EasyChair Preprint

№ 2104

Property Model Methodology: key principles and benefits for future aircraft programs

Pascal Paper, Patrice Micouin, Louis Fabre and Thomas Razafimahefa

December 5, 2019

X-Method: key principles and benefits for future aircraft manufacturer programs

Abstract. In the growing level of complexity of systems, classical MBSE descriptive approaches have shown their limitations vs strong development program expectations in terms of quality, costs and lead-time improvements. This paper is presenting an innovative Modelling & Simulation method called X-METHOD. This method fosters an effective preventive approach in the end2end system development cycle (ie: system specification, design and physical test and integration). This preventive improvement is emerging from the sound integration, by the X-METHOD, of key System Engineering principles with Modelling & Simulation means. We will present X-METHOD principles and process steps, then the result of a first real evaluation on an aircraft manufacturer pilots confirming the pertinence of X-METHOD vision. Finally, next steps to target a future generalization are exposed in conclusion.

Main feedbacks from previous aircraft manufacturer programs

On several aircraft manufacturer developments (Xxxx, Zxxx,...), requirements are often captured as **texts,** and operational scenarios - when captured - are either formalized in MS Office tools or in descriptive MBSE tools. The consequences are a huge number of requirements (Eg: 240000 on Xxxx program) which are, **overall and between levels**, incomplete, incorrect and incoherent. To illustrate this situation, let us take a simple but representative example. This is the specification of green arrows states indicating to the pilot when the Landing Gear System is fully extended (See Fig 1).

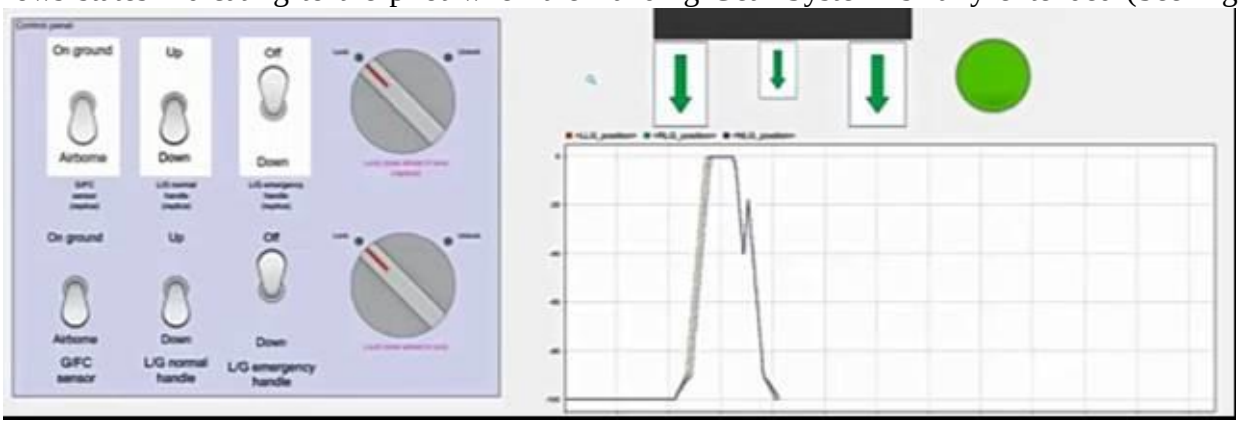


Fig 1

REQ1 and REQ2 are two specifications captured in textual approach. They are supposed to define the Green Arrow state (ON/OFF) vs normal and back-up extension command:

REQ1 : When Normal_Cmd is Up=> Green_Arrows shall be Off after 0.05 s

REQ2 : When Back_Up_Cmd is Down => Green_Arrows shall be On after 15 s

These two specifications are incompletes, as we do not know what happens on Green_Arrow state when Normal_Cmd is Down and Back_Up_Cmd is up. They are also incoherent since if Normal_Cmd is up and Back_Up_Cmd is down the Green_Arrows should be both On and Off after 90 s. Finally, REQ2 is incorrect as requiring 15 seconds will trigger system over costs while 90 seconds is enough.

This result is caused by the current formalization approach, where there are no **simulation based validation** on requirements or scenarios. We are missing there the opportunity to rely on a sound modelling & simulation methodology to introduce an effective **preventive** approach to produce an early & robust **validation of the system* specification and their refinements before starting system design.** Another key issue to tackle, is that formal textual requirement formalization do not enable to perform **concurrent trade-offs** on the **problem space,** aiming to better frame and characterize it before entering in **solution space** exploration.

On the other hand, the **solution space** of the system to be developed is in general formalized in MBSE descriptive approach, but not using executable language and associated tools. Sometimes executable language tool (e.g. DYMOLA, AMESIM, SIMULINK…) are used but with no clear global modelling goals (i.e. design and functional chain verification).
So, we are again missing a **preventive** way to support our System Design Responsible (SDR) to deliver, **at all levels** of the System Of Interest (SOI), **robust design verification**** in the left side of the V and then robust **product verification**** in the right side of the V during test & integration phase.

All in all, not using sound **M**odelling and **S**imulation **B**ased **S**ystem **E**ngineering (**MSBSE** [Ref 12]) methodology to reduce specification and system design errors on the left side of the V, creates a high risk of late rework in downstream development phases (left & right side of the V). And this late rework will in turn impact the high level aircraft program ambitions: development costs & lead time reduction objectives, maturity and safety objectives... The later these errors will be discovered the bigger will be the impact. The above analysis can be applied not only to the product but also to associated industrial & services as they can be also be considered as a SYSTEM according to INCOSE definition of systems [Ref 15].

*: It is to be noticed we are here mentioning specification validation according to ARP4754A principles (ie: specification validation is answering the upper level need) [Ref3].

**: It is to be noticed we are here mentioning Design & Product verification according to ARP4754A principles (ie: design is compliant with its associated validated specification [Ref 3].

