Towards an effective interoperability of models within the 'Systems Engineering' applied to aeronautics

Eugenio Brusa, Ambra Calà, Sergio Chiesa, Francesco De Vita, Davide Ferretto

Dept. Mechanical and Aerospace Engineering, Politecnico di Torino, Italy

Corso Duca degli Abruzzi, 24 – 10129 Torino, ITALY

email: eugenio.brusa@polito.it; ambra.cala@polito.it, sergio-chiesa@polito.it, francesco.devita@polito.it, davide.ferretto@polito.it

Abstract. Aeronautics is a typical field of application of the Systems Engineering, since aircraft includes many on–board equipments. The Systems Engineering provides some suitable tool for an effective description of their functional behavior but a detailed design needs a quantitative investigation. This task is performed by resorting to modeling techniques, which define all the equations required to predict the dynamic behavior in operation. Physical phenomena are described by numerical models, which nowadays have to be connected to the tools of the Systems Engineering to proceed with a really integrated product life management. This task is a fascinating feature of the so–called 'inter–operability', which can be implemented among methods, models and numerical tools. A test case is herein shown and concerns the modeling of a de–icing system for a regional turboprop. A brief description of the modeling activity is proposed, then tools of the System Engineering are applied to perform a review of requirements. Limits of functional models are explored as well as some weak information about functions and requirements in the physical models is detected. Region of inter–operability of the two modeling environments is consequently defined. The available methodologies for interoperating the design tools are discussed, by resorting to the tasks of the ARTEMIS–CRYSTAL project.

Introduction

Accumulation of ice on wings, nacelles, tail and instruments is definitely one of the most dangerous risks for the aircraft flight at different altitudes. Ice accretion is a very heterogeneous phenomenon and depends on several environmental parameters as well as upon some properties of the aircraft. Very often ice is a consequence of a concentration of super–cooled water within clouds, because it is liquid even below 0 °C. Water droplets hit the aircraft and often stick upon the surfaces and freeze very fast or instantly, thus causing a reduction of lift and of the angle of attack which might be considered a limit for the stall. They increase drag and weight, by causing some adverse effects on other control surfaces and induce a flow disruption. Therefore several anti–icing systems are used to avoid the ice accumulation or alternately some de–icing systems are applied to reduce the ice accretion. Among the technologies implemented, very well-known are the electro–thermal system based on the heat produced by resistors and the aero–thermal one which exploits a hot air stream directly coming from the engines. They are used both either as an anti–icing or de–icing system. Nevertheless, de–icing systems are also designed to apply a suitable action to break the ice layers covering the aerodynamic surfaces, by means of boots or actuators. To develop herein a test case the well-known Goodrich's de–icing system will be analyzed. In this case some boots distributed over the most exposed surfaces of the aircraft are either periodically or on demand inflated to break the ice.

