THALES

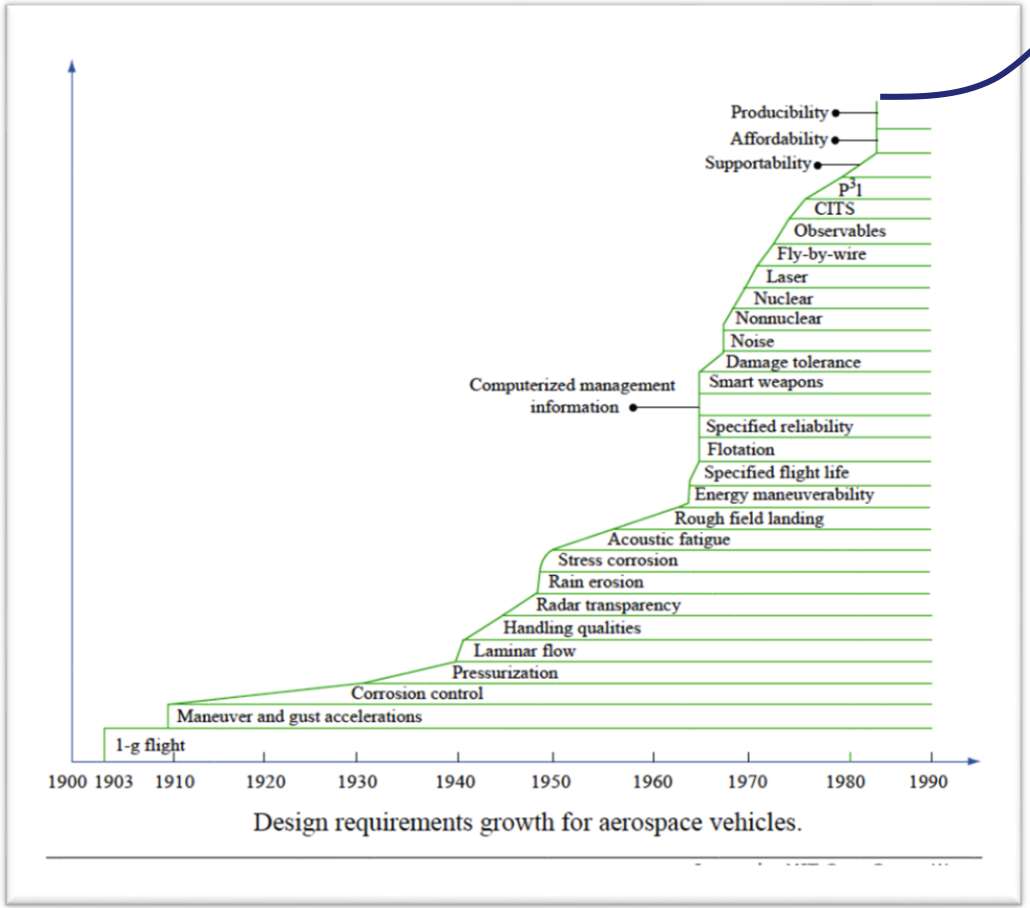
MBSE with Arcadia and Capella

Reconciling with the past and moving towards the future

Juan Navas
Thales Corporate Engineering
juan.navas@thalesgroup.com

650

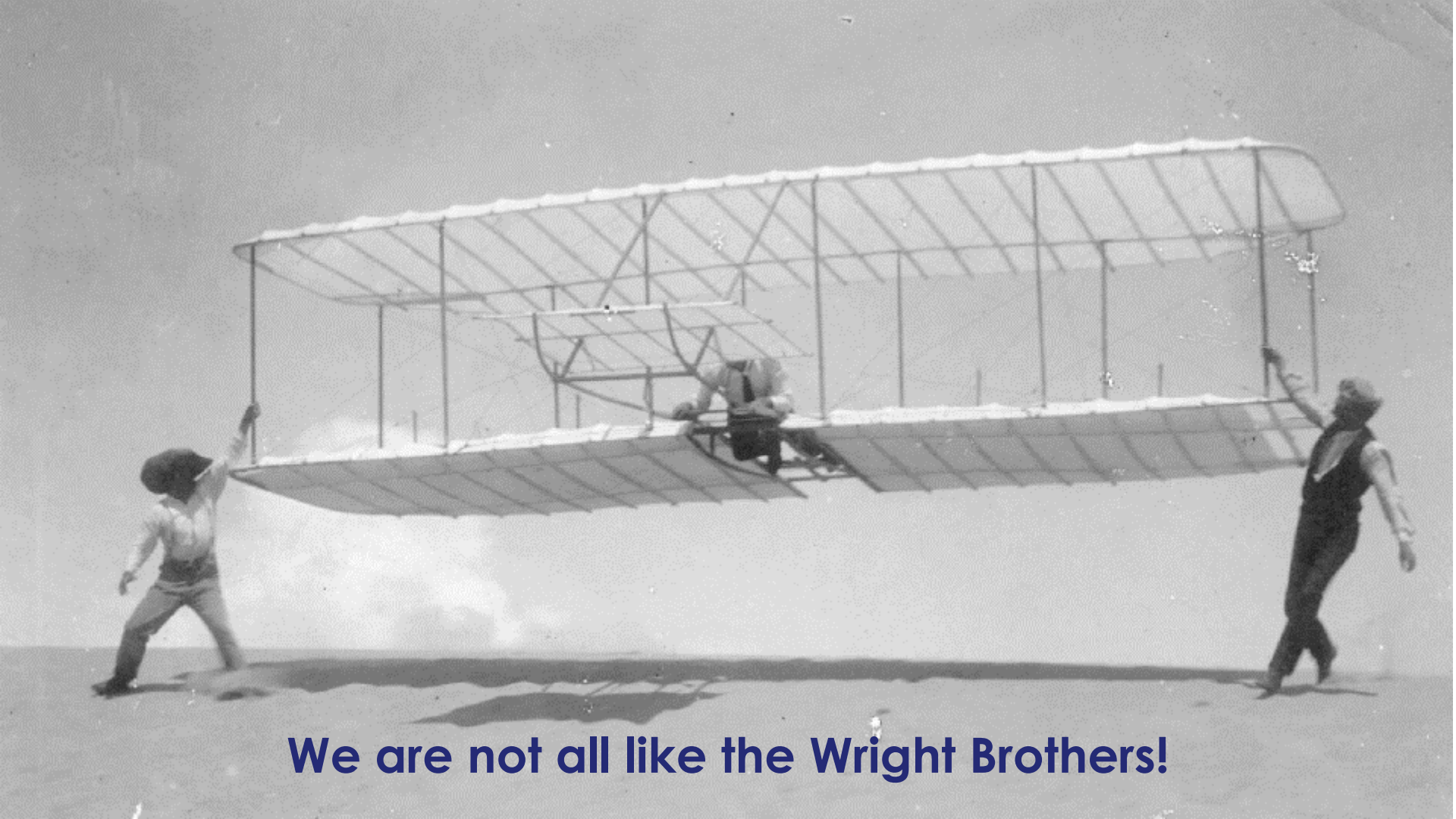
participants registered for the Capella Days 2020



Design requirements growth for aerospace vehicles.

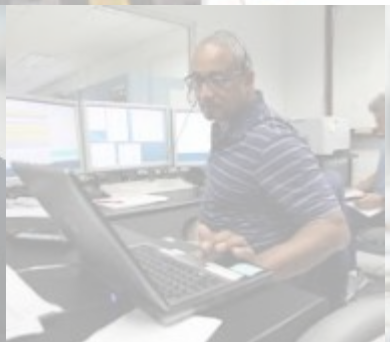
