POLITECNICO DI TORINO
Repository ISTITUZIONALE

Industrial implementation of the MBSE applied to design of an aircraft landing gear: current issues, solutions and tools

Original
Industrial implementation of the MBSE applied to design of an aircraft landing gear: current issues, solutions and tools / Brusa, E., Dagna, A., Delprete, C.. - ELETTRONICO. - (2025), pp. 1-8. (2025 IEEE International Symposium on Systems Engineering (ISSE) Palaiseau (FRA) 28–30 Ottobre 2025) [10.1109/ISSE65546.2025.11370106].

Availability:
This version is available at: 11583/3006218 since: 2025-12-29T16:45:34Z

Publisher:
IEEE

Published
DOI:10.1109/ISSE65546.2025.11370106

Terms of use:

This article is made available under terms and conditions as specified in the corresponding bibliographic description in the repository

Publisher copyright
IEEE postprint/Author's Accepted Manuscript

©2025 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collecting works, for resale or lists, or reuse of any copyrighted component of this work in other works.

(Article begins on next page)

14 November 2026

Industrial Implementation of the MBSE Applied to Design of an Aircraft Landing Gear: Current Issues, Solutions and Tools

Eugenio Brusa
Dept. of Mechanical and Aerospace Engineering
Politecnico di Torino
Torino, Italy
eugenio.brusa@polito.it
ORCID: 0000-0001-7478-0650

Cristiana Delprete
Dept. of Mechanical and Aerospace Engineering
Politecnico di Torino
Torino, Italy
cristiana.delprete@polito.it
ORCID: 0000-0002-0220-4707

Alberto Dagna
Dept. of Mechanical and Aerospace Engineering
Politecnico di Torino
Torino, Italy
alberto.dagna@polito.it
ORCID: 0000-0003-0407-6283

***Abstract*—Design of safety critical systems currently faces an increasing complexity of layout and requires a full traceability of product development. The Model-Based Systems Engineering (MBSE) allows reducing complexity, while supporting the product design by a suitable framework and an effective methodology. It drives designer to define the so-called Digital Twin of product. Nevertheless, the conceptual stage of design might still suffer an arbitrariness, being linked to operator, and the effectiveness of modelling activity might be affected by system complexity. Arbitrariness can be avoided by automating the requirements elicitation, by resorting, for instance, to the NASA's FRET (Formal Requirement Elicitation Tool) and structuring the requirement analysis. System complexity can be decomposed by resorting to two levels of modelling, namely the 'sizing model' for preliminary trade-off and definition of system layout, and the 'dynamic system model' to investigate deeper its performance. Those approaches are applied to design of a landing gear of a commercial aircraft, by analysing those critical issues in implementing the MBSE and demonstrating the effectiveness of proposed approaches.**

Keywords—Model-Based Systems Engineering, Requirement analysis, Functional modeling, Physical modeling, Industrial tool chain

I. INTRODUCTION

Complexity and cost of safety critical systems, in many industrial applications, are rising up and require a holistic approach to design [1]. The "Model-Based Systems Engineering" (MBSE) provides a standard [2] as effective methodology [3], suitable to trace customer needs and requirements to product development [4]. It includes some engineering methods, a specific model of process for product lifecycle development [5], and can resort to several commercial software. Moreover, as it proceeds with a functional and numerical modeling of system, respectively, some dedicated languages, as the SysML (System Modeling Language) [6,7], allow representing the functional model by digital records, thus assuring a complete digitalization of the whole product development [8]. Nevertheless, the MBSE implementation was required to overcome some practical issues. Interoperability of software tools [9], being conceived for different goals, from requirements elicitation, through functional modeling, to numerical simulation, has been assured by the Functional Modeling Unit (FMU) and Interfaces (FMI) for co-simulation [10] as well as by some new software, as the Dassault Systèmes "No Magic Cameo Systems Modeler" [11], aiming at including in just one platform the whole toolchain to improve interoperability. An effective orchestration [12] of tools is even required, and some solutions were provided, for instance, by the Ansys ModelCenter [13] and Dakota software [14].

Apparently, industrial designer can now proceed easily through the system development, by resorting to those tools. Nevertheless, some issues are still unsolved [15]. A deeper integration between product development and sustainability is still matter of investigation [16]. Setting-up of a secure tool chain, to be exploited by all the operators, even within consortia, as well as within systems of systems is key [17]. Particularly, a residual arbitrariness of the functional modeling is still suffered, especially in the requirements elicitation, as it depends upon the operator. It could be mitigated by introducing a driving framework, with a detailed list of steps, as well as by resorting to a disciplined and automated elicitation, based upon specific rules, as described by the NASA's FRET (Formal Requirement Elicitation Tool) [18-20]. This allows then transferring to machine language content, to be embedded into numerical models. Moreover, modeling activity must simultaneously support the trade-off analysis and the whole system definition, but models conceived for those separated purposes are quite different. Particularly, building a digital twin within that technical domain [21], providing high fidelity by definition, is possible only through a detailed numerical modelling activity, which needs a very precisely defined architecture, being assessible only if a suitable preliminary sizing action has been performed [22]. Therefore, conceiving two levels of modeling as namely the 'sizing' and 'dynamic or development' ones could help in distinguishing the depth of information and make fit-to-purpose the approach.

In this paper, the example of the product development of a landing gear of commercial aircraft is exploited to represent some critical issues faced by industrial operators in design, to introduce a structured requirement – functional – logical – physical sequence of analyses for product development, to apply the FRET to requirement elicitation, and then show the structured modeling approach based on sizing and development models of the system of interest.

