

Extending Model-Based Systems Engineering into Integrated Platform Designed for Civil Aircraft Development

Wenhao Bi^{1,*} , Wenhao Wang^{1,2} , An Zhang¹ , Yunong Wang¹ 

1.Northwestern Polytechnical University – School of Aeronautics – Department of Control and Information Engineering – Shaanxi/ Xi'an – China. **2.**Aviation Industry Corporation of China – Xi'an Flight Automatic Control Research Institute – Center of Innovation – Shaanxi/ Xi'an – China.

*Corresponding author: biwenhao@nwpu.edu.cn

ABSTRACT

With the continuous development and application of modeling technology, civil aircraft designers urgently need to apply model-based systems engineering (MBSE) theory and methods into the field of civil aircraft development. However, the existing modeling tools, which combined with MBSE methodologies, only focus on one particular area, such as requirement modeling, functional modeling and design synthesis. This kind of low coupling design system makes it difficult to link the upstream and downstream designers in the early stage of system development. This paper introduces the MBSE development process of civil aircraft and then establishes an integrated platform composed of six work stations for civil aircraft development based on this process. Commercially available modeling tools, such as IBM Rational DOORS and enterprise architect (EA), are conducted in the platform. The development of flight control system is taken as an example to test the feasibility and efficiency of our integrated platform in logic layer. This platform can complete the activities of aircraft life cycle function analysis, requirement and model management. The loose coupling between the modules and the high cohesive nature allows different system engineers to maintain their design independence and consistency at different stages of development.

Keywords: System Engineering; Integrated Platform; Flight Control System.

INTRODUCTION

Civil aircraft development is a complex system engineering problem, including the identification and capture of stakeholders and system requirements, establishing functional structure of aircraft systems and the derivation of functional analysis (Anderson 2009; Mordecai et al. 2018; Zhang et al. 2018). The entire research and development (R&D) process involves various stakeholders, professional tools in different subjects, constraints between different system functions, as well as interactions between hardware and software. Model-based system engineering (MBSE) has been used in many complex systems engineering fields, such as aircraft design, weapon system development and embedded system development (Anwar et al. 2017, 2019, 2020; Madni and Sievers 2018; Ramos et al. 2011).

Received: Oct. 19, 2020 | Accepted: Mar. 08, 2021

Peer Review History: Single Blind Peer Review

Section Editor: Alison Moraes



This is an open access article distributed under the terms of the Creative Commons license.

With the wide spread recognition of MBSE theory in the development of civil aircraft, related methods and tools are urgently needed to apply this modern theory into civil aircraft development process (Andersson 2009; Anwar et al. 2017; Dickerson and Mayris 2013; Mordecai et al. 2018; Zhang et al. 2018). The purpose of designing the civil aircraft MBSE integration platform is to achieve standardized management of all processes in the whole life cycle of the civil aircraft through modeling and digitization. Conflicts, infeasible requirements and functions can be found and modified at the beginning of the project, which is of great significance to reduce the development cycle and cost.

Model-based system engineering has a major transformation in modeling language, modeling ideas and modeling tools comparing to traditional text-based systems engineering (TSE) (Madni and Sievers 2018; Jing et al. 2016). It is the most versatile and efficient R&D mode for the development of large and complex systems (Cloutier et al. 2015; Piaszczyk 2011). Researchers have proposed various methodologies in MBSE theory and modeling processes, such as Harmony-SE, INCOSE object-oriented systems engineering method (OOSEM) and VITECH MBSE (Estefan 2007). Commercially available modeling tools, such as IBM Rational DOORS for requirements management developed by IBM, rhapsody for functional modeling and simulation, enterprise architect (EA) for requirement analysis and functional modeling, were also developed by software vendors. These tools are highly specific, only focusing on one period of system development. Whereas, these exiting tools lack support for the development process throughout the life-cycle of the system and require a high level of user expertise. Thus, we are here to design an integrated platform that can fully realize the civil aircraft system design and development by combining the commercial modeling tools, MBSE methodologies and our own software.

The novelty of our platform lies in the integration of MBSE concepts and methods, in the form of LAN database connection to replace the traditional text file transmission, which allows different system engineers to maintain design independence and consistency at different stages of development. Also, the integration of requirement management, functional modeling and process management can establish traceability between requirements, functions and design products, which is not available in a single modeling software, such as IBM rational rhapsody, MagicDraw and EA. In addition, the integration of commercial software can be replaced in the subsequent development work with a certain degree of independent intellectual property rights and adaptability for different projects.

CIVIL AIRCRAFT DEVELOPMENT PROCESS BASED ON MBSE

Systems engineering is an interdisciplinary field between engineering and engineering management which focuses on how to design and manage complex systems over their life cycle. Figure 1 shows the typical system engineering workflow.

As shown in Fig. 1, considering requirements as the beginning of system design, stakeholders and shareholder requirements are also the ultimate goals and constraints of design and development in the whole life cycle. The requirements and development constraints are determined through requirement analysis. Then the analysis and allocation of functions are carried out, decomposing the functions to the next level and defining the functional interfaces. After that, the functional architecture is designed and used as input to the comprehensive design, defining alternative system solutions, configuration items, physical interfaces and output physical architecture. Products which ultimately meet customer needs are iteratively obtained through all levels of design confirmation processes (Hoffmann 2011).

However, the traditional text-based system engineering is becoming more and more difficult to satisfy the complex system engineering requirements, when MBSE has emerged (Andersson 2009; Dickerson and Mayris 2013; Madni and Sievers 2018). Model-based system engineering is very different from traditional TSE methods in the design and development of large-scale civil aircraft. The main differences, also the advantages, can be summarized as follows:

Firstly, in the requirements development process, the requirements are linked to models which describe the system behavior (dynamic) and structure (static) of the operational scenario. The relationship between requirements and models is realized through the model traceability link, association matrix and other processes. This process should ensure that requirements can be accurately identified and decomposed, and traceability relationship can be established with the function of the system (Topper and Horner 2013).