More recently – Cybersecurity, Resilience, User Experience, …

Increasing number of disciplines involved on complex systems engineering



We are not all like the Wright Brothers!

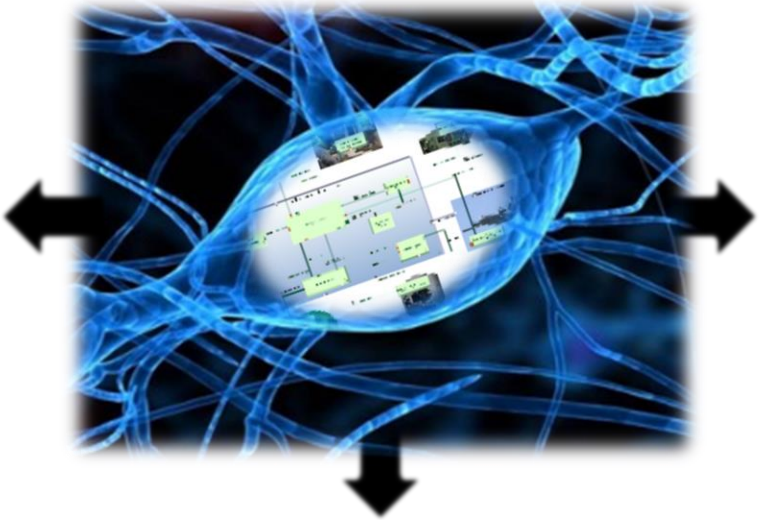
Complex Systems Engineering today requires a collective intelligence




Enable production/consumption of the right information
by the right person for the right use at the right time