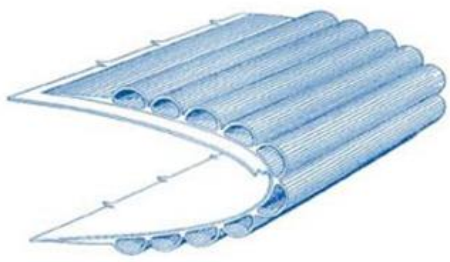

Figure 1. Sketch of airfoil equipped with the Goodrich de–icing system

The Goodrich de–icing system and control

The Goodrich system is often selected for the turboprop aircrafts, because it implies low power consumption. Some inflatable boots are located on the surface potentially affected by ice accumulation. They are activated in sequence by means of compressed air coming from the aircraft engines. A critical issue of design concerns the control strategy applied to the inflation of boots. If they are inflated too early the ice might be so thin that it will not break, being somehow elastic, when boots are inflated too late, they could not break thick ice layers. Ice accretion rate is specified by some International Standards and Regulations. Design activity usually starts by listing the high level requirements of the de–icing system. Principal functions are defined by the so–called functional requirements, while some operational requirements are written by analyzing the flight profile and mission. Safety requirements are imposed by the Regulations, as the CS–25 Appendix C. Very often it looks difficult defining any physical requirement which could be compatible with some selected components, because the architecture of the whole system is still under definition. Industrial practice suggests of specifying at this stage some general constraints for the system characteristics and some requirements for maintenance and installation. Defining a preliminary set of performance requirements is even recommended.
To proceed straightforward with the requirements specification a functional model is required. It allows investigating the system behavior in terms of functions and performing a trade–off among the proposed architectures. Only a detailed physical model focused on the quantitative performance of the system components can help the designer in the dimensioning activity. This process is fundamental to define, allocate, trace and verify each requirement within the product cycle development, especially when it is updated during the sequence of actions foreseen by the workflow.

The role of the functional model

This system can be effectively modeled by following the Model–Based Systems Engineering approach within the IBM Rational Rhapsody®. Different tools lead to define the system structure and behavior, starting from the analysis of the customer needs. System level is analyzed as first, then components and parts are specified and either dimensioned or selected among some available commercial products.

Customer needs are formalized as a list of requirements by resorting to the IBM Rational Doors®, then they are imported into the Rhapsody® model. Requirements must be analyzed, clarified and refined during all the modeling process, eventually by structuring their description into main and derived requirements. System will fit each requirement through a specific function operated by some component. The SysML language allows visualizing those components and their functions by means of its typical diagrams.

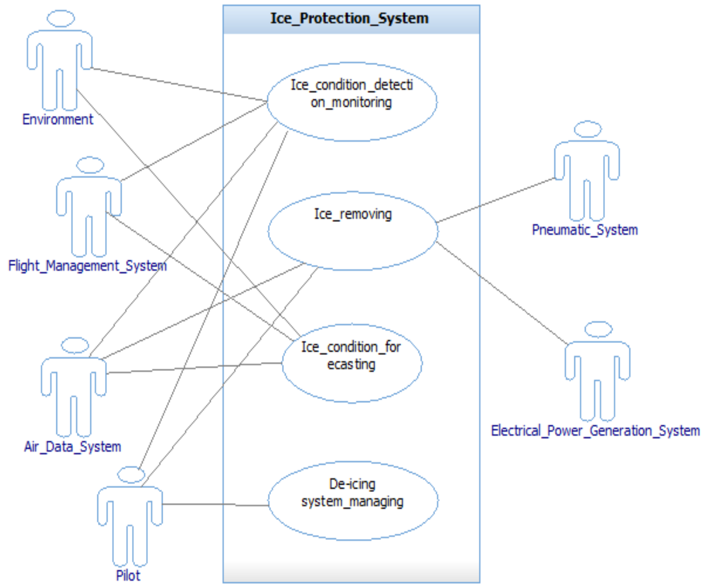

Figure 2. Use cases of the Goodrich de–icing system

Once the main specifications are defined, a Use Case Diagram is drawn to analyze the missions of the system, as in Fig.2. The context in which the system operates is depicted by defining its neighboroughs, the functions provided and the actors involved who directly interact with the system.
In the test case it can be easily realized that four main missions are foreseen and they correspond to the use cases above described and functions to be implemented are all related to those uses. A management of the de–icing operation, through a continuous monitoring, a suitable ice removing and a forecasting capability to prevent the risk of stall are required. In particular, the actors (namely stakeholders) strongly interact to each other. The de–icing system must detect and monitor the icing condition in which the aircraft operates. The air data system picks up the inputs from the environment and from the ice condition forecasting, while the flight management system evaluates this inputs, providing real time information to the pilot for those activities. The pilot acts on the system through a control panel, as soon as he receives the inputs collected by the other systems. Once the ice is detected the ice removing mission is performed by the pneumatic system which inflates the boots.
The Use Case diagram describes the functions performed by the system, while interacting with the stakeholders. The system behavior is predicted by the State Machine diagram, which provides a functional analysis of each use case. This is a qualitative description of the configurations exploited to perform the required functions. It does not need that the system architecture is defined neither that components are yet selected. The State Machine diagram shown in Fig.3 represents how the ice removing mission is performed in terms of transitions from state to state,

being each one triggered by a known event. In case of manual mode state, when the de–icing system is switched on, the pilot can manually activate the control panel the protection system of both the tail and the wings through.

