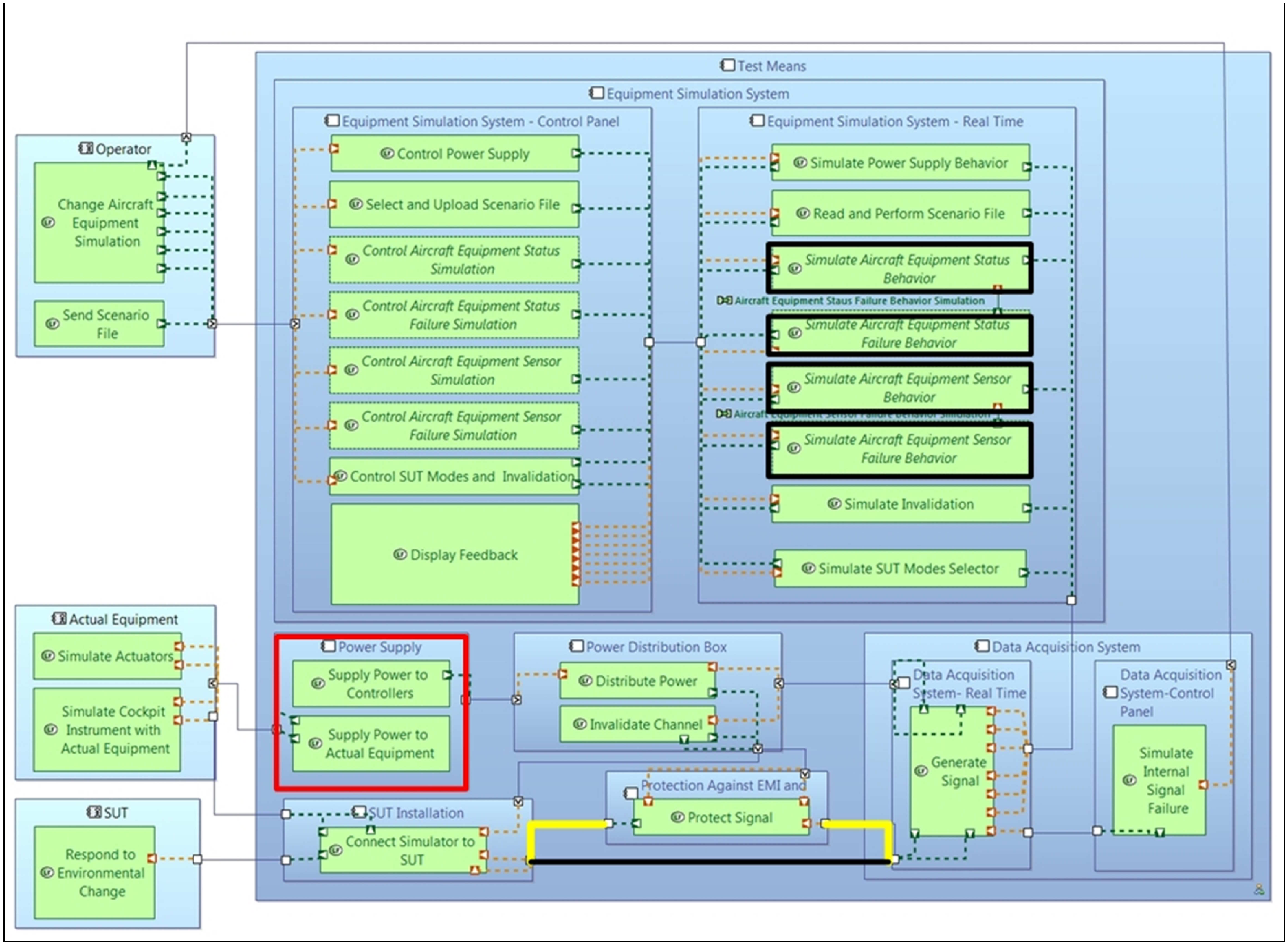
The control panel provides platform for scenario file to be selected and uploaded. It is then executed and read by the real time simulation.

The function "display feedback" in control panel of simulation system indicates the representation of simulation data to the operator through control panel.

In this diagram, some of the functions of general model (Figure 28) have been broken into sub-functions. For example, the black-framed blocks represent the sub-functions of the generic logical function "Simulate aircraft equipment" in real-time simulation system in Figure 28.

As mentioned in section 3.2.2., when there is actual equipment, there should be a separated power supply for the actual equipment. The type of the power supplies for actual equipment is hydraulic and electrical. That is why there are two different functions for power sources shown in red-framed block in Figure 29.

The process starts by operator performing the test through control panel. Simulation data is sent to real-time part of data acquisition system. Invalidation and power supply simulation signals proceeds to PDB and others is sent to SUT through “SUT installation”. If there is a protection the signals pass through that before entering SUT installation (yellow line). If not it goes straight to SUT installation (black line).



Test Means
Equipment Simulation System
Equipment Simulation System - Control Panel
Control Power Supply
Select and Upload Scenario File
Control Aircraft Equipment Status Simulation
Control Aircraft Equipment Status Failure Simulation
Control Aircraft Equipment Sensor Simulation
Control Aircraft Equipment Sensor Failure Simulation
Control SUT Modes and Invalidation
Display Feedback
Equipment Simulation System - Real Time
Simulate Power Supply Behavior
Read and Perform Scenario File
Simulate Aircraft Equipment Status Behavior
Aircraft Equipment Staus Failure Behavior Simulation
Simulate Aircraft Equipment Status Failure Behavior
Simulate Aircraft Equipment Sensor Behavior
Simulate Aircraft Equipment Sensor Failure Behavior
Simulate Invalidation
Simulate SUT Modes Selector
Operator
Change Aircraft Equipment Simulation
Send Scenario File
Actual Equipment
Simulate Actuators
Simulate Cockpit Instrument with Actual Equipment
SUT
Respond to Environmental Change
Power Supply
Supply Power to Controllers
Supply Power to Actual Equipment
Power Distribution Box
Distribute Power
Invalidate Channel
Protection Against EMI and
Protect Signal
SUT Installation
Connect Simulator to SUT
Data Acquisition System
Data Acquisition System- Real Time
Generate Signal
Data Acquisition System-Control Panel
Simulate Internal Signal Failure

Figure 29. [LAB] Simulating signal

[LAB] Detecting signal

As mentioned in section 4.1.2., there are two types of signal detection. One detection of the response from the SUT and for detection of the simulation. Figure 30 shows the logical architecture of "detecting signal" capability which traces back to Figure 24. The function "acquire signal" is performed by real time part of data acquisition system. It receives both type of signals and sends them to the control panel of the acquisition system for display. It also sends the response signal to the simulation system. As explained in system level, the acquisition of SUT's response does not exist in environmental protection test means,

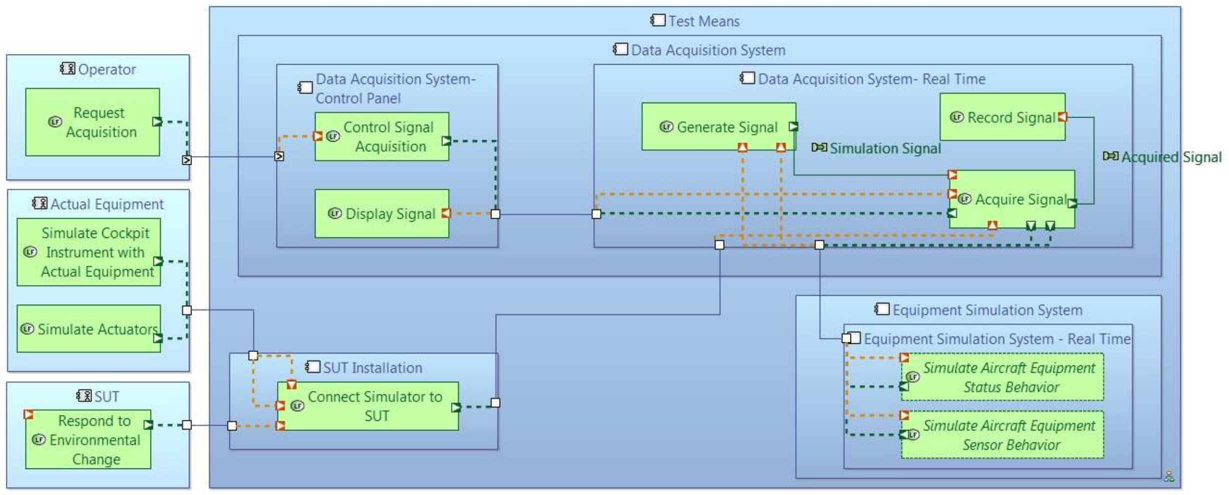

Test Means
Data Acquisition System
Operator
Request Acquisition
Data Acquisition System- Control Panel
Control Signal Acquisition
Display Signal
Data Acquisition System- Real Time
Generate Signal
Record Signal
Simulation Signal
Acquired Signal
Acquire Signal
Actual Equipment
Simulate Cockpit Instrument with Actual Equipment
Simulate Actuators
SUT
Respond to Environmental Change
SUT Installation
Connect Simulator to SUT
Equipment Simulation System
Equipment Simulation System - Real Time
Simulate Aircraft Equipment Status Behavior
Simulate Aircraft Equipment Sensor Behavior

Figure 30. [LAB] Detecting response diagram

[LAB] Managing SUT

As described in section 4.1.2., the "managing SUT" capability involves six functions. For each function, there is a software with an interface. Figure 31 shows the logical architecture diagram of this capability. Each software is represented by one block and one function. There is a component called “SUT managing signal server” which has two functions: generation and acquisition of signals which is the real time part of the SUT management system. In general model (Figure 28) there is one function representing these two functions which is “generate and acquire SUT management signal”.

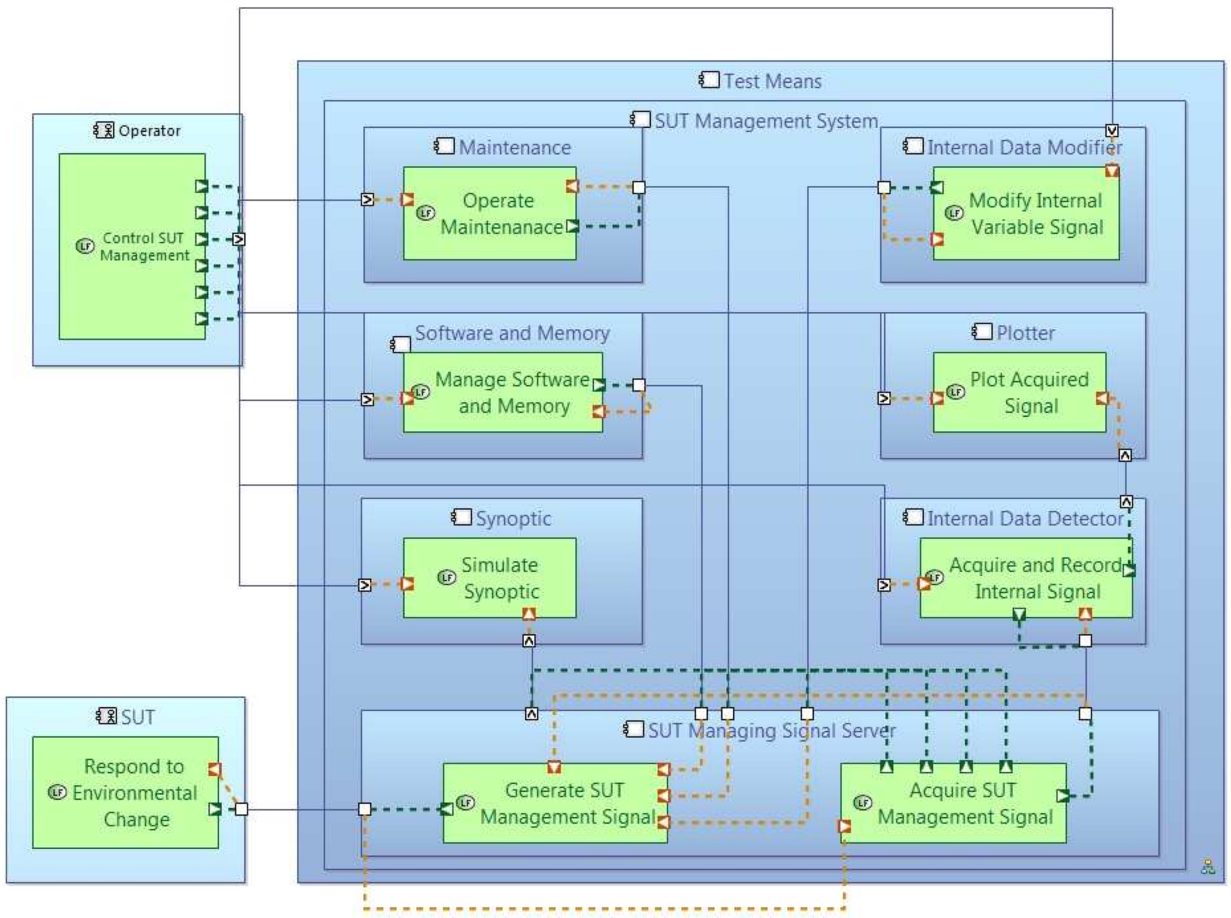


Figure 31. [LAB] Managing SUT diagram

b. [LDFB] Functional dataflow diagram

The dataflow type of diagram represents functions and functional exchanges. In logical architecture type of diagram, functional exchanges between components and between components and actors are not shown. In dataflow diagrams, all of functional exchanges are shown. Note that Green blocks represent functions of test means while Blue blocks are for functions of entities. Same as system level, there are three diagrams of dataflow type modelled in this level which are:

1. [LDFB] Simulating signal

2. [LDFB] Detecting response

3. [LDFB] Managing SUT

[LDFB] Simulating signal

The Figure 32 represents the difference between the use of the scenario file and the manual input. In the scenario file simulation, the test means reads and performs the scenario file, then the command is sent to real time part to produce simulation data. In simulation through control panel, the command is sent by the operator. In Figure 32, the blue lines represent the procedure of simulation using control panel while the red line shows the simulation process through scenario file.

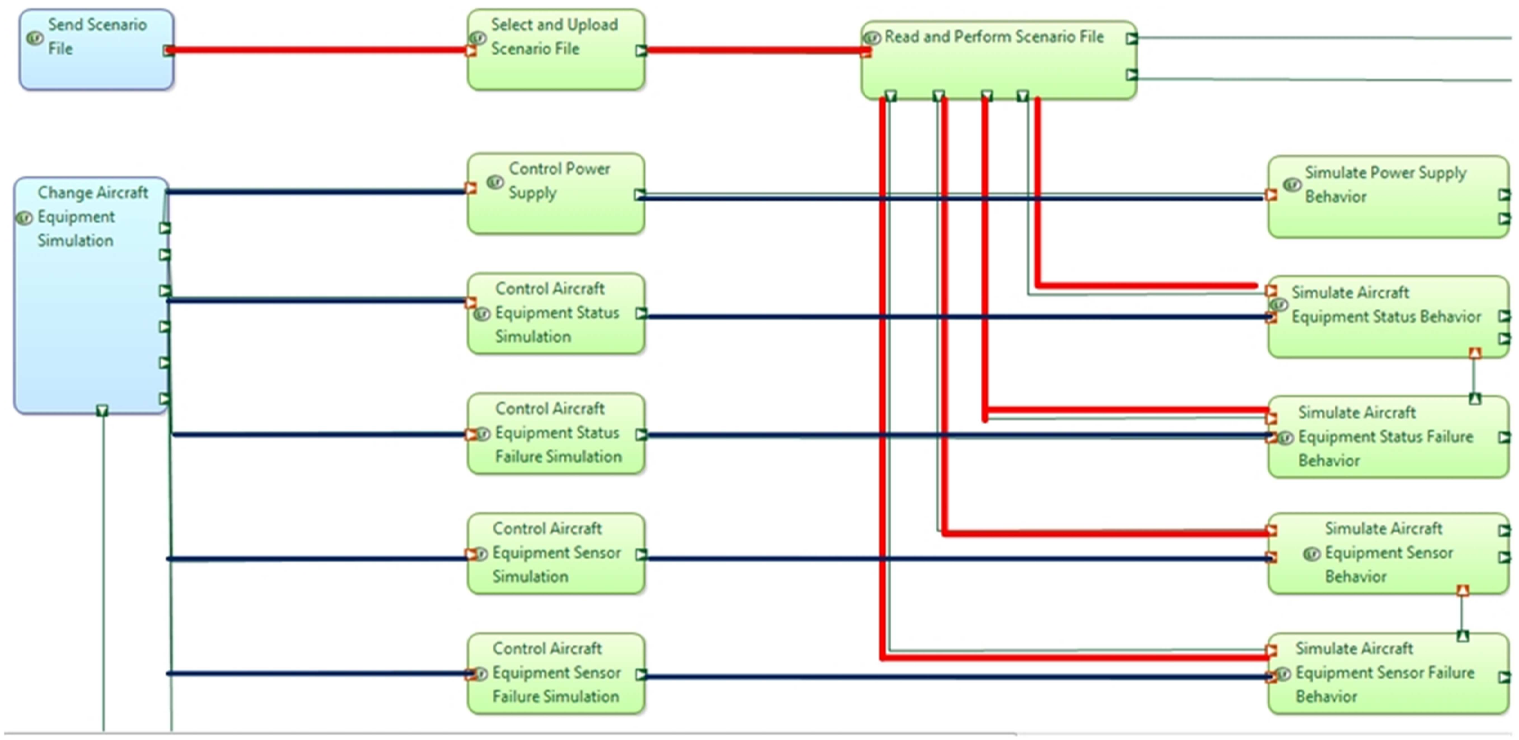

Send Scenario File
Select and Upload Scenario File
Read and Perform Scenario File
Change Aircraft Equipment Simulation
Control Power Supply
Simulate Power Supply Behavior
Control Aircraft Equipment Status Simulation
Simulate Aircraft Equipment Status Behavior
Control Aircraft Equipment Status Failure Simulation
Simulate Aircraft Equipment Status Failure Behavior
Control Aircraft Equipment Sensor Simulation
Simulate Aircraft Equipment Sensor Behavior
Control Aircraft Equipment Sensor Failure Simulation
Simulate Aircraft Equipment Sensor Failure Behavior

Figure 32. [LDFB] Scenario file vs. manual simulation

[LDFB] Detecting signal

As mentioned earlier, there are two types of signal detection: SUT`s response detection and simulation signal detection. These two types of detection are shown using functional chains (functional chain describes a path through the functions) which are explained in this section.

- SUT response detection functional chain

The response signal is sent to the acquisition system (acquire signal function) through the SUT installation. Then it is recorded and displayed to the operator. Figure 33 represents this procedure.

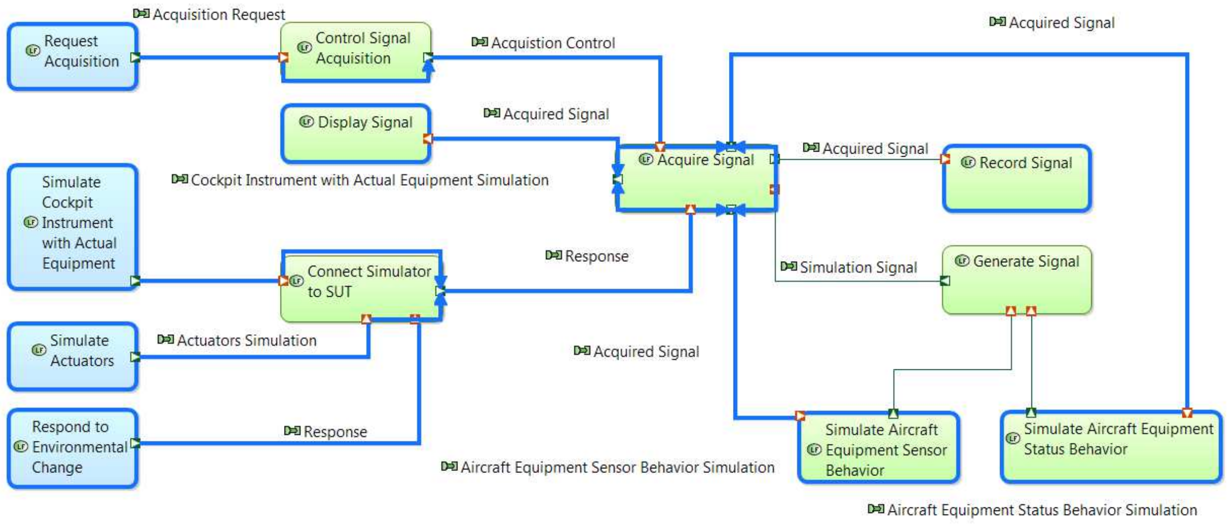

Request Acquisition
Acquisition Request
Control Signal Acquisition
Acquistion Control
Acquired Signal
Display Signal
Acquired Signal
Acquire Signal
Acquired Signal
Record Signal
Simulate Cockpit Instrument with Actual Equipment
Cockpit Instrument with Actual Equipment Simulation
Connect Simulator to SUT
Response
Simulation Signal
Generate Signal
Simulate Actuators
Actuators Simulation
Acquired Signal
Respond to Environmental Change
Response
Aircraft Equipment Sensor Behavior Simulation
Simulate Aircraft Equipment Sensor Behavior
Simulate Aircraft Equipment Status Behavior
Aircraft Equipment Status Behavior Simulation

Figure 33. [LDFB] Detecting response signal

- Simulation signal detection functional chain

The simulation data is sent to generation signal and then is acquired by the system to be displayed to operator. Figure 34 shows this procedure in the "detecting signal" of dataflow type of diagram.

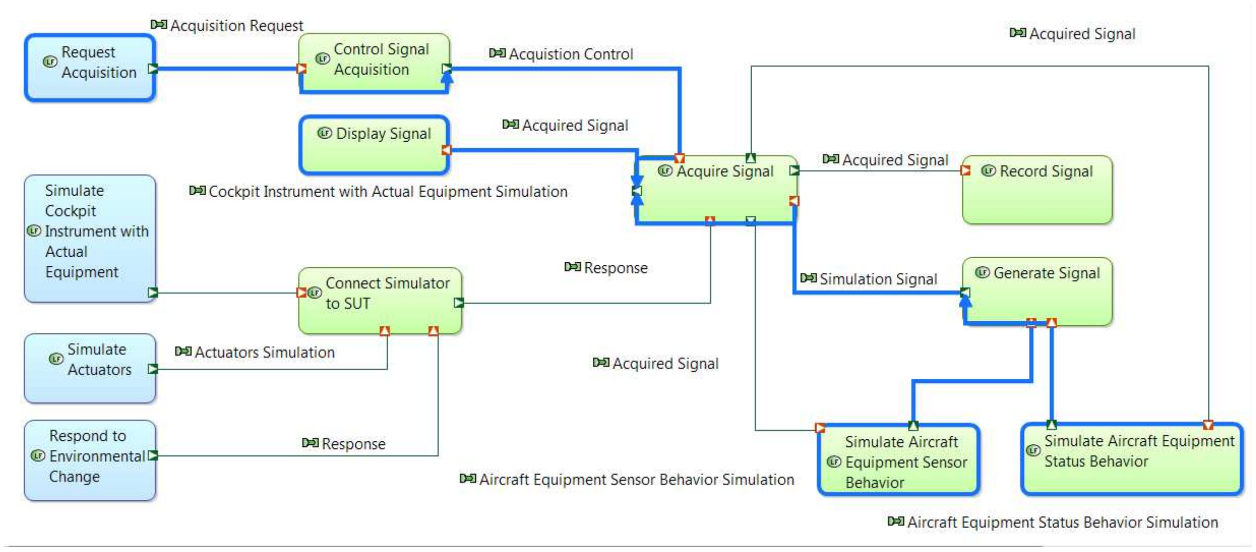

Request Acquisition
Acquisition Request
Control Signal Acquisition
Acquistion Control
Acquired Signal
Display Signal
Acquired Signal
Acquire Signal
Acquired Signal
Record Signal
Simulate Cockpit Instrument with Actual Equipment
Cockpit Instrument with Actual Equipment Simulation
Connect Simulator to SUT
Response
Simulation Signal
Generate Signal
Simulate Actuators
Actuators Simulation
Acquired Signal
Respond to Environmental Change
Response
Aircraft Equipment Sensor Behavior Simulation
Simulate Aircraft Equipment Sensor Behavior
Simulate Aircraft Equipment Status Behavior
Aircraft Equipment Status Behavior Simulation

Figure 34. [LDFB] Detecting simulation signal

c. [LFBD] Functional breakdown diagram

In the generic model (Figure 28), two functions represent the simulation of equipment: "Control Aircraft Equipment Simulation" and "Simulate Aircraft Equipment". Each of these functions is broken down into four functions in "simulating signal" model. The relations are well defined in breakdown diagrams. There are two breakdown diagrams at this level. One showing the sub-functions of "Control Aircraft Equipment Simulation" and one for "Simulate Aircraft Equipment":

[LFBD] Aircraft equipment simulation breakdown: It is for real-time simulation part is where the simulation data is generated. The response from SUT is detected by the same part and based on that new simulation data is calculated and the loop of simulation continues.

[LFBD] Aircraft equipment simulation control breakdown: It is for simulation control of the equipment simulation system which is the control panel enabling operator to command and receive the simulation data parameter.

As mentioned in section 3.2.2, there are two types of data to simulate: status and sensor. Status simulation refers to the simulation of the data like air data (temperature, pressure, etc.) or inertial reference data (position of the aircraft). Sensor simulation refers to data that is sent to the FCC by the sensors in aircraft such as cockpit sensor or actuation sensor. Note that some equipment has both sensor and status simulation. The status and sensor data must have the same value of parameter. However, as the status affects other sensors, in test means status and sensor data are separated to be able to change or disable the sensor without changing the status. However, change/failure of one sensor parameter does not affect the status data and the other sensors. Thus, they have been separated to avoid conflict. Figure 35 shows the breakdown of the simulation functions of control panel. At first, the function "control aircraft equipment simulation" is broken into four sub-functions: status, sensors and their failures. These four are presented in Figure 29. However, in general model (Figure 28) only the high-level function, "control aircraft equipment simulation", is depicted. Cockpit instrument, inertial reference system, actuators and surfaces are the systems that have both status and sensor part. The air data system, however, is only status and aircraft system is only sensors.