Global target improvement expected for new programs

Our experience tells us that aircraft manufacturer development teams are not always applying systemic and systematic modeling & simulation methods to capture, trade and validate the problem space (i.e. from set of requirements at all levels). The consequence is that this effort of requirement elicitation & validation is drifting to the design activities with the consequences of a wider/incomplete/incorrect/incoherent problem to solve (see fig 2) as input to design activities. Due to program timeline pressure, we can foresee that many inaccurate/ambiguous specifications will be generated with this **open problem space/solution space loop process**. The lack of preventive verification process and the associated program performance consequences to discover these errors during the test & integration or even worse during in service phase.

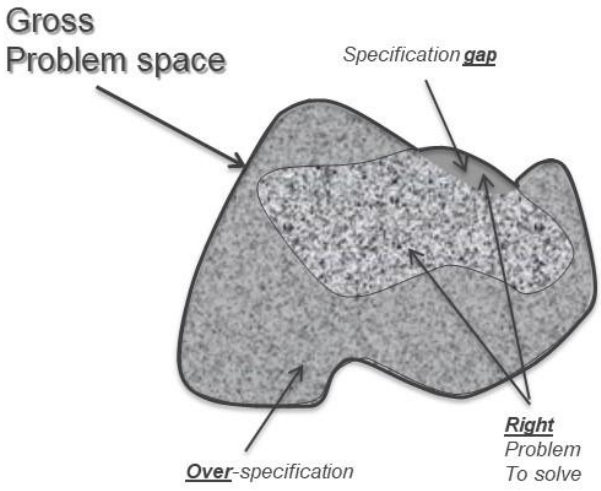


Figure 2

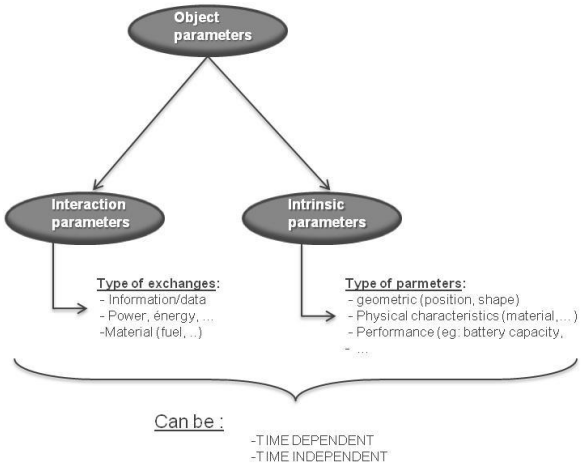


Figure 3

To avoid these issues and contribute to future programs expectations, we recommend in this paper a systemic preventive approach, consisting in modelling the whole problem space in a parametric way. The problem space parameters used can be split in two main families: **interaction** & **intrinsic** specific parameters (see fig 2. These parameters can also be either time dependent or not. So, the problem space domain will be described by the variation of the problem space parameters within the **authorized domain of these parameters including the parameters interrelations - either time dependent or not**. This modeling effort will enable to perform trades on this problem space model, and to perform its formal & factual validation. Such a more rigorous approach will lead to a more robust Solution Space Exploration and down selection of the preferred solution (see fig 4). It shall provide downstream design teams with far clearer specifications, better reflecting customers and other stakeholders needs (i.e. **the problem to solve)** – as opposed to the initial wide and ambiguous gross problem space. In addition, this segregation of Problem Space vs Solution Space opens the opportunity for agile methods (eg: SAFE [ref 14]) application and so to accelerate development cycle.

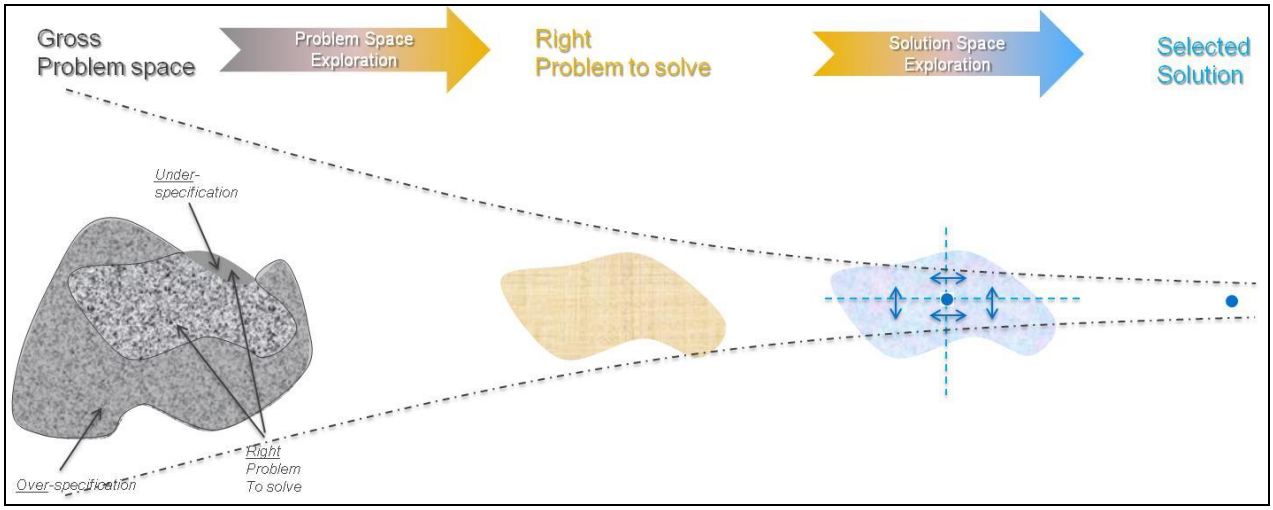


Figure 4

We shall describe in next chapters more in detail, X-METHOD CONCEPT and X-METHOD [Ref 2] more globally as System & Simulation System Engineering driven methodology are proposing a simple and efficient methodological solution answer to the above objectives.