II. A REFINED MODEL OF PRODUCT LIFECYCLE DEVELOPMENT

The MBSE has already consolidated a reference model of the process of product lifecycle development, specifically applied to industrial product requiring material processing and assembling, which can be sketched by resorting, for instance, to the 'V-diagram' [23], being specifically adopted by standards [24], as shown in Fig.1 [16]. Each 'V' covers a specific step of life, as development, service and end of life. As is known, each element of the 'V' defines a sequence of

XXX-X-XXXX-XXXX-X/XX/$XX.00 ©20XX IEEE

activities from top to bottom (yellow arrows) and bottom to top (green), but at each step designer takes care of following steps, being remarked by the horizontal red arrows. The process is recursive, as it applies first to the whole system, then to subsystems and components, as it reaches the details of design synthesis. Moreover, key actions are associated in Fig.1 to each element, to explore contents of the design activity. Along the development, designer performs the design of product, and then sets up the manufacturing activity, including prototyping and testing. Life includes training of operators, and then monitoring, prognosis, diagnosis and maintenance in system operation [25]. Decommissioning is based upon several actions, mainly disassembling to refurbish and remanufacture product, and then a reconfiguration, which resorts to reuse of components and recycling of materials. That last activity feeds flow in circularity, as components and materials are exploited to produce new systems, in a sort of closed loop.

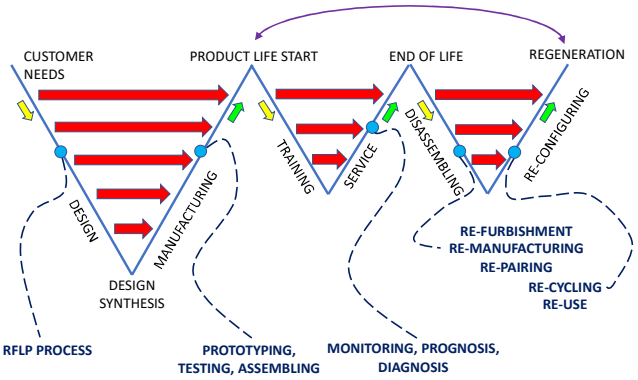

Fig. 1. Multiple 'V' diagram exploited to describe the circular product lifecycle development and main contents of the design activity.

Acronym 'RFLP' in Fig.1 states for 'Requirement, Functional, Logical and Physical' analyses. They are performed in sequence, according to the MBSE, and assure the traceability of requirements, as customer needs are first converted into some technical requirements, and then each requirement is allocated to one or more function, to be allocated in series to logical and physical (real) components, respectively. That allocation assures correspondence between requirements and design synthesis. Design must fulfil all requirements, as confirmed by the verification of requirements, and then product is validated, i.e. it must satisfy all customer needs.

A. Structuring the RFLP process and conversion into machine language within the product development

In that process, activities associated to each item are never obvious. A first source of arbitrariness linked to operator is found by companies in the requirements elicitation. To drive that action, it is known that requirements are classified not only into functional, operational and architectural, being somehow standard by methodology [4], but even in more articulated classes, as shown in Table 1, for the aeronautic product. Some are very general, although they are conceived for transportation systems and flight. Other ones are expressively distinguished within the technical domain of product, and very often are defined by each manufacturer. This classification surely drives the requirements elicitation, but cannot be easily automated yet, nor looks free of arbitrariness, i.e. description and interdependency strongly belong the operator performing this task. Furthermore, some software tools help in collecting and digitalizing requirements, as the IBM® Engineering Requirements Management DOORS [26,27], or even to check their consistency in terms of syntax and attributes, as the RAT–AUTHORING Tool by REUSE® Comp. [28], but elicitation is up to the operator.

TABLE I. EXAMPLE OF REQUIREMENTS CLASSIFICATION APPLIED IN THE AERONAUTICAL ENGINEERING

Requirements classes		
Standard	***General***	***Domain depending***
Functional	Policies and regulations	Performance
Operational	Cost and scheduling constraints	Usability
Architectural	Physical constraints	Interface
	Design constraints	Modes and states
		Adaptability
		Environmental conditions
		Logistics
		Dispatchability

Requirements are verified to be specific, measurable, achievable, and relevant [29,30]. Nevertheless, the degrees of freedom left to operator are still too many for an easy automation of this task, as they are expressed through a text. Therefore, a deeper structuring of the elicitation step is herein proposed by resorting to the space engineering experience, as deployed by the NASA within the FRET language. To understand its role, it be remarked that it is in between the natural language of textual document and the mathematical expressions. Its benefits are a higher level of grammar rules and a deeper semantic level than natural language, which make human readability harder, but more conditioned and suitable for at least a preliminary level of automation, as it becomes interpretable to machine [31]. The FRET is based on a specific sentence building approach, including following elements:

- **Scope** specifies the mode or phase of system operation, in which requirement is relevant
- **Conditions** are identified to be true to have the requirement applicable
- **Component** specifies the (sub)system, which must fulfil the requirement
- **'Shall'** is the verbal statement, imposing that requirement must be fulfilled
- **Timing** specifies the time at which the requirement must be fulfilled
- **Responses** define the actual conditions to be found, in correspondence of which requirement that must be satisfied.

An example of implementation is provided in Fig.2. The usual textual approach based on natural language (on left) is compared to the elicitation described by sentences structured according to the FRET language (middle), which feed then the FMU embedded into physical modelling (right). Each requirement identifies first the subsystem involved, that must satisfy the condition expressed, even according to standards as the EASA CS-25 [32]. Immediately after the specific time for

requirement fulfilment is defined. Finally, the condition to be found for fulfilment is described. In the LG-5.1.1, a specific additional input condition is indicated, in orange, since the wheels of landing gear are locked.