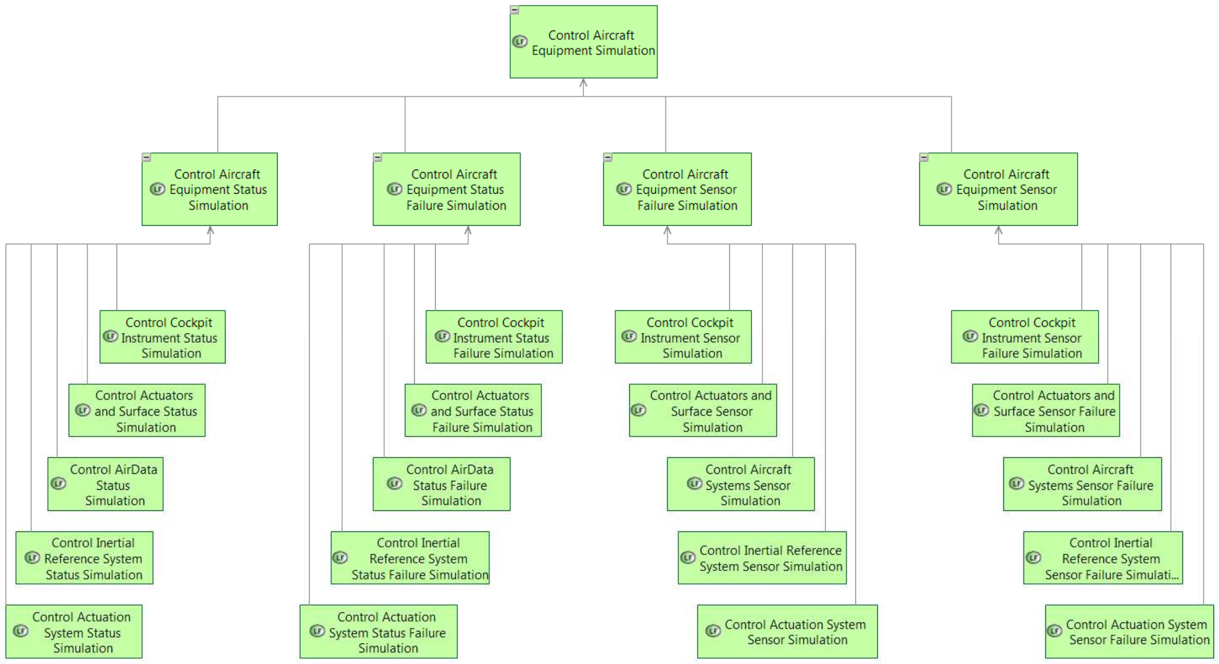

Control Aircraft Equipment Simulation
Control Aircraft Equipment Status Simulation
Control Aircraft Equipment Status Failure Simulation
Control Aircraft Equipment Sensor Failure Simulation
Control Aircraft Equipment Sensor Simulation
Control Cockpit Instrument Status Simulation
Control Actuators and Surface Status Simulation
Control AirData Status Simulation
Control Inertial Reference System Status Simulation
Control Actuation System Status Simulation
Control Cockpit Instrument Status Failure Simulation
Control Actuators and Surface Status Failure Simulation
Control AirData Status Failure Simulation
Control Inertial Reference System Status Failure Simulation
Control Actuation System Status Failure Simulation
Control Cockpit Instrument Sensor Simulation
Control Actuators and Surface Sensor Simulation
Control Aircraft Systems Sensor Simulation
Control Inertial Reference System Sensor Simulation
Control Actuation System Sensor Simulation
Control Cockpit Instrument Sensor Failure Simulation
Control Actuators and Surface Sensor Failure Simulation
Control Aircraft Systems Sensor Failure Simulation
Control Inertial Reference System Sensor Failure Simulati...
Control Actuation System Sensor Failure Simulation

Figure 35. [LFBD] Functional breakdown of aircraft equipment simulation control

Figure 36 shows the breakdown of functions of real-time simulation . Any function in this diagram has a counterpart in diagram of Figure 35 which means it exists in both control panel and real time parts of simulation system.

The logical level concludes with the breakdown diagrams. There are more functions in this level that can be found in the Appendix A.3. The next level, which is the physical level, involves detail of functions, specificity of components and their implementation.

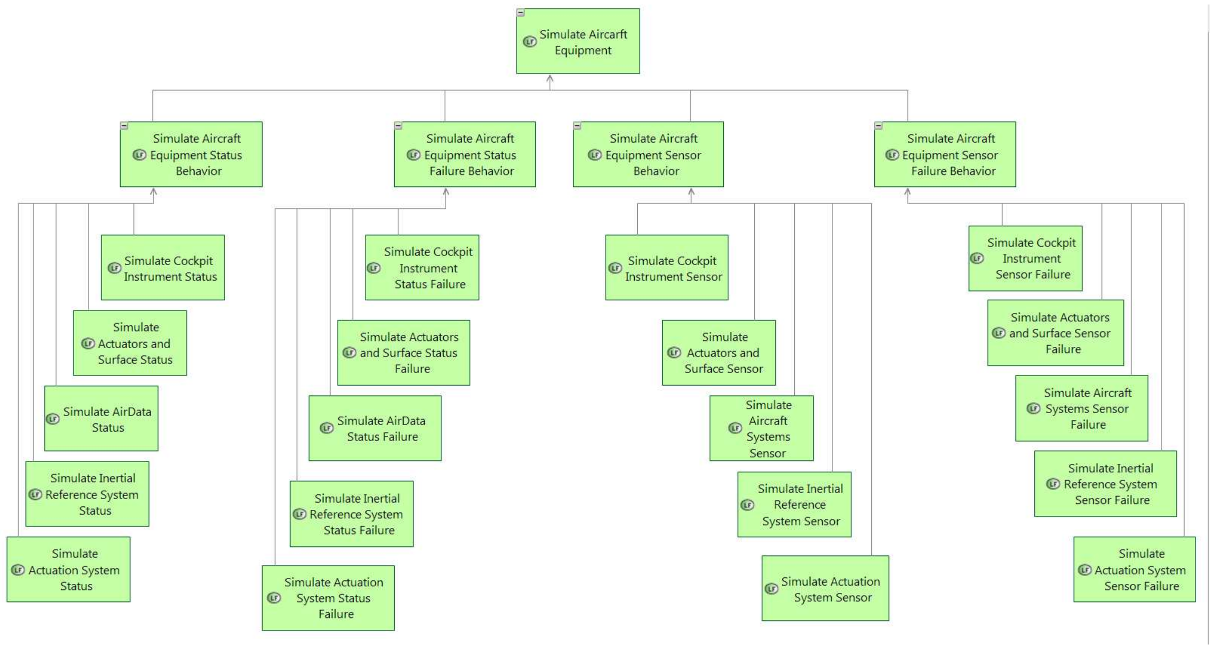

Simulate Aircarft Equipment
Simulate Aircraft Equipment Status Behavior
Simulate Aircraft Equipment Status Failure Behavior
Simulate Aircraft Equipment Sensor Behavior
Simulate Aircraft Equipment Sensor Failure Behavior
Simulate Cockpit Instrument Status
Simulate Actuators and Surface Status
Simulate AirData Status
Simulate Inertial Reference System Status
Simulate Actuation System Status
Simulate Cockpit Instrument Status Failure
Simulate Actuators and Surface Status Failure
Simulate AirData Status Failure
Simulate Inertial Reference System Status Failure
Simulate Actuation System Status Failure
Simulate Cockpit Instrument Sensor
Simulate Actuators and Surface Sensor
Simulate Aircraft Systems Sensor
Simulate Inertial Reference System Sensor
Simulate Actuation System Sensor
Simulate Cockpit Instrument Sensor Failure
Simulate Actuators and Surface Sensor Failure
Simulate Aircraft Systems Sensor Failure
Simulate Inertial Reference System Sensor Failure
Simulate Actuation System Sensor Failure

Figure 36. [LFBD] Functional breakdown of aircraft equipment simulation

4.1.4. Physical architecture level

Physical level is the last modelling level considered in this proposed specification model. It involves the highest level of detail. The following diagrams are used in this level:

- [PAB] Physical architecture diagram
- [PDFB] Physical dataflow diagram
- [PFBD] Physical functional breakdown diagram

In this level, the number of each component, their types (node or behavior) are specified. Node components, which are shown in yellow color, are those components which are not designed by the developer of the system of interest and are purchased from suppliers. Behavior Components, shown as blue, are components that are designed by the developer.

For example, the computer case of test means is purchased. Thus, it is a Node component. The software of the test means is programmed by the developer. therefore, it is a behavior component (Figure 37).

Following section explains the physical level and the diagrams modelled.

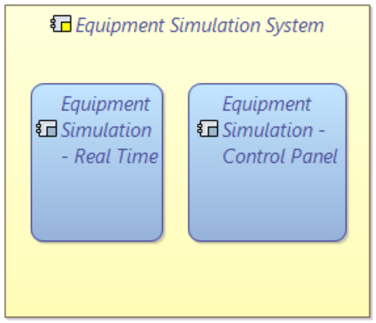

Equipment Simulation System
Equipment Simulation - Real Time
Equipment Simulation - Control Panel

Figure 37. An example of node and behavioral components

a. [PAB] Physical architecture diagram

This type of diagram, contains the components (Node and Behavior component), functions allocated to them and functional exchanges. Same as the two previous levels, there are four diagrams of this type modelled which are:

1. [PAB] General model
2. [PAB] Simulating signal
3. [PAB] Detecting response
4. [PAB] Managing SUT

[PAB] General model

The general model includes all the actors, components (both behavior and node) and their functions. Figure 38 shows the general model of physical level.
There are two SUTs and trays in this model. One tray and one SUT are variants. It means, they only exist in the model if the type of test means is multi-unit. If not, only one of each would be in configuration, representing single-unit type. There could be three SUTs, as well. However, it has not been added, since the implementation is the same.

Comparing Figure 38 and Figure 28, it is perceived that the specified components in the models are the same except for “power supply” and “SUT installation” which are broken into sub-components. In Figure 38, there are two source of power introduced. One for the test means (controller) which is a common component and one for actual equipment which is a variant. There also two functions allocated to actual equipment power supply: hydraulic and electrical. The hydraulic power supply is for actuators (section 3.2.1.).

As specified in Figure 17, there is a SUT installation which includes different components some of which are variant. In logical level, the SUT installation is represented as a single component. However, at this level, it is broken into three components: break-out box and two trays (Figure 39). All these components have the same function, which is to provide installation of SUT to the test means and to route the signal. However, they are different physical implementations. That is why they are introduced as a single component in logical level and broken into detail in physical level.

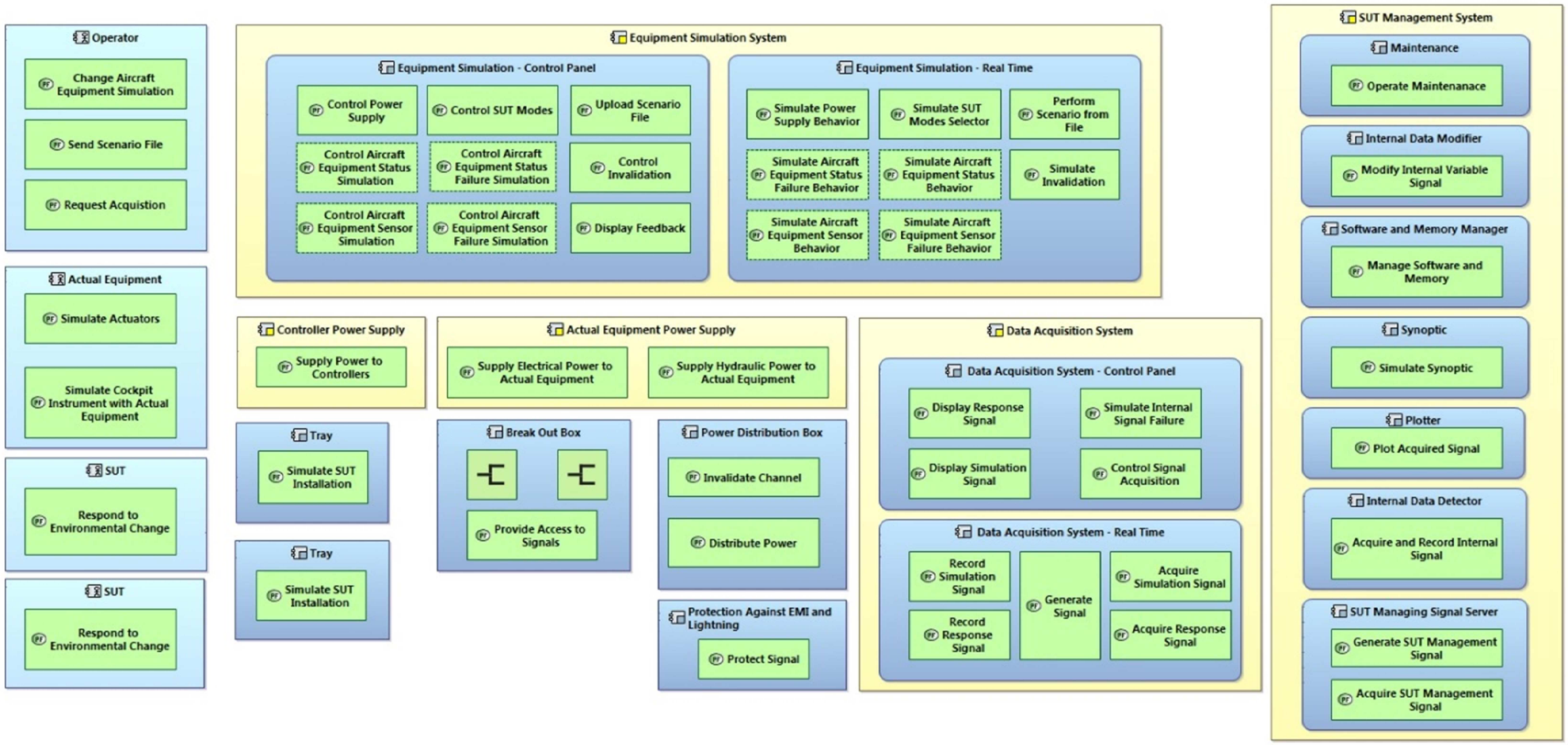

Operator
Change Aircraft Equipment Simulation
Send Scenario File
Request Acquistion
Actual Equipment
Simulate Actuators
Simulate Cockpit Instrument with Actual Equipment
SUT
Respond to Environmental Change
SUT
Respond to Environmental Change
Equipment Simulation System
Equipment Simulation - Control Panel
Control Power Supply
Control SUT Modes
Upload Scenario File
Control Aircraft Equipment Status Simulation
Control Aircraft Equipment Status Failure Simulation
Control Invalidation
Control Aircraft Equipment Sensor Simulation
Control Aircraft Equipment Sensor Failure Simulation
Display Feedback
Equipment Simulation - Real Time
Simulate Power Supply Behavior
Simulate SUT Modes Selector
Perform Scenario from File
Simulate Aircraft Equipment Status Failure Behavior
Simulate Aircraft Equipment Status Behavior
Simulate Invalidation
Simulate Aircraft Equipment Sensor Behavior
Simulate Aircraft Equipment Sensor Failure Behavior
Controller Power Supply
Supply Power to Controllers
Actual Equipment Power Supply
Supply Electrical Power to Actual Equipment
Supply Hydraulic Power to Actual Equipment
Tray
Simulate SUT Installation
Tray
Simulate SUT Installation
Break Out Box
Provide Access to Signals
Power Distribution Box
Invalidate Channel
Distribute Power
Protection Against EMI and Lightning
Protect Signal
Data Acquisition System
Data Acquisition System - Control Panel
Display Response Signal
Simulate Internal Signal Failure
Display Simulation Signal
Control Signal Acquisition
Data Acquisition System - Real Time
Record Simulation Signal
Generate Signal
Acquire Simulation Signal
Record Response Signal
Acquire Response Signal
SUT Management System
Maintenance
Operate Maintenanace
Internal Data Modifier
Modify Internal Variable Signal
Software and Memory Manager
Manage Software and Memory
Synoptic
Simulate Synoptic
Plotter
Plot Acquired Signal
Internal Data Detector
Acquire and Record Internal Signal
SUT Managing Signal Server
Generate SUT Management Signal
Acquire SUT Management Signal

Figure 38. [PAB] General model diagram

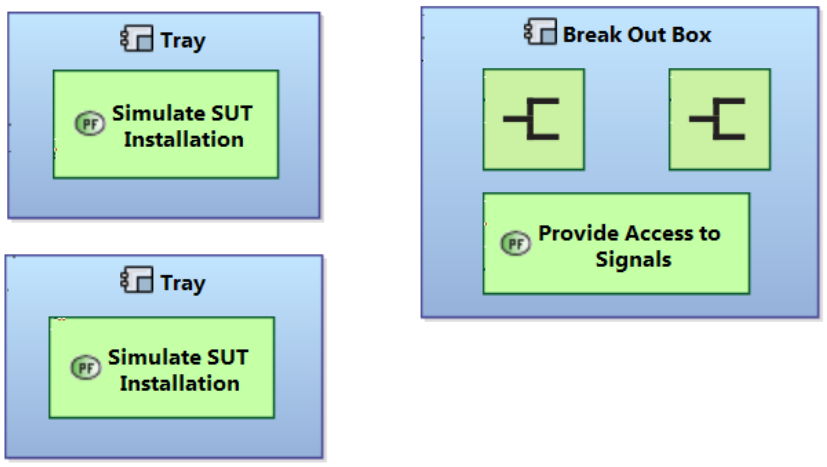
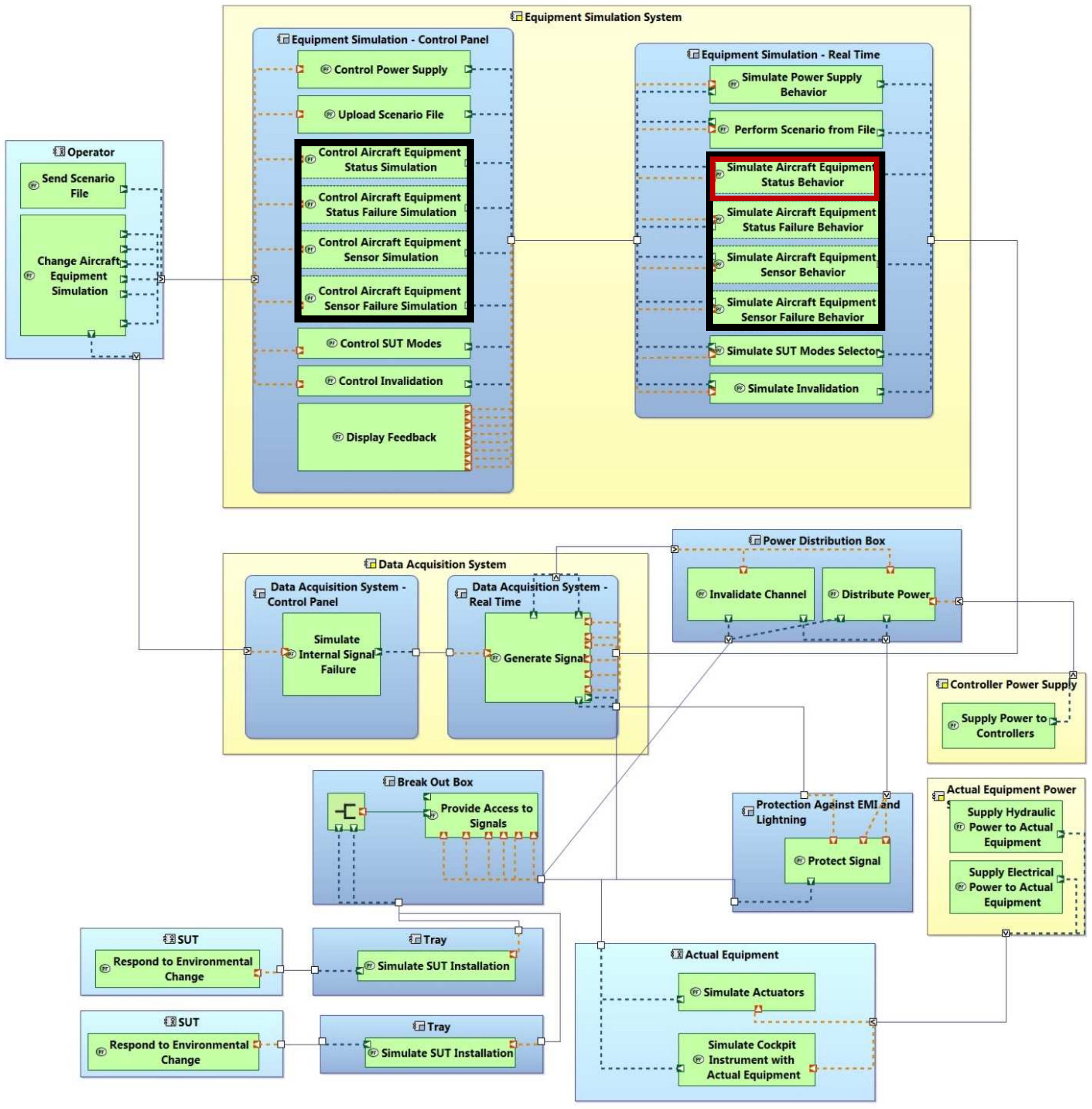

Tray
Simulate SUT Installation
Tray
Simulate SUT Installation
Break Out Box
Provide Access to Signals

Figure 39. [PAB] General model/ breakdown of the SUT installation component

[PAB] Simulating signal

Figure 40 shows the physical architecture for “simulating signal” capability. There are two SUTs and trays shown in the figure representing the multi SUT test means (one of them is variant). The installation of SUT to the test means and actual equipment is through the break-out box.

The simulation is controlled by the operator through two control panels: simulation system and data acquisition system. As shown the data acquisition system’s control panel is for internal signal failure simulation. The data is sent from real time part of the simulation system to signal generation. The generated signals move forward to PDB, actual equipment and the SUT passing the break-out box and trays. If there is a protection, signals move through the protection before entering the break-out box.



Equipment Simulation System
Equipment Simulation - Control Panel
Control Power Supply
Upload Scenario File
Control Aircraft Equipment Status Simulation
Control Aircraft Equipment Status Failure Simulation
Control Aircraft Equipment Sensor Simulation
Control Aircraft Equipment Sensor Failure Simulation
Control SUT Modes
Control Invalidation
Display Feedback
Equipment Simulation - Real Time
Simulate Power Supply Behavior
Perform Scenario from File
Simulate Aircraft Equipment Status Behavior
Simulate Aircraft Equipment Status Failure Behavior
Simulate Aircraft Equipment Sensor Behavior
Simulate Aircraft Equipment Sensor Failure Behavior
Simulate SUT Modes Selector
Simulate Invalidation
Operator
Send Scenario File
Change Aircraft Equipment Simulation
Data Acquisition System
Data Acquisition System - Control Panel
Simulate Internal Signal Failure
Data Acquisition System - Real Time
Generate Signal
Power Distribution Box
Invalidate Channel
Distribute Power
Controller Power Supply
Supply Power to Controllers
Break Out Box
Provide Access to Signals
Protection Against EMI and Lightning
Protect Signal
Actual Equipment Power
Supply Hydraulic Power to Actual Equipment
Supply Electrical Power to Actual Equipment
SUT
Respond to Environmental Change
Tray
Simulate SUT Installation
SUT
Respond to Environmental Change
Tray
Simulate SUT Installation
Actual Equipment
Simulate Actuators
Simulate Cockpit Instrument with Actual Equipment

Figure 40. [PAB] Simulating signal diagram

In summary, section 4.1 presented an overview of the Capella implementation of the model-based specification of test means. The specification model spans the operational level, system level, logical level and physical level. The model starts by specifying the actors and their capability, determining the mission and capability of test means and high level function. Then, the components are introduced and functions are allocated to them. Finally, in physical level, there is a detail representation of the test means architecture and its implementation.

Now that the generic and re-usable model is developed, it is essential to be able to use the model efficiently for specification. Following section discusses the tools and methods deployed to fulfill this purpose.

c. [PFBD] Physical functional breakdown diagram

Each simulation function represented in Figure 40 (the black-framed boxes) is broken into sub-functions in the functional breakdown diagram. For example, the function "simulate aircraft equipment status behavior" which is shown in red-framed box is broken into several functions as depicted in Figure 41.

There are five types of equipment status to be simulate. Moreover, there are various types of signal to simulate. The breakdown diagrams involve each status and type of signal as a separate function. For example, the cockpit instrument status can be simulated with four different type of signals. That is why there are four functions regarding this simulation. However, for other types, there are less form of signals used for simulation.

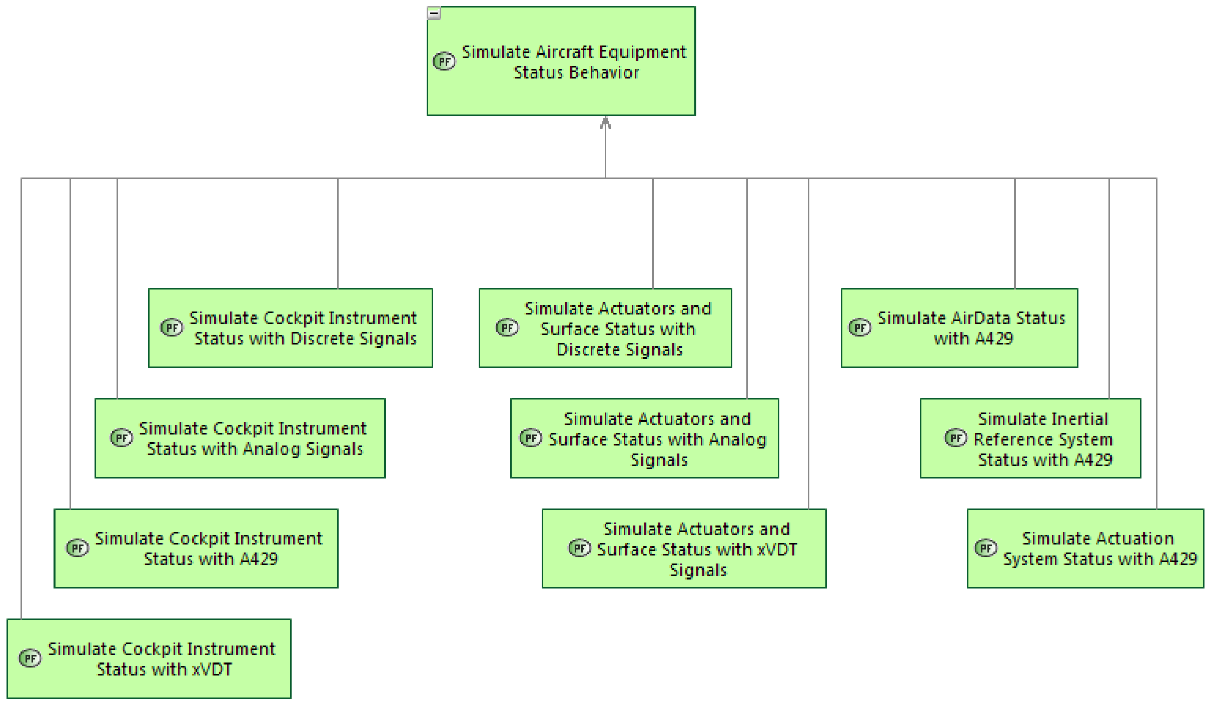

Simulate Aircraft Equipment Status Behavior
Simulate Cockpit Instrument Status with Discrete Signals
Simulate Actuators and Surface Status with Discrete Signals
Simulate AirData Status with A429
Simulate Cockpit Instrument Status with Analog Signals
Simulate Actuators and Surface Status with Analog Signals
Simulate Inertial Reference System Status with A429
Simulate Cockpit Instrument Status with A429
Simulate Actuators and Surface Status with xVDT Signals
Simulate Actuation System Status with A429
Simulate Cockpit Instrument Status with xVDT

Figure 41. Functional breakdown of "simulate aircraft equipment status behavior" function

4.2. Variants management model

As mentioned in section 3.3.2., the developed test means specification model is a generic model that includes all potential features of various test means types. To manage the variants of the test means and to efficiently build a specification of a specific test means, a variability management approach is introduced.