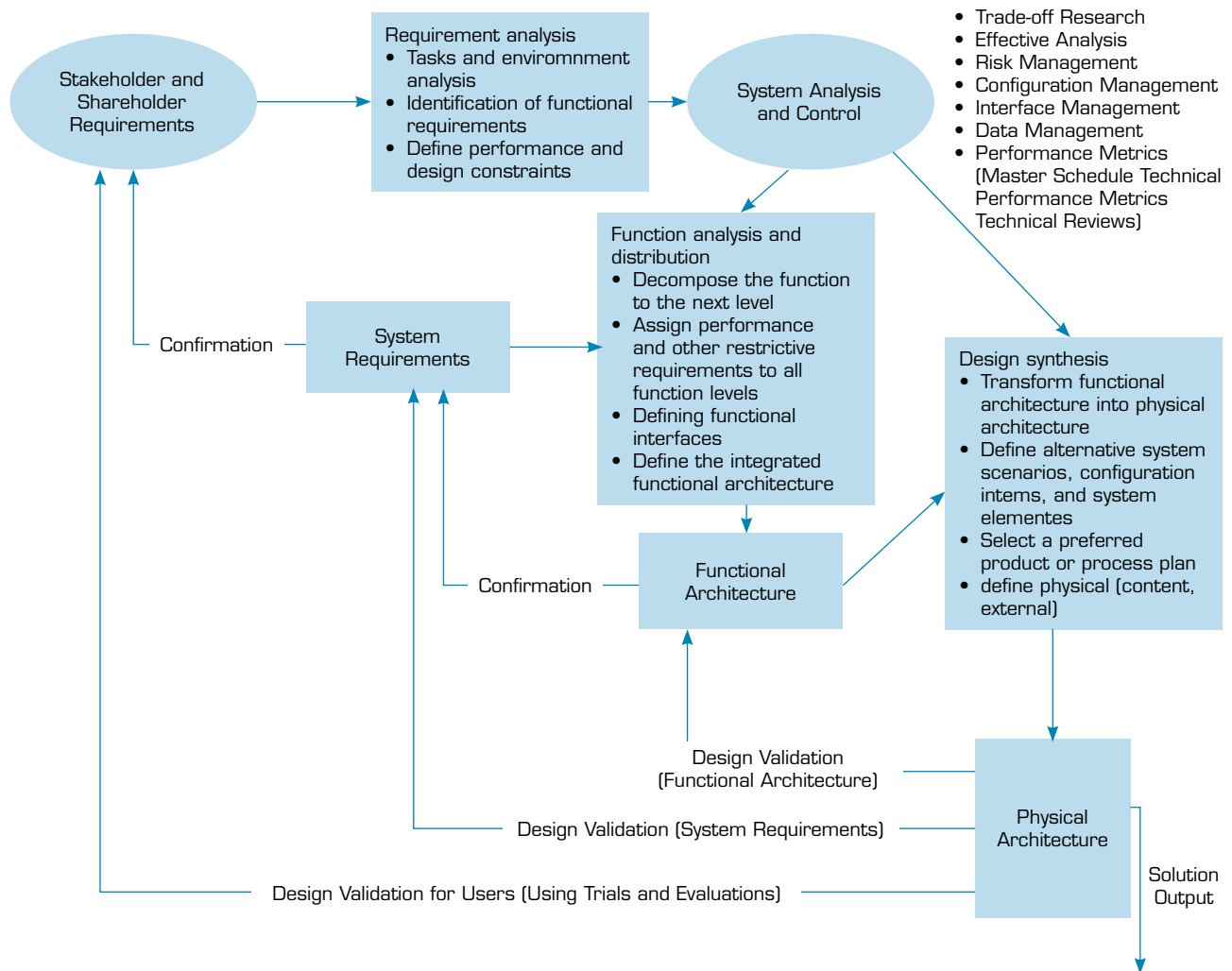


Figure 1. System engineering workflow.

Secondly, in the detailed design of the system, the activities and states of the system are described by intuitive model diagrams. The integrity, coordination and compliance verification of the system functional descriptions can be illustrated through the simulation of the system models, so that the verification and validation can be finished simultaneously with design development. By using this design architecture, it is ready to avoid the continuous impact of the system top-level requirements/design errors on subsequent development (Bozhi and Haomin 2013; Chi et al. 2017; Holt et al. 2015).

Thirdly, extending the system engineering method, MBSE implements the association of the requirement model to the system model. Through the continuous decomposition and expansion of the model, which describes the system and its simulation operation, the forward and consistent development of the system can be realized. Using requirements as constraints and target nodes, the process from use case analysis (task scenario analysis), functional analysis, architecture design (technical path and functional model framework for system function implementation) to physical design (physical implementation form of logical architecture) can be realized. Finally, the model simulation is performed to verify the compliance with the design and requirements within the system life cycle (Cloutier et al. 2014).

Combined with the typical MBSE workflow and the V model, this paper proposes a requirement-based MBSE civil aircraft development process as shown in Fig. 2. This process is a collection of practices that integrates system engineering personnel with work product definitions. After the project is launched, stakeholder requirements, system requirements, architecture analysis and design are transferred to downstream projects, which will be performed in coordinate with the specifications.

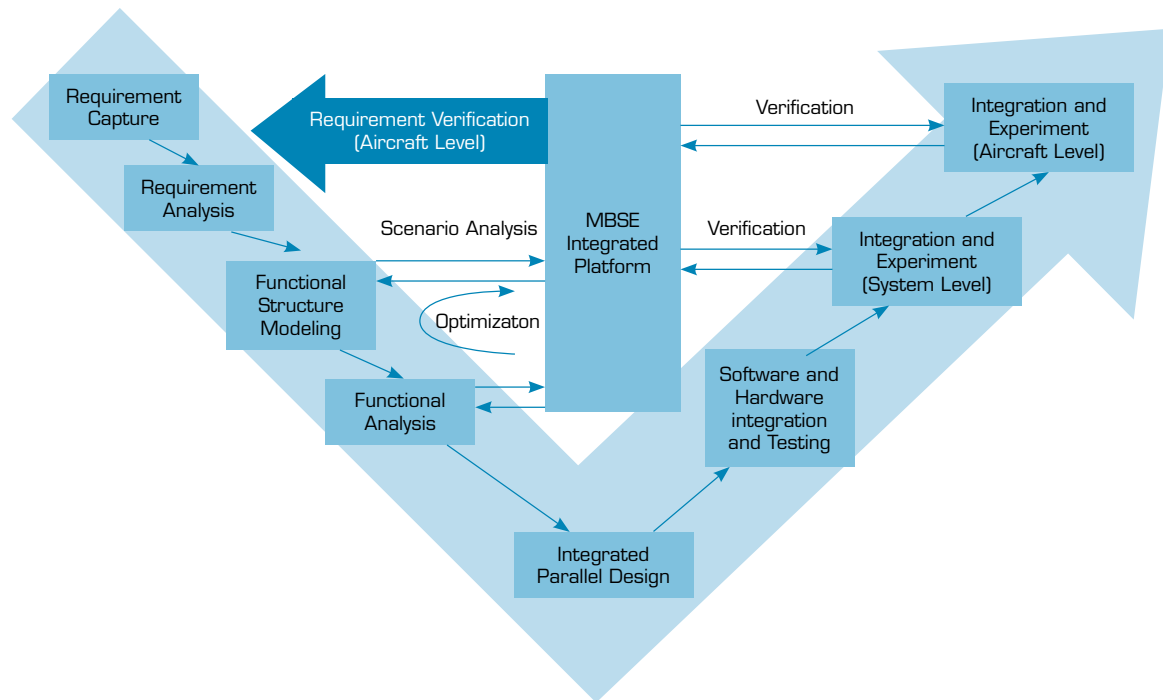


Figure 2. Requirement based civil aircraft MBSE development process.

The difference from the traditional V process is that when there are more top-level requirements, usually the stakeholder requirements, the activities are repeated several times and the development engineering data is incrementally transferred to the downstream project for more detailed development. By using this cycle, we can directly connect the stakeholder requirements with the bottom layer of development, updating products more satisfying and acceptable to the market.

In the following sections, this paper designed the civil aircraft MBSE integration platform and verified with the civil aircraft FCS, using commercial software DOORS to capture the airworthiness requirements and importing to the requirement management and analysis platform for requirements analysis. According to the operational scenario of the FCS, the UML class diagram is used to establish the scenario model and structuralize the functional network. The activity diagram (ACT) is used to describe the internal activities of the FCS. The block definition diagram (BDD) and the internal block diagram (IBD) are used to describe its internal organization. This top-down incremental iteration method can closely combine requirements with functions, simulation and verification, establish traceability of requirements and significantly shorten the development cycle of civil aircraft.

ARCHITECTURAL DESIGN AND IMPLEMENTATION OF THE INTEGRATED PLATFORM

In order to realize the aforementioned process, the platform architecture design has been established, which is shown in Fig. 3.

The platform enables all functional modeling development, verification/evaluation and requirements management, also the integration of functional modeling analysis at the aircraft level. This architecture is based on the MBSE methodologies and proposed in conjunction with the development process of the civil aircraft. From this architecture, the hardware platform modules can be extracted as follows.

As shown in Table 1, there are a total of six platforms/stations. The requirement management and analysis station (RMAS) is used to analyze, verify and review the requirements, which have been imported from outside of the system or created inside the platform. The functional structure and analysis station (FSAS) receives system requirements from RMAS and connects civil aircraft operation scenario with system requirements and build a requirement oriented civil aircraft functional network structure. Inside the functional structure, System Modeling Language (SysML) and commercially available software EA are used to describe its behaviors and structures. During this whole procedure, the Process Management Station (PMS) is used for risk management when requirement changes and the Integrated Display Station (IDS) is used to display the overall project progress and two databases are integrated into the platform simultaneously for model management and data interaction.