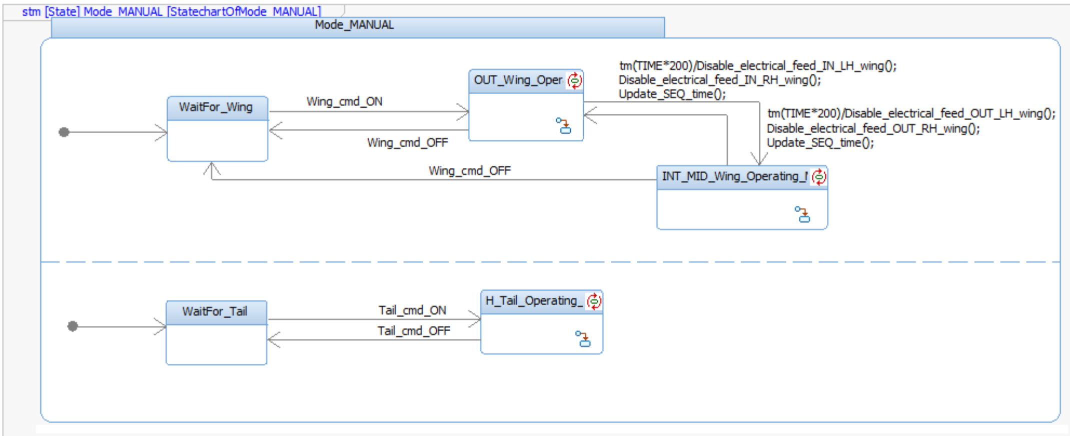

Figure 3. State Machine diagram – Manual Mode

This diagram is powerful when the system is just preliminarily defined, because it allows investigating all the states which have to be considered and how the functions allow the transition between two states. However it could suggest the need for a function or a component (sensor, actuator, connection …) but it is not yet sufficient for a detailed dimensioning of the physical system. This information is somehow contained and utilized in other modeling tool usually applied for the physical modeling, as the Simulink® or the Modelica®.

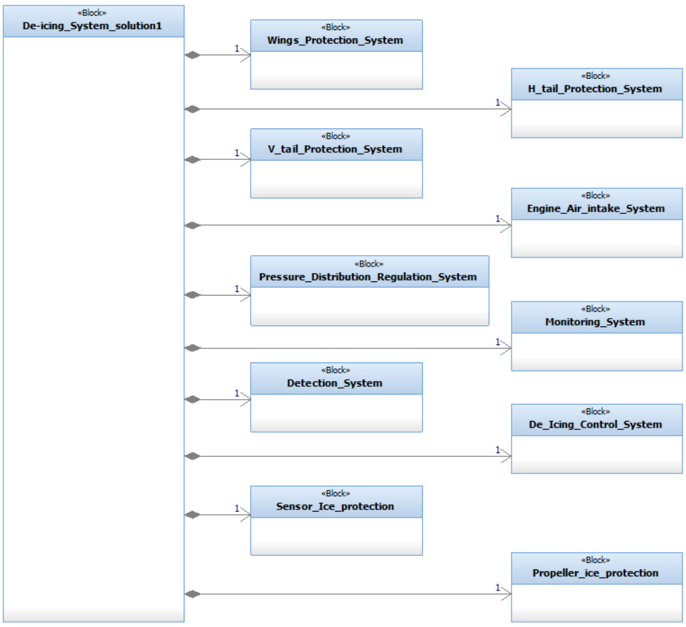

Figure 4. Block Definition Diagram at system level

It can be remarked that functional analysis of systems is completed by resorting to the Activity diagrams, which analyze step by step the actions performed, and also by the Sequence diagrams which describe the sequence of messages exchanged between stakeholders and the system itself as a function of time. To focus on the interoperability the State Machine diagram was modeled for an executable simulation