How the system specification model is captured and validated

Specification model capture: X-METHOD CONCEPT [Ref 9], derived from general system theory [Ref 4] is considering the general **cause-effect** principles to capture the whole expected behavior of a system. To that purpose, X-METHOD CONCEPT propose to capture, through **executable models**, the complete expected behavior of a systems. It is widely recognized in general systems theory, that general systems are following the rule of **cause-effect principle** [Ref 4].

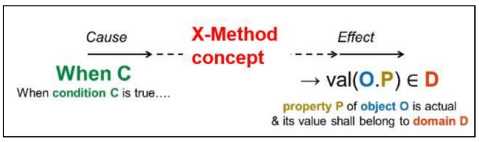


Figure 5

The capture of a system specification modelling sequence, regarding its expected behavior, is a **backward** process from the effects to the causes (see fig 5). The first step is to identify, in black box approach, all inputs and outputs external interfaces of the considered system. The second step, is then to capture the complete expected behaviors of the system under specification. X-METHOD CONCEPT is used to formalize all relations between Outputs & Inputs, Outputs & Outputs and Inputs & Inputs. The last relation type is also called assumptions (see fig 6). It is to be noticed that these expected behavior relations can be either dynamic (time dependent) or static (time independent).

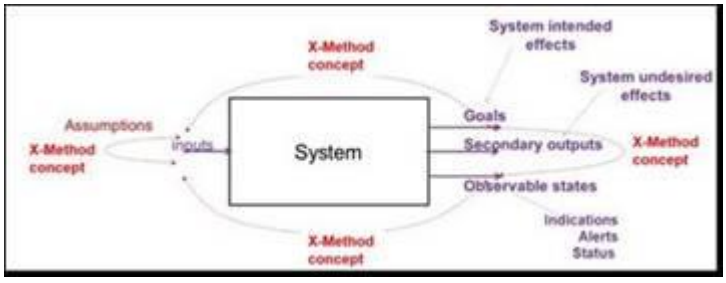


Figure 6.

All these X-METHOD CONCEPT relations can then be modeled using a simulation language like SIMULINK and STATEFLOW. Below is an example of such a simulation for an aircraft landing gear extension function (see fig 7).

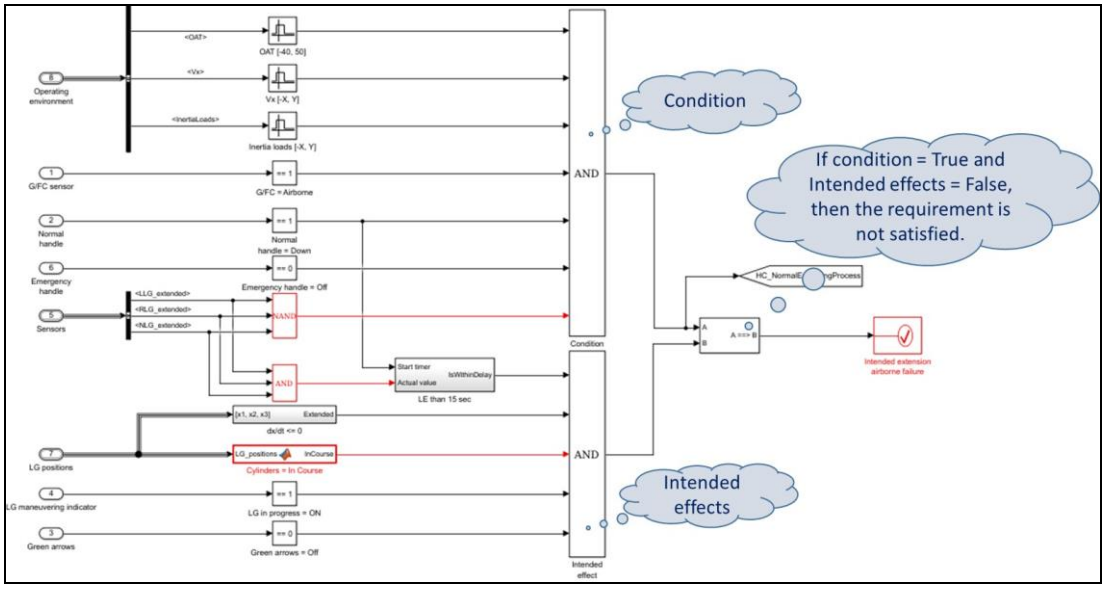


Figure 7.

In addition, to the expected conditions capture, **un**expected conditions linked to system degraded modes can also be captured through X-METHOD CONCEPTs formalism. This opens a very interesting possibility to include in the specification, the modeling of the safety requirements from the beginning of the development. It gives the opportunity to better explore system architectures trades within safety constraints.

Finally, in addition to the capture of expected & unexpected dynamic behavior of the system, through X-METHOD CONCEPT external interface relations it is possible and very useful to extend the specification modeling effort to all other system property conditions of the system. In particular, all conditions on intrinsic property shown in fig2 (e.g. maximal weight, recurrent cost, type of technology/material…) can be added in the system specification model. These properties can be time dependent (e.g. fuel weight vs flight phase) or not (e.g. recurrent cost). These additional properties will be included within X-METHOD CONCEPT specification model execution. In case of not time dependent conditions, the property will be used for condition check within time independent specification validation execution process (e.g. airframe cost & weight will be deduced from technology choice through techno/cost relation already captured in a general cost & weight knowledge data base respectively). The advantage of this global X-METHOD CONCEPT modelling strategy over **T**ext **B**ased **R**equirement (**TBR**), is to get an **overall specification model.** This model shall allow aircraft manufacturer new developments to take benefit of model simulation and execution to perform far more efficient specification validation and including its refinement and then system design verification.