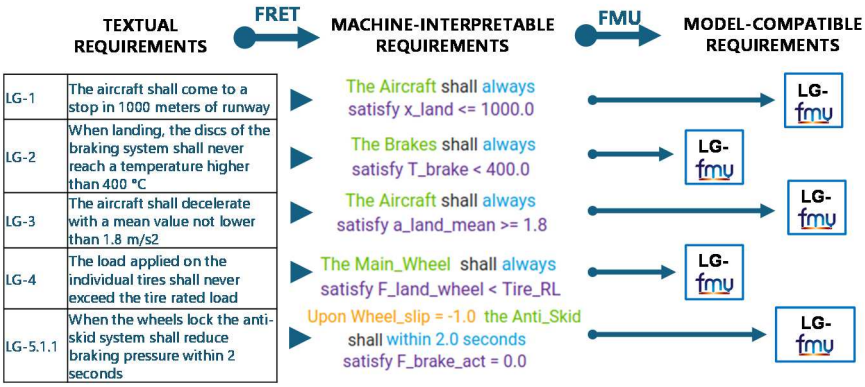

Fig. 2. Implementation of the FRET language in case of the landing gear.

A main benefit of that structuring technique is that conversion of textual requirements into a machine language looks easier, and then it can be exploited by a digital model to monitor system operation and warn the user, when thresholds are reached. Embedding into FMU makes even safe and closed content [33]. This transformation can be performed by means of the Lustre® tool, being a synchronous data-flow language [34]. This language allows associating each textual requirement, being already transformed into a machine language expression, to the physical model of subsystem, and check can be run directly by the numerical model, as the FRET can generate the requirement monitoring action simply as a Simulink block [35]. This approach allows setting-up a driven elicitation of requirement, more formalised and standard to company, almost independent upon the specific user involved. Requirement is even imported and embedded inside the numerical model for a faster and automatic verification, according to the process described in Fig.3.

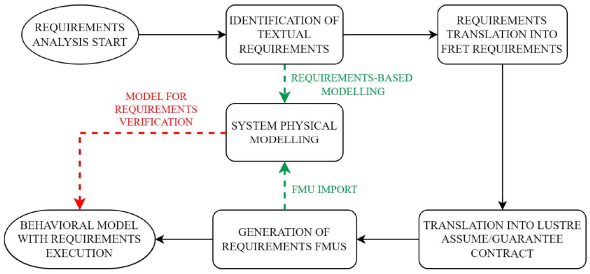

Fig. 3. Implementation of the FRET language in case of the landing gear.

Particularly, the Lustre® tool instructs a Simulink block, within the physical model of subsystem, when it is sufficiently simple and resorts only to that tool. However, standards for interoperability between simulation tools, as the Functional Mock-Up Interface (FMI) allow connecting multiple dynamic models, deployed in several tools, which might use different and mutually incompatible solvers. Those elements are encapsulated into black-box Functional Mock-Up Units (FMU), to be imported into an FMI-compliant simulation. The content of the FMU cannot be accessed, but inputs and outputs are managed by the simulator.

In case of the landing gear, for instance, the simplest model mass-spring-damper with one degree of freedom can be conceived as a Simulink block, to create the FMU, which receives the aircraft longitudinal speed V_{long} and calculates vertical displacement, x_{vert}, speed, V_{vert}, and acceleration, a_{vert}, as a closed block, to be easily embedded into another simulation environment (Fig.4) [36].

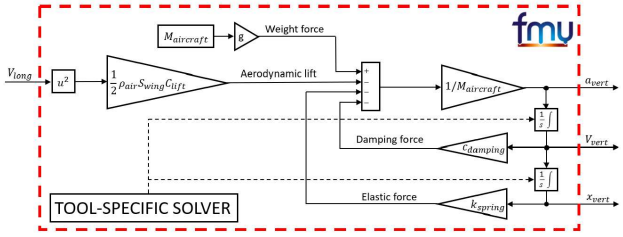

Fig. 4. Example of FMU for cosimulation of 1-DOF model of the landing gear shock absorber.

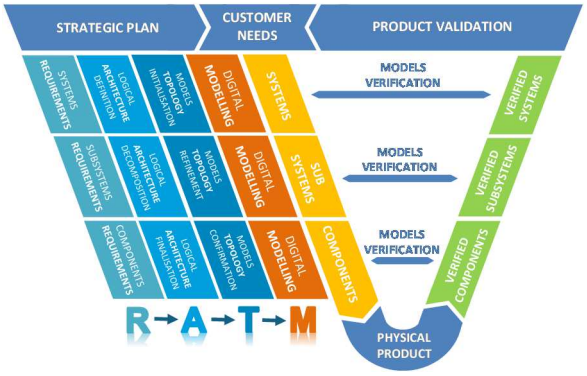

Fig. 5. Evolution of the MBSE-RFLP sequence of analyses into the RATM sequence.

B. Structuring the numerical modeling to include requirements and make the trade-off analysis

The central physical modeling activity described in Fig.3 needs itself to be suitably organized, to allow a consistent activity of design, a precise trade-off analysis between technological solutions and even collaborating different units of the same company or consortium. If a specific classification of requirements, fit-to-purpose for the technical domain as for instance the aeronautical one, and their structuring by FRET allow driving more precisely the operator, it is true that even numerical modeling might suffer an inherent complexity linked to the depth of numerical model. This issue can be mitigated by subdividing physical modeling into a preliminary initialization of system architecture, being supported by sizing models, and then a deeper building of high-fidelity digital

twins, in the simulation of systems dynamics. A simpler approach can support the definition of the system architecture and once that it has been defined, the deep modeling activity can be completed within a more detailed one, by resorting to a defined set of (sub)models, being topologically identified.

First at all the sequence of analyses driven by the MBSE from requirements to design synthesis can be better split into steps as in Fig.5, i.e. by distinguishing the definition of system layout and of related models per each element (T), after logical analysis, from the physical product modelling (M), being including the Digital Twins. According to that approach, the designer is driven through a process which looks more detailed, and somehow standardized, as shown in Fig.6. Particularly, advanced analyses are performed through the dynamic models aiming at investigating the system behavior and performance.