Pure::variants provides the platform for defining different features and linking them to the Capella model. This section discusses the variability management method applied to the thesis using Pure::variants.

Pure::Variants is a software linkable to Capella used to develop the feature model and generate new configuration. In Pure::variants a feature is defined as: "Features are an abstract concept for describing commonalities and variabilities". "A feature can be a requirement, a technical function or function group or a non-functional (quality) characteristic." [41].

Pure ::Variants enables users to define features, sub-features and it types of the features (whether it is mandatory, optional, etc.). It also defines the relation between features and sub-features. For example, one feature could be required by another one, which means that the second feature only exist if the first one does. Moreover, it provides a graphical representation of the generated configuration (distinguishes and shows which components and functions are and are not in the configuration).

There are two models, which have to be created. One is the Feature Model, which is a list of all the features of the generic model. The other one is Variant Description Model (VDM), which is a configuration of the features, which will be in a single product.

1. The “mandatory” feature represents the commonalities. It represents the feature that must exist in the system. However, Optional, Or and Alternative features represent variants.

2. The “optional” features represent the variants in the model. Both mandatory and optional type of features can be sub-features.

3. “Or” and “alternative” features are a group of sub-features which belong to a parent feature. If the types of the sub-features are Alternative, then Only one of them can be selected for one specific configuration. However, for “or” type of sub-features, multiple of them can be chosen. Or and Alternative can only be defined as sub-features. Table 1 shows the sign of these features. Based on these definitions, the features of the test means are defined as shown in Figure 42.

Table 1. Sign of different type of features

!	Mandatory
?	Optional
⇔	Alternative
✕	Or

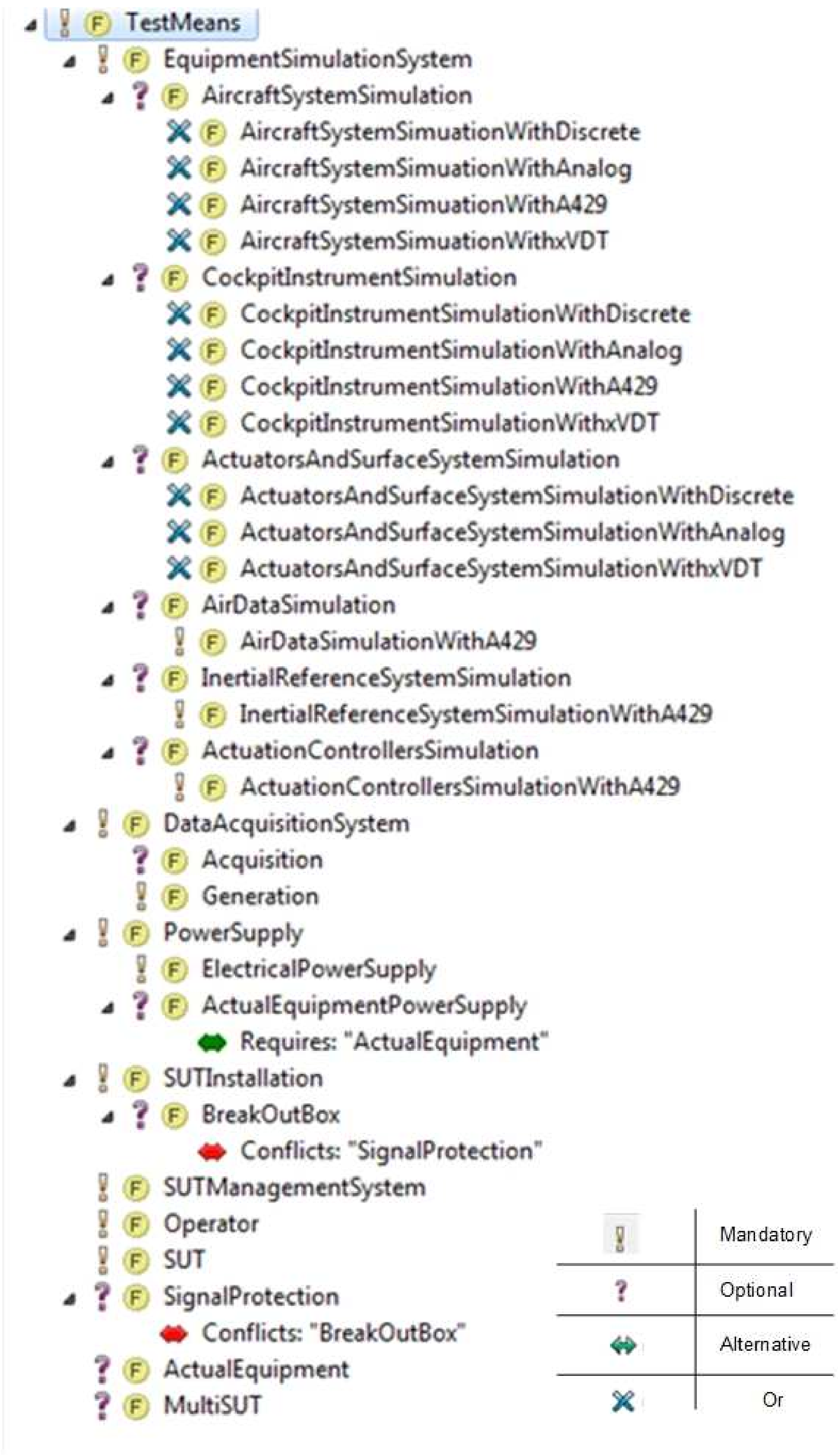

TestMeans
EquipmentSimulationSystem
AircraftSystemSimulation
AircraftSystemSimuationWithDiscrete
AircraftSystemSimuationWithAnalog
AircraftSystemSimuationWithA429
AircraftSystemSimuationWithxVDT
CockpitInstrumentSimulation
CockpitInstrumentSimulationWithDiscrete
CockpitInstrumentSimulationWithAnalog
CockpitInstrumentSimulationWithA429
CockpitInstrumentSimulationWithxVDT
ActuatorsAndSurfaceSystemSimulation
ActuatorsAndSurfaceSystemSimulationWithDiscrete
ActuatorsAndSurfaceSystemSimulationWithAnalog
ActuatorsAndSurfaceSystemSimulationWithxVDT
AirDataSimulation
AirDataSimulationWithA429
InertialReferenceSystemSimulation
InertialReferenceSystemSimulationWithA429
ActuationControllersSimulation
ActuationControllersSimulationWithA429
DataAcquisitionSystem
Acquisition
Generation
PowerSupply
ElectricalPowerSupply
ActualEquipmentPowerSupply
Requires: "ActualEquipment"
SUTInstallation
BreakOutBox
Conflicts: "SignalProtection"
SUTManagementSystem
Operator
SUT
SignalProtection
Conflicts: "BreakOutBox"
ActualEquipment
MultiSUT
Mandatory
Optional
Alternative
Or

Figure 42. Feature model of test means

The optional features are defined based on the variant in Figure 17. According to Figure 17, there are seven optional features in the test means:

Simulation system: the simulation system is a common feature that exists in all type of test means. However, there are various types of equipment to simulate. Moreover, each system can be simulated using different type of signals. In the feature list, the simulation system is defined mandatory, while the equipment to simulate are optional features. For each three type of equipment, there are a group signals defined that are in “Or” relationship, which means that one or more of them can be selected for one specific test means. For the other three type of equipment, there is only one type simulation signal available.

Acquisition and recording: which is a function of data acquisition system along with signal generation.

Actual equipment power supply: In Figure 42., the “requires” relation indicates the fact that actual equipment power supply only exists if there is actual equipment.

Break-out box: which is part of the SUT installation system. It does not exist in environmental qualification test means. That is why it is in conflict with “signal protection” feature.

Signal protection: which is a feature that involves the protection function and component regarding environmental qualification test means.

Actual equipment: this feature is regarding to actual equipment installation to the test means which is specified in Figure 17.

Multi SUT: This feature refers to multiple system under tests.

Once the feature model is created, the components and functions of the Capella model are allocated to the proper features. The selection of features for one specific type of test means is performed through .vdm type of diagram which is discussed in following section.

4.3. Model-based specification process

The objective of the presented thesis is to reduce time of the specification process by creating a generic and re-usable model. In the previous sections, the various aspects of a new, model-based approach have been detailed. This section summarizes and illustrates the steps of generating the specification based on the developed model. Once the feature model is imported to Capella model, selection of the features can be managed using vdm diagram. Figure 43 depicts the model-based specification process which is explained in the coming paragraph.

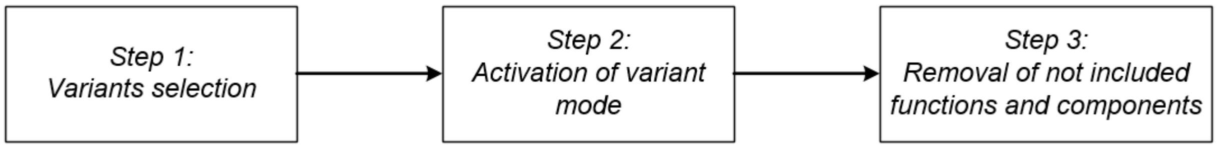
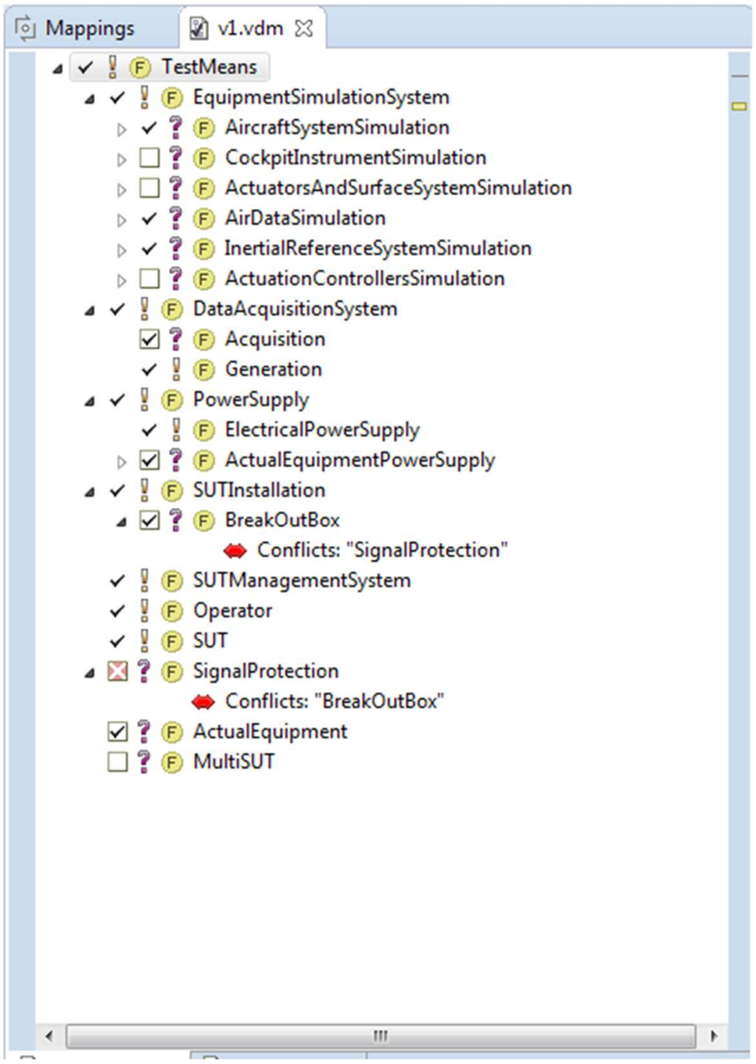

Step 1:
Variants selection
Step 2:
Activation of variant mode
Step 3:
Removal of not included functions and components

Figure 43. Model-based specification process

Step 1: The features of to-be-built test means has to be specified and chosen in vdm file format in instance of Pure::Variants in Capella. Figure 44 shows selection of features in variant description model. For this specific test means, the optional features needed is actual equipment, acquisition and simulation of aircraft, air data and inertial reference system.



```
Mappings    v1.vdm
◢ ✓ ! Ⓕ TestMeans
    ◢ ✓ ! Ⓕ EquipmentSimulationSystem
        ▷ ✓ ? Ⓕ AircraftSystemSimulation
        ▷ ☐ ? Ⓕ CockpitInstrumentSimulation
        ▷ ☐ ? Ⓕ ActuatorsAndSurfaceSystemSimulation
        ▷ ✓ ? Ⓕ AirDataSimulation
        ▷ ✓ ? Ⓕ InertialReferenceSystemSimulation
        ▷ ☐ ? Ⓕ ActuationControllersSimulation
    ◢ ✓ ! Ⓕ DataAcquisitionSystem
          ☑ ? Ⓕ Acquisition
          ✓ ! Ⓕ Generation
    ◢ ✓ ! Ⓕ PowerSupply
          ✓ ! Ⓕ ElectricalPowerSupply
        ▷ ☑ ? Ⓕ ActualEquipmentPowerSupply
    ◢ ✓ ! Ⓕ SUTInstallation
        ◢ ☑ ? Ⓕ BreakOutBox
                  ⬌ Conflicts: "SignalProtection"
      ✓ ! Ⓕ SUTManagementSystem
      ✓ ! Ⓕ Operator
      ✓ ! Ⓕ SUT
    ◢ ☒ ? Ⓕ SignalProtection
              ⬌ Conflicts: "BreakOutBox"
      ☑ ? Ⓕ ActualEquipment
      ☐ ? Ⓕ MultiSUT
```

Figure 44. Variant editor in Capella

Step 2: The variant mode is activated and in each diagram, the functions/components that exist in the specified architecture are shown differently from those which do not exist. The difference is shown using transparency. Those which are not in the architecture are less transparent. Figure 45 (same as Figure 38) shows an example of a diagram while the variant mode activated. As it is depicted, the second SUT and the protection are less transparent. Since they are not in the selected configuration in Figure 44.

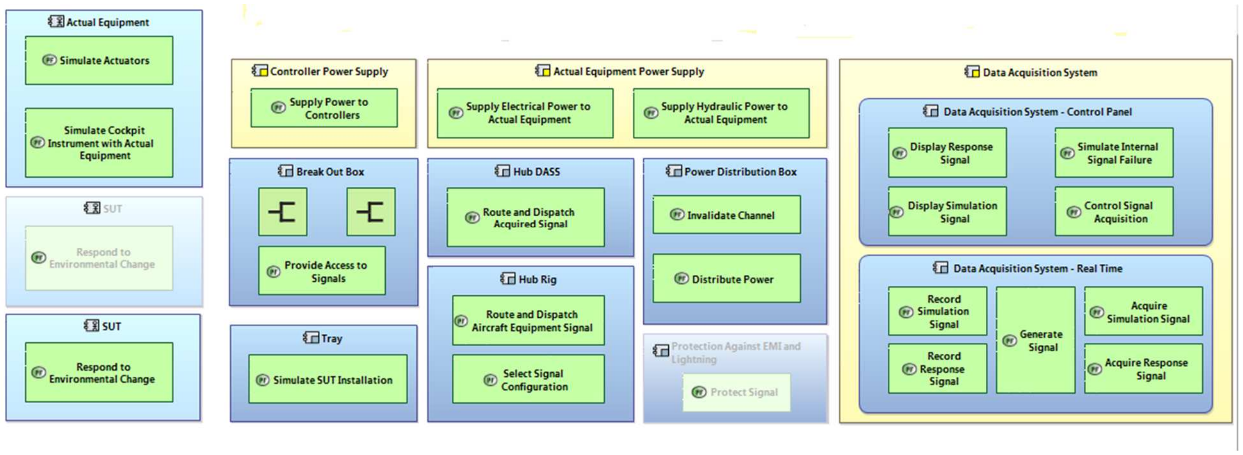

Actual Equipment
Simulate Actuators
Simulate Cockpit Instrument with Actual Equipment
SUT
Respond to Environmental Change
SUT
Respond to Environmental Change
Controller Power Supply
Supply Power to Controllers
Actual Equipment Power Supply
Supply Electrical Power to Actual Equipment
Supply Hydraulic Power to Actual Equipment
Break Out Box
Provide Access to Signals
Hub DASS
Route and Dispatch Acquired Signal
Power Distribution Box
Invalidate Channel
Distribute Power
Tray
Simulate SUT Installation
Hub Rig
Route and Dispatch Aircraft Equipment Signal
Select Signal Configuration
Protection Against EMI and Lightning
Protect Signal
Data Acquisition System
Data Acquisition System - Control Panel
Display Response Signal
Simulate Internal Signal Failure
Display Simulation Signal
Control Signal Acquisition
Data Acquisition System - Real Time
Record Simulation Signal
Generate Signal
Acquire Simulation Signal
Record Response Signal
Acquire Response Signal

Figure 45. A sample of the diagram while variant mode is activated.

Step 3: In this step, functions and components that are not in the architecture can be removed from the model. Then, a model is obtained which is consistent and includes all the features needed for the new project. From the final model, a specification document is generated and handed over to the test means designer.

Figure 46 shows another example of the process. In the left side of the figure, there is the variant description model. Based on the selected features, the actual equipment power supply does not exist in the specific architecture. The right side represents the Capella diagram of physical architecture type. As it is shown, the actual equipment power supply and its functions are less transparent comparing to controller power supply (power supply of the test means which is electrical). Using different transparency, the Pure::Variants provides a representation distinguishing the component existing in a single product from those that do not. The Capella model shown in Figure 46 in the same as Figure 38. A complete step-by-step approach is described in the Appendix B.

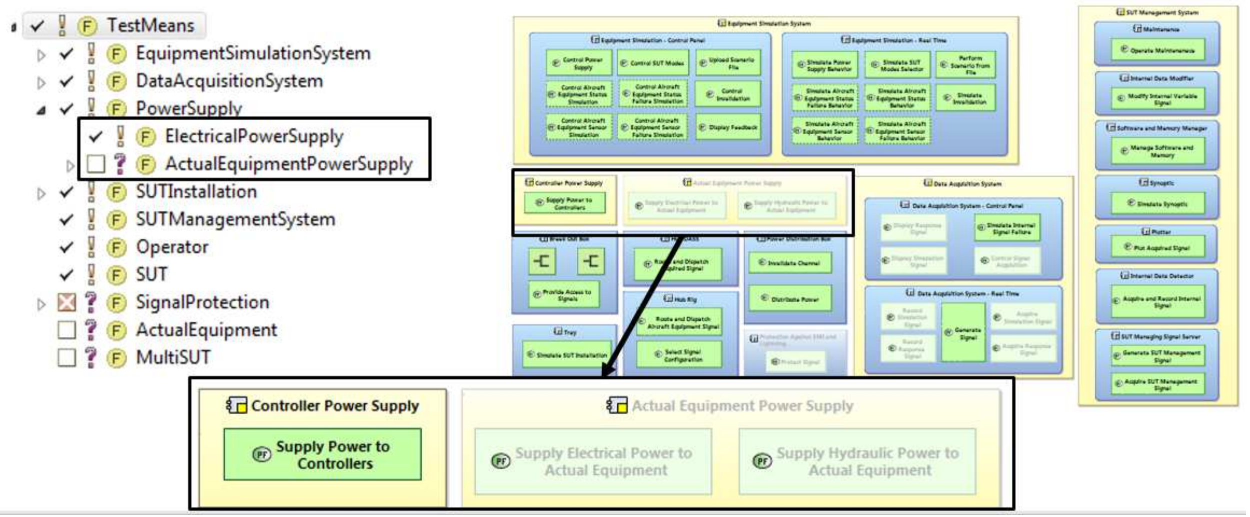

TestMeans
EquipmentSimulationSystem
DataAcquisitionSystem
PowerSupply
ElectricalPowerSupply
ActualEquipmentPowerSupply
SUTInstallation
SUTManagementSystem
Operator
SUT
SignalProtection
ActualEquipment
MultiSUT
Controller Power Supply
Supply Power to Controllers
Actual Equipment Power Supply
Supply Electrical Power to Actual Equipment
Supply Hydraulic Power to Actual Equipment

Figure 46. Example of model-based specification process

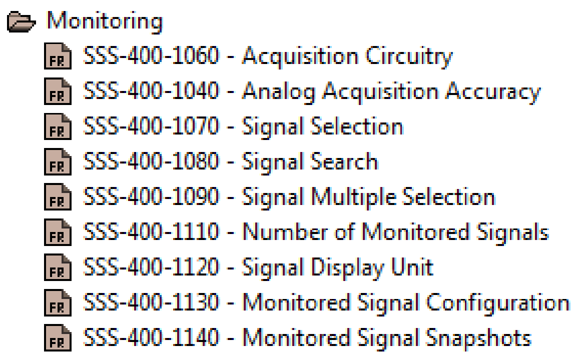
4.4. Additional aspects for model-based specification

As mentioned in section 2.4, the purpose of the thesis is to optimize the process of test means development. In addition to variability management, there are other tools used in the model to facilitate the process of defining a new specification. This section explains the additional aspects.

4.4.1. Linking requirements

As mentioned in section 1.3., the requirements are identified in left hand-side of the V diagram and the specification is written based on the identified requirements. Requirements are in different levels and of various types: safety requirements, customer requirements, performance requirements, etc. [20]. The requirements of test means are also in different levels. Thus, they are allocated to functions or components at different levels in the Capella model. Some requirements are not needed to be defined separately as they are included in the architecture of the test means.

Definition of requirements is one of the Capella's feature. However, it is basic and does not provide a good platform to manage requirements. Requirements are defined and allocated to functions and components. At each level, requirements are defined and allocated to elements of that level. However, it is possible to allocate requirements to components and functions of lower level too. Figure 47. Sample of requirements shows a sample of requirements of the test means in system analysis level.



Monitoring
- SSS-400-1060 - Acquisition Circuitry
- SSS-400-1040 - Analog Acquisition Accuracy
- SSS-400-1070 - Signal Selection
- SSS-400-1080 - Signal Search
- SSS-400-1090 - Signal Multiple Selection
- SSS-400-1110 - Number of Monitored Signals
- SSS-400-1120 - Signal Display Unit
- SSS-400-1130 - Monitored Signal Configuration
- SSS-400-1140 - Monitored Signal Snapshots

Figure 47. Sample of requirements definition in Capella

It is also possible to link requirement management tools like Doors [42] to Capella model. However, for this project, the available feature was enough to illustrate the functionality.

4.4.2. Documentation generation

The developed model involves several diagrams in different level. the model needs to be generated in document format for the users who are not familiar with Capella or do not have access to it. There are different tools available to generate a document format of the model.

Capella has features itself or provides connection to generate documentation from the developed model. The type of document that Capella generates is in HTML format. There are other tools like M2Doc by Obeo which is an open source tool that generates file in MS Word format from the Capella model [43]. It takes a template of Word file and inserts the screenshots of the diagrams, the functions/components and their requirements. The HTML format is easier to work with as it enables the user to navigate through diagrams faster. However, the MS Word is more advisable for documentation since it is the common format for all documents and enables the user to add or omit information. Appendix C explains how to generate a document using M2Doc.

This chapter overviewed the diagrams of Capella in different levels. Moreover, additional aspects of the model and variability management approach has been discussed. Next chapter discusses the result and benefits of the created model.

Chapter 5. Conclusion and future work

5.1 Summary and conclusion

This thesis aims to improve the development process of test means of flight control computers by implementing a model-based system engineering approach. This is achieved by first analyzing the current process of test means development and its challenges, as part of a case study at Thales Canada. Based on the studies, the current process can become more efficient by creating a generic and re-usable model that provides an overview of all the potential features and different implementations of test means. Then, the thesis proceeds to identifying the types of the test means and their common and variant components. Test means are classified into three categories and three sub-categories. Each category and sub-category has its own features (function/component). Finally, the test means are modelled using Capella based on the ARCADIA methodology.

The developed generic model of the test means enables a efficient design process by providing an overview of the architecture and elements of the test means. Moreover, with use of variant management software, the model of one specific test means is generated within few minutes. The generated model covers four levels, starting by the mission and external system interacting with the test means, to the detailed functions and component while it is consistent in all levels.

Another result of the project is the complete analysis on test means architecture and categorization of different types of test means based on their missions, functions and components. The categories and sub-categories are defined and used in variant management tool, which eases the process of identifying the required features for test means.

There are two benefits of the developed generic model in test means development: more efficient and effective specification process by reducing the time and ambiguity. As it was explained in section 2.4. Objective of the thesis, the developed model aims to reduce the specification process. Having the model alleviates the challenge of identifying the needed features in to-be-built test means. Therefore, the process becomes faster. Furthermore, the categorization of various types provides a clear overview of the different possible implementation for test means, which eliminates any vagueness about the architecture.

However, the model has not yet been applied in design process of a new test means. Hence, the benefits are predicted, not proven. There are potentially more benefits. For example, the generic and consistent model assist members of the test means team with different tasks to have a better knowledge of the system they are developing.

In general, the developed specification model is valid beyond its application in Thales; it is generic enough to be considered valuable for the development of test means for flight control computers.

5.2 Future work

In terms of the Capella model, additional diagrams can be generated to improve the specification of test means and to ease subsequent activities such as validation and verification. For example, scenario diagrams (which indicates the sequence of functions happening), interface diagrams (which provides information on what is being exchanged between functions), data models, etc. can be explored in the future. Moreover, components and functions can be modeled in more detail. The same applies to the Pure Variants model.

For generating the MS Word file from the model, the template was written according to the M2Doc 1.0.0. This template enables automatic generation of reports, but it not very user-friendly. Future versions of M2Doc will provide more features to manage the document more efficiently.

In conclusion, this project is the first step towards model-based development of test means. As such, it provides a starting point for more efficient test means development.

References

[1] J. Schmidt, “A Major Aerospace Concern: Manufacturing Delays And Ways To Solve Them.” [Online]. Available: https://www.manufacturing.net/article/2014/02/major-aerospace-concern-manufacturing-delays-and-ways-solve-them. [Accessed: 03-Mar-2019].

[2] Telegraph;, “Boeing 787 Dreamliner: a timeline of problems,” Jul-. [Online]. Available: https://www.telegraph.co.uk/travel/comment/Boeing-787-Dreamliner-a-timeline-of-problems/. [Accessed: 03-Mar-2019].

[3] J.-C. Mare, *Aerospace Actuators 2: Signal-by-Wire and Power-by-Wire*. John Wiley & Sons, 2017.

[4] R. Fielding, C; Luckner, “Flight Control Systems,” W. Pratt, Ed. 2012.

[5] H. Al-lami, A. Aslam, T. Quigley, J. Lewis, R. Mercer, and P. Shukla, “The Evolution of Flight Control Systems Technology Development, System Architecture and Operation,” 2015.

[6] J. P. Sutherland, “Fly-By-Wire Flight Control Systems,” 1968.

[7] G. Bartley F., “Boeing 777: Fly-by-wire Flight Controls,” in *The Avionics Handbook*, 2001.

[8] United States Federal Aviation Administration, *Pilot’s Handbook of Aeronautical Knowledge*. Washington, D.C: U.S. Dept. of Transportation, Federal Aviation Administration, 2016.