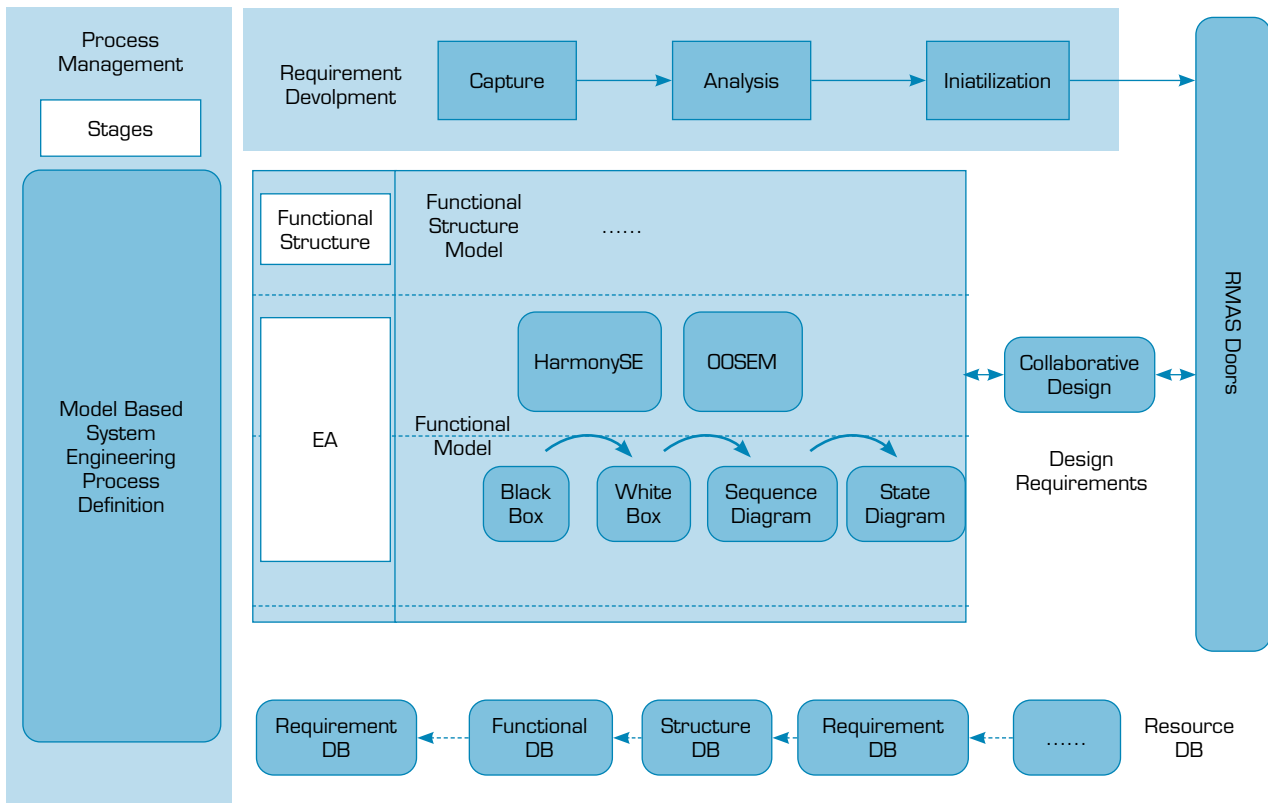


Figure 3. Civil aircraft MBSE platform architecture design.

Table 1. Hardware platform design and related function.

Platform name	Platform function
RMAS	Requirements management and analysis
FSAS	Functional analysis and modelling and operational scenario definition
PMS	Process management including risk assessment and requirement coverage calculation
IDS	Display the overall project progress and trace the changes in requirements
Req & Model DB	Requirements and model database

Based on the abovementioned platform architecture design, we built the platform hardware architecture shown in Fig. 4 and the information flow, as well as the interface between all six stations are shown in Fig. 5. Each station implements specific functions in the civil aircraft MBSE process and associated with corresponding software tools, such as EA, DOORS and specific software developed by MFC. The solid line between each station represents the data stream and format.

The specific information flow labeled in solid line is shown as follows. As an instruction for Fig. 5.

1. Requirements can be either created inside or imported from outside of the platform.
2. The rich text format (RTF) report is exported from the requirement database to the RMAS for generating the verification word report and the CSV format file exported from DOORS, which describes the detailed requirements information, is for the requirement editing and modification.
3. Requirements are imported into the FSAS in CSV UTF-8 format for constructing the functional network structure and functional analysis.

4. The functional model established in the FSAS are exported into the model database in XML format for data interaction.
5. The complete and incomplete functional models can be extracted from the model database to the FSAS still in XML format.
6. The functional models connected with system requirements are exported to the PMS for risk assessment and requirement coverage calculation.
7. The test reports describing the completion of the functional flow by FSAS are exported to the requirement database.
8. The verification reports are exported in Word format and the requirements in the RMAS are synchronized to the requirements database in CSV format.
9. Trace the completion of requirements and display them in IDS.

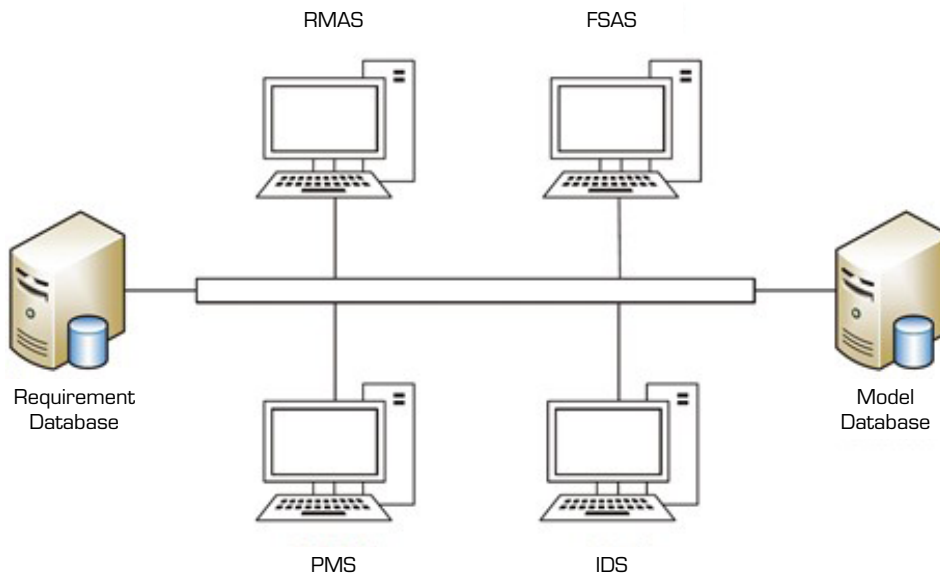


Figure 4. Integrated platform hardware architectural design.

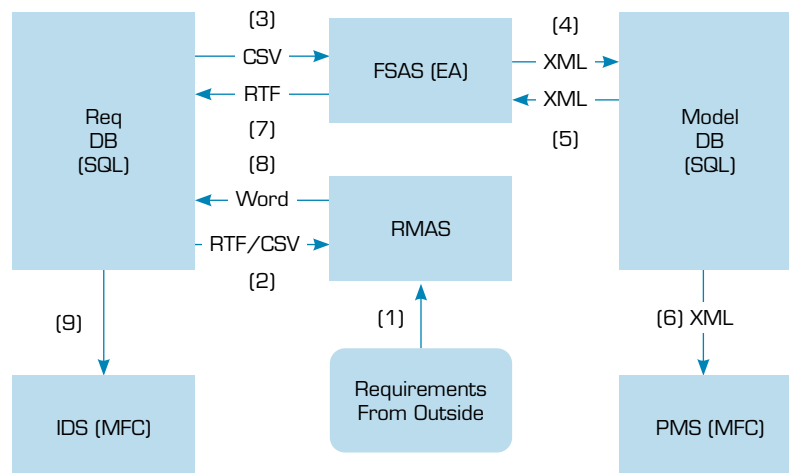


Figure 5. Information flow.

Began with RMAS, we captured stakeholder requirements and converted to system requirements. With certain modification exported to FSAS, we connected civil aircraft operation scenes with system requirements and built a requirement-oriented civil aircraft functional network structure. System modeling language was used to describe its behaviors and structures.

The completion of requirements was calculated and exported back to RMAS to generate the requirement verification report. During the whole procedure, the PMS was used to control project risk when requirement changes. Integrated display station was used to display the overall project progress and the two databases were integrated into the platform simultaneously for model management.