Specification model validation: The purpose of specification model validation is to take full benefits of simulation & proof tools for reaching early & robust specification model validation. Specification validation is two folds: establish **formal & factual truth** [Ref 11] of the specification.

a) **Formal validation:** the goal is to prove specification model logical coherence & completeness based on a **proof tool,** which allows to detect and correct logical holes (e.g. are there any inputs not addressing at least one system output?) and logical contradictions (e.g. are there two functions trying to address simultaneously one same display which can manage only one signal at a given time?).

b) **Factual validation:** the goal is to prove **compliance of specification model vs stakeholder needs**. In particular, it will enable to detect missing or wrong inputs, outputs or system behaviors. Here, the specification models are simulated to share, through model review with relevant system user representative's panel, the assumptions of expected behavior of the system. This is a **concurrent process** review on the specification model. The simulation specification is presented to users through a simplified but representative human interface, on which the user can experiment on inputs (e.g. actual & virtual control on fig 8 as an example of A/C LGS extension/retraction function specification) and see dynamics effects (e.g. actual & virtual outputs on fig 8).

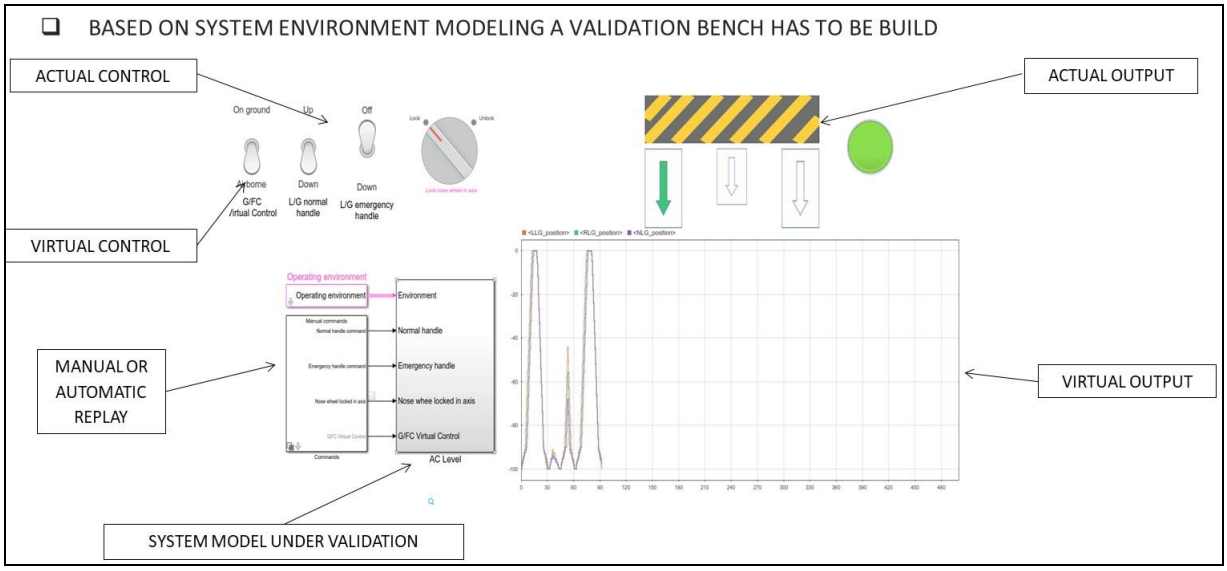


Figure 8.

After the first review, agreed specification gaps & deviations are triggering corrective actions on the specification model. Then, a second review is performed to be sure that corrections are not generating new specification gaps and so on until no specification errors or gaps are remaining. It is important to underline that the method & tool must be considered as a **help** to specification review stakeholder's panel, to reach a shared understanding of the assumptions specification model as captured by the System Development Responsible. The experience, "critic eye" and collective intelligence of the stakeholders' panel attending to this review are of a prime importance to detect the relevant specification model issues. To capture "hidden" specifications and support the convergence of this process, it is recommended to support this collective work with creativity and non-regression methods based for instance on DFMEA approach.

How the system architecture model is captured and validated

Architecture design capture: Once system architecture specification validated, the architecture design team can start to develop design architecture solutions by modelling executable architecture models. During this phase, the focus is to select components candidates to the architecture solution, identify **external** interfaces and elaborate associated **dynamic functional chain**. You can see an example below on fig 9, which represents a functional architecture choice for a landing gear extension/retraction functions.

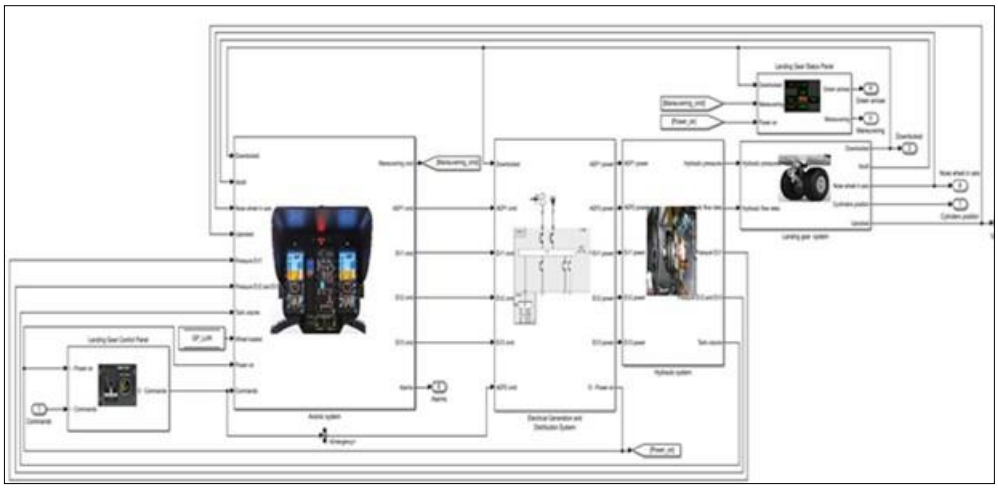


Figure 9.