[9] EclipseAerospace, “ECLIPSE 500 Aircraft Overview.”

[10] A. Garg, R. I. Linda, and T. Chowdhury, “Evolution of Aircraft Flight Control System and Fly-By-Light Flight Control System,” *Certif. J.*, vol. 3, no. 12, 2013.

[11] I. Moir, A. Seabridge, and M. Jukes, *Civil Avionics Systems*, 2nd ed. 2013.

[12] E. Uzuncaova and M. Ayala, “Boeing 777 Flight Control System,” pp. 1–16, 2003.

[13] “Flight Control Computers | Flight Control Systems.” [Online]. Available: https://www.curtisswrightds.com/applications/aerospace/mission-management/flight-control.html. [Accessed: 06-Mar-2019].

[14] “System Specification Airbus A330/A340 Flight Control System Contents,” 2001.

[15] Airbus, “A330 Flight Crew Operating Manual - Flight Controls.”

[16] I. Moir and A. Seabridge, *Aircraft Systems*, 3rd ed. John Wiley & Sons, 2008.

[17] D. Briere and P. Traverse, “Airbus A320/A330/A340 Electrical Flight Controls,” pp. 616–623, 1993.

[18] J. D. Aplin, “Primary flight computers for the Boeing 777,” *Microprocess. Microsyst.*, pp. 473–478, 1997.

[19] D. McLean, *Automatic Flight Control Systems*, 1st ed. United Kingdom: Prentice Hall International, 1990.

[20] Society of Automotive Engineers, “Guidelines For Development Of Civil Aircraft and Systems,” 2010.

[21] A. Moir, Ian; Seabridge, *Design and Development of Transport Aircraft Systems*, Second Edi. 2013.

[22] “Verification vs Validation - Software Testing Fundamentals.” [Online]. Available: http://softwaretestingfundamentals.com/verification-vs-validation/. [Accessed: 11-Mar-2019].

[23] A. Kossiakoff, W. Sweet, and S. Seymour, *Systems Engineering: Principles and Practice*, 2nd ed. 2003.

[24] RTCA, “Software Considerations in Airborne Systems and Equipment Certification,” 2012.

[25] NASA, “NASA Systems Engineering Handbook,” 2007.

[26] Department of Defence, “Military Standard Engineering Management,” 1974.

[27] INCOSE, “Systems engineering vision 2020,” 2007.

[28] C. Piaszczyk, “Model Based Systems Engineering with Department of Defense Architectural Framework,” 2011.

[29] G. Belcher, D. McIver, and K. Szalai, “Validation of Critical Flight Control Systems,” 1991.

[30] F. Webster and T. D. Smith, “Flying Qualities Flight Testing of Digital Flight Control Systems,” 2001.

[31] “What is UML | Unified Modeling Language.” [Online]. Available: https://www.uml.org/what-is-uml.htm. [Accessed: 14-Apr-2019].

[32] “SysML Open Source Project - What is SysML? Who created SysML?” [Online]. Available: https://sysml.org/. [Accessed: 14-Apr-2019].

[33] J.-L. Viorin, *Model-based System and Architecture Engineering with the Arcadia Method*. ISTE Press Ltd, Elsevier Ltd, 2018.

[34] RTCA, “Environmental Conditions and Test Procedures for Airborne Equipment,” 2010.

[35] P. Roques, *Systems Architecture Modeling with the Arcadia Method*, 1st ed. United Kingdom: ISTE Press and Elsevier, 2018.

[36] THALES GLOBAL SERVICES., “Capella Guide,” 2017.

[37] R. Capilla, J. Bosch, and K. C. Kang, Eds., *Systems and software variability management: Concepts, tools and experiences*. 2013.

[38] K. Schmid and I. John, “A customizable approach to full lifecycle variability management,” pp. 259–284, 2004.

[39] “pure-systems - The leading provider of software for product line and variant management tools | pure::variants.” [Online]. Available: https://www.pure-systems.com/products/pure-variants-9.html. [Accessed: 15-Mar-2019].

[40] pure-systems GmbH, “pure :: variants Connector for Capella Manual,” 2017.

[41] pure-systems GmbH, “pure :: variants User ’ s Guide,” 2017.

[42] “Overview of Rational DOORS.” [Online]. Available:

https://www.ibm.com/support/knowledgecenter/en/SSYQBZ_9.6.1/com.ibm.doors.requirements.doc/topics/c_welcome.html. [Accessed: 10-Apr-2019].

[43] "Reference Documentation - M2Doc." [Online]. Available: http://www.m2doc.org/ref-doc/nightly/index#overview. [Accessed: 05-Apr-2019].

[44] "Cessna Citation Sovereign Flight Controls," 2017.

[45] "Flight Control System | pritamashutosh." [Online]. Available: https://pritamashutosh.wordpress.com/2012/11/17/flight-control-system/. [Accessed: 04-Mar-2019].

[46] Aviation Dictionary, "fly-by-light." [Online]. Available: http://aviation_dictionary.enacademic.com/2976/fly-by-light. [Accessed: 05-Mar-2019].

Appendix A: Capella diagrams

This section provides an overview of all the diagrams in Capella model developed for this thesis. There are different types of diagrams for various purposes available in Capella. For example, scenario diagram which provides an overview of the sequence of functions or breakdown diagrams which specifies the sub-functions of functions. Based on the system of interest, some or all of them are used.

A.1. Operational analysis

The first level of ARCADIA methodology where the users of the system's capabilities and activities are identified. There are two diagrams for this level.

A.1.1. Operational capabilities [OCB]

Operational capabilities diagram shows the entities and actors of the system. Moreover, it identifies the capabilities of them. "operational capability" is the term used in Capella to refer to capabilities of the system stakeholders (OC). Figure A. 1 shows the operational capability diagram.

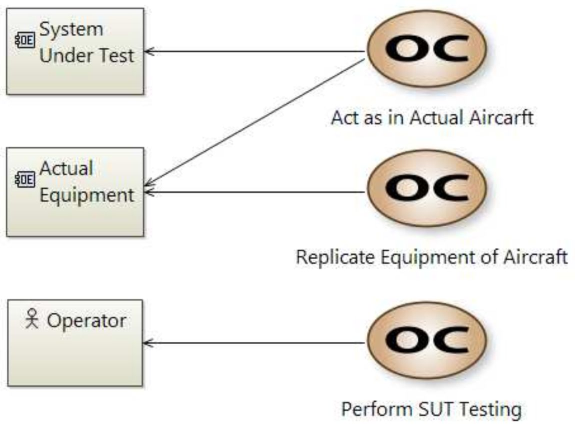


Figure A. 1. Operational capability diagram [OCB]

A.1.2. Operational architecture [OAB]

This diagram depicts the operational activities which are the activities performed by actors and entities. Figure A. 2 shows the operational architecture diagram.

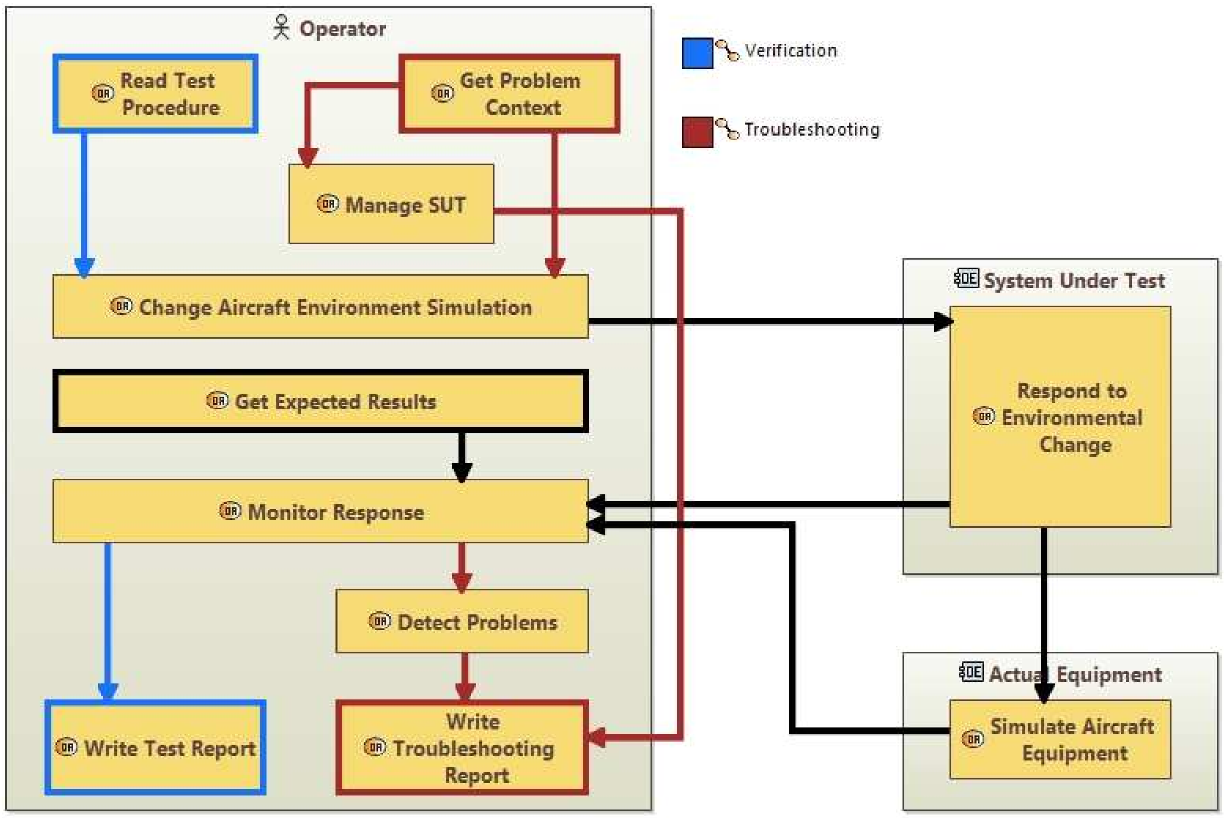


Figure A. 2. Operational architecture diagram [OAB]

A.2. System analysis

This level introduces the system to the model and defines its high level functions.

A.2.1. Missions and capabilities [MCB]

This diagram specifies the missions and capabilities of the test means. Figure A. 3 shows this diagram. Entities and actors of the operational level (previous level) are all considered actors in system level.

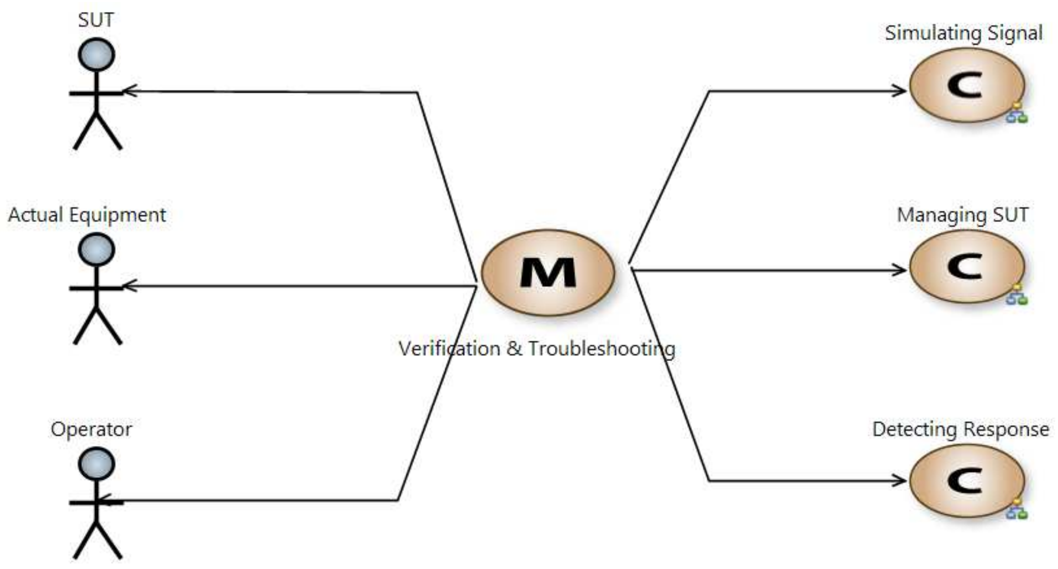


Figure A. 3. Mission and capabilities diagram [MCB]

A.2.2. System architecture [SAB]

This kind of diagram illustrates the relation of the test means and its actors and allocate the functions of actors and the test means to them.

There are four diagrams of this type in the developed model:

A.2.2.1. General model

The general model involve the high level functions of the test means. Figure A. 4 shows the general model of system architecture type of diagram.

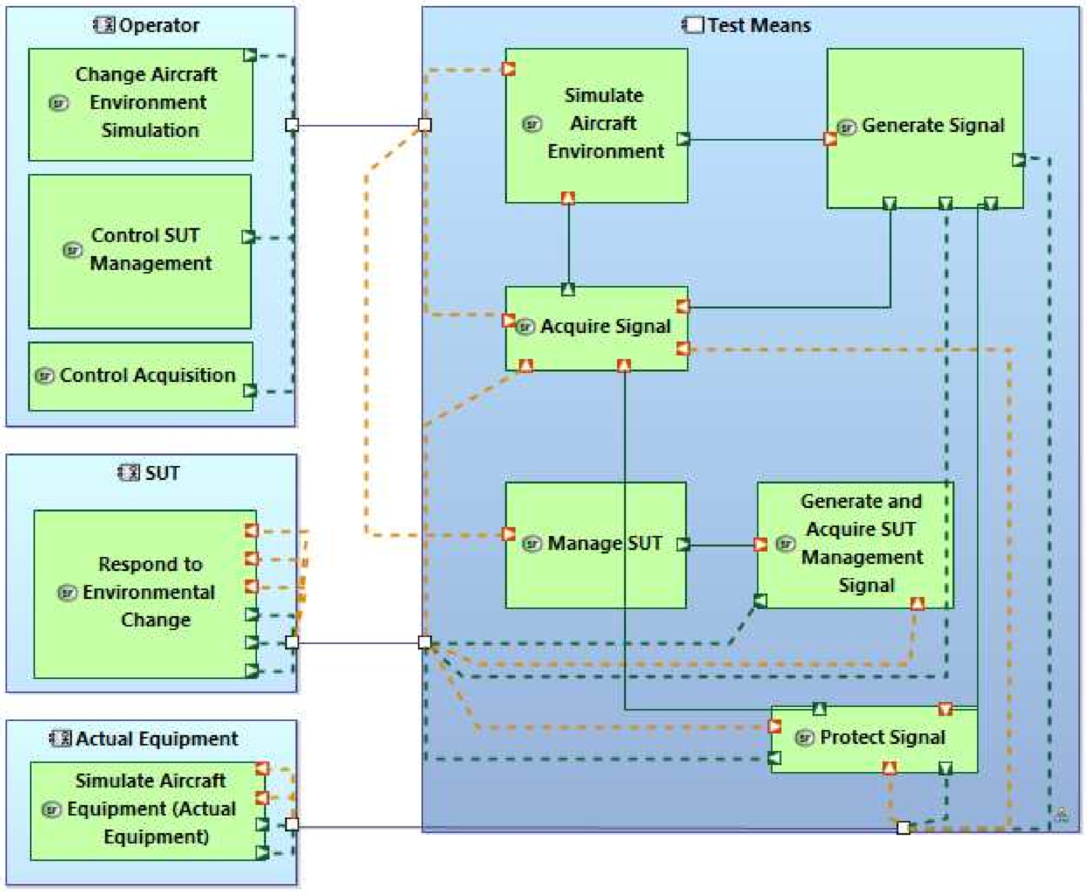

Operator
Change Aircraft Environment Simulation
Control SUT Management
Control Acquisition
SUT
Respond to Environmental Change
Actual Equipment
Simulate Aircraft Equipment (Actual Equipment)
Test Means
Simulate Aircraft Environment
Generate Signal
Acquire Signal
Manage SUT
Generate and Acquire SUT Management Signal
Protect Signal

Figure A. 4. General model diagram [SAB]

A.2.2.2. Simulating signal [SAB]

This diagram illustrates the process of simulation with four high-level simulation functions of test means. Figure A. 5 the simulating signal diagram of system architecture type.

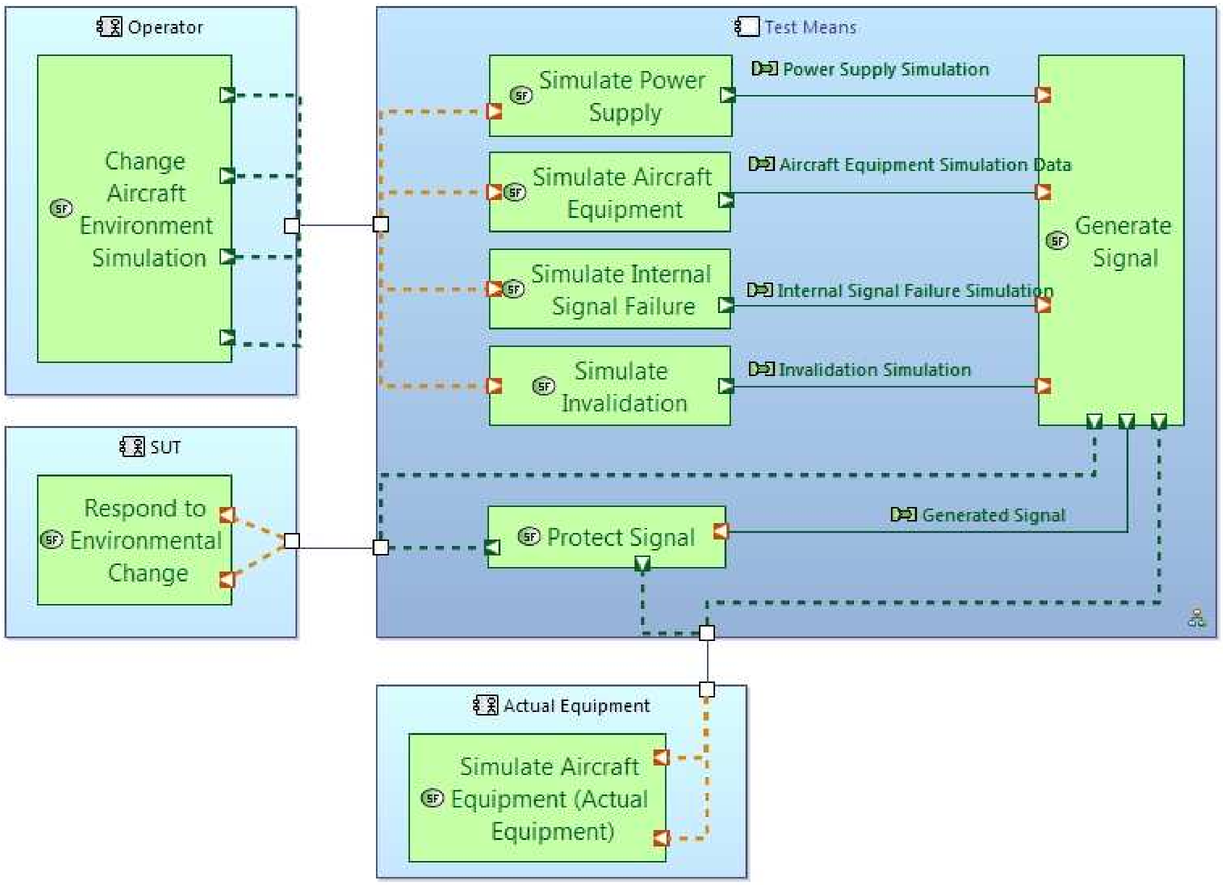
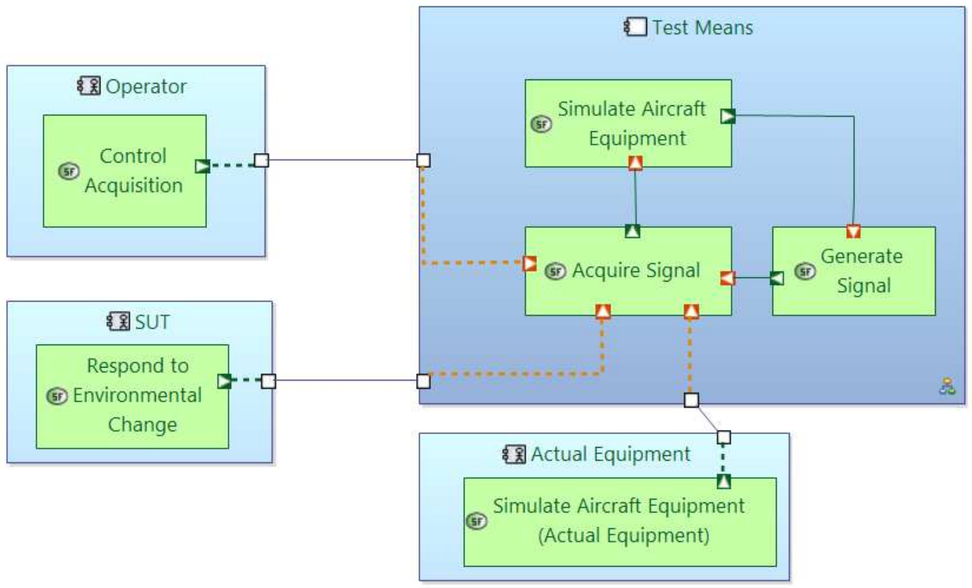

Operator
Change Aircraft Environment Simulation
Test Means
Simulate Power Supply
Power Supply Simulation
Simulate Aircraft Equipment
Aircraft Equipment Simulation Data
Simulate Internal Signal Failure
Internal Signal Failure Simulation
Simulate Invalidation
Invalidation Simulation
Generate Signal
SUT
Respond to Environmental Change
Protect Signal
Generated Signal
Actual Equipment
Simulate Aircraft Equipment (Actual Equipment)

Figure A. 5. Simulating signal diagram [SAB]

A.2.2.3. Detecting signal [SAB]

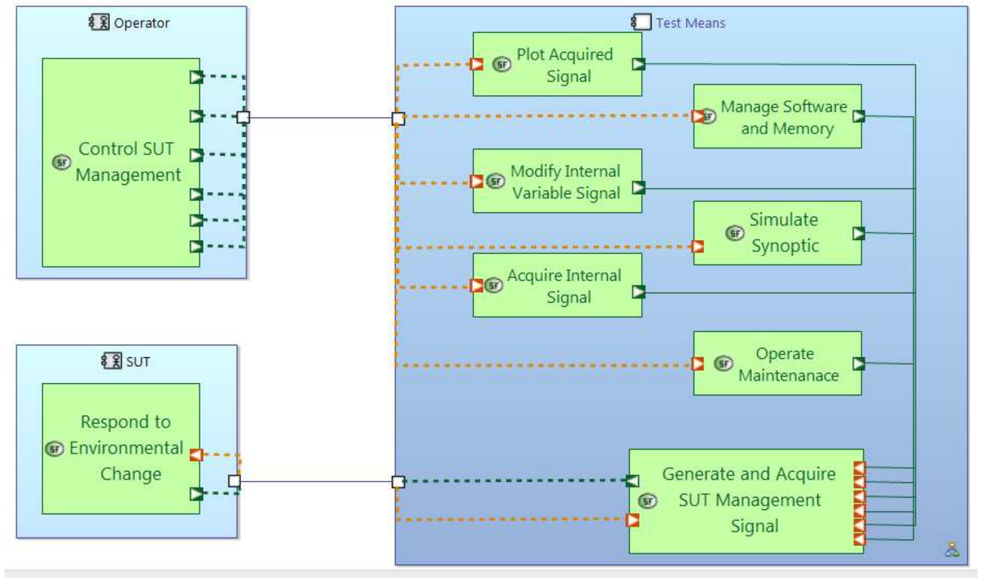
This action is for detecting the response signal of the SUT. These signals would be read by the operator to check the SUT and also would be detected to keep the simulation of the environment which is according to the SUT response. Therefore, there is a functional exchange between the “acquire signal” and “simulate aircraft equipment”. Figure A. 6 shows the detecting signal diagram of system architecture type.



Test Means
Operator
Control Acquisition
Simulate Aircraft Equipment
Acquire Signal
Generate Signal
SUT
Respond to Environmental Change
Actual Equipment
Simulate Aircraft Equipment (Actual Equipment)

Figure A. 6. Detecting response diagram [SAB]

A.2.2.4. Managing SUT [SAB]



Operator
Control SUT Management
Test Means
Plot Acquired Signal
Manage Software and Memory
Modify Internal Variable Signal
Simulate Synoptic
Acquire Internal Signal
Operate Maintenanace
SUT
Respond to Environmental Change
Generate and Acquire SUT Management Signal

Figure A. 7. Managing SUT diagram [SAB]

Figure A. 7 shows the managing SUT diagram of system architecture type.

A.2.3. Functional dataflow [SDFB]

This type of diagram depicts the functions and functional exchanges. Sam as architecture type, there are four types of this diagram as well.

A.2.3.1. General signal [SDFB]

The general model of dataflow type depicts the functional exchanges of the functions defined in Figure A. 4.

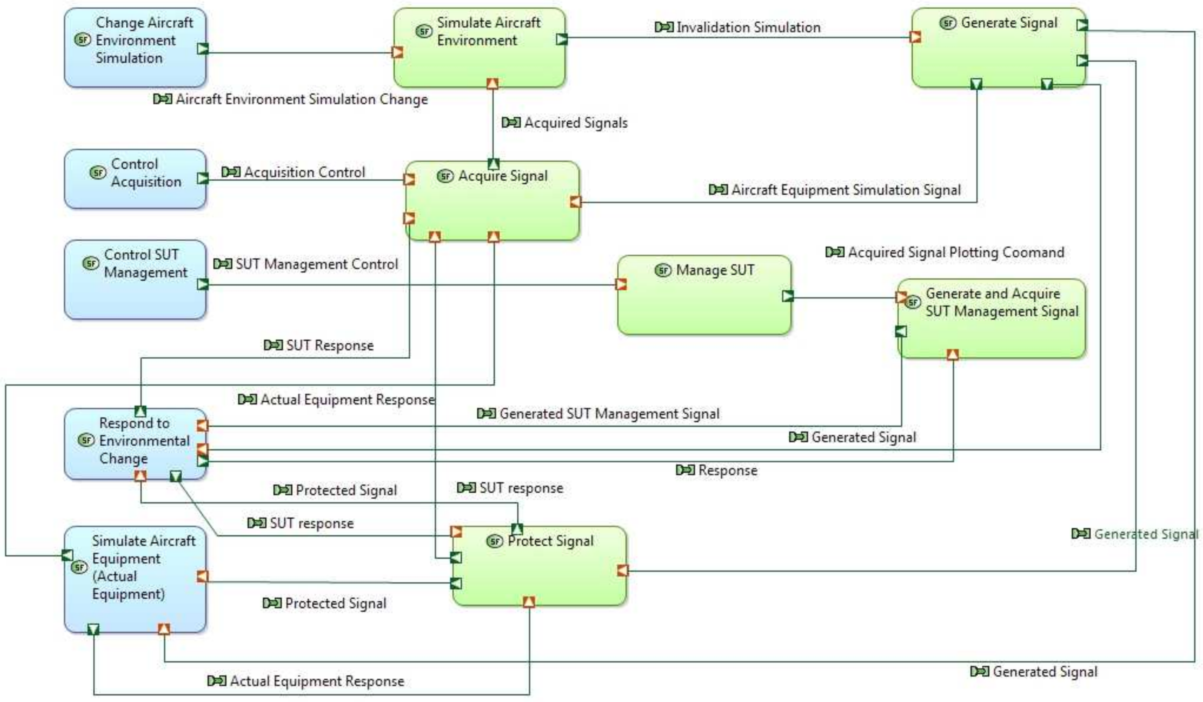

Change Aircraft Environment Simulation
Simulate Aircraft Environment
Invalidation Simulation
Generate Signal
Aircraft Environment Simulation Change
Acquired Signals
Control Acquisition
Acquisition Control
Acquire Signal
Aircraft Equipment Simulation Signal
Control SUT Management
SUT Management Control
Manage SUT
Acquired Signal Plotting Coomand
Generate and Acquire SUT Management Signal
SUT Response
Actual Equipment Response
Generated SUT Management Signal
Respond to Environmental Change
Generated Signal
Response
Protected Signal
SUT response
SUT response
Generated Signal
Simulate Aircraft Equipment (Actual Equipment)
Protect Signal
Protected Signal
Actual Equipment Response
Generated Signal

Figure A. 8. General model diagram [SDFB]

A.2.3.2. Simulating signal [SDFB]

The simulating signal diagram of dataflow type is shown in Figure A. 9.. This diagram show the functional exchanges of Figure A. 5..