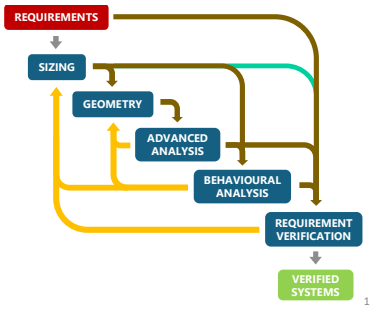

Fig. 6. Evolution of the MBSE design approach when sizing and behavioral models are distinguished.

C. Sizing models

In case of the aircraft landing gear, the sizing modeling is based on a preliminary design activity concerning the system components, as detailed in the flow chart of Fig.7.

The manufacturer practice within the technical domain suggests the initialization of some typical parameters [37], and then some preliminary rules allow sizing all the main elements. In the landing gear, for instance, the diameter of the wheel rim is found by the Sforza's formula [38] as:

$$D_{rim} = 1.4\left(F_{main,vert}\frac{2.20468}{g}\right)^{1/4}\ [in] \qquad (1)$$

where $F_{main,vert}$ is the action applied upon the landing gear, and g the gravitational constant. That parameter allows selecting several other dimensions by catalogue, and determining the contact patch on ground, pressure and properties of tyres, as stiffness. Similarly, a minimum thermal mass in braking system, $m_{brake,eq}$ is defined considering the energy developed in landing, E_{land} :

$$m_{brake,eq} = \frac{E_{land}}{C_b\left(T_{b,des} - T_{env}\right)} \qquad (2)$$

where C_b is the specific heat capacity of brake material, $T_{b,des}$ is the design brake temperature, to be reached at the end of braking action, and T_{env} is the temperature of environment in operation. In this case, that parameter defines the reference mass required, while a specific procedure described by Bailey [39] allows defining all the geometric parameters of the brake housing and rotor, and consequently the braking torque applied.

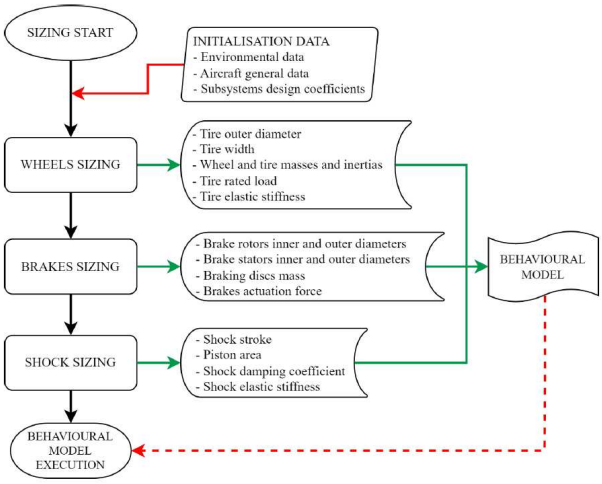

Fig. 7. Sizing modeling for the landing gear of commercial aircraft.

A sizing model supports even the design of the shock absorber, as a first approximation of the maximum stroke required is predicted as [40]:

$$S_{shock} = \frac{V_{Vert}^2}{2g} + S_{tire}\frac{1 - LF_a - LF_t\eta_{tire}}{-(1 - LF_a - LF_t\eta_{shock})} \qquad (3)$$

where V_{vert} is vertical speed, S_{tire} is the tire deflection at rest, LF_a is the aircraft load factor (lift-to-weight ratio), LF_t is the is the landing gear loading factor, η_{tire} and η_{shock} are the tire and shock absorber efficiencies, respectively. Piston area, volumes of chambers, damping and stiff coefficients are then derived.

D. Dynamic/development models for behaviour prediction

Once that the overall geometry of landing gear has been found, a detailed model of its dynamic behavior can be set up, to investigate its performance. This physical model is developed into a dynamic simulator as the Simulink®. In terms of architecture of blocks it looks like in Fig.8.

According to proposed approach, that kind of structure for modelling allows organizing the overall product development and requirement traceability and verification as shown in Fig.9. Particularly, the product development includes a preliminary elicitation of requirements in textual mode, based upon customer needs, resorting to classes (1). FRET allows transforming those requirements into some organized standard sentences, easily translated into machine language and embedded within numerical models (2). Models are developed at sizing and then development stages respectively, interoperated between software and orchestrated, through the FMU (3). Verification is based on some checks, being organized in model blocks, directly monitoring the requirements fulfilment (4).