Architecture design verification: Once executable architecture design model realized, it is possible to run a first preliminary verification process. This one is performed by co-simulating together architecture design and specification models by re-injecting design model inputs & outputs in the specification model (see fig 10). As the specification model "knows" through validated set of X-METHOD CONCEPT the authorized conditions between outputs and inputs, in case of design model non-compliance, a flag will warn the architecture designer on his design error, and for which time of the simulation scenario. This approach is drastically reducing the risk of design errors. It is to be noticed that simulation scenarios can be either manual, coming from flight data, recorded & replayed or automatically generated from the same proof tool used during validation step (see bottom left of fig 10)

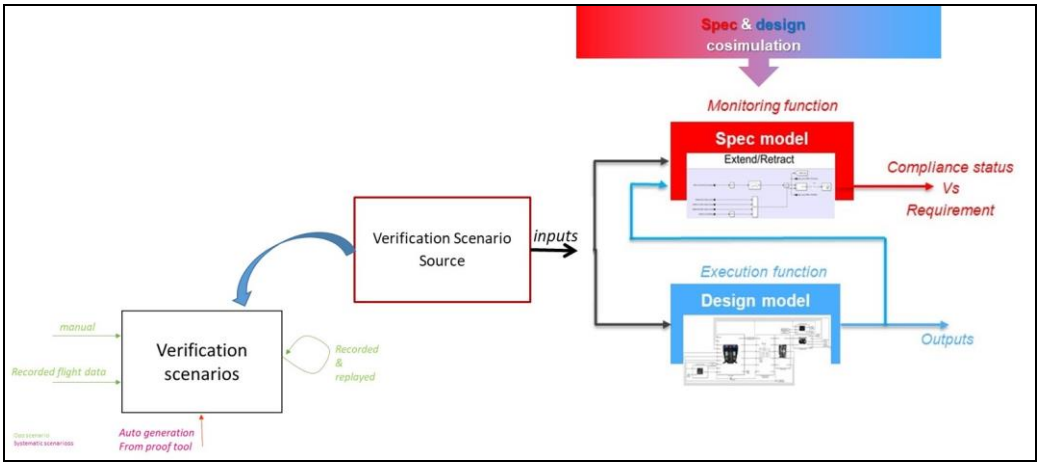


Figure 10.

How the system requirement refinement is captured and validated

System requirement refinement capture: From the architecture choice, architecture decomposition choices are made. In figure 11, extension function is decomposed in three system contributors (i.e. avionic, hydraulic & mechanical). The architecture specification will then be **refined** within three **contributing functions**. For instance, the extension requirement in less than 15'' will be distributed wisely in the three contributing system (e.g. 0.4'' for avionic, 2'' for hydraulics and 12'' for mechanical). For more complex physical problems like the noise, an efficient refinement can only be performed through a **robust architecture study** to enable architecture components variations quantification in term of their wave's characteristics, positions & orientations, etc…

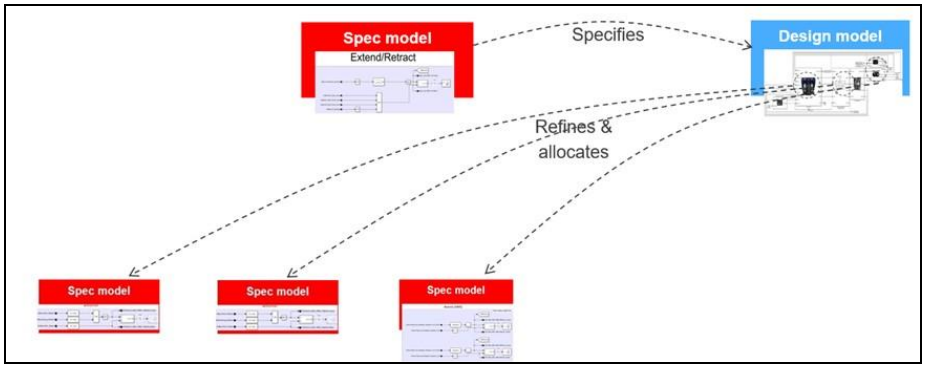


Figure 11.

System requirement refinement validation: Here is another benefit of requirement modelling. The contributing systems specifications refinement can be validated by **proof** means. To get a valid refinement, envelop of the three contributing systems specifications must be at least **more restrictive than the upper level specification** (see fig 12).

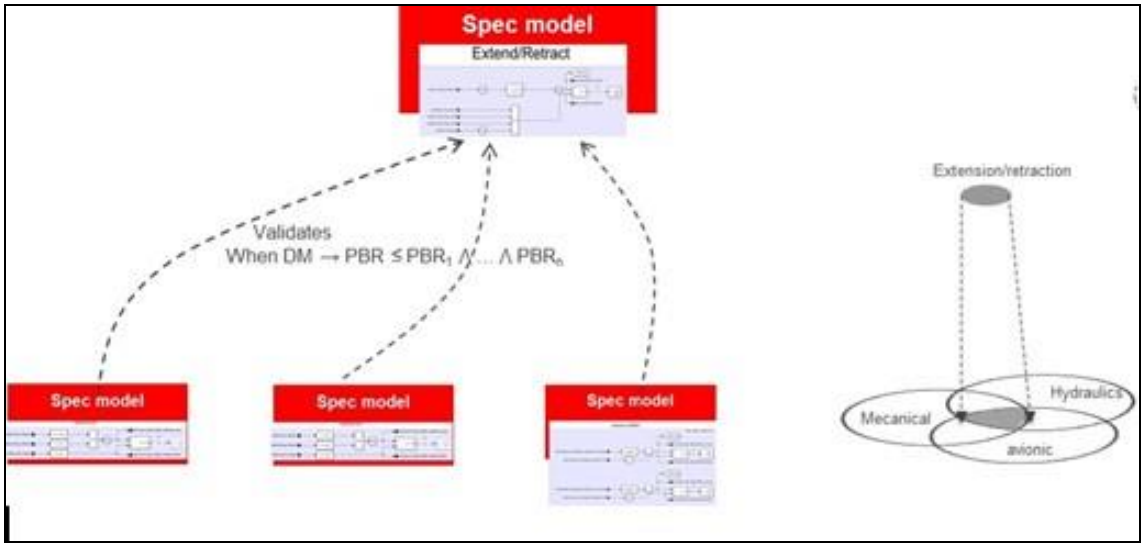


Figure 12.