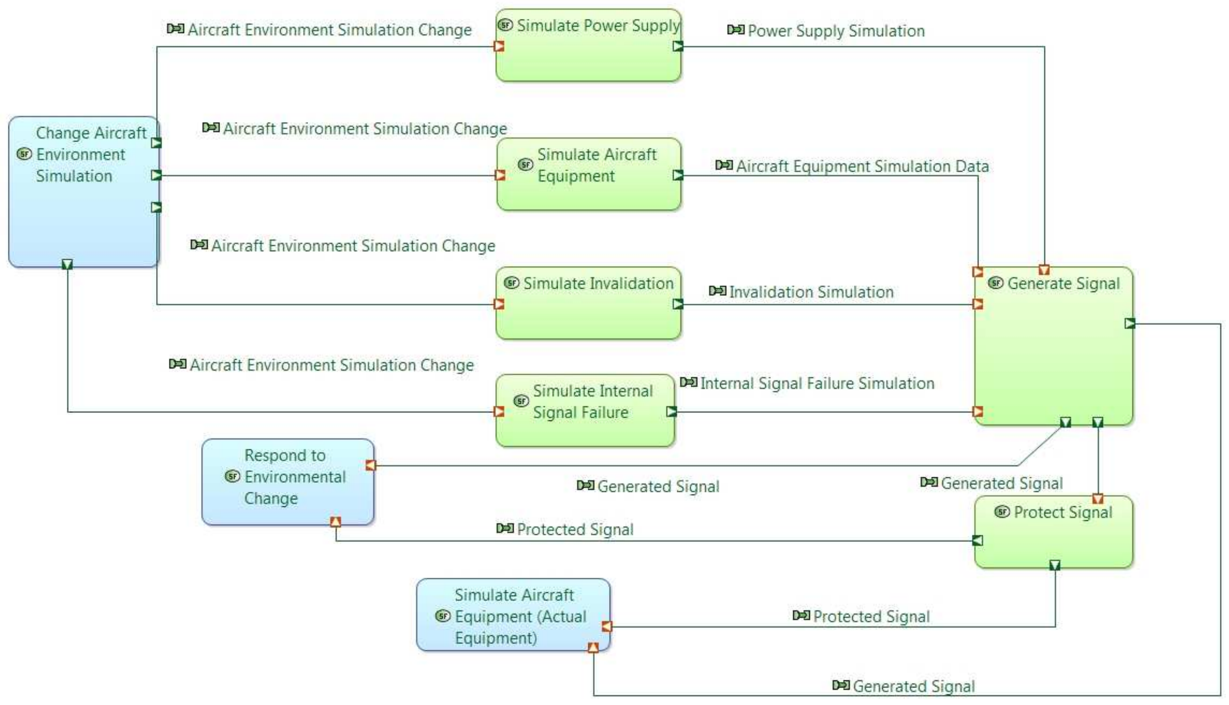
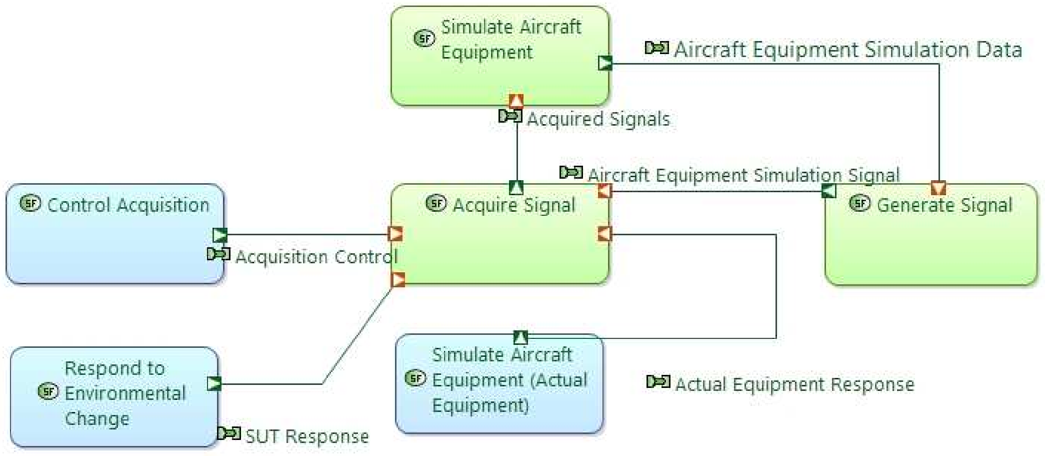

Aircraft Environment Simulation Change
Simulate Power Supply
Power Supply Simulation
Change Aircraft Environment Simulation
Aircraft Environment Simulation Change
Simulate Aircraft Equipment
Aircraft Equipment Simulation Data
Aircraft Environment Simulation Change
Simulate Invalidation
Invalidation Simulation
Generate Signal
Aircraft Environment Simulation Change
Simulate Internal Signal Failure
Internal Signal Failure Simulation
Respond to Environmental Change
Generated Signal
Generated Signal
Protect Signal
Protected Signal
Simulate Aircraft Equipment (Actual Equipment)
Protected Signal
Generated Signal

Figure A. 9. Simulating signal diagram [SDFB]

A.2.3.3. Detecting response [SDFB]

The detecting response diagram of dataflow type is shown in Figure A. 10. This diagram show the functional exchanges of Figure A. 6..



Simulate Aircraft Equipment
Aircraft Equipment Simulation Data
Acquired Signals
Aircraft Equipment Simulation Signal
Control Acquisition
Acquire Signal
Generate Signal
Acquisition Control
Simulate Aircraft Equipment (Actual Equipment)
Respond to Environmental Change
Actual Equipment Response
SUT Response

Figure A. 10. Detecting response diagram [SDFB]

A.2.3.4. Managing SUT [SDFB]

The managing SUT diagram of dataflow type is shown in Figure A. 11. This diagram show the functional exchanges of Figure A. 7..

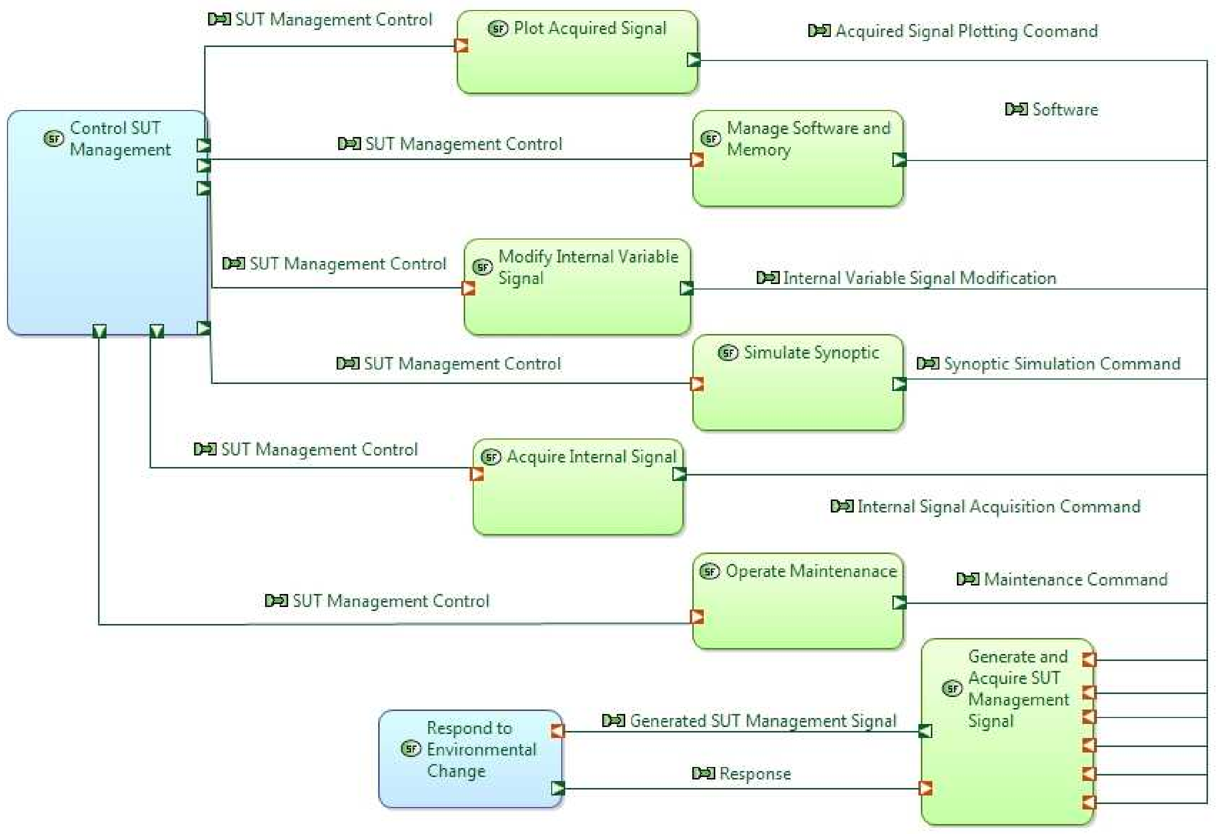

SUT Management Control
Plot Acquired Signal
Acquired Signal Plotting Coomand
Control SUT Management
Manage Software and Memory
Software
SUT Management Control
SUT Management Control
Modify Internal Variable Signal
Internal Variable Signal Modification
Simulate Synoptic
SUT Management Control
Synoptic Simulation Command
SUT Management Control
Acquire Internal Signal
Internal Signal Acquisition Command
Operate Maintenanace
Maintenance Command
SUT Management Control
Generate and Acquire SUT Management Signal
Respond to Environmental Change
Generated SUT Management Signal
Response

Figure A. 11. Managing SUT [SDFB]

A.2.4. Functional breakdown [SFBD]

Functional breakdown diagram shows the sub-functions of the high level functions. High level functions are mostly used in genera model while the sub-functions are used in detail diagrams which are for specific capability.

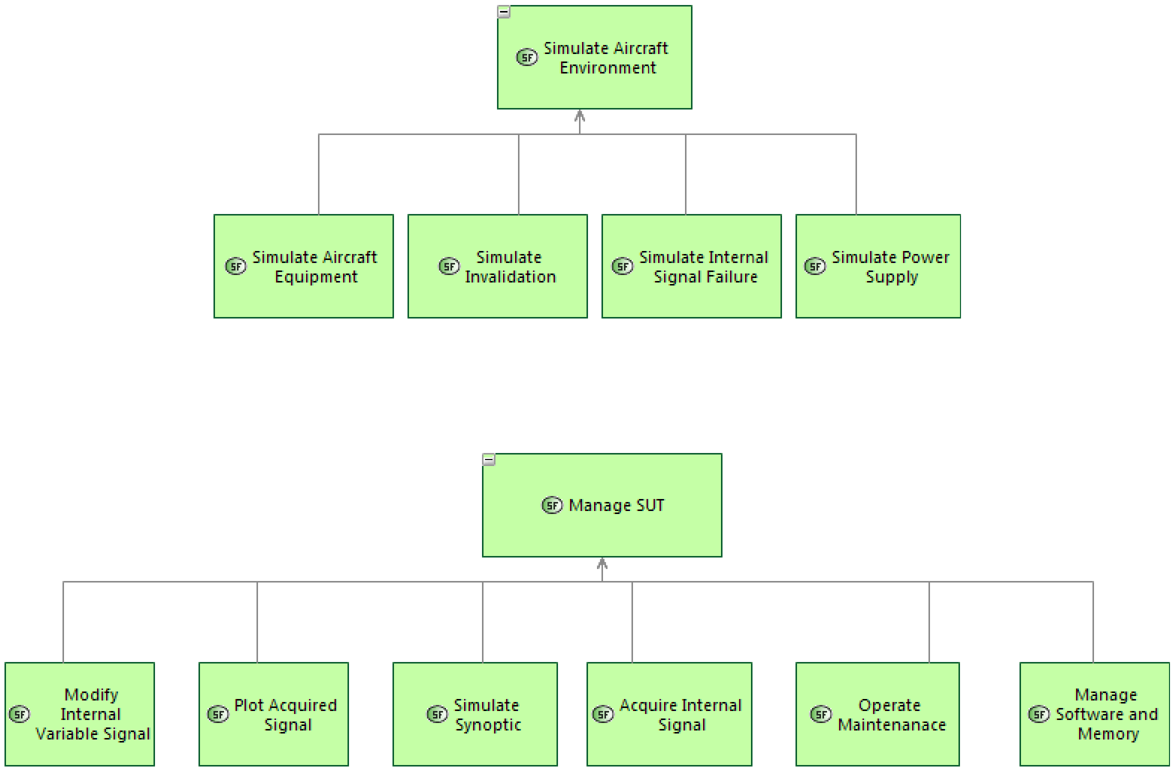

Simulate Aircraft Environment
Simulate Aircraft Equipment
Simulate Invalidation
Simulate Internal Signal Failure
Simulate Power Supply
Manage SUT
Modify Internal Variable Signal
Plot Acquired Signal
Simulate Synoptic
Acquire Internal Signal
Operate Maintenanace
Manage Software and Memory

Figure A. 12. System functional breakdown diagram [SFBD]

A.3. Logical architecture

Logical Architecture is a diagram consisting the components and allocated functions. It is the most completed diagram. Since having all functions and components in one diagram is not feasible, there are four different architecture diagrams. These diagrams are similar to those in upper level, but in more detail. There are three types of diagrams used in this level to model the test means.

A.3.1. Logical architecture [LAB]

Logical architecture type of diagram specifies the components and allocates the functions of each component to it.

A.3.1.1. General model [LAB]

The general model involves all the main functions of the system in logical level. The purpose of this diagram is to have an overview of the logical components and their function at once. Figure A. 13 represents all the logical component of the test means and their functions.

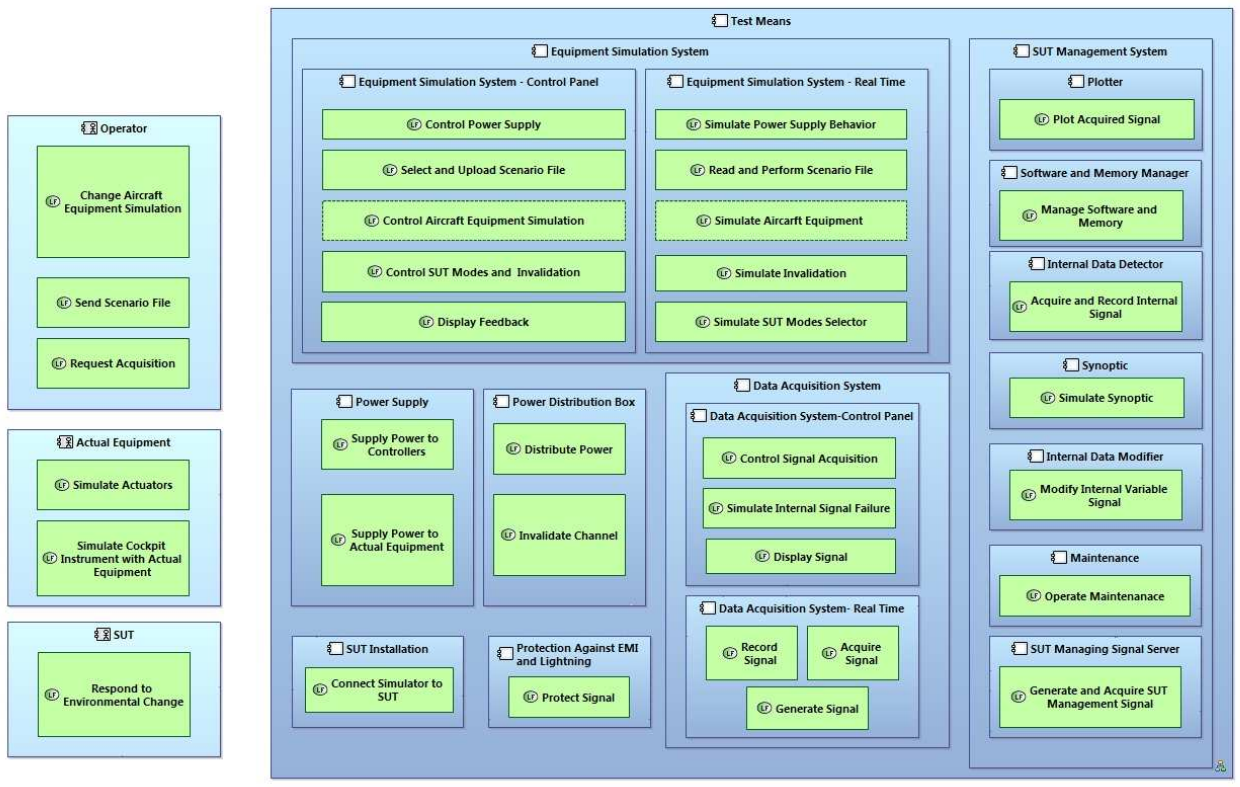
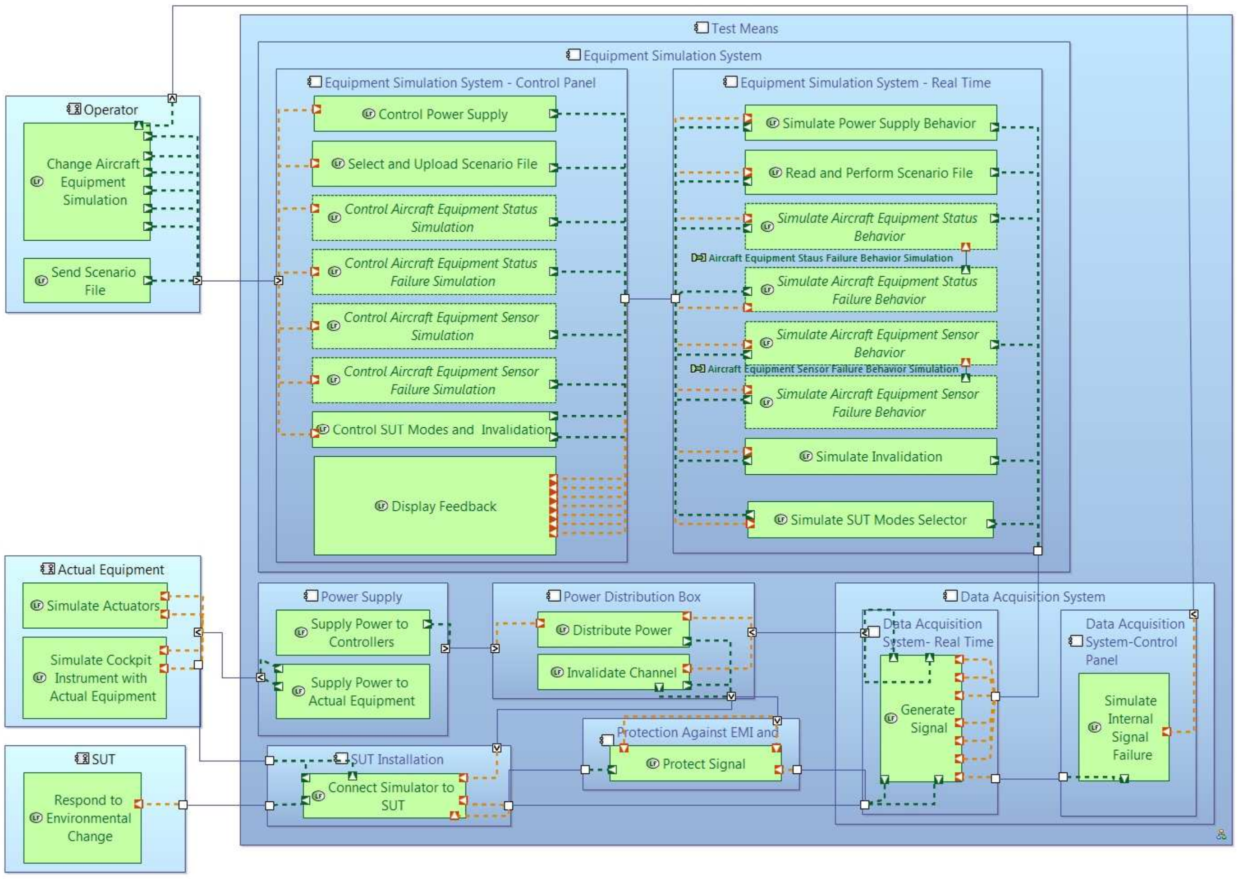

Operator
Change Aircraft Equipment Simulation
Send Scenario File
Request Acquisition
Actual Equipment
Simulate Actuators
Simulate Cockpit Instrument with Actual Equipment
SUT
Respond to Environmental Change
Test Means
Equipment Simulation System
Equipment Simulation System - Control Panel
Control Power Supply
Select and Upload Scenario File
Control Aircraft Equipment Simulation
Control SUT Modes and Invalidation
Display Feedback
Equipment Simulation System - Real Time
Simulate Power Supply Behavior
Read and Perform Scenario File
Simulate Aircarft Equipment
Simulate Invalidation
Simulate SUT Modes Selector
Power Supply
Supply Power to Controllers
Supply Power to Actual Equipment
Power Distribution Box
Distribute Power
Invalidate Channel
SUT Installation
Connect Simulator to SUT
Protection Against EMI and Lightning
Protect Signal
Data Acquisition System
Data Acquisition System-Control Panel
Control Signal Acquisition
Simulate Internal Signal Failure
Display Signal
Data Acquisition System- Real Time
Record Signal
Acquire Signal
Generate Signal
SUT Management System
Plotter
Plot Acquired Signal
Software and Memory Manager
Manage Software and Memory
Internal Data Detector
Acquire and Record Internal Signal
Synoptic
Simulate Synoptic
Internal Data Modifier
Modify Internal Variable Signal
Maintenance
Operate Maintenanace
SUT Managing Signal Server
Generate and Acquire SUT Management Signal

Figure A. 13. General model diagram [LAB]

A.3.1.2. Simulating signal [LAB]

Figure A. 14 represents the architecture of “Simulating Signal” capability in logical level.



Test Means
Equipment Simulation System
Equipment Simulation System - Control Panel
Equipment Simulation System - Real Time
Operator
Change Aircraft Equipment Simulation
Send Scenario File
Control Power Supply
Select and Upload Scenario File
Control Aircraft Equipment Status Simulation
Control Aircraft Equipment Status Failure Simulation
Control Aircraft Equipment Sensor Simulation
Control Aircraft Equipment Sensor Failure Simulation
Control SUT Modes and Invalidation
Display Feedback
Simulate Power Supply Behavior
Read and Perform Scenario File
Simulate Aircraft Equipment Status Behavior
Aircraft Equipment Staus Failure Behavior Simulation
Simulate Aircraft Equipment Status Failure Behavior
Simulate Aircraft Equipment Sensor Behavior
Aircraft Equipment Sensor Failure Behavior Simulation
Simulate Aircraft Equipment Sensor Failure Behavior
Simulate Invalidation
Simulate SUT Modes Selector
Actual Equipment
Simulate Actuators
Simulate Cockpit Instrument with Actual Equipment
SUT
Respond to Environmental Change
Power Supply
Supply Power to Controllers
Supply Power to Actual Equipment
Power Distribution Box
Distribute Power
Invalidate Channel
Data Acquisition System
Data Acquisition System- Real Time
Generate Signal
Data Acquisition System-Control Panel
Simulate Internal Signal Failure
Protection Against EMI and
Protect Signal
SUT Installation
Connect Simulator to SUT

Figure A. 14. Simulating signal diagram [LAB]

A.3.1.3. Detecting signal [LAB]

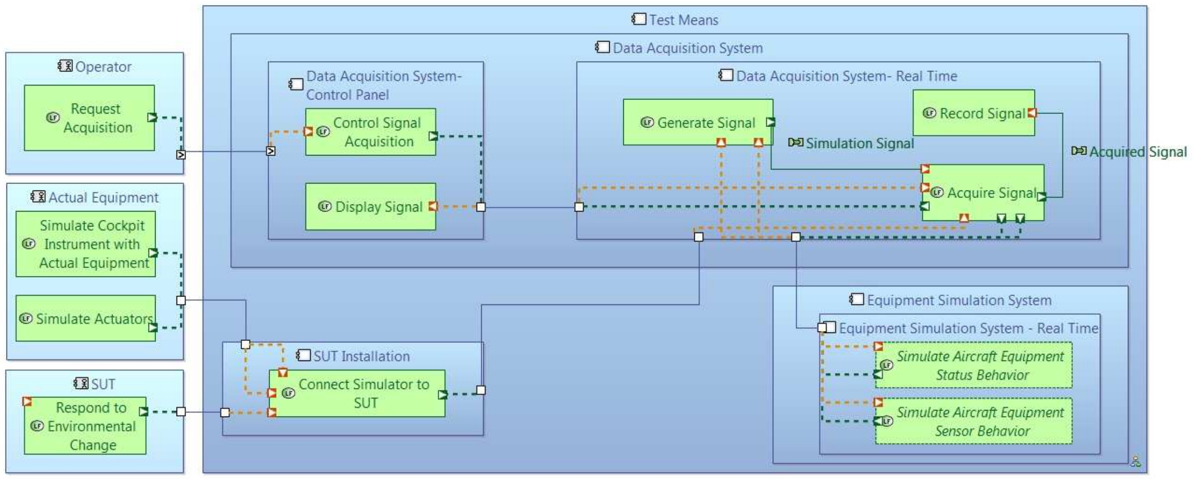

Test Means
Data Acquisition System
Data Acquisition System- Control Panel
Control Signal Acquisition
Display Signal
Data Acquisition System- Real Time
Generate Signal
Simulation Signal
Record Signal
Acquired Signal
Acquire Signal
Operator
Request Acquisition
Actual Equipment
Simulate Cockpit Instrument with Actual Equipment
Simulate Actuators
SUT
Respond to Environmental Change
SUT Installation
Connect Simulator to SUT
Equipment Simulation System
Equipment Simulation System - Real Time
Simulate Aircraft Equipment Status Behavior
Simulate Aircraft Equipment Sensor Behavior

Figure A. 15. Detecting response diagram [LAB]

A.3.1.4. Managing SUT [LAB]

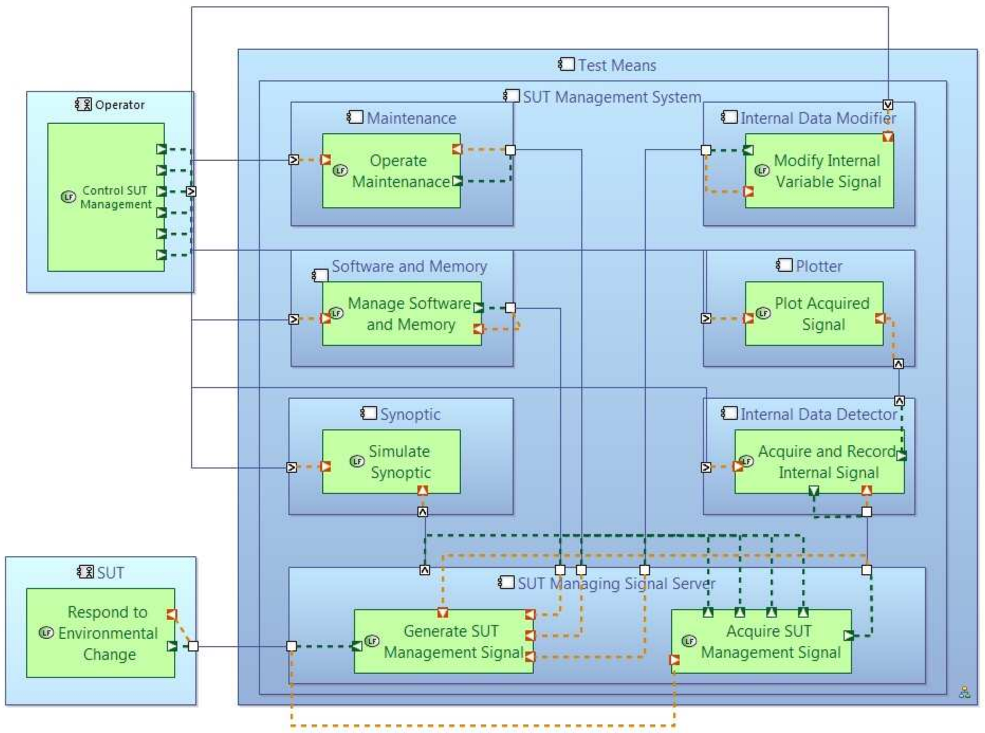


Figure A. 16. Managing SUT diagram [LAB]

A.3.2. Logical dataflow [LDFB]

Dataflow diagram shows the functional exchanges between the functions allocated to components in architecture diagram.