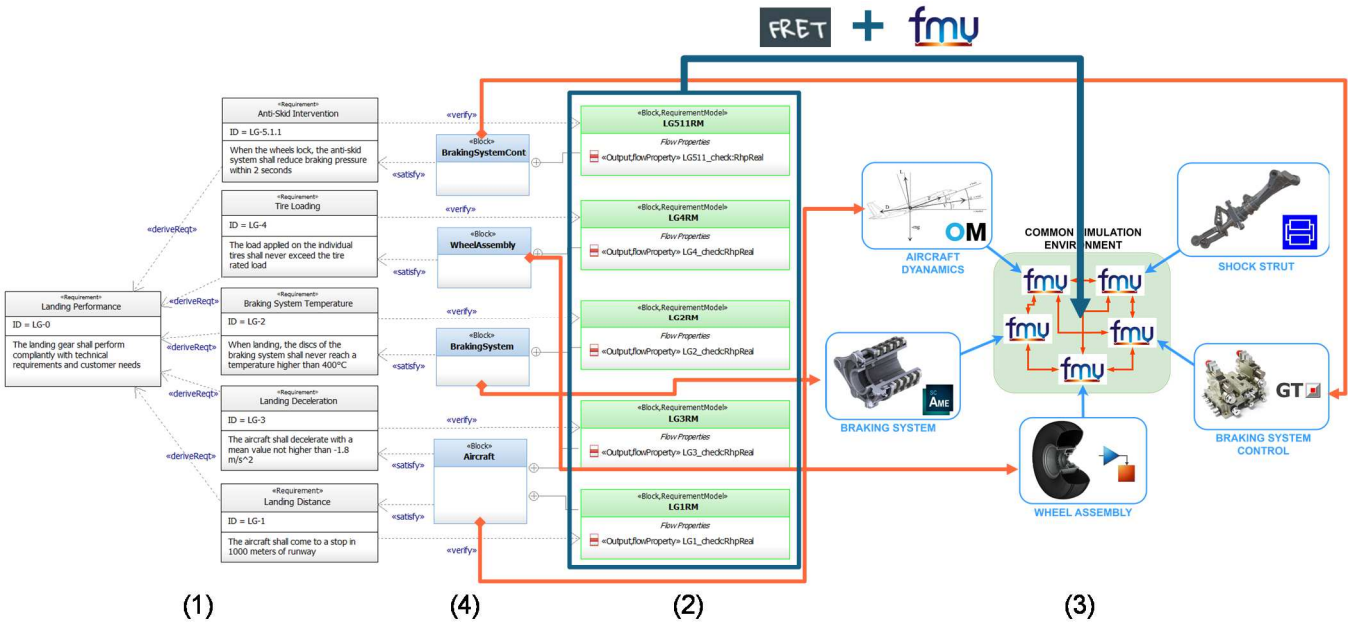

Fig. 8. Overview upon the functional and numerical models of the landing gear of commercial aircraft.

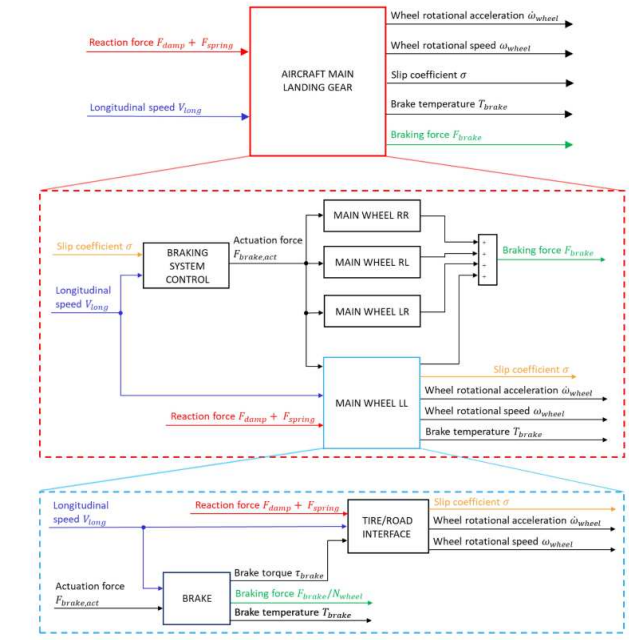

Fig. 9. Dynamic model for the landing gear of commercial aircraft.

III. INDUSTRIAL IMPLEMENTATION AND PRELIMINARY RESULTS

The above-mentioned approach has been preliminarily implemented and validated, at the industrial partner, to be finalized to improve and consolidate the use of MBSE, in the aircraft engineering and design. It can be noticed that building a dedicated platform for operators, to be easily accessible and available on internal cloud to industrial partners has been a key target and highly motivated this study. Although the platform is still under development, the approach was tested at several levels [41]. Particularly, the elicitation of requirements has been organized as above described and numerical models developed to embed those requirements and to perform the verification. At current step, a key performance indicator of the implementation of that approach consists in testing the requirements verification activity through the models and comparing to previous and even more standard organization of the system development. Some outcomes have been detected.

A. Outcomes of implementation

A first useful benefit consists in the immediate visualization of the requirement verification path followed, within the software tool, which looks like in Fig.10.

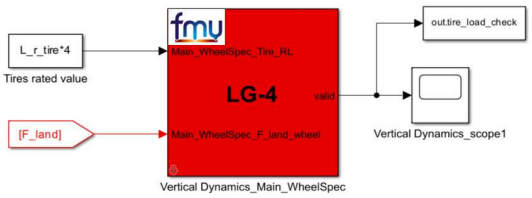

Fig. 10. Example of requirement verification path set-up within the system models.

Second, for each requirement, monitoring is always active within the model, i.e. the corresponding Boolean value (TRUE=1 or FALSE=0) is plotted and visualized, together with measured parameters (Fig.11).

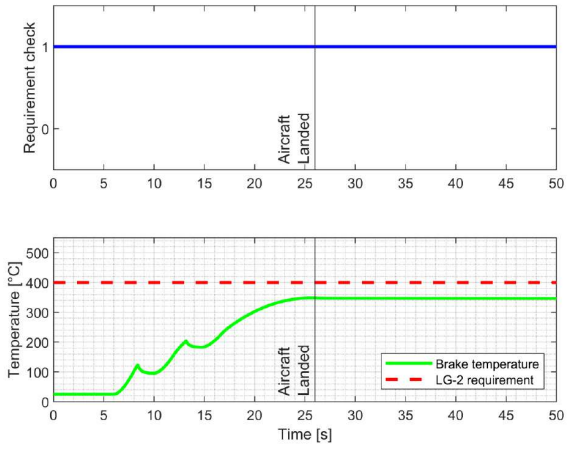

Fig. 11. Example of requirement verification applied to brake temperature of landing gear with Boolean value displayed.