Specification/Design Zigzagging down to items level

Once contribution system refinement done, the process is an iterative specification-design zigzagging process in line with [Ref 1, 8, 10]. Each contributing system specifications are formally & factually validated as done previously at upper level. See fig 13:

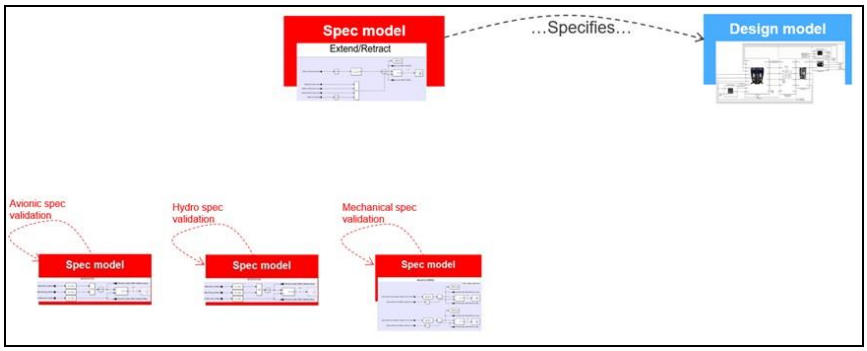
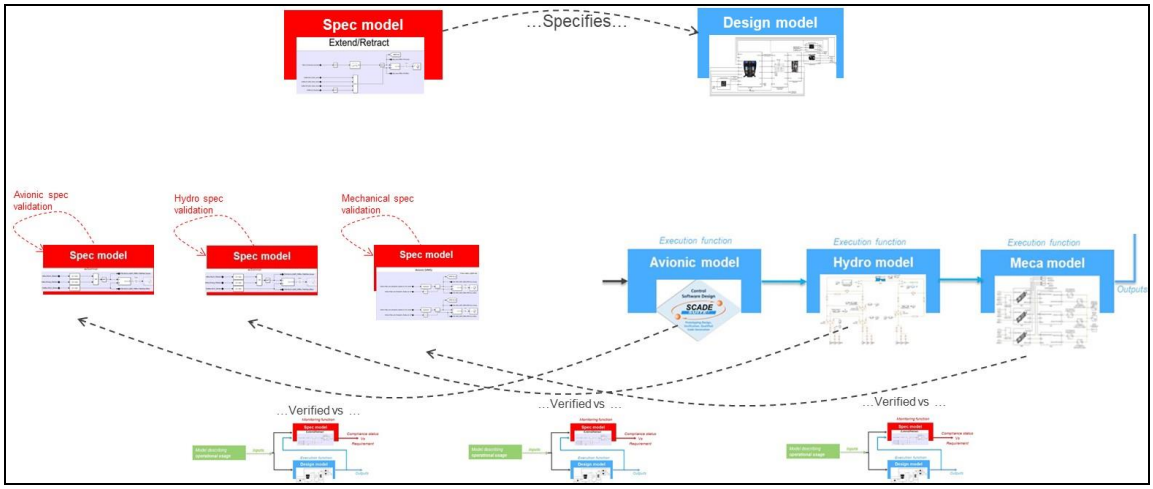


Figure 13.

Then, the design of each contributing systems is done and they are verified vs their associated specifications. See figure 14.



.

Figure 14.

This process goes down until we reach the lower level of decomposition (item level) where there is no need of deeper design (ie: known system component already development and reused).

Product verification during test & integration phase

Another benefit of the specification modelling effort is during test & integration phases. Here, the specification model and operational scenarios build during design phase can be reused to streamline the test preparation & execution phases. Once the product item are produced (hardware & software), their tests can star (see fig 15).

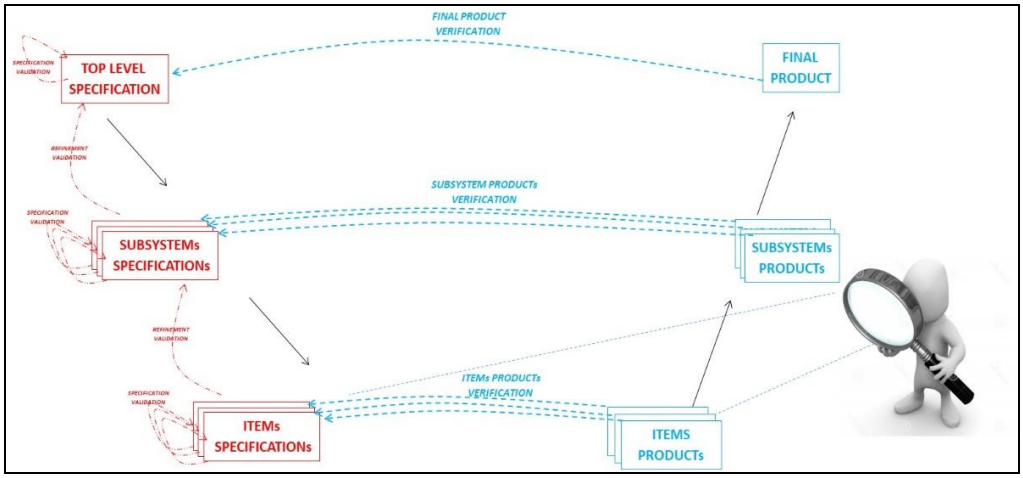


Figure 15.

The specification model and operational scenarios are loaded on the item test bench and the item inputs & outputs are physically mapped with test means (e.g. SPEEDGOAT) on which the specification model is loaded (see fig 16). With this approach we reduce the test preparation time and the risk of interpretation specification errors during a classical test preparation through test programming activities. In addition, as specification and operational scenario models are shared with design teams, there is a better integration between test & design teams, and left part models & teams can support debugging of right part verification in case of errors found (especially when global error are found, simulation can significantly help to drive investigations in physical product by providing information of what the normal state should be at any point in the model / product).