A.3.2.1. Simulating signal [LDFB]

The simulating signal diagram of dataflow type in logical level is shown in Figure A. 17. This diagram show the functional exchanges of Figure A. 5.

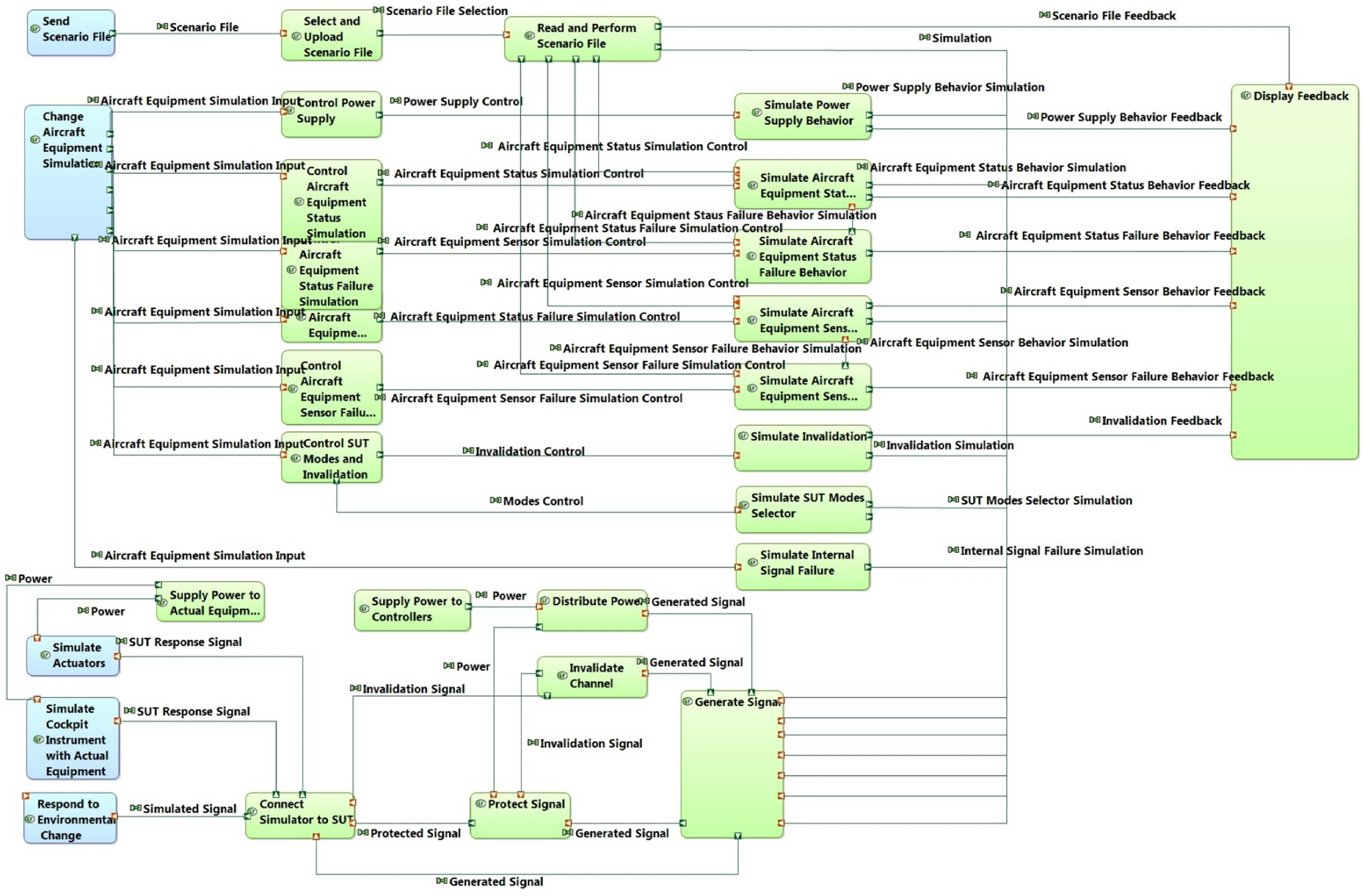

Send Scenario File
Scenario File
Select and Upload Scenario File
Scenario File Selection
Read and Perform Scenario File
Scenario File Feedback
Simulation
Change Aircraft Equipment Simulation
Aircraft Equipment Simulation Input
Control Power Supply
Power Supply Control
Simulate Power Supply Behavior
Power Supply Behavior Simulation
Power Supply Behavior Feedback
Display Feedback
Aircraft Equipment Status Simulation Control
Control Aircraft Equipment Status Simulation
Simulate Aircraft Equipment Stat...
Aircraft Equipment Status Behavior Simulation
Aircraft Equipment Status Behavior Feedback
Aircraft Equipment Staus Failure Behavior Simulation
Aircraft Equipment Status Failure Simulation Control
Aircraft Equipment Sensor Simulation Control
Aircraft Equipment Status Failure Simulation
Simulate Aircraft Equipment Status Failure Behavior
Aircraft Equipment Status Failure Behavior Feedback
Aircraft Equipment Sensor Behavior Feedback
Aircraft Equipme...
Simulate Aircraft Equipment Sens...
Aircraft Equipment Sensor Failure Behavior Simulation
Aircraft Equipment Sensor Behavior Simulation
Aircraft Equipment Sensor Failure Simulation Control
Control Aircraft Equipment Sensor Failu...
Aircraft Equipment Sensor Failure Behavior Feedback
Invalidation Feedback
Simulate Invalidation
Invalidation Simulation
Control SUT Modes and Invalidation
Invalidation Control
Simulate SUT Modes Selector
Modes Control
SUT Modes Selector Simulation
Simulate Internal Signal Failure
Internal Signal Failure Simulation
Power
Supply Power to Actual Equipm...
Supply Power to Controllers
Distribute Power
Generated Signal
Simulate Actuators
SUT Response Signal
Invalidate Channel
Invalidation Signal
Simulate Cockpit Instrument with Actual Equipment
Generate Signal
Respond to Environmental Change
Simulated Signal
Connect Simulator to SUT
Protected Signal
Protect Signal

Figure A. 17. Simulating signal diagram [LDFB]

A.3.2.2. Detecting response [LDFB]

The detecting response diagram of dataflow type in logical level is shown in Figure A. 18.. This diagram show the functional exchanges of Figure A. 5..

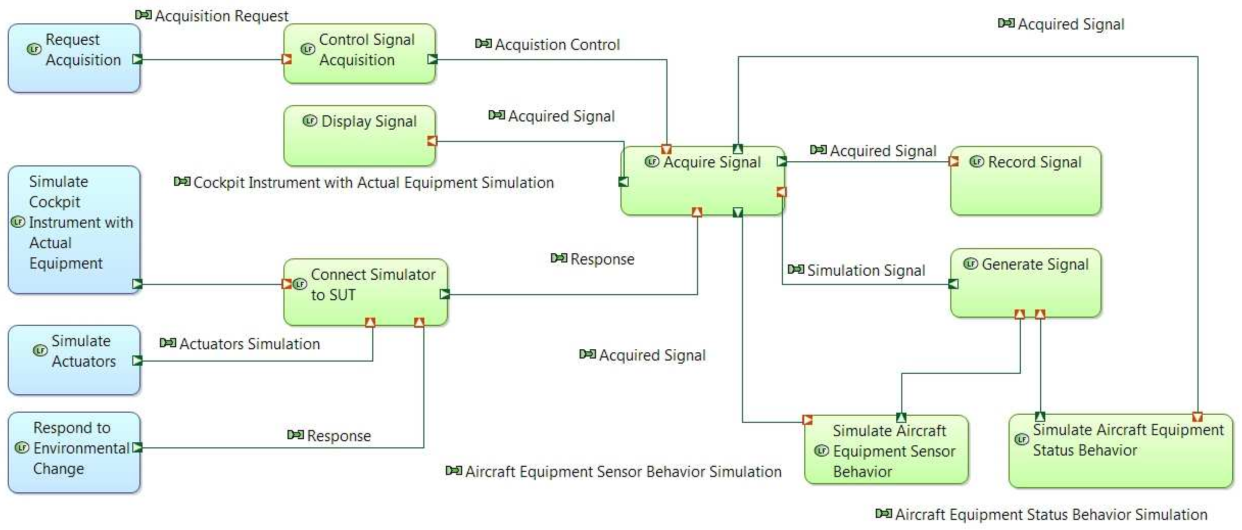

Acquisition Request
Request Acquisition
Control Signal Acquisition
Acquistion Control
Acquired Signal
Display Signal
Acquired Signal
Acquire Signal
Acquired Signal
Record Signal
Simulate Cockpit Instrument with Actual Equipment
Cockpit Instrument with Actual Equipment Simulation
Connect Simulator to SUT
Response
Simulation Signal
Generate Signal
Simulate Actuators
Actuators Simulation
Acquired Signal
Respond to Environmental Change
Response
Aircraft Equipment Sensor Behavior Simulation
Simulate Aircraft Equipment Sensor Behavior
Simulate Aircraft Equipment Status Behavior
Aircraft Equipment Status Behavior Simulation

Figure A. 18. Detecting response diagram [LDFB]

A.3.2.3. Managing SUT [LDFB]

The managing SUT diagram of dataflow type in logical level is shown in Figure A. 19. This diagram show the functional exchanges of Figure A. 16..

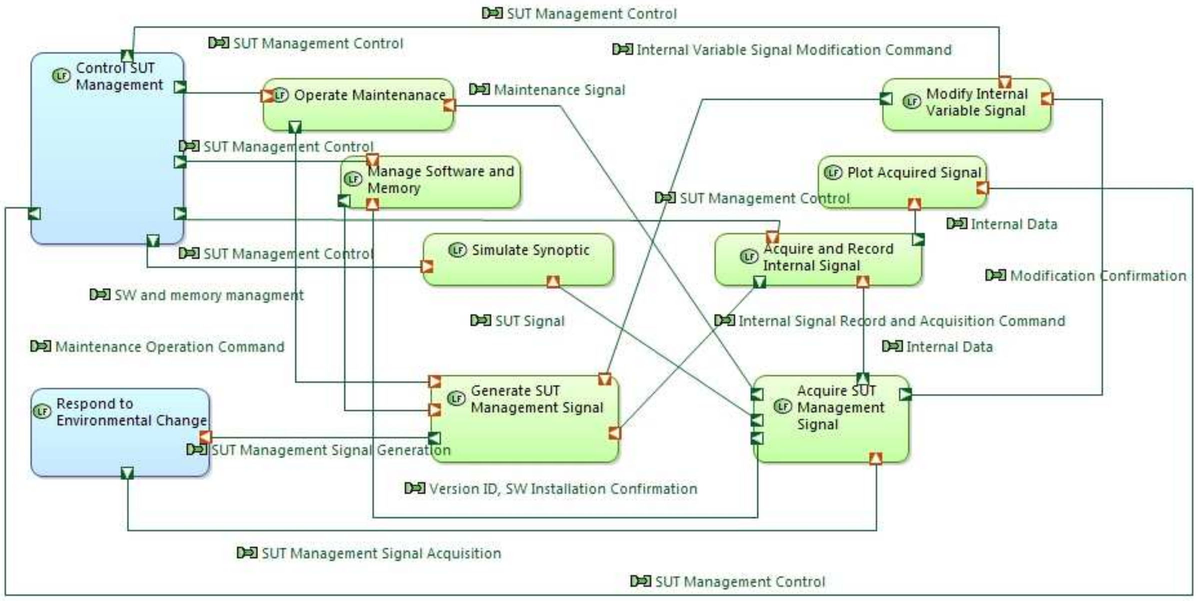

SUT Management Control
SUT Management Control
Internal Variable Signal Modification Command
Control SUT Management
Operate Maintenanace
Maintenance Signal
Modify Internal Variable Signal
SUT Management Control
Manage Software and Memory
Plot Acquired Signal
SUT Management Control
Internal Data
Simulate Synoptic
Acquire and Record Internal Signal
SUT Management Control
Modification Confirmation
SW and memory managment
SUT Signal
Internal Signal Record and Acquisition Command
Maintenance Operation Command
Internal Data
Generate SUT Management Signal
Acquire SUT Management Signal
Respond to Environmental Change
SUT Management Signal Generation
Version ID, SW Installation Confirmation
SUT Management Signal Acquisition
SUT Management Control

Figure A. 19. Managing SUT diagram [LDFB]

A.3.3. Logical breakdown [LFBD]

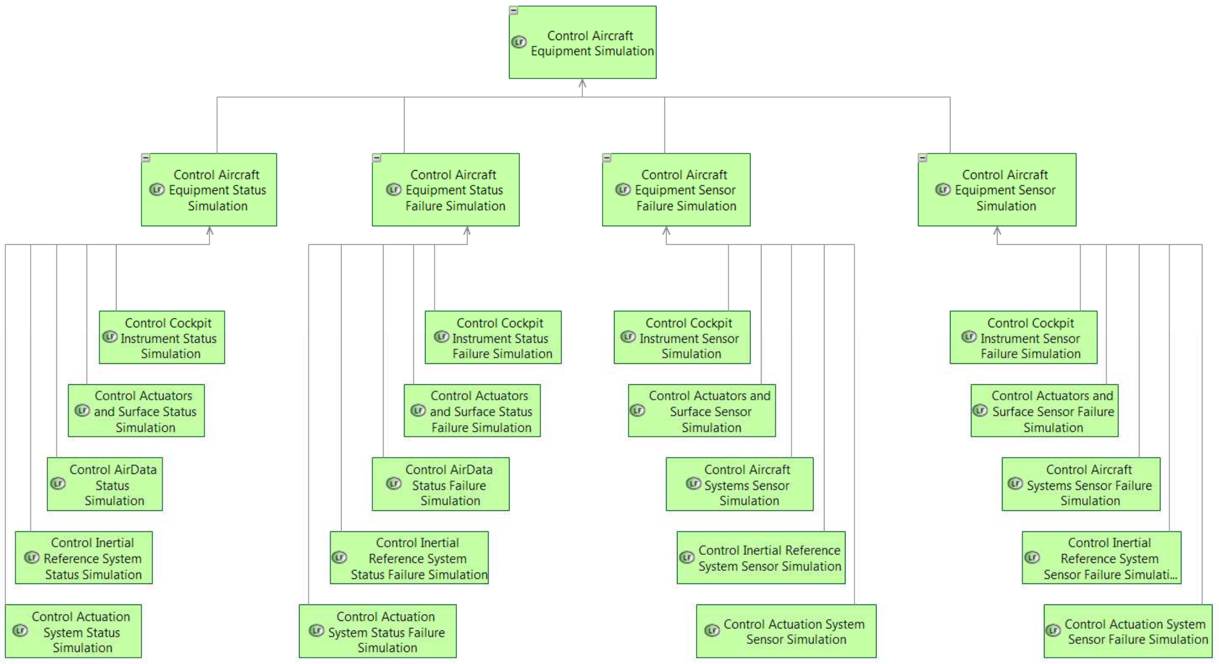

Control Aircraft Equipment Simulation
Control Aircraft Equipment Status Simulation
Control Aircraft Equipment Status Failure Simulation
Control Aircraft Equipment Sensor Failure Simulation
Control Aircraft Equipment Sensor Simulation
Control Cockpit Instrument Status Simulation
Control Actuators and Surface Status Simulation
Control AirData Status Simulation
Control Inertial Reference System Status Simulation
Control Actuation System Status Simulation
Control Cockpit Instrument Status Failure Simulation
Control Actuators and Surface Status Failure Simulation
Control AirData Status Failure Simulation
Control Inertial Reference System Status Failure Simulation
Control Actuation System Status Failure Simulation
Control Cockpit Instrument Sensor Simulation
Control Actuators and Surface Sensor Simulation
Control Aircraft Systems Sensor Simulation
Control Inertial Reference System Sensor Simulation
Control Actuation System Sensor Simulation
Control Cockpit Instrument Sensor Failure Simulation
Control Actuators and Surface Sensor Failure Simulation
Control Aircraft Systems Sensor Failure Simulation
Control Inertial Reference System Sensor Failure Simulati...
Control Actuation System Sensor Failure Simulation

Figure A. 20.Functional breakdown of aircraft equipment simulation control [LFBD]

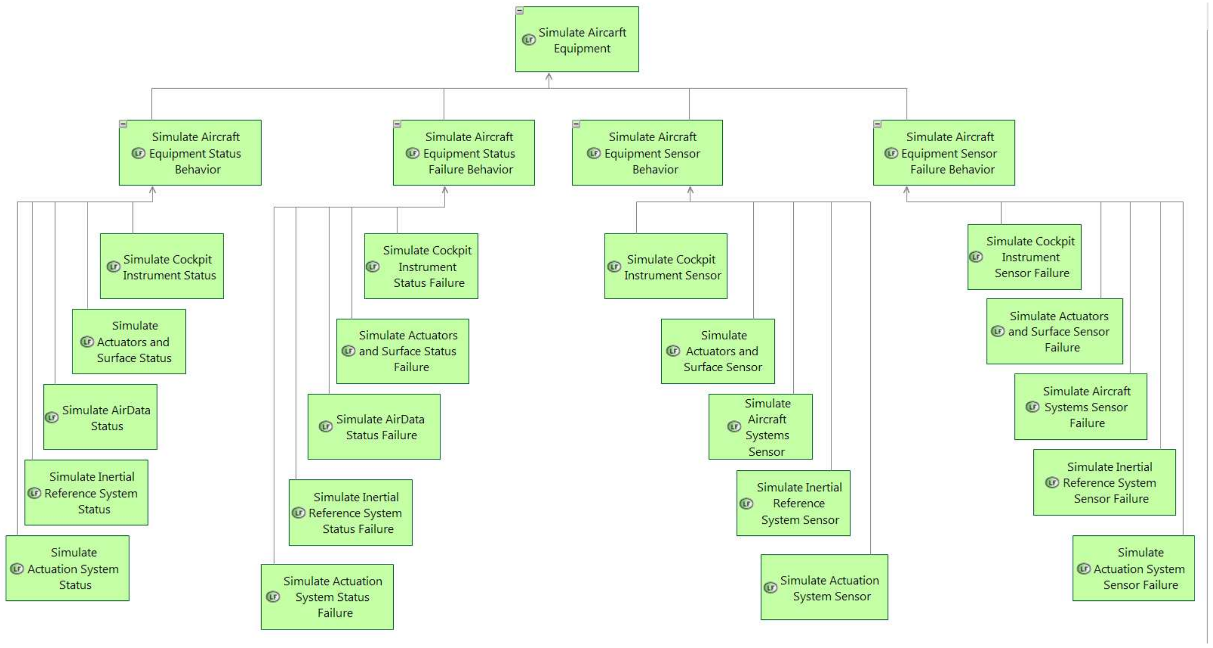

Simulate Aircraft Equipment
Simulate Aircraft Equipment Status Behavior
Simulate Aircraft Equipment Status Failure Behavior
Simulate Aircraft Equipment Sensor Behavior
Simulate Aircraft Equipment Sensor Failure Behavior
Simulate Cockpit Instrument Status
Simulate Actuators and Surface Status
Simulate AirData Status
Simulate Inertial Reference System Status
Simulate Actuation System Status
Simulate Cockpit Instrument Status Failure
Simulate Actuators and Surface Status Failure
Simulate AirData Status Failure
Simulate Inertial Reference System Status Failure
Simulate Actuation System Status Failure
Simulate Cockpit Instrument Sensor
Simulate Actuators and Surface Sensor
Simulate Aircraft Systems Sensor
Simulate Inertial Reference System Sensor
Simulate Actuation System Sensor
Simulate Cockpit Instrument Sensor Failure
Simulate Actuators and Surface Sensor Failure
Simulate Aircraft Systems Sensor Failure
Simulate Inertial Reference System Sensor Failure
Simulate Actuation System Sensor Failure

Figure A. 21. Functional breakdown of aircraft equipment simulation [LFBD]

A.4. Physical architecture

Physical level is the last level of the model. It involves the highest level of detail of the model. Diagrams which are used in this level, are physical architecture [PAB], physical dataflow [PDFB] and functional breakdown diagram [PFBD]. The number of each component, their types (node or behavior) are determined in this level.

A.4.1. Physical architecture [PAB]

Physical architecture type of diagram depicts the components, their types, implementations and quantity. There are four diagram of this type modeled in physical level.

A.4.1.1. General model [PAB]

The general model includes all the actors, components (both behavior an node) and their functions. Figure A. 22 shows the general model of physical level (The same picture as Figure 38).