A third interesting feature is that monitoring the verification of requirement drives the user in windowing some signals to explore better some transitional effects, which might be otherwise underestimated. In case of requirement LG-4 of Fig.2, for instance, being related to tire rated load, the simulation highlights the aircraft touchdown, by a reaction force on tires so large that limit of tire rated load (manufacturer's definition of the nominal maximum load applicable to tire), as defined by requirement. For given set-up, verification could switch to FALSE during that time, as in Fig.11. Particularly, the numerical simulation just puts in evidence the concentration of load, being quite typical for that specific time of operation. Designer could underestimate that condition in terms the rate of load, without monitoring the verification index, as in Fig.11 is done. The energy absorption is out of range and thus that tire dimensioning is incompatible with requirement, as the Boolean value of verification suggests, therefore the overloading is critical in that sense, although it could be considered typical.

Further consideration can be applied to requirement LG-5.1.1. It defines the time to release the brake, when braking the landing gear wheel. It must be done within 1 s, for instance, since the wheel looks locked. To investigate fulfilment of this requirement in classical approach a time-consuming numerical computation should be performed, being based on data extraction and analysis, and requires checking the entire simulation to identify when locking of occurs. In this case, that check is automatically and continuously performed, therefore locking effect is easily detected and the operator is immediately warned about that (Fig.13).

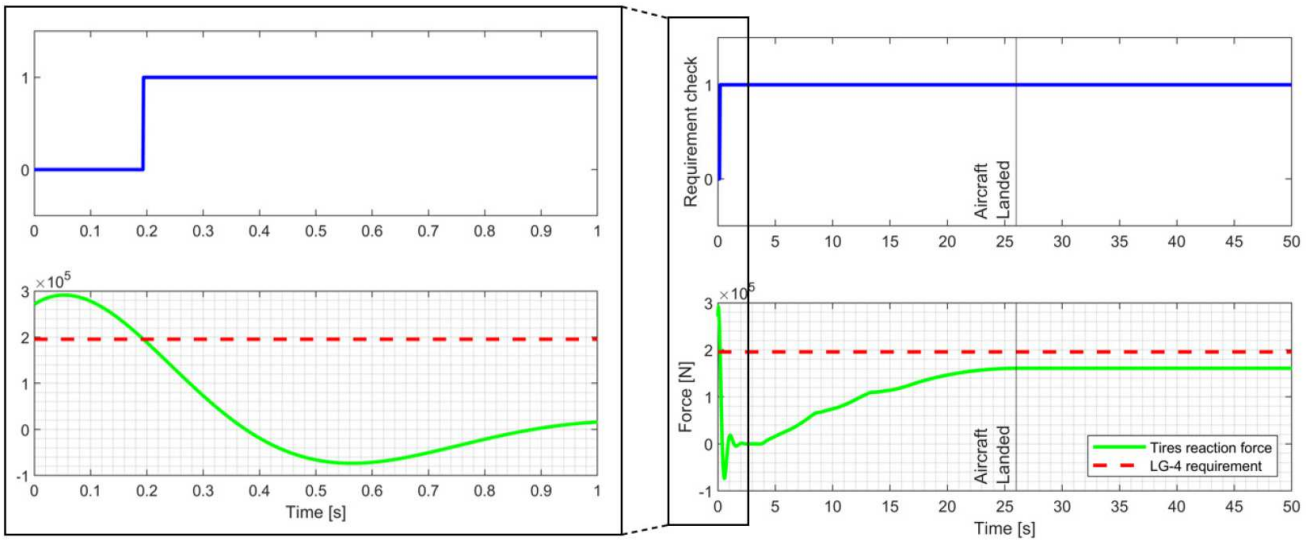

Fig. 12. Example of verification highlighting 'False' during touchdown of the aircraft and monitoring of force upon landing gear tires.

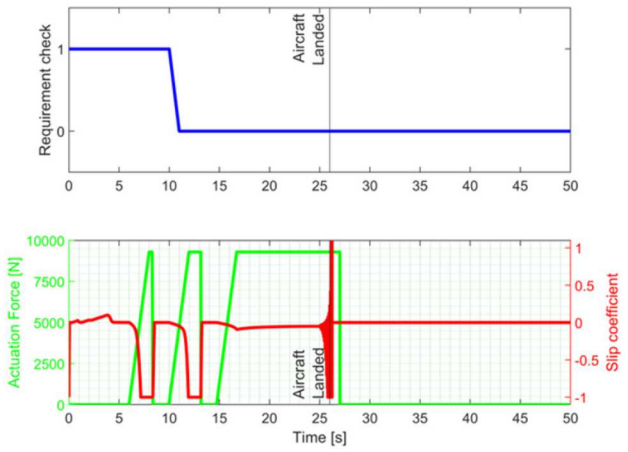

Fig. 13. Example of prediction of longitudinal wheel slip coefficient and brake actuation force during landing, to verify the LG-5.1.1 requirement.

B. *Cost of implementation*

The simulation performance has been even investigated. That approach increases the number of FMU to be implemented as separated modules, including a lower number of data. This structure may affect the time required to perform the simulation. In Fig.14 the simulation time is investigated for several computational cases. Three main parameters are depicted, i.e. the number of FMUs exploited for the landing gear model, that for the system requirements and the communication step between FMUs, i.e. the time required to exchange data between FMUs. As that communication step is required to be smaller, simulation time increases exponentially. However, the time to perform the simulation itself is never the main cause of that increasing, while time to save results is more critical. This is due to fact that tool saves results using a print interval equal to the communication step. Therefore, the number of data points for a shorter step is higher. A key parameter is even the number of FMUs used to handle requirements. A test was done, to investigate its role. Particularly, the model has been run without requirements embedded, with all requirements packaged into a single FMU and then in five separate FMUs. It was expected that simulation time could increase for higher number of FMUs, but it is worth noticing the scale of that increasing. Moreover, it could be applied the separation of FMUs even for the physical model in addition to requirements. As it can be appreciated in Fig.14, when the simulation time significantly increases, when all the five system elements (aircraft dynamics, braking system, braking system control, wheel assembly, shock absorber) are separately handled (right side of plot) with respect to case in which they are all embedded into just one FMU (left side). It can be appreciated that time increasing is not proportional to number of FMUs used, thus leaving room to define a suitable strategy to combine modularity of units and computational performance.