of the system, within the constraints above evidenced that simulation can only describe the state at which the system is set up for a given boundary condition. A link between the states and the quantitative values of the system parameters is a matter of the interoperability among tools.
Moreover, once the operations of the de–icing system are defined a trade–off among the proposed architectures has to be performed. This task is done by resorting to the structural diagrams, which include blocks, subsystems, components and parts where requirements can be allocated. Identifying the blocks allows depicting one or more scenarios where the system performs its functions. The Block Definition diagram shows both the composition and the classification of structural elements. This representation is useful to completely define the system components and their features. Interoperability may allow integrating the SysML Block diagrams and the numerical simulator for a complete analysis.
It is remarkable that in Fig.4 each block represents an element of the main system, as a part of the protection system, the distribution valves and the control system. Nevertheless, very often a single block corresponds to a subsystem being itself composed by several parts. Therefore to catch completely the interactions among those parts and what kind of information they exchange it is usually drawn the Internal Block diagram (Fig.5). It defines the internal structure of the details of the de–icing system. For the test case it is necessary considering several Internal Block diagrams, to compare functional and physical models, respectively, for instance as they appear within the Simulink® simulation.

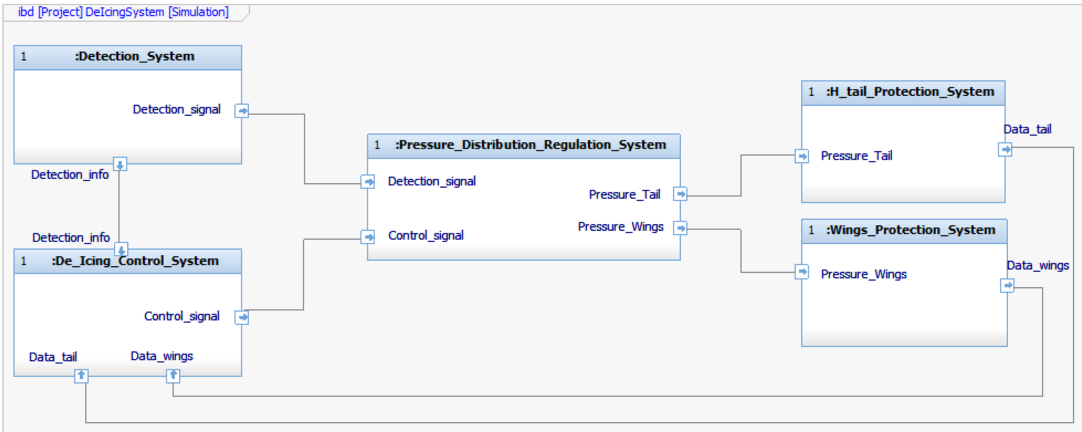

Figure 5. Internal Block Diagram

Diagram shown in Fig.5 represents an executable scenario. It contains the whole control block. All the elements are connected to each other in the desired configuration through some flow ports and connectors. The behavior of each part is described in the State Machine diagrams. Flow ports are used to explicit the messages exchanged between parts. Flow direction is defined and it is required to run the simulation.

The role of the physical model

The functional model, as described in the previous section, can provide a detailed description of the system architecture and its behavior. However this approach is limited to some qualitative aspects and does not predict the physical ones. No mathematical model is included. Therefore the functional model is unable to perform numerical simulations, although this task is required for a full performance analysis.