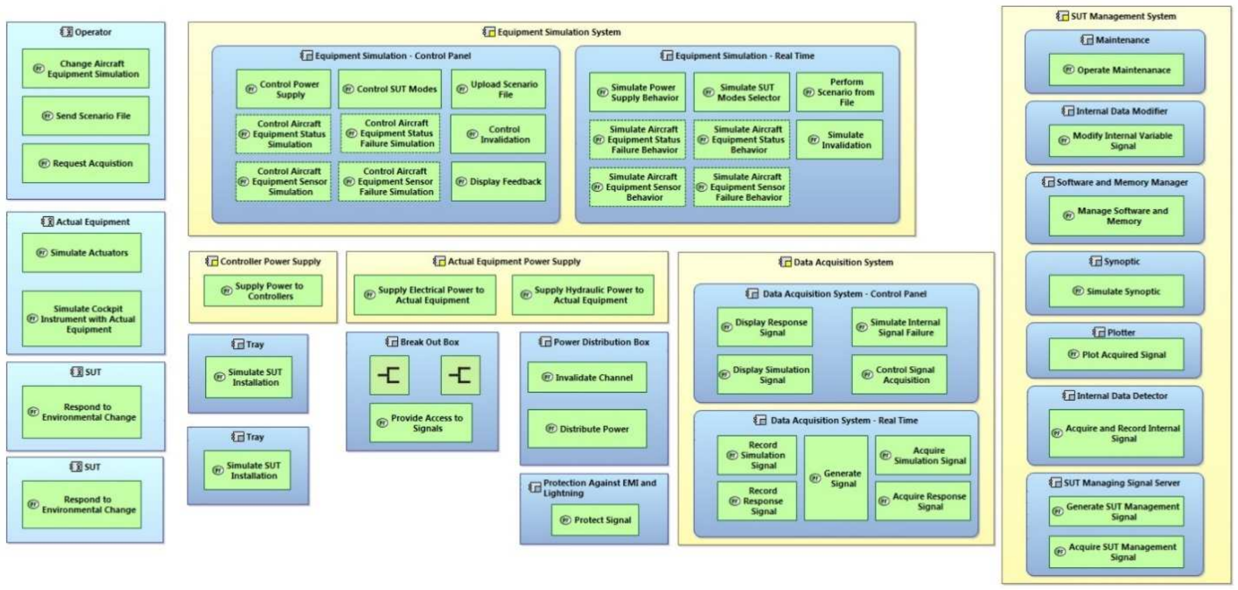
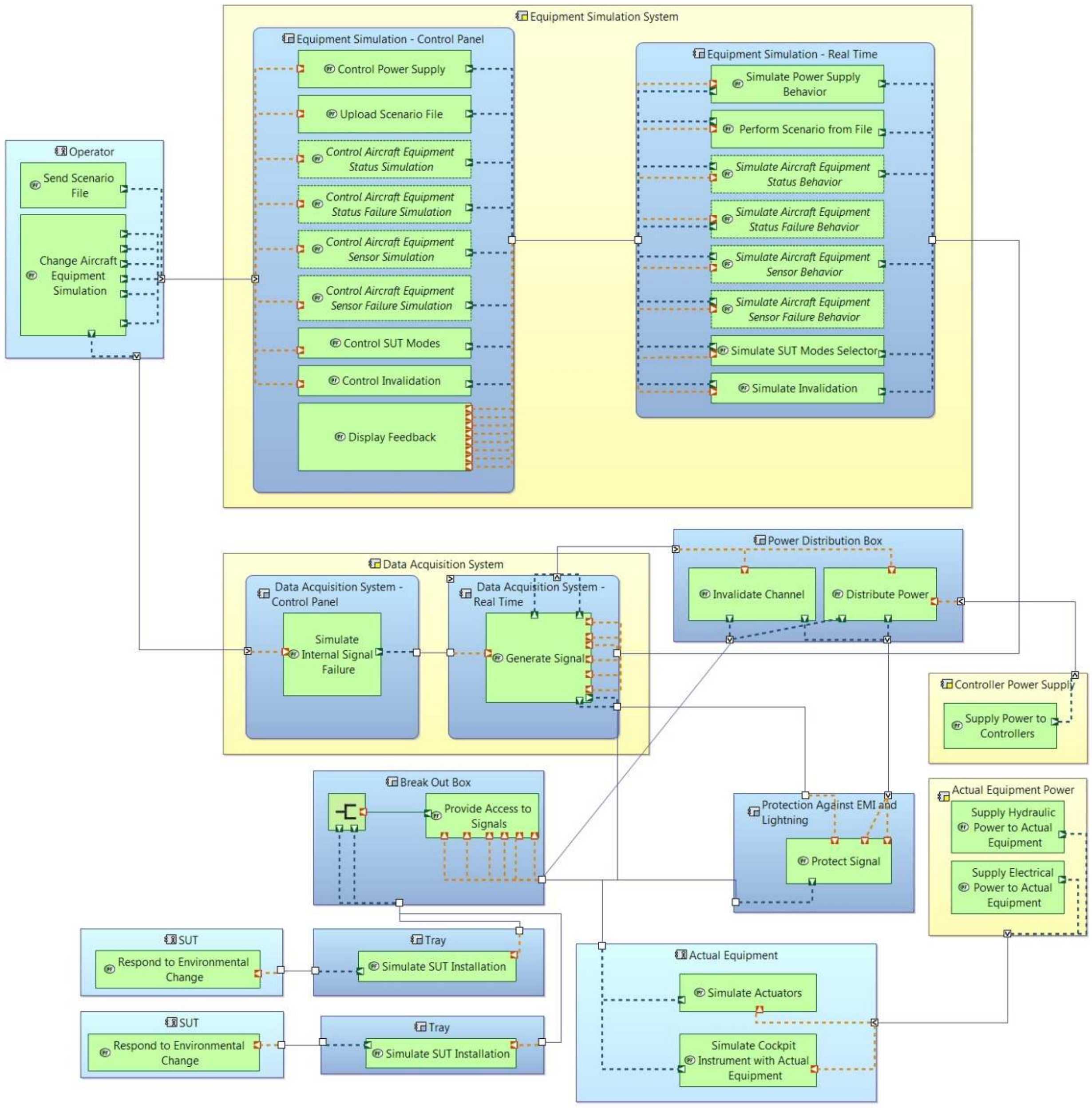

Operator
Change Aircraft Equipment Simulation
Send Scenario File
Request Acquistion
Actual Equipment
Simulate Actuators
Simulate Cockpit Instrument with Actual Equipment
SUT
Respond to Environmental Change
SUT
Respond to Environmental Change
Equipment Simulation System
Equipment Simulation - Control Panel
Control Power Supply
Control SUT Modes
Upload Scenario File
Control Aircraft Equipment Status Simulation
Control Aircraft Equipment Status Failure Simulation
Control Invalidation
Control Aircraft Equipment Sensor Simulation
Control Aircraft Equipment Sensor Failure Simulation
Display Feedback
Equipment Simulation - Real Time
Simulate Power Supply Behavior
Simulate SUT Modes Selector
Perform Scenario from File
Simulate Aircraft Equipment Status Failure Behavior
Simulate Aircraft Equipment Status Behavior
Simulate Invalidation
Simulate Aircraft Equipment Sensor Behavior
Simulate Aircraft Equipment Sensor Failure Behavior
Controller Power Supply
Supply Power to Controllers
Actual Equipment Power Supply
Supply Electrical Power to Actual Equipment
Supply Hydraulic Power to Actual Equipment
Tray
Simulate SUT Installation
Tray
Simulate SUT Installation
Break Out Box
Provide Access to Signals
Power Distribution Box
Invalidate Channel
Distribute Power
Protection Against EMI and Lightning
Protect Signal
Data Acquisition System
Data Acquisition System - Control Panel
Display Response Signal
Simulate Internal Signal Failure
Display Simulation Signal
Control Signal Acquisition
Data Acquisition System - Real Time
Record Simulation Signal
Generate Signal
Acquire Simulation Signal
Record Response Signal
Acquire Response Signal
SUT Management System
Maintenance
Operate Maintenanace
Internal Data Modifier
Modify Internal Variable Signal
Software and Memory Manager
Manage Software and Memory
Synoptic
Simulate Synoptic
Plotter
Plot Acquired Signal
Internal Data Detector
Acquire and Record Internal Signal
SUT Managing Signal Server
Generate SUT Management Signal
Acquire SUT Management Signal

Figure A. 22. General model diagram [PAB]

A.4.1.2. Simulating signal [PAB]



Equipment Simulation System
Equipment Simulation - Control Panel
Control Power Supply
Upload Scenario File
Control Aircraft Equipment Status Simulation
Control Aircraft Equipment Status Failure Simulation
Control Aircraft Equipment Sensor Simulation
Control Aircraft Equipment Sensor Failure Simulation
Control SUT Modes
Control Invalidation
Display Feedback
Equipment Simulation - Real Time
Simulate Power Supply Behavior
Perform Scenario from File
Simulate Aircraft Equipment Status Behavior
Simulate Aircraft Equipment Status Failure Behavior
Simulate Aircraft Equipment Sensor Behavior
Simulate Aircraft Equipment Sensor Failure Behavior
Simulate SUT Modes Selector
Simulate Invalidation
Operator
Send Scenario File
Change Aircraft Equipment Simulation
Data Acquisition System
Data Acquisition System - Control Panel
Simulate Internal Signal Failure
Data Acquisition System - Real Time
Generate Signal
Power Distribution Box
Invalidate Channel
Distribute Power
Controller Power Supply
Supply Power to Controllers
Break Out Box
Provide Access to Signals
Protection Against EMI and Lightning
Protect Signal
Actual Equipment Power
Supply Hydraulic Power to Actual Equipment
Supply Electrical Power to Actual Equipment
SUT
Respond to Environmental Change
Tray
Simulate SUT Installation
SUT
Respond to Environmental Change
Tray
Simulate SUT Installation
Actual Equipment
Simulate Actuators
Simulate Cockpit Instrument with Actual Equipment

Figure A. 23. Simulating signal diagram [PAB]

A.4.1.3. Detecting response [PAB]

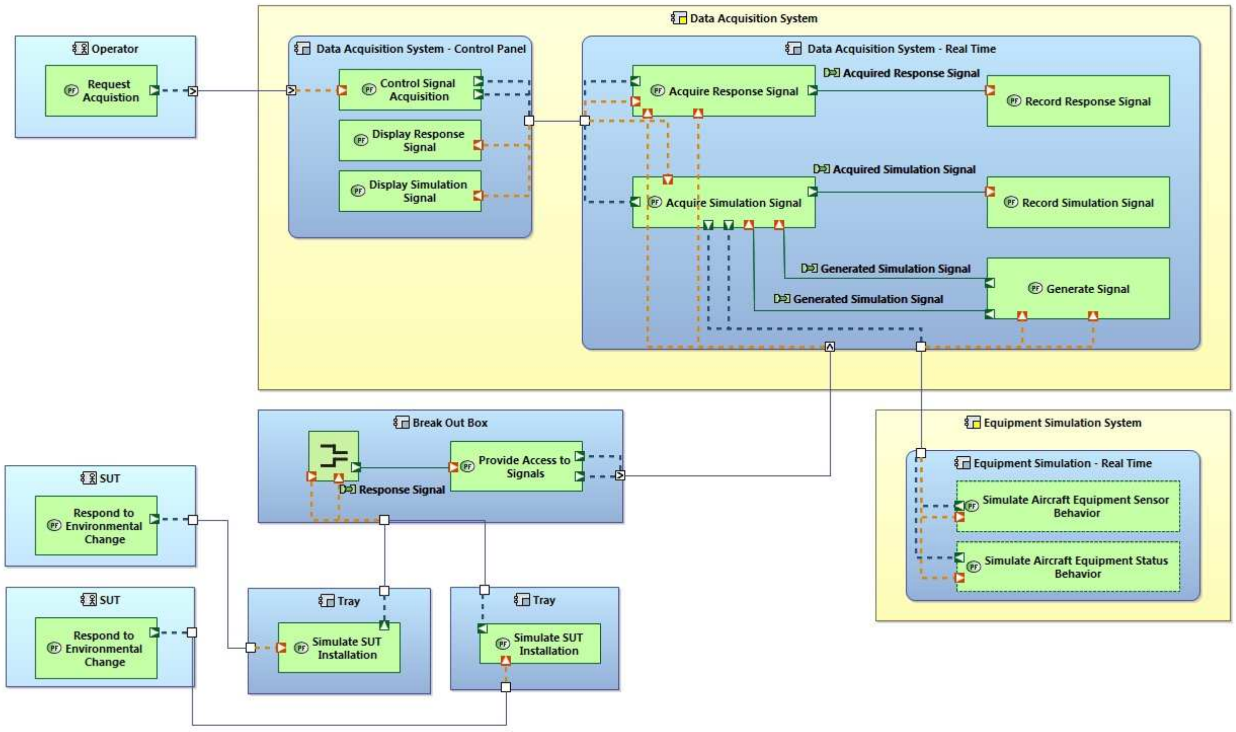

Data Acquisition System
Operator
Request Acquistion
Data Acquisition System - Control Panel
Control Signal Acquisition
Display Response Signal
Display Simulation Signal
Data Acquisition System - Real Time
Acquire Response Signal
Acquired Response Signal
Record Response Signal
Acquired Simulation Signal
Acquire Simulation Signal
Record Simulation Signal
Generated Simulation Signal
Generated Simulation Signal
Generate Signal
Break Out Box
Provide Access to Signals
Response Signal
Equipment Simulation System
Equipment Simulation - Real Time
Simulate Aircraft Equipment Sensor Behavior
Simulate Aircraft Equipment Status Behavior
SUT
Respond to Environmental Change
SUT
Respond to Environmental Change
Tray
Simulate SUT Installation
Tray
Simulate SUT Installation

Figure A. 24. Detecting response diagram [PAB]

A.4.1.4. Managing SUT [PAB]

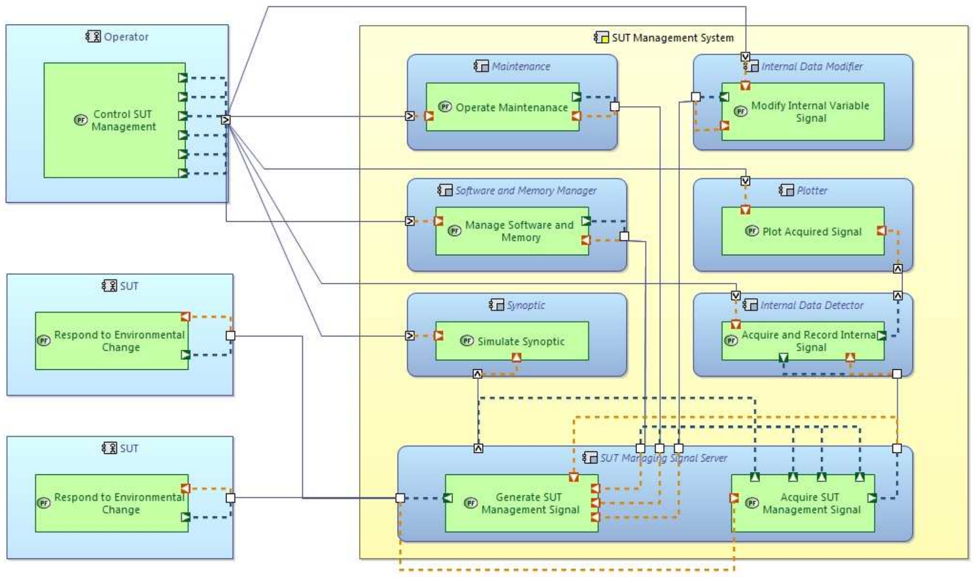

Operator
Control SUT Management
SUT
Respond to Environmental Change
SUT
Respond to Environmental Change
SUT Management System
Maintenance
Operate Maintenanace
Internal Data Modifier
Modify Internal Variable Signal
Software and Memory Manager
Manage Software and Memory
Plotter
Plot Acquired Signal
Synoptic
Simulate Synoptic
Internal Data Detector
Acquire and Record Internal Signal
SUT Managing Signal Server
Generate SUT Management Signal
Acquire SUT Management Signal

Figure A. 25. Managing SUT diagram [PAB]

A.4.2. Physical dataflow [PDFB]

Dataflow diagram shows the functional exchanges between the functions allocated to components in architecture diagram (section A.4.1.). At this level, there are two diagrams of this type modelled.

A.4.2.1. Simulating signal [PDFB]

The simulating signal diagram of dataflow type in physical level is shown in Figure A. 26.. This diagram show the functional exchanges of Figure A. 5..

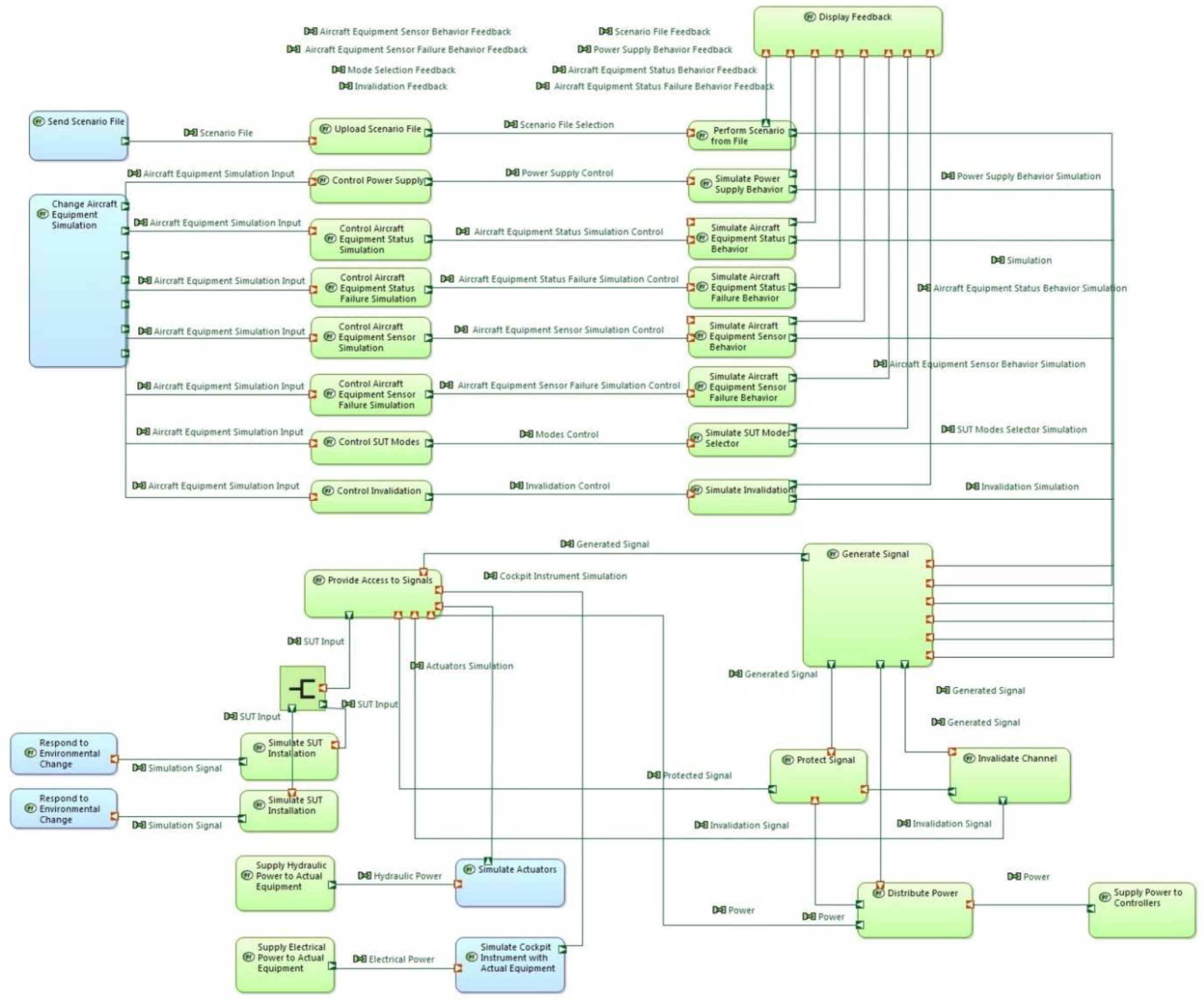

Display Feedback
Aircraft Equipment Sensor Behavior Feedback
Aircraft Equipment Sensor Failure Behavior Feedback
Mode Selection Feedback
Invalidation Feedback
Scenario File Feedback
Power Supply Behavior Feedback
Aircraft Equipment Status Behavior Feedback
Aircraft Equipment Status Failure Behavior Feedback
Send Scenario File
Scenario File
Upload Scenario File
Scenario File Selection
Perform Scenario from File
Change Aircraft Equipment Simulation
Aircraft Equipment Simulation Input
Control Power Supply
Power Supply Control
Simulate Power Supply Behavior
Power Supply Behavior Simulation
Control Aircraft Equipment Status Simulation
Aircraft Equipment Status Simulation Control
Simulate Aircraft Equipment Status Behavior
Simulation
Control Aircraft Equipment Status Failure Simulation
Aircraft Equipment Status Failure Simulation Control
Simulate Aircraft Equipment Status Failure Behavior
Aircraft Equipment Status Behavior Simulation
Control Aircraft Equipment Sensor Simulation
Aircraft Equipment Sensor Simulation Control
Simulate Aircraft Equipment Sensor Behavior
Aircraft Equipment Sensor Behavior Simulation
Control Aircraft Equipment Sensor Failure Simulation
Aircraft Equipment Sensor Failure Simulation Control
Simulate Aircraft Equipment Sensor Failure Behavior
Control SUT Modes
Modes Control
Simulate SUT Modes Selector
SUT Modes Selector Simulation
Control Invalidation
Invalidation Control
Simulate Invalidation
Invalidation Simulation
Generated Signal
Generate Signal
Provide Access to Signals
Cockpit Instrument Simulation
SUT Input
Actuators Simulation
Generated Signal
Generated Signal
Generated Signal
Respond to Environmental Change
Simulation Signal
Simulate SUT Installation
Respond to Environmental Change
Simulation Signal
Simulate SUT Installation
Protected Signal
Protect Signal
Invalidate Channel
Invalidation Signal
Invalidation Signal
Supply Hydraulic Power to Actual Equipment
Hydraulic Power
Simulate Actuators
Power
Distribute Power
Supply Power to Controllers
Power
Power
Supply Electrical Power to Actual Equipment
Electrical Power
Simulate Cockpit Instrument with Actual Equipment

Figure A. 26. Simulating signal diagram [PDFB]

A.4.2.2. Detecting response [PDFB]

The detecting response diagram of dataflow type in physical level is shown in Figure A. 27.. This diagram show the functional exchanges of Figure A. 24..

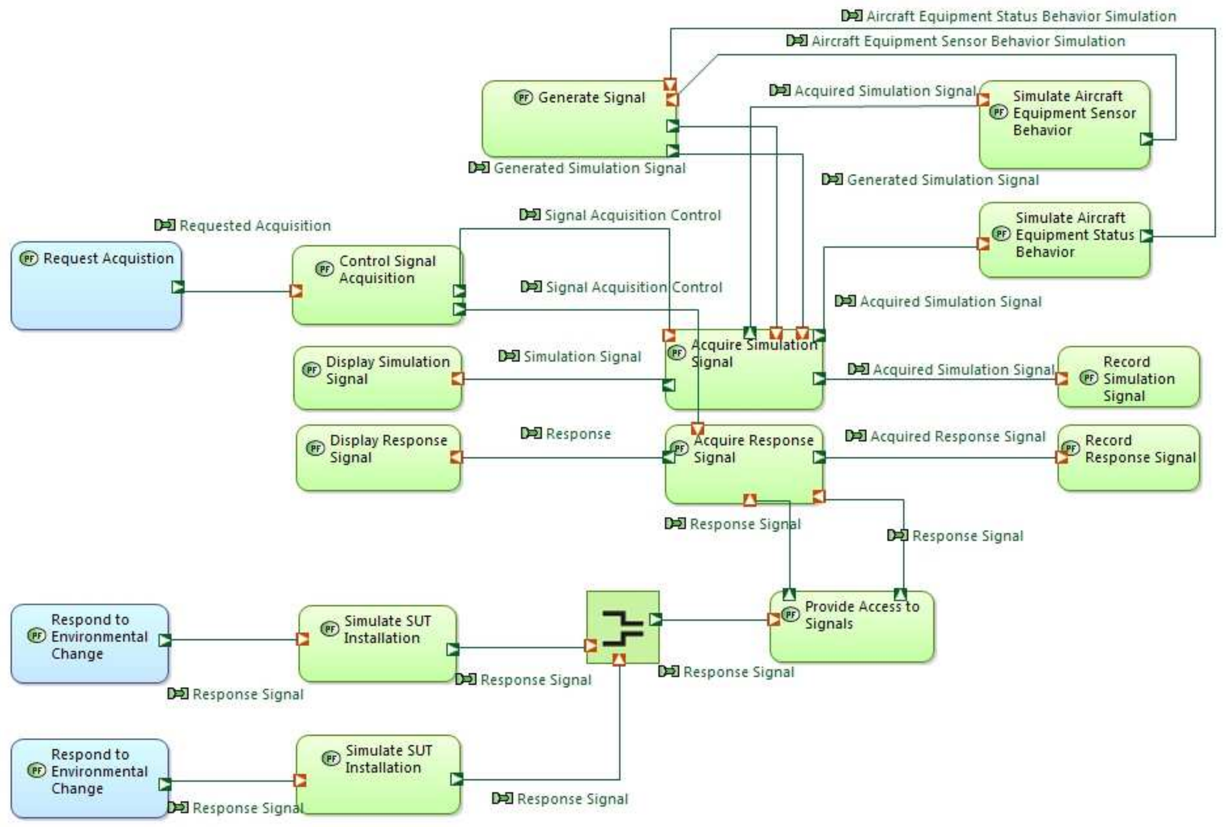

Aircraft Equipment Status Behavior Simulation
Aircraft Equipment Sensor Behavior Simulation
Generate Signal
Acquired Simulation Signal
Simulate Aircraft Equipment Sensor Behavior
Generated Simulation Signal
Generated Simulation Signal
Requested Acquisition
Signal Acquisition Control
Simulate Aircraft Equipment Status Behavior
Request Acquistion
Control Signal Acquisition
Signal Acquisition Control
Acquired Simulation Signal
Display Simulation Signal
Simulation Signal
Acquire Simulation Signal
Acquired Simulation Signal
Record Simulation Signal
Display Response Signal
Response
Acquire Response Signal
Acquired Response Signal
Record Response Signal
Response Signal
Response Signal
Respond to Environmental Change
Simulate SUT Installation
Provide Access to Signals
Response Signal
Response Signal
Response Signal
Respond to Environmental Change
Simulate SUT Installation
Response Signal
Response Signal

Figure A. 27. Detecting response diagram [PDFB]

A.4.3. Physical functional breakdown [PFBD]

There are eight breakdown of functions in the physical level.