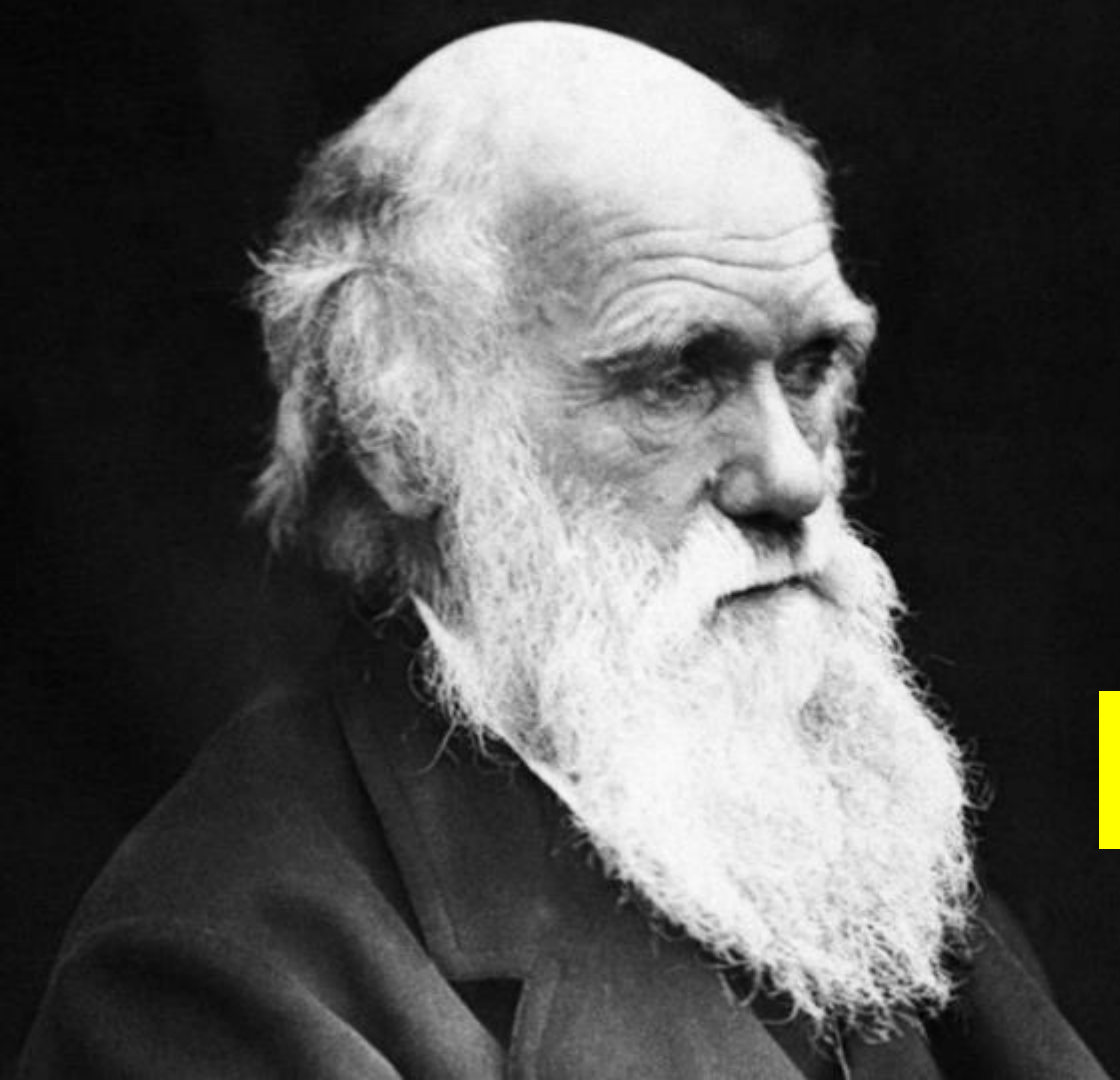
Models are key enablers of Digital Engineering