This weakness motivates the resorting to the numerical modeling as in Simulink®. As Fig.6 shows a Simulink® model of the Goodrich system includes a logical flow behind the mathematical model which describes the system behavior. Each action and its effects in time are predicted by solving the equations enclosed into the blocks which appear in the depicted model. For each block inputs and outputs are numerical values of some design parameters. The tool allows exploring also the hierarchy of the operations performed by the blocks, eventually linked together to form a macro, or a subsystem. This approach allows a detailed analysis of complex systems. It looks effective because they are modeled as a tree of blocks.

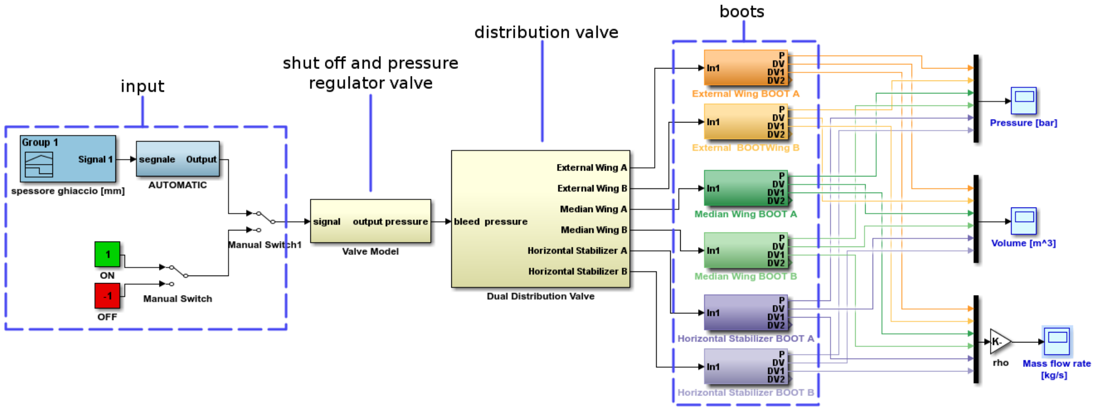

Figure 6. Physical model

A key issue for simulation is the data management and the exchange of information among blocks. Input variables can be stored in an external file, to be loaded as soon as the code starts up. However inputs and outputs can be matter of exchange between the Simulink® and another tool eventually co–simulated. Right now this is a well-known option of engineering, for instance when the multi–body dynamics is analyzed simultaneously with a control system operation. Since the appearing of the Systems Engineering a challenging goal looked co–simulating functional models and Simulink®. This task requires that data management somehow is performed across the numerical and the functional modeling environments. In the test case the first group of blocks Fig.6 defines the input section which allows to simulate either an automatic or a manual mode operation. Performing the simulation through the automatic mode is possible to predict the ice accretion on the ice–detector, while in the manual mode the de–icing system is activated through a manual switch. If the Simulink® model could be operated in connection with the Parametric diagram of the system, according to Systems Engineering, the effects of manual operations, triggered through the panel of the functional simulator in the Rhapsody® environment, could be quantitatively predicted and visualized.
Outputs of the first section in Fig.6 are sent to the valve models, then results of such block are inputs for the distribution valve, which implements the logic of the inflation of boots. In the last set of blocks the second order model depicted in Fig.7 is implemented for each component. It can be appreciated that in this physical model outputs of the simulation are numerical values of air pressure, of volume of boots chamber and of flow rate per cycle. Those results allow understanding quantitatively the critical levels of the design parameters and lead to a detailed design activity.