The existing modeling tools, such as IBM rational rhapsody, EA and MagicDraw designed for the system development are hardly able to establish the traceability among stakeholder requirements, system requirements, functional requirements, functional model and downstream design products. Thus, our platform focusses on the whole process of civil aircraft development instead of requirements or functional modeling only, which significantly improves the development efficiency of the complex system by integrating requirements, models and processes in one single design system. Through the distributed but integrated way to achieve the system architecture, we can improve the participation of all system related designers at the early stage in system design.

PLATFORM VERIFICATION OF CIVIL AIRCRAFT FLIGHT CONTROL SYSTEM

Typical civil aircraft systems include FCS, Landing Gear System (LGS), Environmental Control System (ECS), Fuel System and Fire Protection System (FPS). Modern civil aircraft FCS is the core of the whole aircraft airborne system, and it is also one of the most complex systems of the whole aircraft. Thus, here we chose FCS as the target case study to verify the aforementioned development process and platform.

Based on the previous research of civil aircraft development process and integrated platform, this section carries out the research on MBSE requirements and functional modeling for the FCS of civil aircraft. The verification process is mainly divided into the following three steps:

1. According to the airworthiness requirements of the FCS, the requirements template is defined by (DOORS eXtension Language) DXL language and loaded into DOORS for further editing.
2. Using the scene meta-model, a class diagram is created in the EA to describe the operational scenario of the FCS. Based on the function of the FCS, a use case model of the flight scenario is established.
3. According to the relationship between functions and different scenario, the corresponding requirements are associate with functional network. The ACT is used to describe the activities (dynamic) of the FCS, the BDD and the IBD are used to describe its internal organization structure. Finally, the effectiveness and correctness are verified by logical simulation.

The first step is finished in RMAS and the last two are complete in FSAS with two databases participated in the whole process. In the MBSE requirement and functional modeling of the civil aircraft FCS, the PMS is used to display the coverage of the requirements, identifying and managing the risk in the iteration process and the completion report of the project is realized on the IDS. This section applies the civil aircraft MBSE integrated platform, taking the FCS as an example to test the feasibility and efficiency of our integrated platform for further development and verification. In the meantime, the databases integrated in this platform are created in SQL Server 2008. And the tables are created in binary format to store the models and requirements into the databases in their original format of XML and CSV.

Flight control system requirements modeling

Requirement is the first to be analyzed and defined in civil aircraft design process. Requirement modeling involves two steps: requirement capture and requirement analysis (Bourque and Fairley 2014) and the requirement capture process can be generally decomposed by the following two steps:

Identify the stakeholder

Stakeholders refer to any living or nonliving bodies that are affected by the system. Currently, the domain method is to divide the relevant stakeholders into affirmative stakeholders, prospective stakeholders and potential stakeholders according to the number of stakeholders' legality, rights and urgency attributes to the company. And this classification method is easier to implement in operation. For civil aircraft development, the affirmative stakeholders are mainly aircraft developers, civil aviation companies, civil aviation bureaus, airports, government departments, pilots and various flight attendants. The prospective stakeholders are

mainly the people who have boarded the aircraft and the competitors and the potential stakeholders are mainly all the citizens (Chi et al. 2017; Bozhi and Haomin 2013).

Capture the stakeholder expectations (requirements)

After identifying the stakeholders of the project, the original needs have to be obtained from the stakeholders through various research techniques. Typical research techniques include field research, questionnaires, interviews and joint requirements planning (JRP).

In order to design the system, detailed and formal definitions of these stakeholder requirements are required to obtain aircraft-level requirements. The analysis of civil aircraft requirements is to translate the requirements and constraints put forward by stakeholders into a language that the system designers can understand. Converting these languages into the functions and constraints required by the system and then provide a detailed theoretical basis for the subsequent system design.

Users can extend the DOORS function based on DXL, not only to extend the DOORS user interface in the form of plug-ins, but also to support programmatic integration with peripheral systems. For end users, DXL can be used to develop applications that seamlessly interface with graphical user interfaces (Wei 2012). The requirements modeling process is completed in the RMAS and template is edited with DXL language and loaded into DOORS for improvement. As shown in Fig. 6, according to the airworthiness requirements, the FCS requirements template is written in DXL language and loaded into DOORS to complete the requirements modeling.

```

1 // Flight Control System (FCS)
2
3 //Copyright (c) 2018 Chrisphy
4 //See DOORS Example to Generate DXL Module
5 //Copy this file to a different location before modifying it.
6
7 /*
8 */
9 Template FCS = Template "Airworthiness Requirements" <<
10   "CCAR 28.671 Regulation" <<
11   "Each control device and operating system must be simple, smooth and accurate for its function" <>
12   "CCAR 25.672 Stabilization system and automatic and powered control system" <>
13   "CCAR 25.675 Stopper" <>
14   "CCAR 25.679 Control system ventilation lock" <>
15   "CCAR 25.681 Limit load static test" <>
16   "CCAR 25.683 Operational test" <>
17   "CCAR 25.685 Detail design of the control system" <>
18   "CCAR 25.689 Cable system" <>
19   "CCAR 25.693 Joint interface" <>
20   "CCAR 25.697 Lift and resistance devices and their operating components" <>
21   "CCAR 25.699 Lift and resistance device indicator" <>
22   "CCAR 25.701 Intersection of flaps and ailerons" <>
23   "CCAR 25.1301 Function and installation" <>
24   "CCAR 25.1309 Equipment, systems and installation" <> >>
25   "System functional requirements" <>
26   "Performance requirements to be achieved" <>
27   "Interface (physical, installation, and functional) requirements with other systems" <>
28   "Security requirements" <>
29
30 Instance FCS

```

Figure 6. Flight control system requirements template.

Flight control system functional network structure modeling

The development of functional network structure is based on the capture of operational scenario, which is the direct environment for identifying and confirming the requirements. Therefore, the analysis and development of operational scenario is an important activity of system engineering. A scenario encompasses the expected behavior of a system or system of systems, and all activities of the system are generally not adequately described by a single scenario. Thus, a behavioral model built on a single scenario only describes the lower bounds of the expected behavioral boundaries of the system. The attribute is a description of the characteristics that the system behavior must conform to, and provides the upper limit of the behavior boundary for the behavior modeling of the expected system based on attribute completion. This paper uses the scenario meta-model to establish the basic three operational scenarios for the FCS, which are flying, environment and failure. And the flying scenario is shown in Fig. 7.