Provide consistent sources of truth



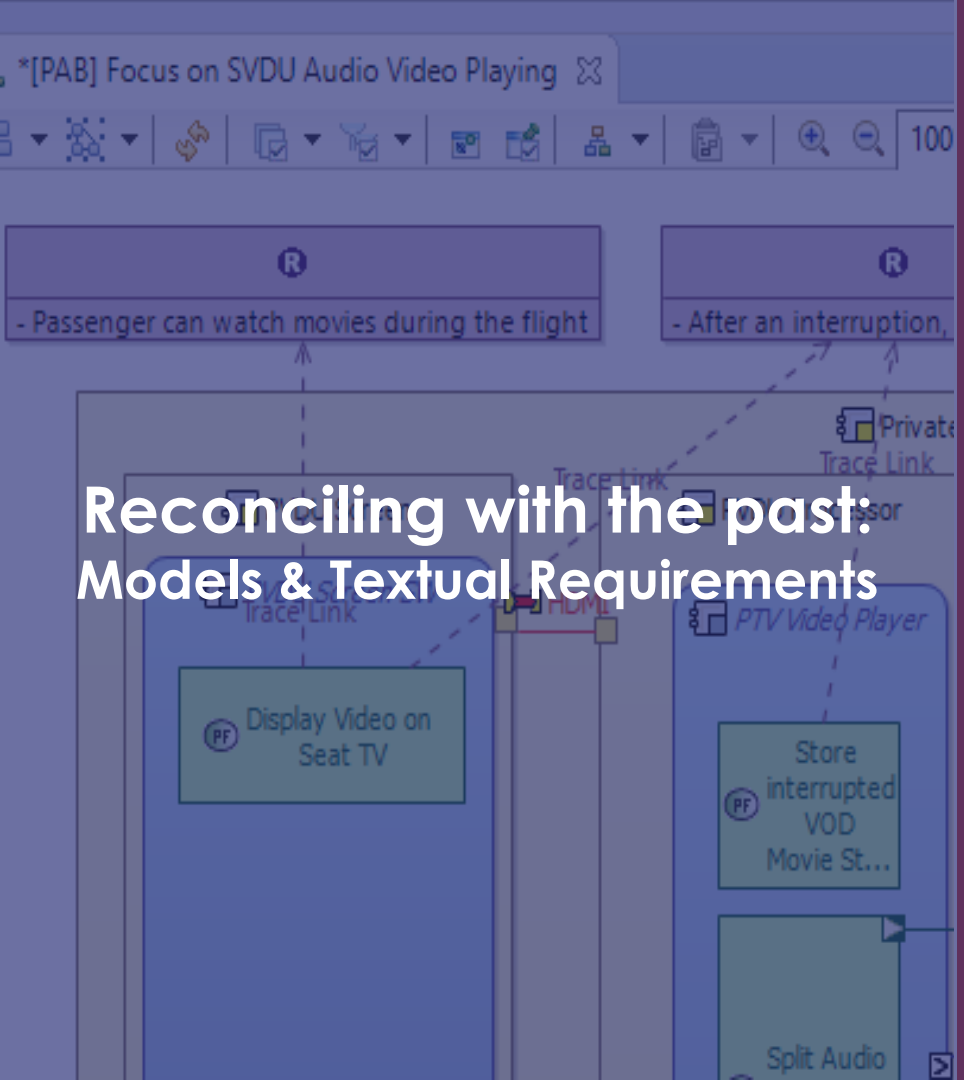
Enable communication and collaboration across stakeholders