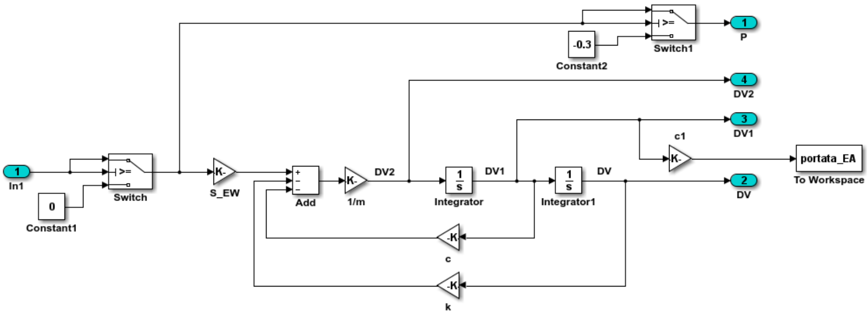

Figure 7. Second order model for the numerical simulation of dynamics

The challenge of interoperability

The State Machine diagram built within Rhapsody® and the Simulink® model represent two types of executable simulation. Languages and purposes are different. Simulators of systems are built up by implementing the whole model including all its parts within a certain modeling environment, through a specific tool. This allows handling a unique or 'homogeneous' environment. However, in many industrial applications it happens that systems exhibit a certain complexity associated to the number of composing subsystems and of functions, typically obeying to physical laws belonging to various engineering fields like in mechatronics. This complexity leads to the practical impossibility of simulating the system in a unique environment and to the need of performing several analyses by means of different approaches, supported by different tools. In the test case the functional behavior is easily analyzed through the SysML, while the dynamic performance can be investigated only through a physical model. The so–called heterogeneous simulation environments are therefore growing up as a key feature of the Model–Based Systems Engineering. Nevertheless, designer needs to integrate the different environments as soon as the design synthesis is required. This action can be performed if some connectors between two specific tools are developed and applied. This is the goal of the interoperability standards. In the test case interoperating the Rhapsody® and the Simulink® environments it is required to perform a complete simulation of the de–icing system. Some key issues of design cannot be investigated only by the State Machine diagram. For instance a critical task within the simulator is the prediction of the ice accretion. Setting up a suitable time period for inflating the boots in case of automatic de–icing is rather difficult if this action is only based on the transitions between states. It is required a physical modeling of the system and of the ice behavior, described into a set of equations, which could be solved as a function of time, to test the effectiveness of control. To perform this activity the model should be derived from the Block diagrams of the SysML in such a way that any change in requirements could be updated in the functional model and transferred to the physical one, almost automatically.

Dedicated connectors are currently and widely developed to provide some customized interface to allow the interoperation between two simulation environments. Unfortunately, they are applicable case by case only to the specific tools used. Looking for an interoperability standard is currently a hot topic of the research activity. It could allow a deeper integration among the tools of the Model–Based Systems Engineering. One of the most promising standards currently available and implemented by some tool developers is the 'Functional Mock-up Interface' (FMI). It is an open standard that enables the integration of different models, being using different languages and semantics.