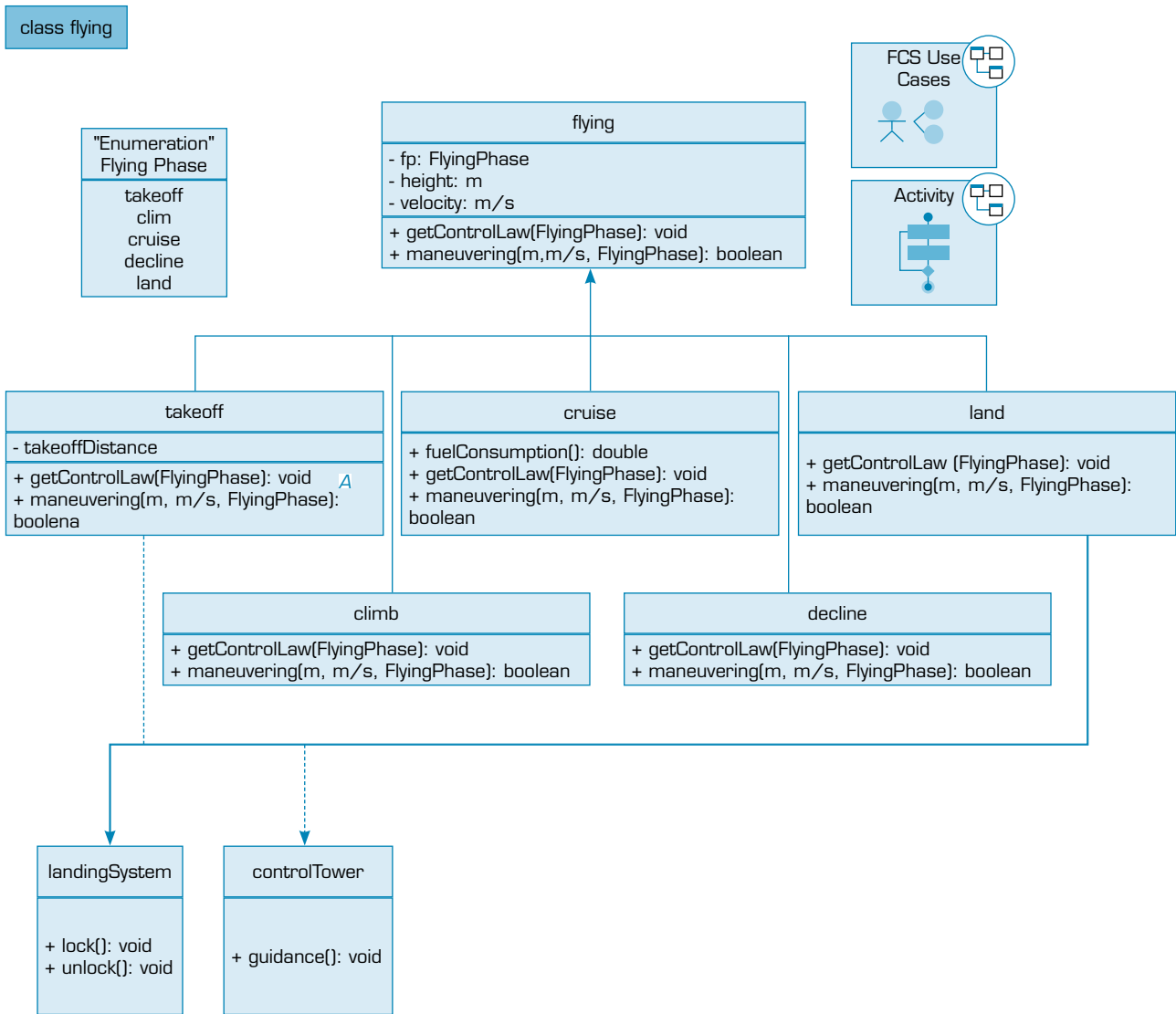


Figure 7. Flight control system flying scenario.

The FCS mainly realizes two functions of automatic flight and stability control of the aircraft. And the automatic flight can be divided into the following five subfunctions: autopilot, flight guidance, automatic throttle, automatic trim and yaw damping. The Automatic Flight Control System (AFCS) not only protects the flight attitude and safe flight in the manual flight mode, but also automatically controls the scheduled flight path under the condition which the autopilot (AP) is connected. When the AP is connected, the AFCS has full authority to control the aircraft, and the signals collected by the sensors are used to monitor the flight path and flight attitude of the aircraft (Yuve and Yan 2014). A use case model is established to describe the FCS flying scenario, as shown in Fig. 8.

Functional modeling analysis is finished in the FSAS by using commercial software EA. Civil aircraft design activities need to analyze the top and bottom requirements, determine the relationships between functions according to different scenarios. After establishing a functional model, it is necessary to confirm and supply the requirements to some extent. This paper uses SysML to complete the initial modeling of the FCS and uses the ACT to describe the system activities. The BDD and IBD are used to describe the internal organization of system. Finally, the correctness is verified by simulation.

uc FCS Use Cases

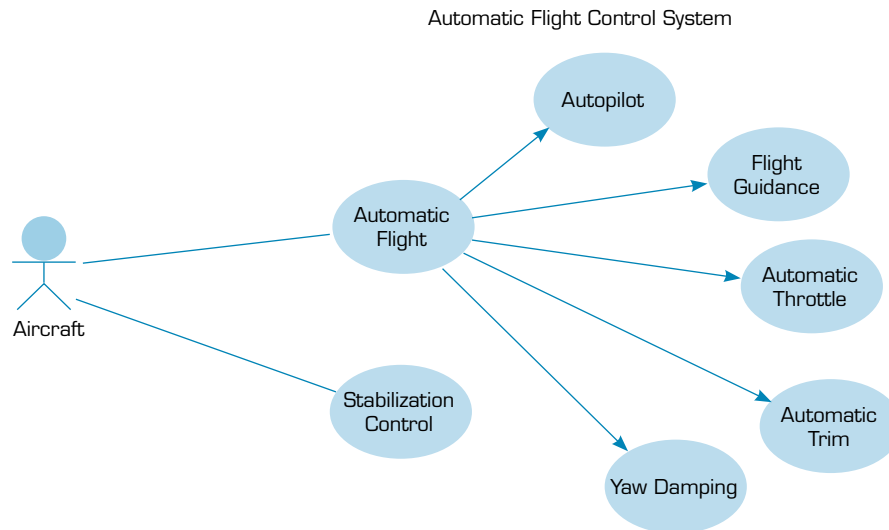


Figure 8. Flight control system flying scenario use case model.

Flight control system functional analysis modeling

Flight Control System ACT

The typical FCS consists of three parts: the rudder circuit, the stabilization circuit, and the control circuit. A steering gear, a feedback component and an amplifier build up the rudder circuit, which is used to improve the quality of the steering gear to meet the requirements of the FCS, and the rudder circuit is essentially a follow-up system that keeps the output signal synchronized with the input signal. The measuring component that measures the aircraft's flight attitude plus the rudder circuit constitutes the AP. The AP and the controlled object (aircraft) form the stabilization circuit, which is mainly responsible for track stability and aircraft flight attitude control. The control circuit consists of a stabilization circuit, a kinematics section describing the geometric relationship of the aircraft's spatial position, and a position measuring component of the center of gravity of the aircraft, which is responsible for stabilizing and controlling the trajectory of the aircraft (Sentang and Yuhua 2005).

The above-mentioned control method is currently used by most aerospace vehicles, and the FCS ACT is built upon this, as shown in Fig. 9. Note that the Fork/Join node in Fig. 9 represents synchronous execution, that is, synchronously executing the deflection rudder surface and measuring feedback information after driving the steering gear; after the aircraft motion is controlled by the rudder surface deflection the calculation of the kinematics of aircraft space position and the measurement of the track attitude are simultaneously performed. The control circuit is responsible for judging the exit condition, which is whether the current height h is equal to the expected height H , and the guards of the two exits are: $[H==h]$ and $[else]$, respectively.

The acquisition of the driving rod instruction, the aircraft attitude deviation calculation, and the angular position rate deviation calculation in Fig. 9 belong to the action, while the other rounded rectangles represent the activity. The activity in the ACT can create a child diagram, such as a Composite Structure Diagram, an ACT, a State Machine; and Action as the smallest unit of the ACT can only add Child Elements, such as Operation pin (Action Pin). The angular position rate deviation calculation operation completes the control signal calculation of the rudder circuit (achieved by the Effect of the Action unit), and the eccentric angle of the rudder surface is obtained by inverse Fourier transform, and is transmitted to the rudder surface via the amplified signal.

act [package] FCS Activity [FCS Activity]