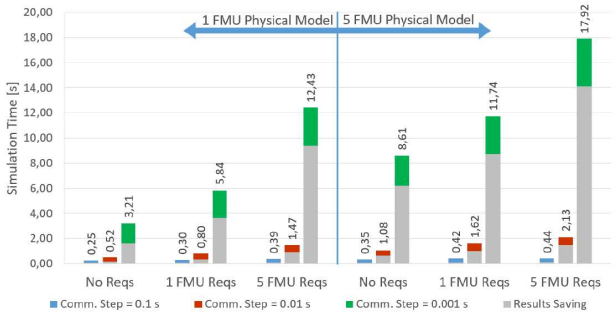

Fig. 14. Sensitivity analysis of simulation time upon variable communication step between FMUs, number of requirements and physical model FMUs.

IV. CONCLUSION

The paper investigates some practical issues found by companies in implementing the MBSE approach to system design and some beneficial effects of strategies currently applied to overcome those. The technical domain involved is the aircraft design, considering the specific case of landing gear. A more detailed path in implementing the product lifecycle development has been applied, by resorting to a preliminary sizing modelling and then a deeper dynamic modelling of subsystems, i.e. by distinguishing architecture, models' topology and system details. Nevertheless, the main target explored is the automatic verification of requirements. That action required resorting not only to a suitable syntax in elicitation, but even to a structured formalism, applying the NASA FRET approach. It could be then interpreted and translated into a machine language to build up some blocks to check quantitatively the requirements fulfilment. Therefore, the standard Functional Mock-Up Units (FMU) and Interfaces (FMI) which assure the interoperability between functional and numerical models are even more exploited to embed requirements metrics and verification conditions inside the physical model to be simulated. The proposed approach looks effective, easily implementable and assures interoperability and easy orchestration by the operator, whose arbitrariness in modelling is practically nulled. Some strategies to optimise the time consuming is simulation are required, although the complexity of systems is even more effectively decomposed in modules. Next step is finalizing the design of an industrial software platform supporting the whole product lifecycle development, and implementing the approaches herein preliminarily tested.

REFERENCES

[1] I.D. Stef, G. Draghici, A. Draghici. "Product design process model in the digital factory context", Procedia Technology, 2013, Vol. 9, pp.451–462.

[2] D.D. Walden, T.M. Shortell, G.J. Roedler, B.A. Delicado, O. Mornas (editors). Systems Engineering Handbook: A Guide for System Life Cycle Processes and Activities, 5th ed., Hoboken NJ (USA): John Wiley and Sons, 2023, pp.344.

[3] J.M. Borky, T.H. Bradley. Effective Model-Based Systems Engineering. New York: Springer, 2019, pp.799.

[4] E. Brusa, A. Calà, D. Ferretto. Systems Engineering and Its Application to Industrial Product Development, 1st ed., Cham (Switzerland), Springer, 2018, pp.376.

[5] A. Kossiakoff, S.M. Biemer, S.J. Seymour, D.A. Flanigan. Systems Engineering Principles and Practice. 3rd Ed. Hoboken NJ (USA): John Wiley and Sons, 2020, pp.688.

[6] L. Delligatti. SysML distilled: A Brief Guide to the Systems Modeling Language. Upper Saddle River, NJ (USA): Addison-Wesley, 2014, pp.300.

[7] M. Debbabi, F. Hassaïne, Y. Jarraya, A. Soeanu, L. Alawneh. Verification and Validation in Systems Engineering: Assessing UML/SysML Design Models. Berlin Heidelberg: Springer, 2010, pp.276

[8] A. Madni, C. Madni, S. Lucero. "Leveraging digital twin technology in model-based systems engineering." Systems 2019, 7, 7; pp.1-13; doi:10.3390/systems7010007

[9] M. Sjarov, D. Kißkalt, T. Lechler, A. Selmaier, J. Franke. "Towards 'Design for Interoperability' in the context of Systems Engineering", Procedia CIRP, Vol. 96, 2021, pp.145-150; doi: 10.1016/j.procir.2021.01.067.

[10] T. Blochwitz, et al. "The functional mockup interface for tool independent exchange of simulation models" in Proceedings from the 8th Int. Modelica Conference, Technical University of Dresden, Dresden, March 20–22, 2011, The Modelica Association and Linköping University Electronic Press, June 2011, pp.105-114.

[11] Dassault Systèmes. "No Magic Cameo Systems Modeler (CATIA)", 2025. [Online]. Available: https://www.3ds.com/products/catia/no-magic/cameo-systems-modeler.

[12] E. Brusa, A. Dagna, C. Delprete, R. Gentile. "An orchestration method for integrated multi-disciplinary simulations in Digital Twin applications", Aerospace, 2023, Vol. 10, 601(7), pp.1-25; doi:10.3390/aerospace10070601.

[13] Ansys. "Ansys ModelCenter", 2025. [Online]. Available: https://www.ansys.com/it-it/products/connect/ansys-modelcenter.

[14] Dakota Software. "EHS Compliance Software", 2025. [Online]. Available: https://www.dakotasoft.com/.

[15] J. Heidrich et al. Systems engineering: challenges and best practices. Kaiserslautern (Germany): Fraunhofer IESE, 2016.

[16] E. Brusa, C. Delprete, C. Gastaldi, L. Giorio, "A roadmap to the Integration Between Systems Engineering and Circular Design to Develop Sustainable Industrial Product", IEEE Systems Journal, vol.18 (3), September 2024, pp.1693–1704; doi:10.1109/JSYST.2024.3435025.