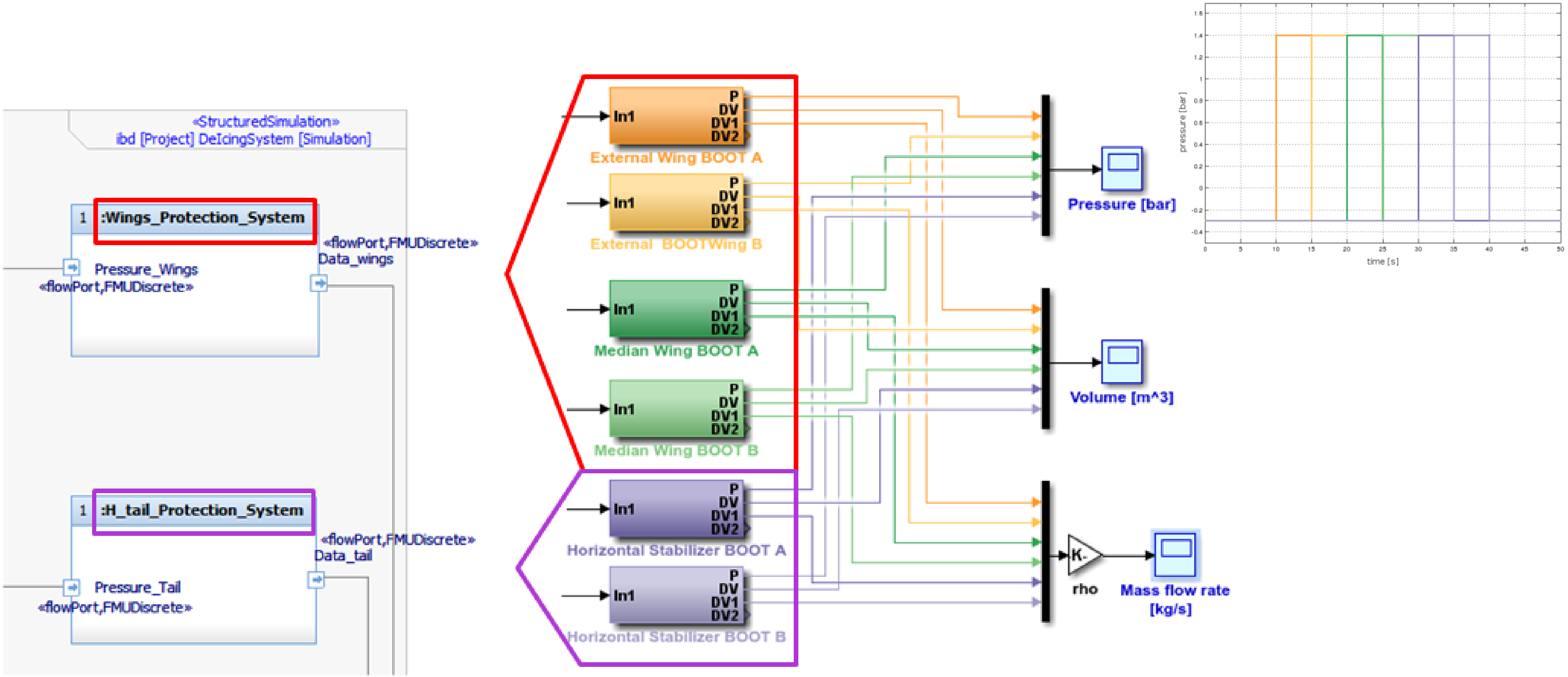

Figure 8. Interoperation of tools with FMI Stereotypes

The specific interface created is referred to as 'Functional Mock-up Unit' (FMU) and is aimed at allowing the communication exchange between tools. Two approaches are foreseen and supported by the FMI. A 'model exchange mode' directly utilizes the equations solver of an hosting numerical tool while the 'co–simulation' mode fully exploits the skills of interoperability, because in this case an embedded solver is used and the hosting tools have only to assure the synchronization of all the data handled by the heterogeneous simulation environment. The state of arts does provide an assessed connector for the interoperation between the Rhapsody® and Simulink®, nevertheless the FMI 2.0 was selected as a standard reference to test at least the exporting capabilities and to verify the consistence of FMI stereotypes, when executing the same simulation in both the above mentioned tools. The real goal will be co–simulating the tools as soon as the connectors will be available.
Application of the FMI to the test case is fairly linear. The first step is characterizing the system composition and the simulation parameters defined in Rhapsody® through the SysML language by using the FMI. This is meant to have a common structure. System composition is obtained from the Internal Block Diagram, where interactions between the blocks are specified by the flow ports and characterized by means of the FMI stereotypes. In particular, each block is marked with the "FMU Export" stereotype and is described by an associated State Machine Diagram, which shows his functional behavior. All the relevant attributes of blocks are marked with "FMU Parameter" stereotype whilst "FMU Ignore" is assigned to the attributes and ports which are not considered. The above procedure assures that the system is composed by a structured block in which its subsystems have a well-defined behavior and a common semantics, that can be used to compile the FMI project and for exporting. The stereotype "Structured Simulation" is in fact applied to the parts of the model that contain lower levels blocks, as it usually happens in Simulink®. For the reasons exposed above, the Simulink® model is re–organized and synchronized with the FMI structure as well as the Rhapsody® model. The generated FMU file is indicated in the Rhapsody® browser as "controlled file". This strategy leads to perform the simulation directly within Rhapsody® once that the external model is imported. In this case, time management has to be taken into account because even if the FMI standard uses a double precision floating point value (seconds), Rhapsody® represents it as an integer in unit of milliseconds. This could lead to unacceptable behaviors, especially for state-based dynamics, and variables trend has to be checked. The final simulation environment will be composed by different SysML blocks connected to each other with flow ports representing the variables in input or output. These variables are managed inside each block, which is a sort of "black box" that includes the behavior specified in the original external model. This means that the FMU is only a container and it does not provide nor connection recommendations neither the structure of the scenario. As a matter of fact, unlike the majority of the numerical simulation tools, FMI standard has not yet a library and the set-up of the simulation scenario is a responsibility of the engineer.