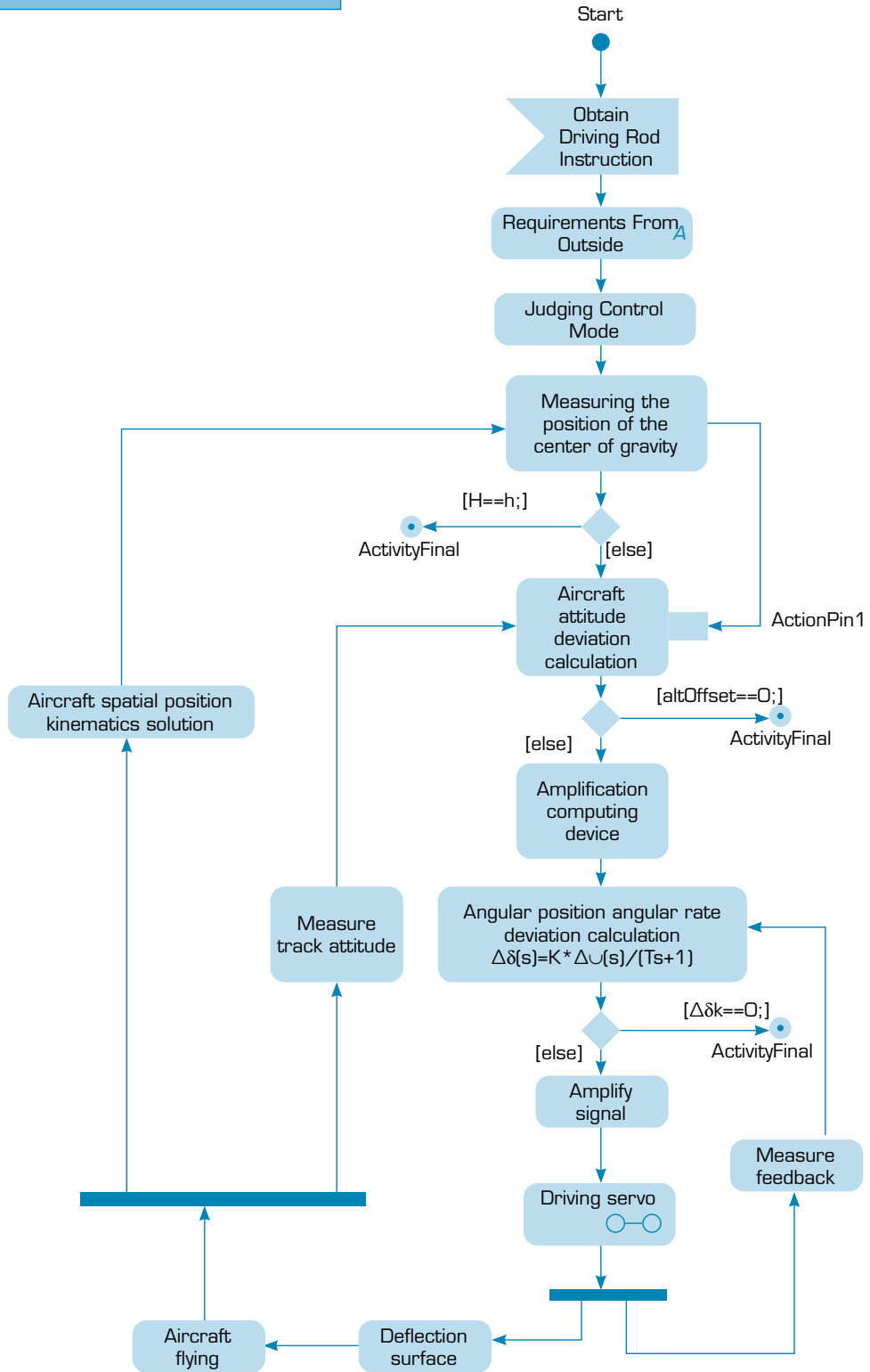


Figure 9. Flight Control System ACT.

In order to improve the reliability of the teletype control system, the redundancy technology is widely used in the design of the steering gear in the design of modern civil aircraft. For the driving servo activity in Fig. 9, a subdiagram as shown in Fig. 10 is constructed as a complex combination structure diagram of the double redundancy steering gear. The two channels of the servo are composed of a potentiometer, a power amplifier, etc., and the output of the two-channel motor shaft is collectively referred to as one by using a speed reduction mechanism. The digital servo controller also undertakes servo control and margin management. The controller can control the servo motion of the servo by integrating the output signal of the position sensor (Zhou et al. 2017).

act [activity] Driving servo [Driving Servo]

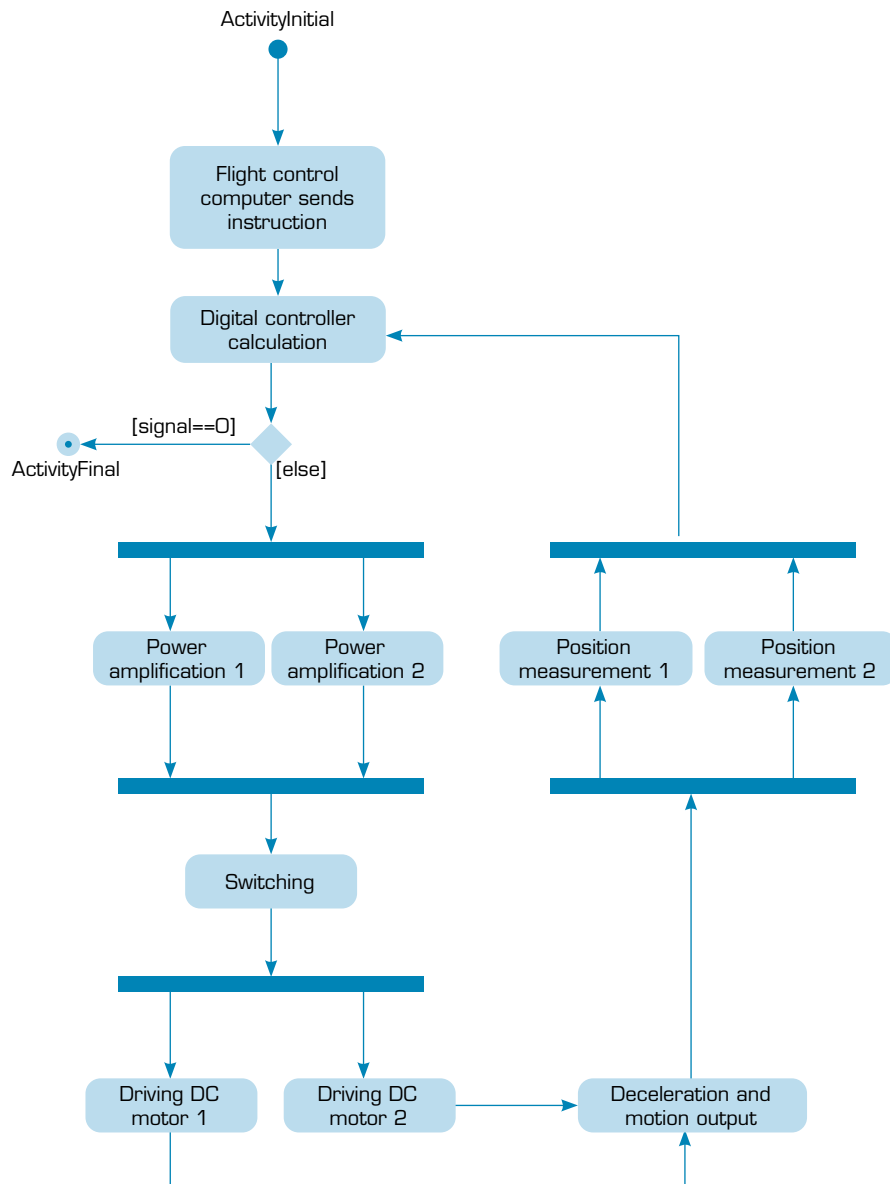


Figure 10. Double redundancy driving servo.

Flight Control System BDD

The FCS is generally composed of a sensor subsystem for measuring the flight state of the aircraft, a computer subsystem for performing data processing and transmitting control commands, a servo actuating subsystem for driving the deflection of the control surface, a control display subsystem for the human-computer interaction interface, and an internal self-test subsystem. The typical civil aircraft FCS generally includes five subsystems: a flight control computer system, a servo actuation system, a sensor system, a control display system and an internal self-test system.