Provide digital representations of the system across life-cycle stages

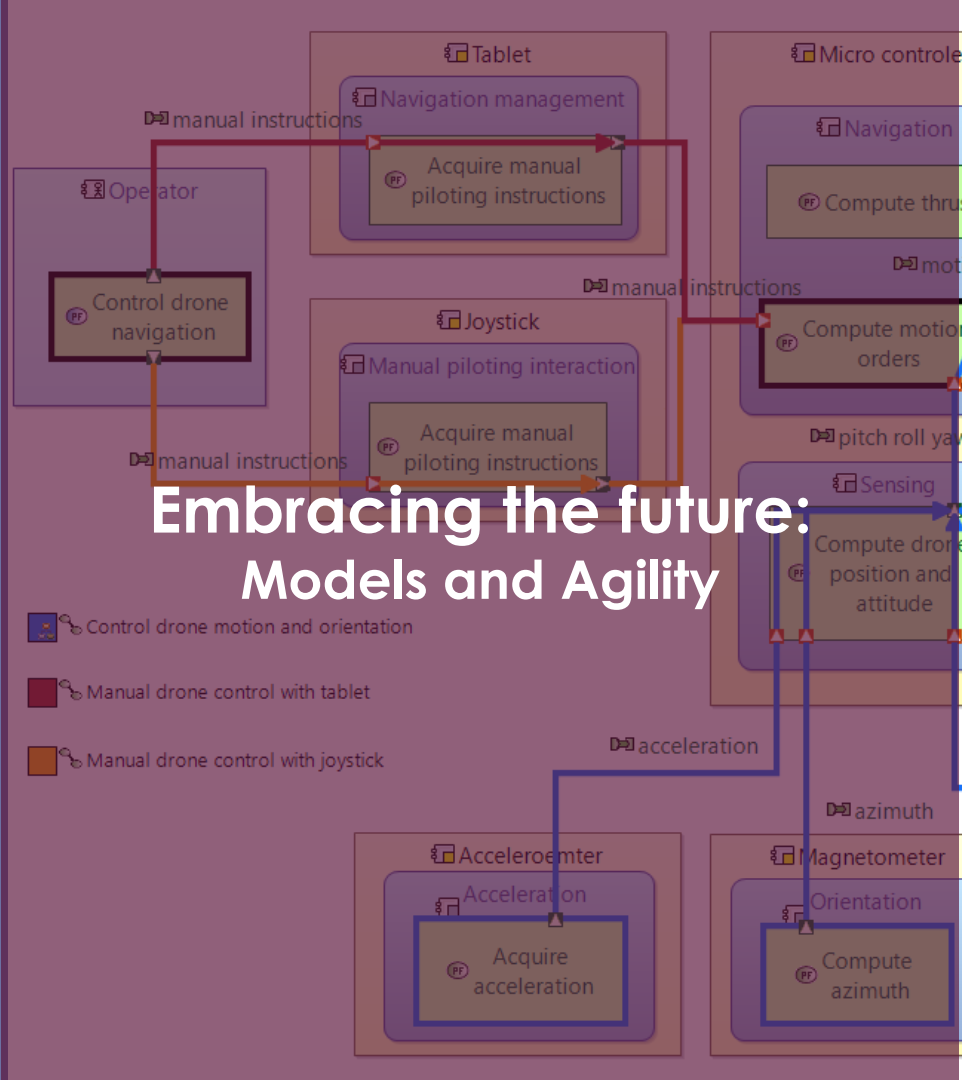


It is not the most intellectual of the species that survives; it is not the strongest that survives; but the species that survives is the one that is able best to **adapt and adjust to the changing environment** in which it finds itself