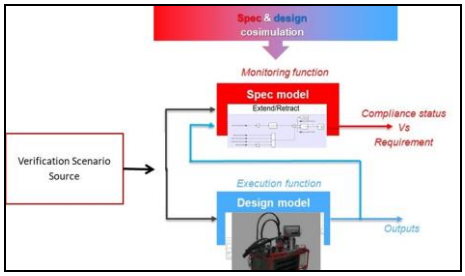


Figure 16.

Final product verification

The product & test process is a bottom – up iterative process until reaching the upper level which is the final product (see fig 17)

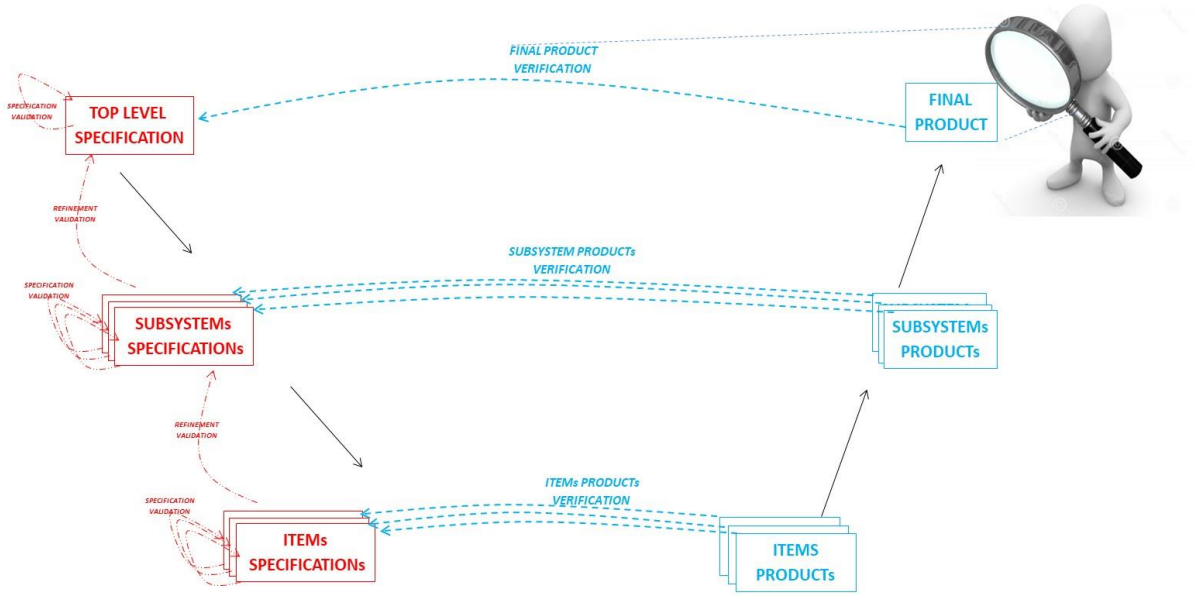


Figure 17.

At this level, the architecture specification model is used for end product verification as done previously at lower levels except that we are testing here complete physical functional chain (see fig 18). We acknowledge the same kind of benefits than those mentioned at lower level physical tests.

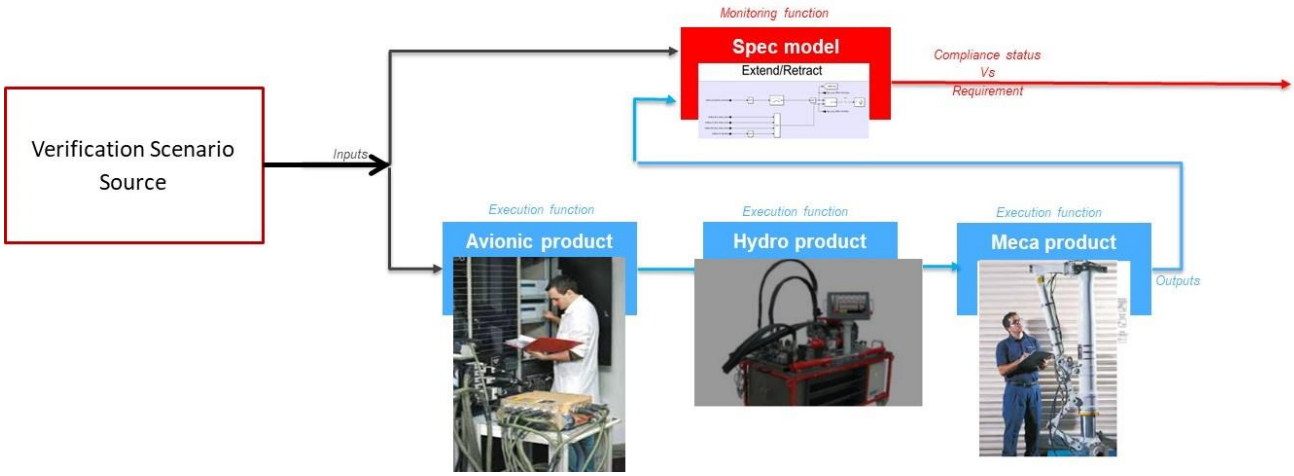


Figure 18.

Conclusion and perspectives

X-METHOD principles has been evaluated [Ref 5, 6, 13] through proofs of concept and pilot projects related to operational aircraft manufacturer projects phase on several engineering disciplines like electric power generation, landing gear, fuel and avionics (see fig 19). The goal of these evaluations was to demonstrate the relevance of X-METHOD method and also to evaluate the level of learning effort of our key users to reach autonomy in its application.