The BDD of the FCS is shown in Fig. 11, and its subsystems typically have three or more design margins. The redundancy information is represented in a multiplicity manner in the BDD. As illustrated in Fig. 11, the flight control computer subsystem (fccs) and the FCS are aggregated. [+ fccs2.4] indicates that the FCS has 2 to 4 computer subsystems, and + indicates that the access rights are public.

bdd [package] BDD [FCS BDD]

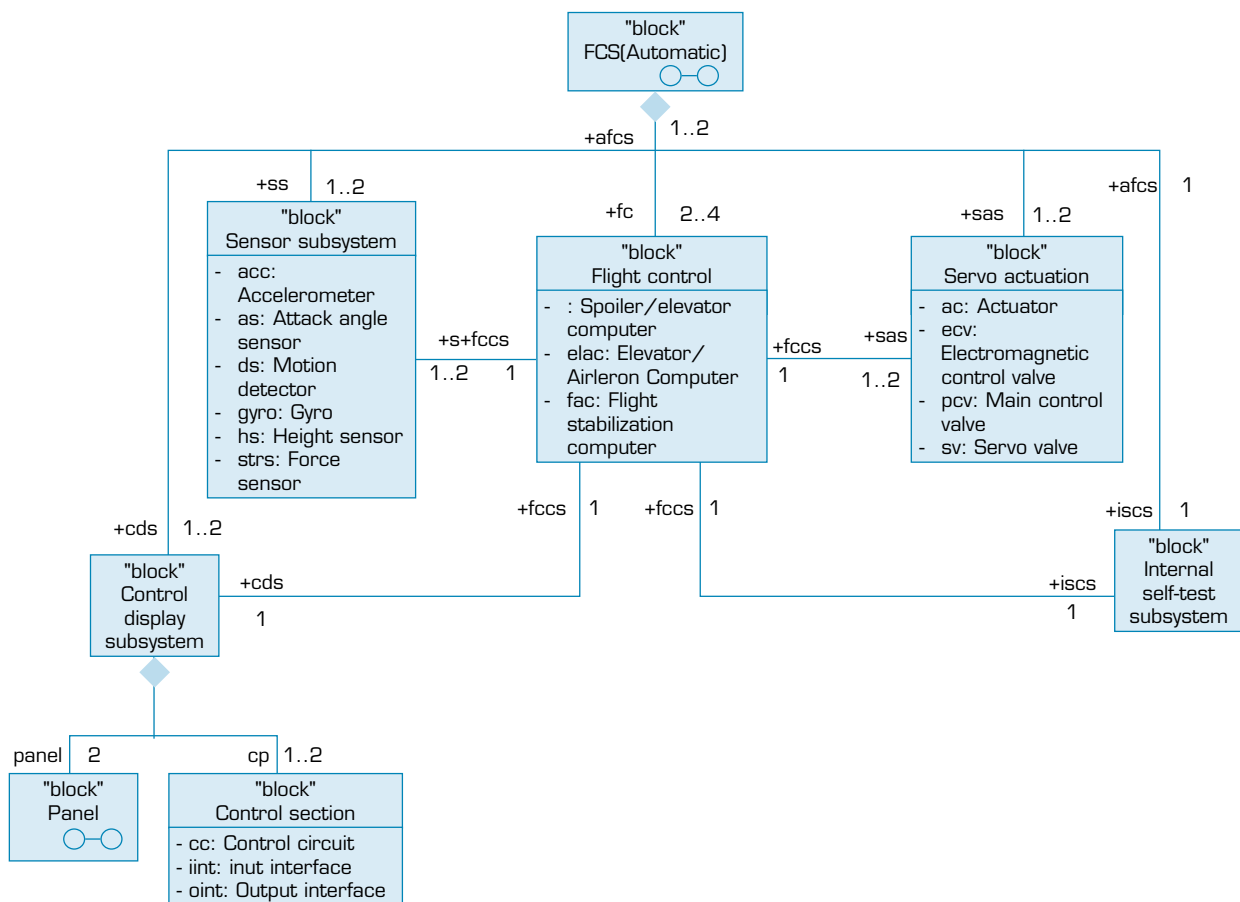


Figure 11. Flight Control System BDD.

Flight Control System IBD

The IBD describes the blocks inside the system, the nesting relationship between the blocks, the information exchange between the blocks, and the data interface between the system and the external, as shown in Fig. 12. If the FCS fails in a short period of time, a mechanical backup of the rudder and horizontal stabilizer is used to complete the safe flight and landing mission of the aircraft. The computer automatically assigns system tasks, and when one computer processes the task, the other computers are in standby. When the working computer is damaged, another replacement work is arranged in accordance with a pre-set priority order, and the switching process does not cause a sudden deflection of the steering surface.

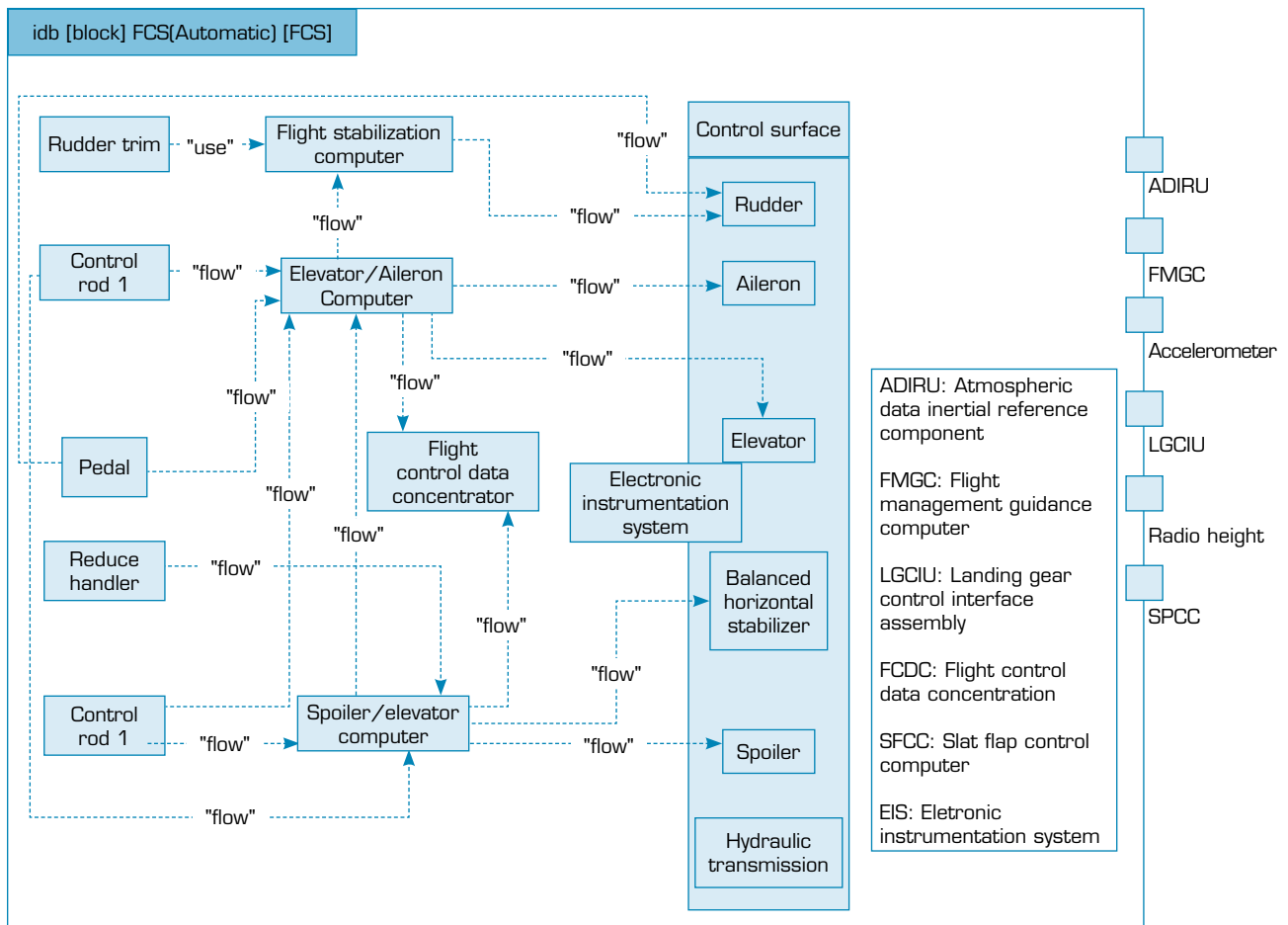


Figure 12. Flight Control System IBD.

Process Management and Integrated Display

The PMS platform is based on C++ language. It controls project activities, including the following three nested parallel tasks.