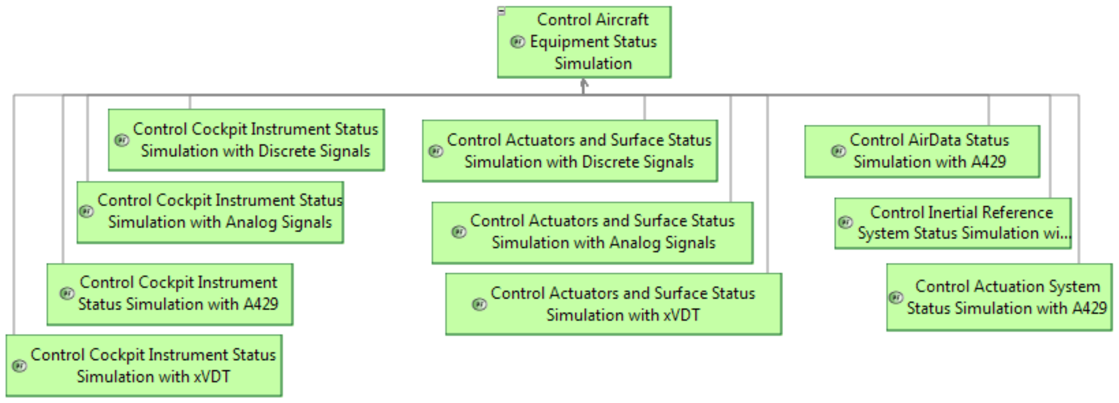

Control Aircraft Equipment Status Simulation
Control Cockpit Instrument Status Simulation with Discrete Signals
Control Cockpit Instrument Status Simulation with Analog Signals
Control Cockpit Instrument Status Simulation with A429
Control Cockpit Instrument Status Simulation with xVDT
Control Actuators and Surface Status Simulation with Discrete Signals
Control Actuators and Surface Status Simulation with Analog Signals
Control Actuators and Surface Status Simulation with xVDT
Control AirData Status Simulation with A429
Control Inertial Reference System Status Simulation wi...
Control Actuation System Status Simulation with A429

Figure A. 28. Control aircraft equipment status simulation breakdown diagram [PFBD]

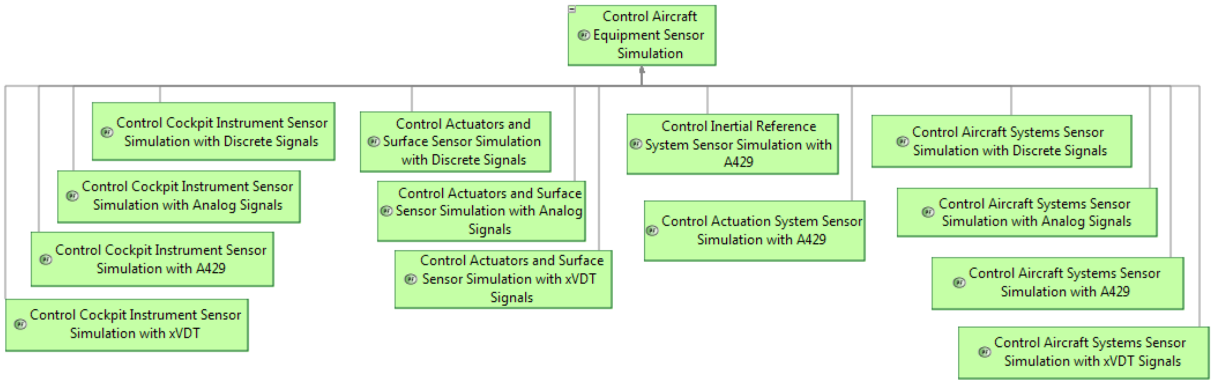

Control Aircraft Equipment Sensor Simulation
Control Cockpit Instrument Sensor Simulation with Discrete Signals
Control Cockpit Instrument Sensor Simulation with Analog Signals
Control Cockpit Instrument Sensor Simulation with A429
Control Cockpit Instrument Sensor Simulation with xVDT
Control Actuators and Surface Sensor Simulation with Discrete Signals
Control Actuators and Surface Sensor Simulation with Analog Signals
Control Actuators and Surface Sensor Simulation with xVDT Signals
Control Inertial Reference System Sensor Simulation with A429
Control Actuation System Sensor Simulation with A429
Control Aircraft Systems Sensor Simulation with Discrete Signals
Control Aircraft Systems Sensor Simulation with Analog Signals
Control Aircraft Systems Sensor Simulation with A429
Control Aircraft Systems Sensor Simulation with xVDT Signals

Figure A. 29. Control aircraft equipment sensor simulation breakdown diagram [PFBD]

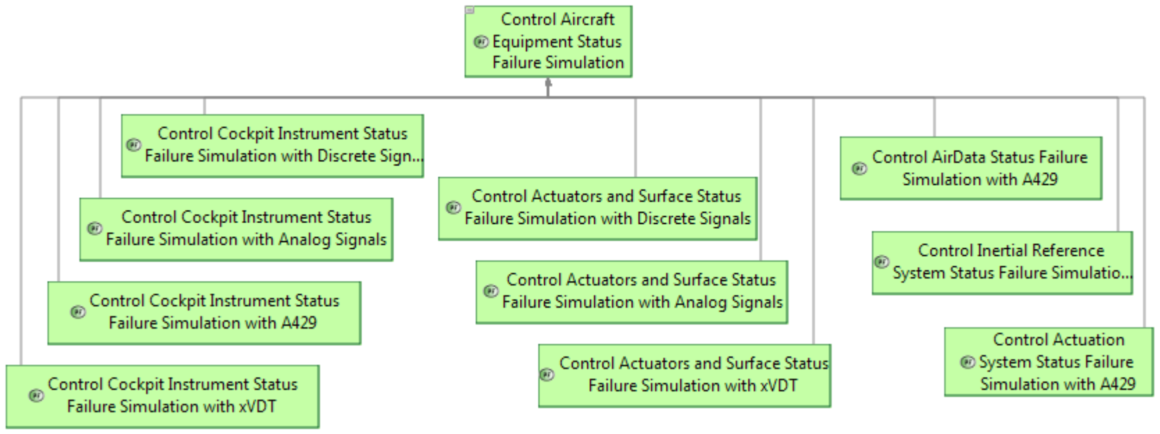

Control Aircraft Equipment Status Failure Simulation
Control Cockpit Instrument Status Failure Simulation with Discrete Sign...
Control Cockpit Instrument Status Failure Simulation with Analog Signals
Control Cockpit Instrument Status Failure Simulation with A429
Control Cockpit Instrument Status Failure Simulation with xVDT
Control Actuators and Surface Status Failure Simulation with Discrete Signals
Control Actuators and Surface Status Failure Simulation with Analog Signals
Control Actuators and Surface Status Failure Simulation with xVDT
Control AirData Status Failure Simulation with A429
Control Inertial Reference System Status Failure Simulatio...
Control Actuation System Status Failure Simulation with A429

Figure A. 30. Control aircraft equipment status failure simulation breakdown diagram [PFBD]

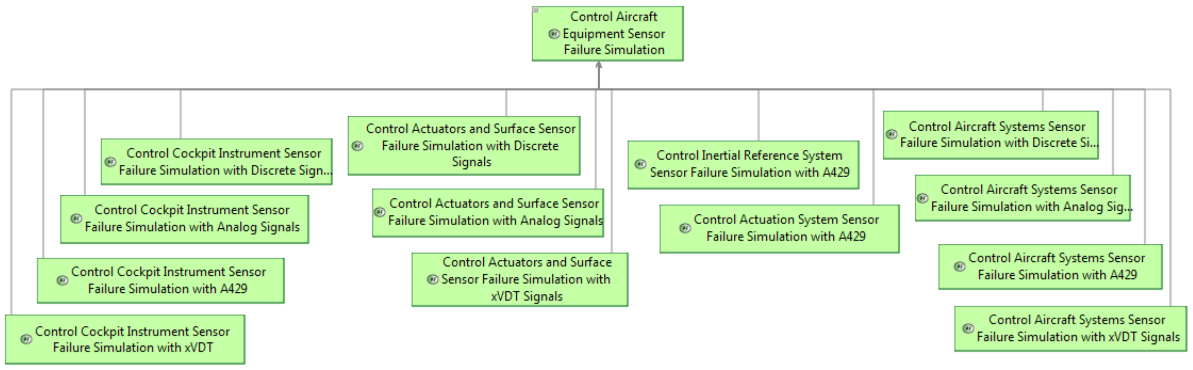

Control Aircraft Equipment Sensor Failure Simulation
Control Cockpit Instrument Sensor Failure Simulation with Discrete Sign...
Control Cockpit Instrument Sensor Failure Simulation with Analog Signals
Control Cockpit Instrument Sensor Failure Simulation with A429
Control Cockpit Instrument Sensor Failure Simulation with xVDT
Control Actuators and Surface Sensor Failure Simulation with Discrete Signals
Control Actuators and Surface Sensor Failure Simulation with Analog Signals
Control Actuators and Surface Sensor Failure Simulation with xVDT Signals
Control Inertial Reference System Sensor Failure Simulation with A429
Control Actuation System Sensor Failure Simulation with A429
Control Aircraft Systems Sensor Failure Simulation with Discrete Si...
Control Aircraft Systems Sensor Failure Simulation with Analog Sig...
Control Aircraft Systems Sensor Failure Simulation with A429
Control Aircraft Systems Sensor Failure Simulation with xVDT Signals

Figure A. 31. Control aircraft equipment sensor failure simulation breakdown diagram [PFBD]

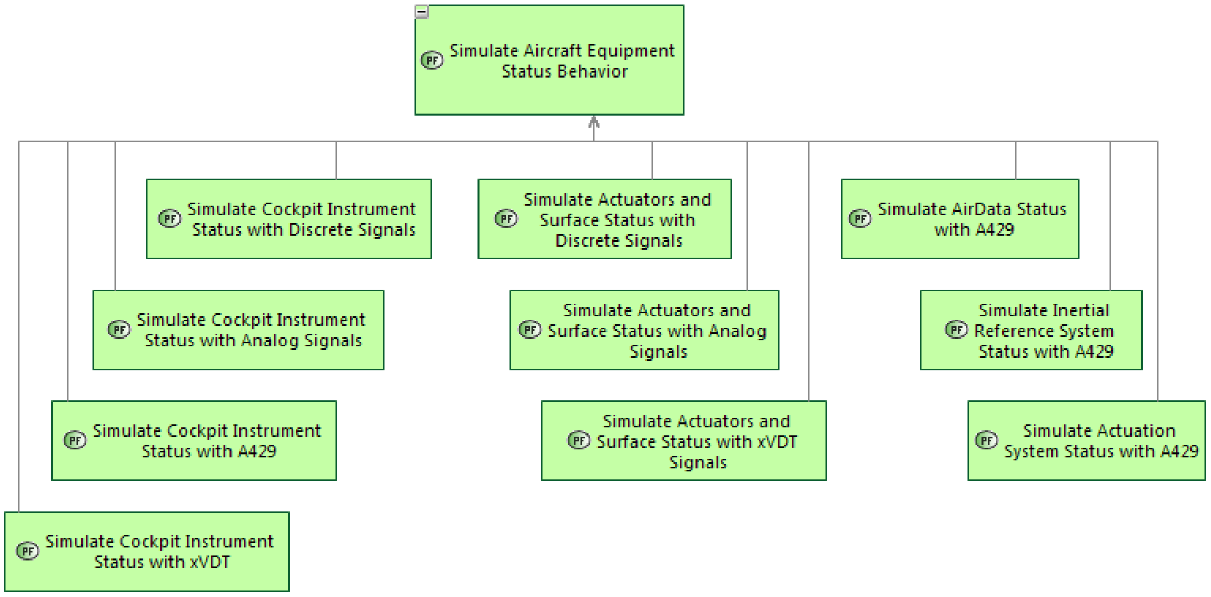

Simulate Aircraft Equipment Status Behavior
Simulate Cockpit Instrument Status with Discrete Signals
Simulate Cockpit Instrument Status with Analog Signals
Simulate Cockpit Instrument Status with A429
Simulate Cockpit Instrument Status with xVDT
Simulate Actuators and Surface Status with Discrete Signals
Simulate Actuators and Surface Status with Analog Signals
Simulate Actuators and Surface Status with xVDT Signals
Simulate AirData Status with A429
Simulate Inertial Reference System Status with A429
Simulate Actuation System Status with A429

Figure A. 32. Aircraft equipment status simulation breakdown diagram [PFBD]

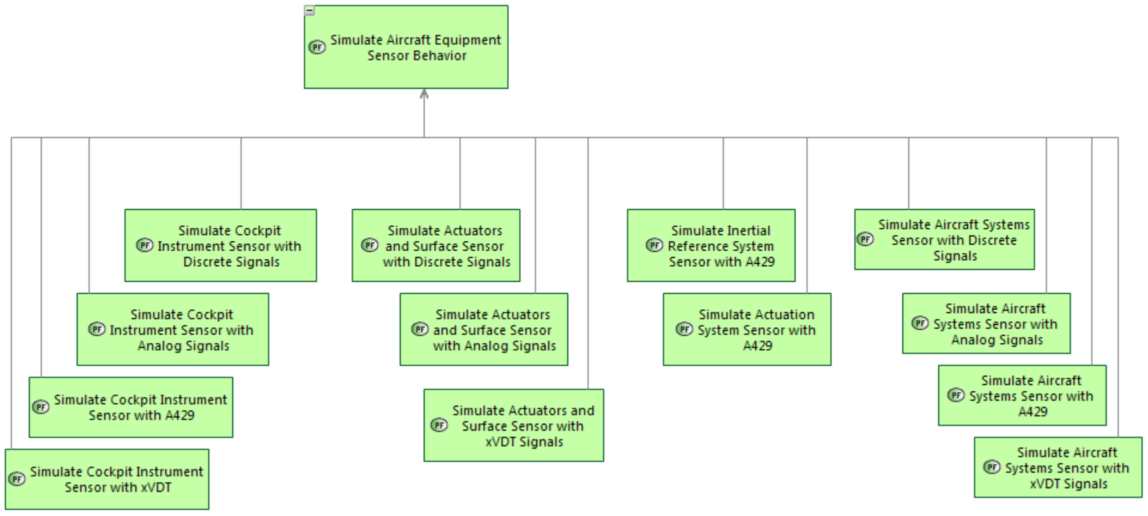
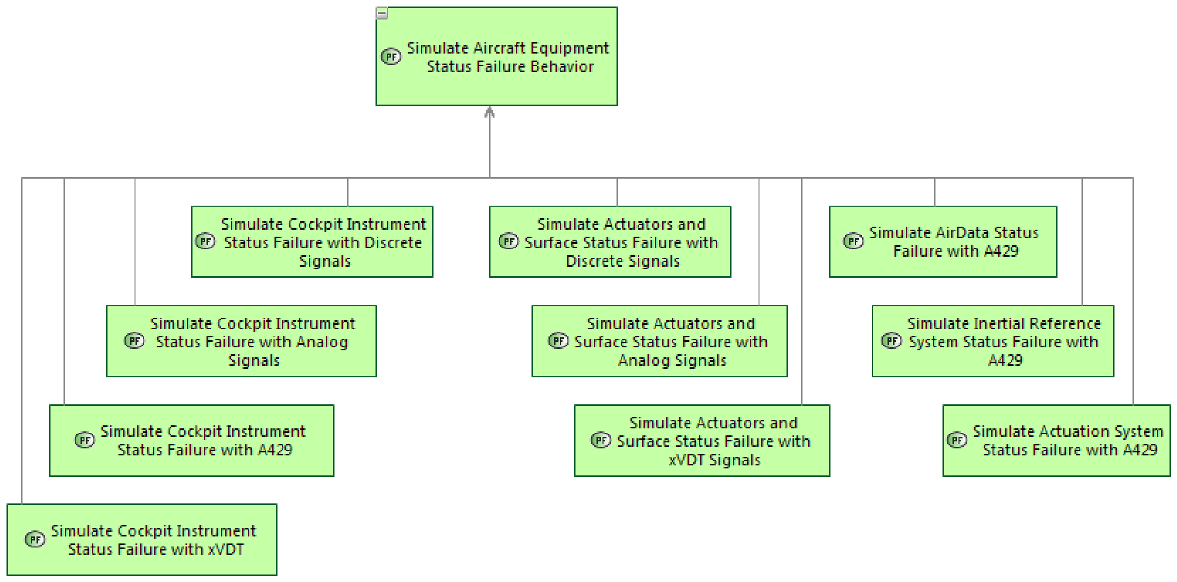
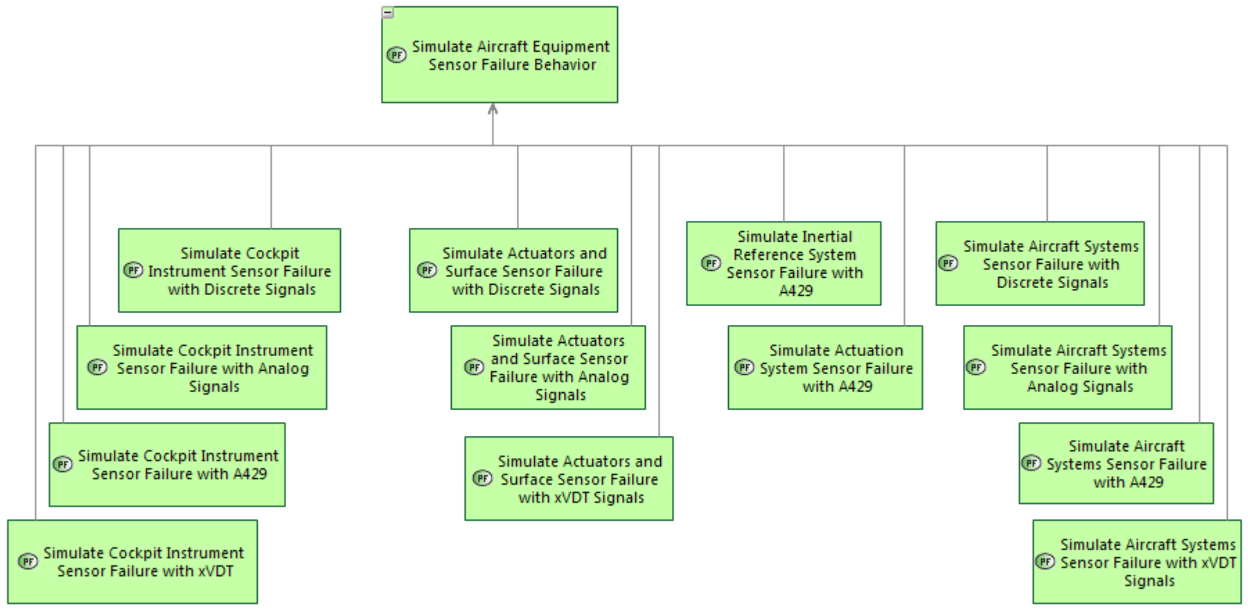

Simulate Aircraft Equipment Sensor Behavior
Simulate Cockpit Instrument Sensor with Discrete Signals
Simulate Cockpit Instrument Sensor with Analog Signals
Simulate Cockpit Instrument Sensor with A429
Simulate Cockpit Instrument Sensor with xVDT
Simulate Actuators and Surface Sensor with Discrete Signals
Simulate Actuators and Surface Sensor with Analog Signals
Simulate Actuators and Surface Sensor with xVDT Signals
Simulate Inertial Reference System Sensor with A429
Simulate Actuation System Sensor with A429
Simulate Aircraft Systems Sensor with Discrete Signals
Simulate Aircraft Systems Sensor with Analog Signals
Simulate Aircraft Systems Sensor with A429
Simulate Aircraft Systems Sensor with xVDT Signals

Figure A. 33. Aircraft equipment sensor simulation breakdown diagram [PFBD]



Simulate Aircraft Equipment Status Failure Behavior
Simulate Cockpit Instrument Status Failure with Discrete Signals
Simulate Actuators and Surface Status Failure with Discrete Signals
Simulate AirData Status Failure with A429
Simulate Cockpit Instrument Status Failure with Analog Signals
Simulate Actuators and Surface Status Failure with Analog Signals
Simulate Inertial Reference System Status Failure with A429
Simulate Cockpit Instrument Status Failure with A429
Simulate Actuators and Surface Status Failure with xVDT Signals
Simulate Actuation System Status Failure with A429
Simulate Cockpit Instrument Status Failure with xVDT

Figure A. 34. Aircraft equipment status failure simulation breakdown diagram [PFBD]



Simulate Aircraft Equipment Sensor Failure Behavior
Simulate Cockpit Instrument Sensor Failure with Discrete Signals
Simulate Actuators and Surface Sensor Failure with Discrete Signals
Simulate Inertial Reference System Sensor Failure with A429
Simulate Aircraft Systems Sensor Failure with Discrete Signals
Simulate Cockpit Instrument Sensor Failure with Analog Signals
Simulate Actuators and Surface Sensor Failure with Analog Signals
Simulate Actuation System Sensor Failure with A429
Simulate Aircraft Systems Sensor Failure with Analog Signals
Simulate Cockpit Instrument Sensor Failure with A429
Simulate Actuators and Surface Sensor Failure with xVDT Signals
Simulate Aircraft Systems Sensor Failure with A429
Simulate Cockpit Instrument Sensor Failure with xVDT
Simulate Aircraft Systems Sensor Failure with xVDT Signals

Figure A. 35. Figure A. 34. Aircraft equipment sensor failure simulation breakdown diagram [PFBD]

Appendix B: How to select a configuration in Pure::Variants

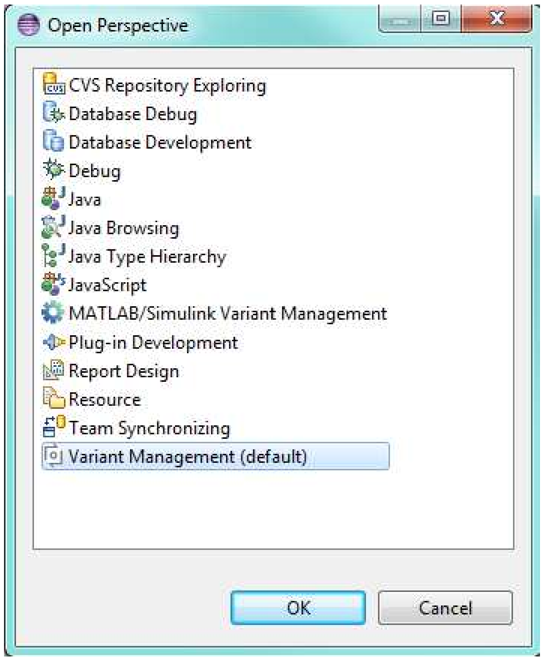
The general model consists of all the features (components, functions and entities) that could exist in the test means. To generate a new model, which means the selection of features needed in the test means, the pure variants is used. There are tow ways to define a new configuration. 1) using the Pure Variants. 2) using the instance of Pure Variants in Capella (Mappings view). In this section, both of these approaches are discussed.

B.1. Pure Variants

The first step to choose the features is to open a .vdm file (Variant Description Model file type) in the project. There are several modes available in Pure Variants. To manage the modes:

Windows → Open Perspective → Other

To generate a VDM, the "Variant Management" mode must be enabled.


Open Perspective

- CVS Repository Exploring
- Database Debug
- Database Development
- Debug
- Java
- Java Browsing
- Java Type Hierarchy
- JavaScript
- MATLAB/Simulink Variant Management
- Plug-in Development
- Report Design
- Resource
- Team Synchronizing
- Variant Management (default)

Figure B. 1 Perspectives in Pure Variants.

In the top-right corner of the Pure Variants window, the selected mode can be checked. Then, right-click on the project "TestMeans" in "Variants Project" view → New → Configuration Space. Note that in other modes will not able you to create a "Configuration Space".

Right-click on the configuration space file → New → Variant Model. this is the "vdm" file. When it is opened, the desired configuration can be selected. Mandatory features are already in the configuration, there is no need to select them. Then, save the file and import copy the project in Capella workspace. To import it: File → import → Existing projects into Workspace → Browse → find the workspace and select the Pure Variants project.

Then, open the Mappings view in Capella.

Windows → Show View → Other → Variant Management → Mappings

B.2. Mappings view in Capella

Mappings view only enables you to edit the file. In other words, it is not possible to create a VDM type of file using Mappings view. The file must be already created in Pure Variants and imported to Capella workspace.

Then, open the Mappings view in Capella.

Windows → Show View → Other → Variant Management → Mappings

Appendix C: How to use M2Doc

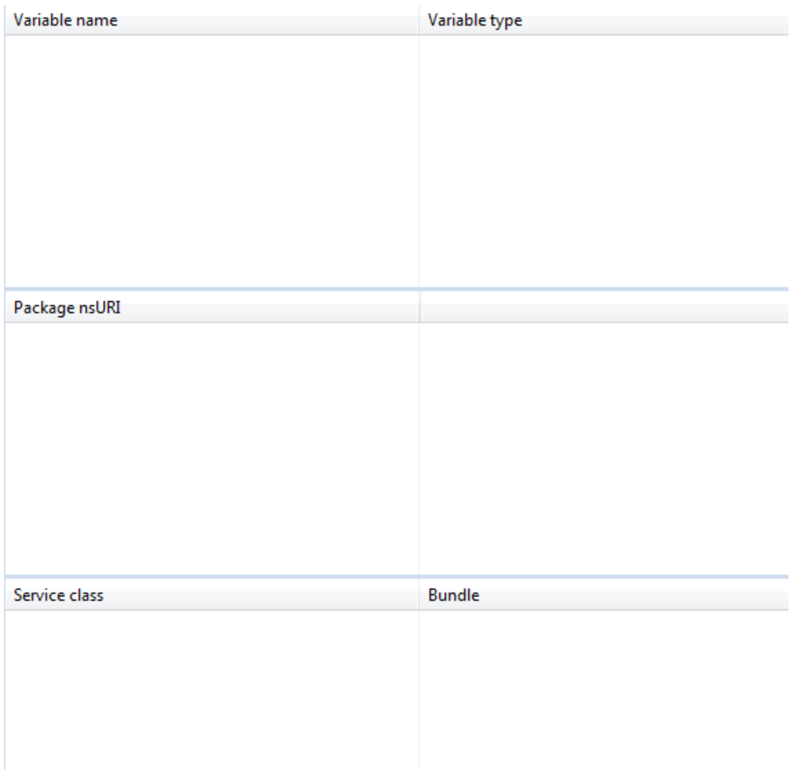
M2Doc is an add-on of Capella that enables the user to generate a Microsoft Word type of file from the model. To generate the document, a template must be created which means a simple MS Word file must be coded. To do so, open a new Word. Click Alt+F9 to change the mode to the coding mode. After coding it, save the file in the project file in Capella's workspace. The add-on needs one other Word file to generate the document. Therefore, there should be two Word files in the project. One the template (the coded) and one other is a blank file which the model document will be generated in.

After creating the template and the blank files inside project, refresh the project in Capella.

Right-click on the project in Project Explorer window in Capella → Refresh. Now the Word files must be visible.

The template must become compatible to Capella. To do that, follow the following procedure:

1. Open the template file in M2Doc editor: right-click on the template file → Open with → Other → M2Doc Template Editor. The opened window has three sections as shown in Figure C. 1
2. Right-click in on “Package nsURI” section → Add → search “ *Capella” and select all the found packages.
3. Right-click on the first block (variables) → Add missing variables. (it will automatically find variables from the word file. Then you need to define the type of variable by selecting a eClass from Capella data model). for the template the missing variable was “capellamodeller::SystemEngineering”.
4. Close the M2Doc editor and right-click on the template Word file in project explorer → Initial Documentation Configurations. A .genconf type of file has been generated. Click on it to get it open. The opened view is shown in Figure C. 2.
5. in “Destination URL” part, determine the destination file (the Word file which you want the model to be generated in).
6. In the selection tab (at the bottom of the window), in resource set, do right click → Load Resource → browse workspace → select the .melodymodeller file of the project.
7. Back in the overview tab, set the variable value: you have to select by hand the model element corresponding to your variable base on the type declared at step 3 (you will be offered the choice only between the model elements matching the type previously selected).
8. Convert your project in modeling project. Right click on the project folder → configure → convert to modeling project
9. In the option block of the .genconf file, right click → Initialize Option.

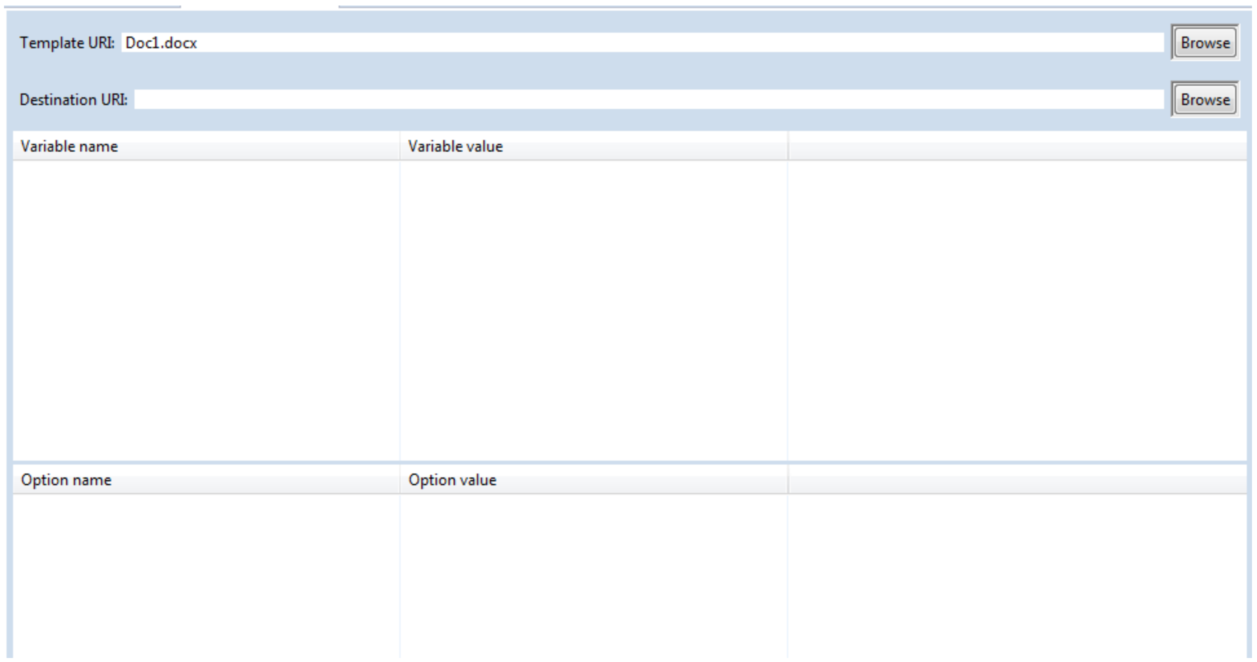


Variable name	Variable type

Package nsURI	

Service class	Bundle

Figure C. 1. . M2Doc template editor



Template URI: Doc1.docx

Destination URI:

Variable name	Variable value	

Option name	Option value	

Figure C. 2. Generated configuration file view in Capella