[17] M. Kirshner. "Model-Based Systems Engineering Cybersecurity for Space Systems", Aerospace, 2023, Vol. 10(2), 116, pp.1-17, doi:10.3390/aerospace10020116.

[18] S.R. Hirshorn et a.. NASA systems engineering handbook. The National Aeronautics and Space Administration, 2019, pp.297.

[19] NASA Software V&V. "FRET - A framework for the elicitation, specification, formalization and analysis of requirements", 2025 [Online]. Available: https://github.com/NASA-SW-VnV.

[20] D. Giannakopoulou, A. Mavridou, J. Rhein, T. Pressburger, J. Schumann, N. Shi. "Formal requirements elicitation with FRET" in Proc. From REFSQ Workshops, Pisa, Italy, March 24, 2020. CEUR Workshop Proceedings: vol. 2584, CEUR-WS.org 2020, pp.1–6.

[21] L. Li, S. Aslam, A. Wileman, S. Perinpanayagam. "Digital Twin in Aerospace Industry: A Gentle Introduction", IEEE Access, 2022, vol.10, pp.9543-9562.

[22] E. Brusa. "Digital Twin: Toward the Integration Between System Design and RAMS Assessment Through the Model-Based Systems Engineering", IEEE Systems Journal, Vol.15 (3), Sept. 2021, pp. 3549–3560; DOI:10.1109/JSYST.2020.3010379.

[23] M. Hoffman, T. Beaumont. Application development: managing the project lifecycle. Carlsbad CA (USA): Midrange, 1997.

[24] SAE Aerospace, ARP4761: Guidelines and methods for conducting the safety assessment process on civil airborne systems and equipment, U.S.A., SAE Committee S-18, Warrendale, PA (USA): Soc. Automotive Eng., 1996.

[25] A. Tundis, D. Ferretto, A. Garro, E. Brusa, M.Muhlhauser. "Dependability Assessment of a Deicing System through the RAMSAS method", in Proc. from IEEE Int. Symp. on Sys. Eng., Vienna, Austria, October 11-13, 2017, pp.1-8; doi: 10.1109/SysEng.2017.8088266.

[26] IBM. "IBM requirement management", 2025 [Online]. Available: https://www.ibm.com/products/requirements-management.

[27] IBM. IBM Rational Rhapsody 9.0.1 User Guide. IBM, Armonk, New York, USA, 9.0.1 edition, 2022.

[28] REUSE. “RAT AUTHORING Tools - requirements and models writing”, 2025 [Online]. Available:https://www.reusecompany.com/rat-authoring-tools.

[29] A. T. Bahill, S.J. Henderson. “Requirements development, verification, and validation exhibited in famous failures.” Systems Engineering, 2005, Vol.8(1),pp.1–14; doi:10.1002/sys.20017

[30] B. Powel Douglass, C. von Holst. Agile Model-Based Systems Engineering Cookbook: Improve system development by applying proven recipes for effective agile systems engineering. 2nd Ed. Birmingham-Mumbai: <Packt>, 2022, pp.600.

[31] Z. Ádám et al. “From Natural Language Requirements to the Verification of Programmable Logic Controllers: Integrating FRET into PLCverif”, in Proc. from NASA Formal Methods (NFM), Houston TX (USA), 16-18 May 2023, [On line] Available: https://ntrs.nasa.gov/api/citations/20220019339/downloads/FINAL%20NFM23_NASA_FRET_PLCverif.pdf

[32] EASA. CS-25 large aeroplanes certification, EASA Standards: October 2003.

[33] C. Wolf, M. Schleipen, G. Frey. “Secure exchange of black-box simulation models using FMI in the industrial context”. in Proc. from the 15th Int. Modelica Conf. 2023, Aachen, October 9-11, 2023. Modelica and Linköping University Electronic Press, December 2023.

[34] P. Caspi, D. Pilaud, N. Halbwachs, J. A. Plaice. “Lustre: a declarative language for real-time programming”. in Proceedings from the 14th ACM SIGACT-SIGPLAN Symp. on Principles of programming languages (POPL ’87), Munich, January 21-23, 1987. New York, NY (USA), ACM Press, 1987.

[35] A. Mavridou, H. Bourbouh, P-L Garoche, D. Giannakopoulou, T. Pressburger, J. Schumann. “Bridging the Gap Between Requirements and Simulink Model Analysis”, in Proc. from Meeting: 26th Int. Working Conf. on Requirements Eng.: Foundations for Software Quality, Pisa, Italy, 23-26 June 2020, [On line] Available: https://ntrs.nasa.gov/citations/20205004070

[36] The MathWorks Inc. Matlab version: 9.14.0 (r2023a), 2023.

[37] C. Delprete, A. Dagna, E. Brusa. “Model-based design of aircraft landing gear system.”, Applied Sciences, 2023, Vol. 13(20), 11465, pp.1-23; doi:10.3390/app132011465.

[38] P. Sforza. “Landing Gear Design” in Commercial Airplane Design Principles. Berlin/Heidelberg (Germany): Elsevier, 2014, pp. 251–300.

[39] D.A. Bailey. Investigation of improvements in aircraft braking design. Ph.D. Thesis, Cranfield University, Cranfield (UK), 2004, p.55.

[40] N.S. Currey. Aircraft Landing Gear Design: Principles and Practices. Reston, VA (USA): American Institute of Aeronautics and Astronautics, 2012, pp.78-110.

[41] E. Brusa, D. Ferretto, A. Calà. “Integration of heterogeneous functional-vs-physical simulation within the industrial system design activity”, in Proc. from IEEE Int. Symp. on Sys. Eng., Rome, September 29–30, 2015, IEEE CFP15SYM-POD, Vol.I, pp.303-310.