1. Overall progress of the project and traceability of requirements;
2. Project pending issues include problem descriptions, applying metrics, and handling project issues;
3. Propose solutions to project problems and carry out risk assessment; conduct daily meeting, coordinate design and management team to reach short-term task consensus.

The PMS is used to store, mark and classify the FCS model established by the MBSE process in the model database, and display the coverage of the requirements in a certain scenario, achieving the traceability from functions to requirements. Risk identification and management of the project during each iteration, prioritizing high-risk activities.

The IDS is mainly responsible for displaying the engineering progress of the MBSE integrated platform, including presenting the work content of other platforms, querying the person in charge of a certain requirement and progress, as well as visualization of engineering data.

CONCLUSION

Based on the V model of traditional system engineering, this paper designed the requirement-based civil aircraft MBSE Research and Development process and built an integrated platform with six work stations, including all the activities in the

development process of complex system. The functions of the platform and all six stations as well as the information flow have been fully discussed. Based on our platform, the requirement modeling, functional modeling and process management were completed for the civil aircraft flight control system. The test results showed the feasibility and efficiency of our platform in complex system development. The comparison with existing modeling tools and integrated platforms verified the advantages of our platform.

AUTHORS' CONTRIBUTION

Conceptualization: Bi W and Zhang A; **Methodology:** Bi W and Wang W; **Writing – Original Draft:** Wang W and Wang Y; **Writing – Review and Editing:** Bi W and Wang W; **Funding Acquisition:** Bi W and Zhang A; **Resources:** Bi W and Zhang A; **Supervision:** Bi W.

DATA AVAILABILITY STATEMENT

All dataset was generated and analyzed in the current study.

FUNDING

National Natural Science Foundation of China

<https://doi.org/10.13039/501100001809>

Grant No. 61903305 and 62073267

Aeronautical Science Foundation of China

Ministry of Industry and Information Technology of the Peoples Republic of China

<https://doi.org/10.13039/501100006579>

Grant No. MJ-2016-F-02

ACKNOWLEDGEMENTS

Not applicable.

REFERENCES

Andersson H (2009) Aircraft Systems Modeling: Model Based Systems Engineering in Avionics Design and Aircraft Simulation. Linköping: Linköping University Electronic Press.

Anwar MW, Rashid M, Azam F, Kashif M (2017) Model-based design verification for embedded systems through SVOCL: an OCL extension for SystemVerilog. Des Autom Embed Syst 21(1):1-36. <https://doi.org/10.1007/s10617-017-9182-z>

Anwar MW, Rashid M, Azam F, Kashif M, Butt WH (2019) A model-driven framework for design and verification of embedded systems through SystemVerilog. Des Autom Embed Syst 23(3):179-223. <https://doi.org/10.1007/s10617-019-09229-y>

- Anwar MW, Rashid M, Azam F, Naeem A, Kashif M, Butt WH (2020) A Unified Model-Based Framework for the Simplified Execution of Static and Dynamic Assertion-Based Verification. *IEEE Access* 8:104407-104431. <https://doi.org/10.1109/ACCESS.2020.2999544>
- Bourque P, Fairley RE (2014) *Guide to the Software Engineering Body of Knowledge Version 3.0*. Washington: IEEE Computer Society Press.
- Bozhi G, Haomin L (2013) Requirement Management in Large Civil Aircraft Design. *Civil Aircraft Design and Research* 2013(4):1. <https://doi.org/10.19416/j.cnki.1674-9804.2013.04.002>
- Chi Y, Shan D, Zhenxin Y, He P (2017) Application of Model-Based simulation design in a flight control system. *J Simul* 29(10). <https://doi.org/10.16182/j.issn1004731x.joss.201710041>
- Cloutier R, Sauser B, Bone M, Taylor A (2014) Transitioning Systems Thinking to Model-Based Systems Engineering: Systemigrams to SysML Models. *IEEE Trans Syst Man Cybern Syst* 45(4):662-674. <https://doi.org/10.1109/TSMC.2014.2379657>
- Cloutier R, Sauser B, Bone M, Taylor A (2015) Transitioning Systems Thinking to Model-Based Systems Engineering: Systemigrams to SysML Models. *IEEE Trans Syst Man Cybern Syst* 45(4):662-674. <https://doi.org/10.1109/TSMC.2014.2379657>
- Dickerson CE, Mavris D (2013) A Brief History of Models and Model Based Systems Engineering and the Case for Relational Orientation. *IEEE Syst J* 7(4):581-592. <https://doi.org/10.1109/JSYST.2013.2253034>
- Estefan JA (2007) Survey of Model-Based Systems Engineering (MBSE) Methodologies. *IncoSE MBSE Focus Group* 25(8):1-70. [accessed Mar 25 2021]. https://www.omg.sysml.org/MBSE_Methodology_Survey_RevB.pdf
- Hoffmann H-P (2011) *Systems engineering best practices with the rational solution for systems and software engineering. Deskbook Release 3.1.2. Model-Based Systems Engineering with Rational Rhapsody and Rational Harmony for Systems Engineering*. IBM Software Group. [accessed Mar 25 2021]. <https://silo.tips/download/solution-for-systems-and-software-engineering>
- Holt J, Perry S, Payne R, Bryans J, Hallerstede S, Hansen FO (2015) A Model-Based Approach for Requirements Engineering for Systems of Systems. *IEEE Syst J* 9(1):252-262. <https://doi.org/10.1109/JSYST.2014.2312051>
- Jing Z, Hui Y, Yahui G, Taiké Y (2016) Summary of Model Based System Engineering. *Aero Engine* 42(4):12-16. <https://doi.org/10.13477/j.cnki.aeroengine.2016.04.003>
- Madni AM, Sievers M (2018) Model-based systems engineering: Motivation, current status, and research opportunities. *Syst. Eng* 21(3):172-190. <https://doi.org/10.1002/sys.21438>
- Mordecai Y, Orhof, O, Dori D (2018) Model-Based Interoperability Engineering in Systems-of-Systems and Civil Aviation. *IEEE Trans Syst Man Cybern Syst* 48(4):637-648. <https://doi.org/10.1109/TSMC.2016.2602543>
- Piaszczyk C (2011) Model Based Systems Engineering with Department of Defense Architectural Framework. *Syst Eng* 14(3):305-326. <https://doi.org/10.1002/sys.20180>
- Ramos AL, Ferreira JV, Barceló J (2011). *Model-Based Systems Engineering: An Emerging Approach for Modern Systems*. *IEEE Trans Syst Man Cybern Syst (Applications and Reviews)* 42(1):101-111. <https://doi.org/10.1109/TSMCC.2011.2106495>
- Sentang W, Yuhua F (2005) *Flight Control System*. Beijing: Beijing University of Aeronautics and Astronautics Press.
- Topper JS, Horner NC (2013) *Model-Based Systems Engineering in Support of Complex Systems Development*. Johns Hopkins APL Technical Digest 32(1):419-432.
- Wei X (2012) A DOORS-based software requirements development process optimization method. *Journal of Computer Applications and Software* 29(9):175-177. <https://doi.org/10.3969/j.issn.1000-386x.2012.09.047>

Yuye C, Yan M (2014) Aircraft System (Twelfth Five-Year Plan for Aviation Majors in General Colleges and Universities). Beijing: National Defence Industry Press.

Zhang JJ, Liu Z, Li F, Dong DY, Meng H, Liu HT, Chai X (2018) Employing Model-Based Systems Engineering (MBSE) on a Civil Aircraft Research Project: A Case Study. In: Zhang X, editor. The Proceedings of the 2018 Asia-Pacific International Symposium on Aerospace Technology. APISAT 2018. Lecture Notes in Electrical Engineering, Vol 459. Singapore: Springer. https://doi.org/10.1007/978-981-13-3305-7_175

Zhou Y, Chen L, Wang J, Zhang C (2017). The controller design for permanent magnet motor with complete electrical double redundancy structure. In: 2017 Prognostics and System Health Management Conference (PHM-Harbin), Harbin, 2017, pp. 1-5. <https://doi.org/10.1109/PHM.2017.8079138>