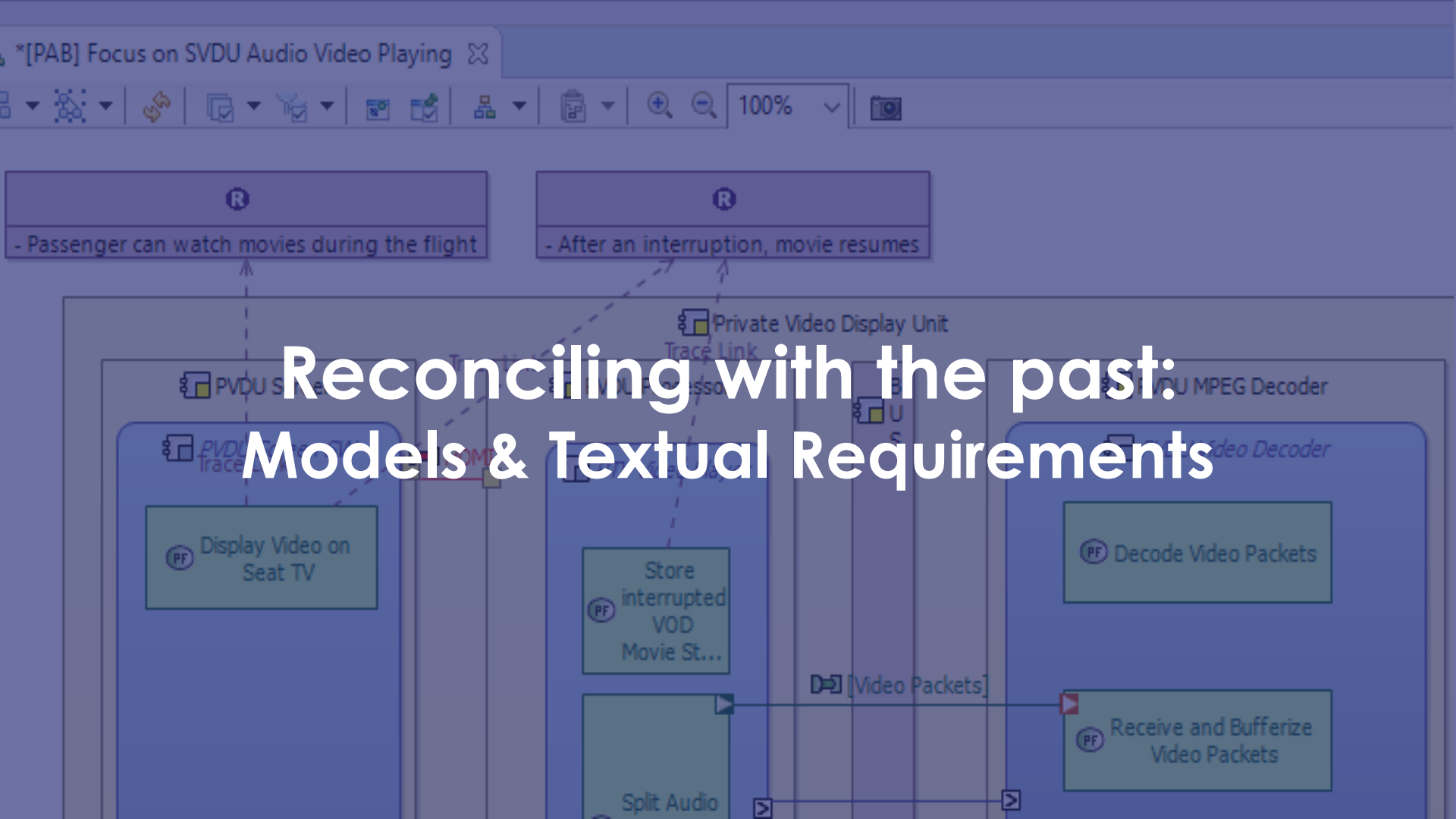

Reconciling with the past
Embracing the future



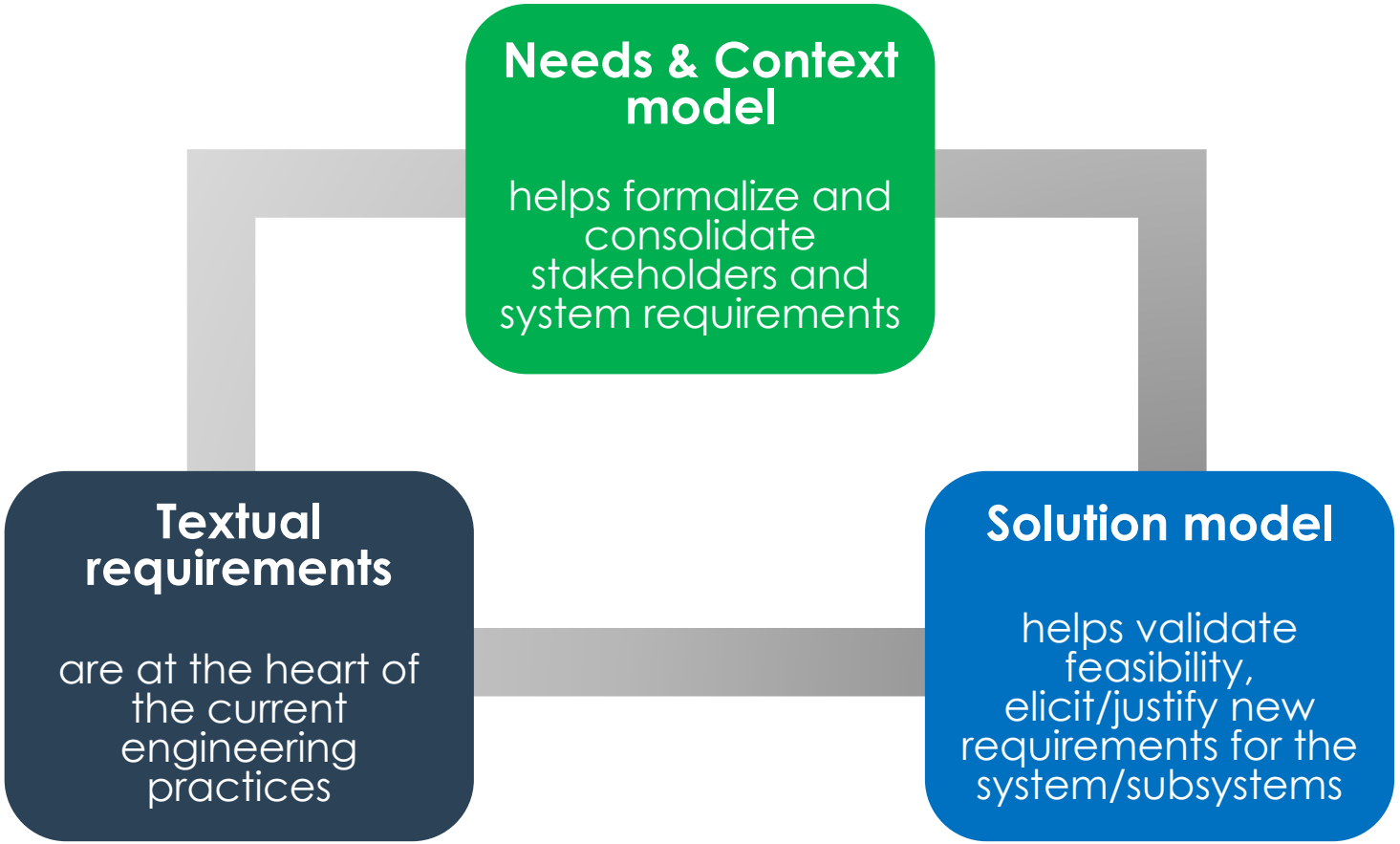
Reconciling with the past:
Models & Textual Requirements