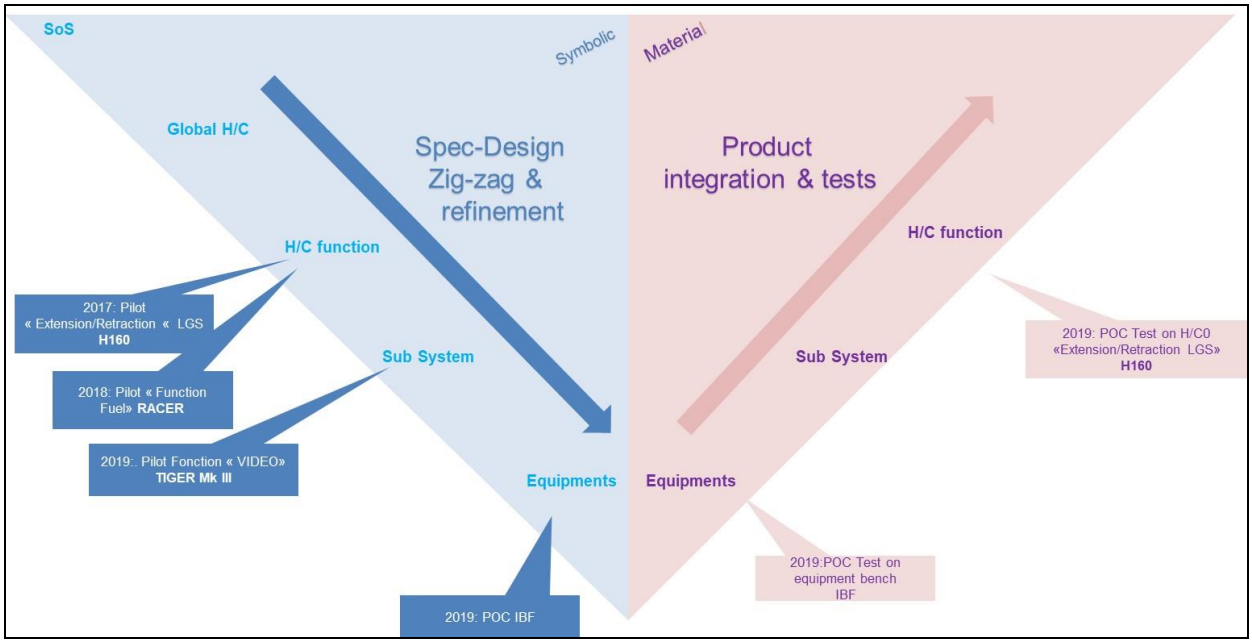


Figure 19.

Performed evaluations are currently not covering the complete perimeter of a new aircraft development. In particular, as shown in figure 19, the System Of Systems and aircraft levels have not been evaluated yet. In addition, many systems remain to be evaluated even if the Landing Gear System, Fuel and Avionic Systems are quite representatives systems. It is also, important to notice that Industrial System and Service System which can be considered also a system (see INCOSE definition), remains to be evaluated to investigate how it could improve their development and co-development together with the aircraft. In addition, it will be of prime importance to evaluate the methodology on a use case involving a System supplier.

However, the mentioned evaluations are showing the pertinence of the proposed Model & Simulation Based System Engineering approach for establishing the targeted end2end preventive approach presented in the introduction of this paper. In addition, the evaluations have shown the effort of modelling is clearly not neutral for our key users but the majority of them demonstrated, on aircraft manufacturer pilots, a good learning curve rate provided a significant effort of training and coaching to support them at the beginning.

To establish a complete demonstration of the global value of this integrated method approach, a significant effort will be necessary by extending the POC to the complete Aircraft development perimeter. In particular, treating with methodology the aircraft level is of prime importance to enable performing a far more robust aircraft system functions allocation trades, and so significantly reducing the silo effect we observe today without aircraft level robust specification capture and validation. In addition, to ensure efficient generalization to all development teams, it is necessary to develop significant adaptations of current Development Process, Method, Tools and Competences referential. We are convinced that this investment will largely benefit to the future aircraft manufacturer Group

developments by contributing to reduce late specifications and design corrections and so contributing to our future Programs Time, Costs, Quality and Performances improvement objectives.

References

1 Suh N.P, 2005, Complexity, Theory and Applications, Oxford University Press.

2 <anonymous>. 2014: <anonymous>, Wiley & ISTE

3 SAE, 2010, ARP4754A, Guidelines for Development of Civil Aircraft and Systems, Warrendale (PA).

4 Bunge M., 2007, Philosophy of science, volume 1: from problem to theory, Chapter 1, the scientific approach, Transaction Publishers, New Brunswick, New Jersey, 4th print.

5 <anonymous>, 2016: X-Method: A First Assessment in the Avionics Domain, ERTSS 2016.

6 <anonymous>, 2017: X-Method: a case study with Modelica, TMCE 2017.

7 ISO/IEC/IEEE, 29148, 2011, "ISO/IEC/IEEE International Standard - Systems and software engineering -- Life cycle processes -Requirements engineering

8 Suh N.P, 2001, Axiomatic Design Advances and Applications, Oxford University Press**.**

9 <anonymous>2008, "Toward a X-Method concept theory: system requirements structured as a semilattice", in: Systems Engineering, Vol. 11-3, pp. 235-245.

10 Lee D.G, Suh N.P, 2006, Axiomatic Design and Fabrication of Composites Structures, Oxford University Press**.**

11 Leibniz, G.W., 1714, "Monadology".
http://home.datacomm.ch/kerguelen/monadology/monadology.html (accessed on 13 March2018)
"There are also two kinds of truths, those of reasoning and those of fact. Truths of reasoning are necessary and their opposite is impossible: truths of fact are contingent and their opposite is possible."

12-Gianni, D., D'Ambrogio, A and Tolk (2015): A. Modeling and Simulation-Based Systems Engineering Handbook, CRC Press, Boca Raton, FL,

13. <anonymous> (2018) <X-Method>: A Landing Gear Operational Use Case

14. SAFE: https://www.scaledagileframework.com/

15. INCOSE System definition:
https://www.incose.org/about-systems-engineering/system-and-se-definition