Conclusion and future works

Defining the architecture and requirements of a complex industrial product needs a multidisciplinary and integrated approach suitable to involve all the necessary technical competences. Functional and numerical issues of design are investigated, as in case of the Goodrich de–icing system. Diagrams of the Systems Engineering allow defining the interactions between the system and all the stakeholders, uses, activities and sequences in operation, thus leading to straight identifying of blocks and of requirements, which are allocated to functions and components. For an effective dimensioning of the whole system a parametric simulation of states in functional models, like in the Rhapsody®, are often insufficient without a physical model aimed at predicting the system evolution in time domain as in the Simulink® code. To enhance the integration of those design tools a reliable communication among functional and physical models has to be developed such as the FMI interface. It could create a permanent link between the SysML and the numerical tools, thus allowing to perform a complete iterative design loop suitable to assess both the requirements and the design parameters. In the Goodrich de–icing system, for instance, the above described inter–operation of tools is useful to define the strategy of inflation of boots, to be compatible with the real ice accretion phenomenon, predicted by some physical model. Future works will focus on performing a complete hybrid simulation, setting up an heterogeneous environment in Rhapsody® in which both the system activation logic, based on SysML state machines, and the physical behavior from Simulink® will be present. The interest for the creation of a reusable library within the FMI standard applied to aeronautics is also expressed.

References

Chiesa S., 1995. *Impianti di bordo per aeromobili: impianti pneumatici, condizionamento, anti-ghiaccio e APU*. CLUT, Torino.

———. 2006. *Investigations of performance of pneumatic deicing boots, surface ice detectors and scaling of intercycle ice.* DOT/FAA/AR-06/48, Washington DC.

Sage, P., Rouse, W., 2008. *Handbook of Systems Engineering and Management.* John Wiley and Sons, New York.

———. 2011. *Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities.* Version 3.2.1 Revised by M. Krueger, D. Walden, and R. D. Hamelin. San Diego, CA (US): INCOSE.

———. 2012. Requirements WG-INCOSE, *Guide for Writing Requirements*, v1.0, INCOSE

2010-006-01, April 2012

———. 2014. EASA CS-25/Amendment 15

Feldman Y. A., Greenberg L., Palachi E., 2014. *Simulating Rhapsody SysML Blocks in Hybrid Models with FMI.* Proceedings of the 10th International Modelica Conference, Lund, Sweden.

Bertsch C., Ahle E., Schulmeister U., 2014. *The Functional Mockup Interface seen from an industrial perspective.* Proceedings of the 10th International Modelica Conference, Lund, Sweden.

Garro A., Groß J., Riestenpatt Gen. Richter M., and Tundi A., 2014. *Reliability Analysis of an Attitude Determination and Control System (ADCS) through the RAMSAS method.* Journal of Computational Science, Amsterdam, The Netherlands.