Embracing the future:
Models and Agility



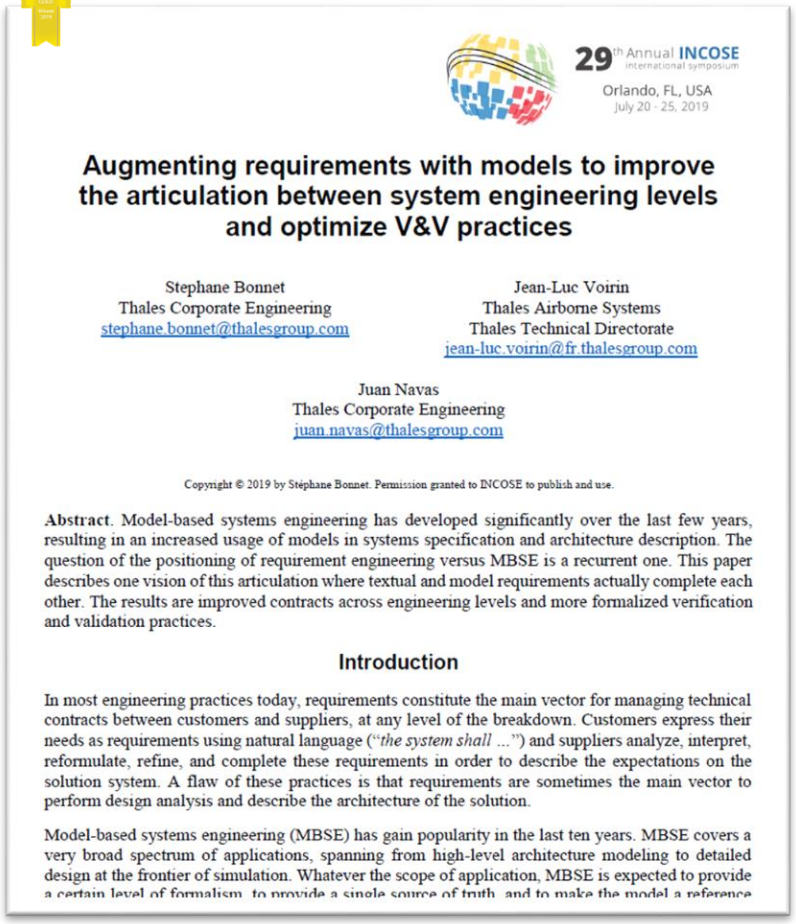
Reconciling with the past: Models & Textual Requirements


Needs & Context model
helps formalize and consolidate stakeholders and system requirements
Textual requirements
are at the heart of the current engineering practices
Solution model
helps validate feasibility, elicit/justify new requirements for the system/subsystems

Models add rigor to needs expression / solution description

Models can be processed to ensure completeness and consistency

… Why not considering that models ARE requirements?



29th Annual INCOSE international symposium
Orlando, FL, USA
July 20 - 25, 2019

Augmenting requirements with models to improve the articulation between system engineering levels and optimize V&V practices

Stephane Bonnet
Thales Corporate Engineering
stephane.bonnet@thalesgroup.com

Jean-Luc Voirin
Thales Airborne Systems
Thales Technical Directorate
jean-luc.voirin@fr.thalesgroup.com

Juan Navas
Thales Corporate Engineering
juan.navas@thalesgroup.com

Abstract. Model-based systems engineering has developed significantly over the last few years, resulting in an increased usage of models in systems specification and architecture description. The question of the positioning of requirement engineering versus MBSE is a recurrent one. This paper describes one vision of this articulation where textual and model requirements actually complete each other. The results are improved contracts across engineering levels and more formalized verification and validation practices.

Introduction

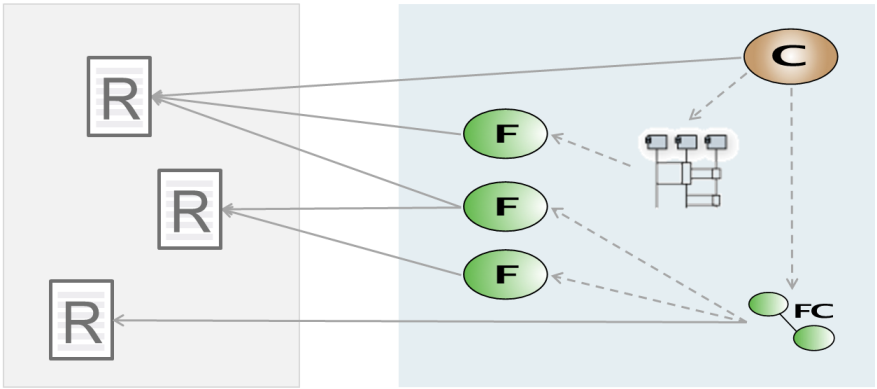
In most engineering practices today, requirements constitute the main vector for managing technical contracts between customers and suppliers, at any level of the breakdown. Customers express their needs as requirements using natural language (“*the system shall …*”) and suppliers analyze, interpret, reformulate, refine, and complete these requirements in order to describe the expectations on the solution system. A flaw of these practices is that requirements are sometimes the main vector to perform design analysis and describe the architecture of the solution.

Model-based systems engineering (MBSE) has gain popularity in the last ten years. MBSE covers a very broad spectrum of applications, spanning from high-level architecture modeling to detailed design at the frontier of simulation. Whatever the scope of application, MBSE is expected to provide a certain level of formalism, to provide a single source of truth, and to make the model a reference

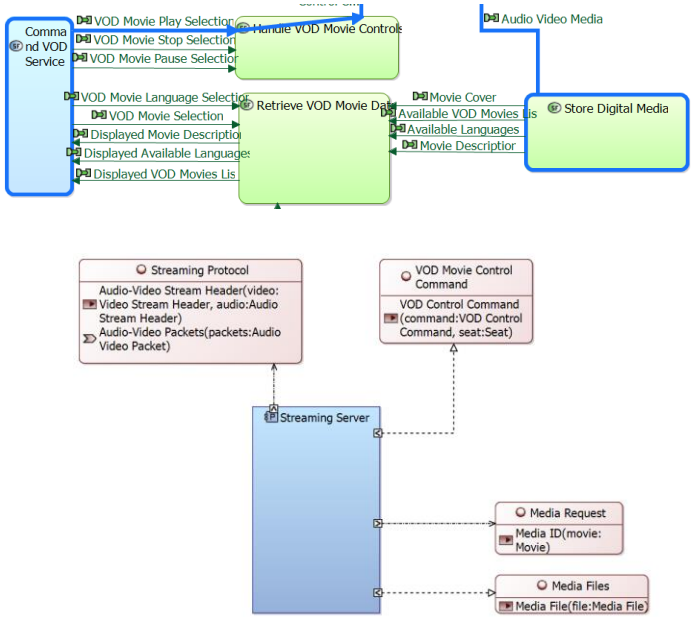
Requirements can either be textual “shall” statements, either model elements: textual and model requirements actually complete each other

Textual Requirements

Model ~~elements~~ Requirements

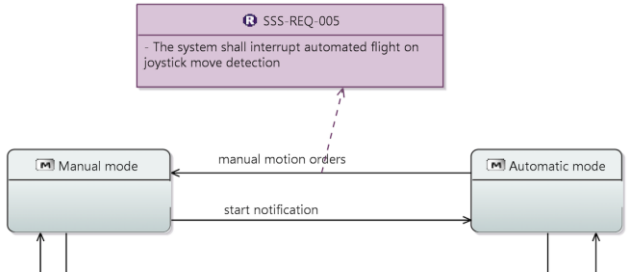


Model requirements

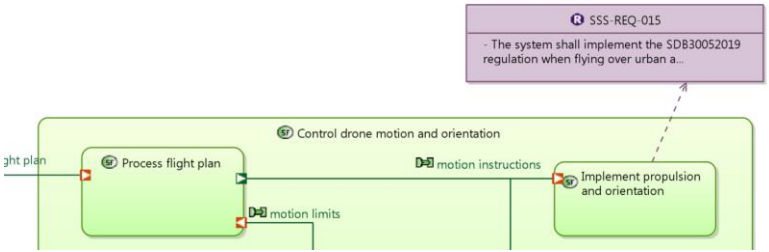


Some of the Arcadia concepts can be considered as **Functional** and **Interfaces** requirements, eventually with related **Performance** requirements.

Model + textual requirements

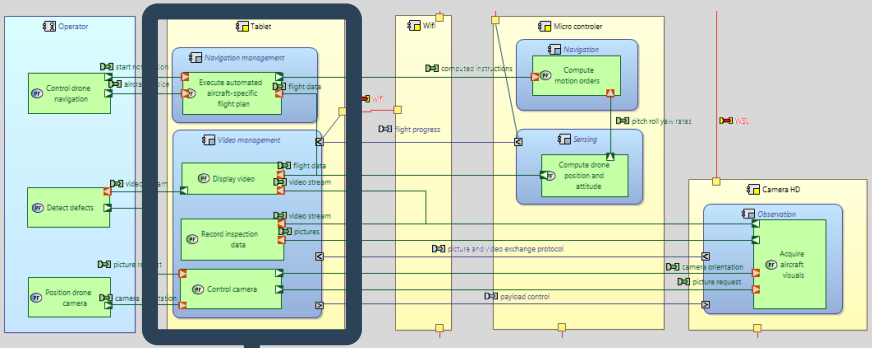


Models formalize **stakeholders** requirements

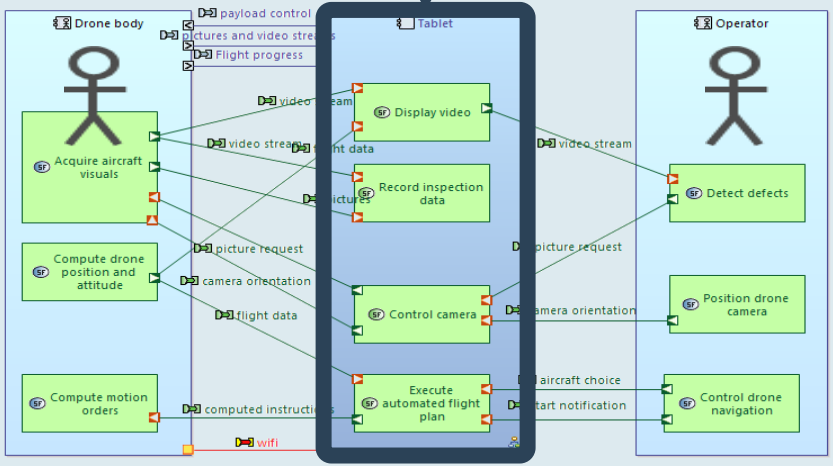


Some expectations (**Environmental**, **Regulations**, etc.) are easier to express with textual descriptions with traceability links to model elements

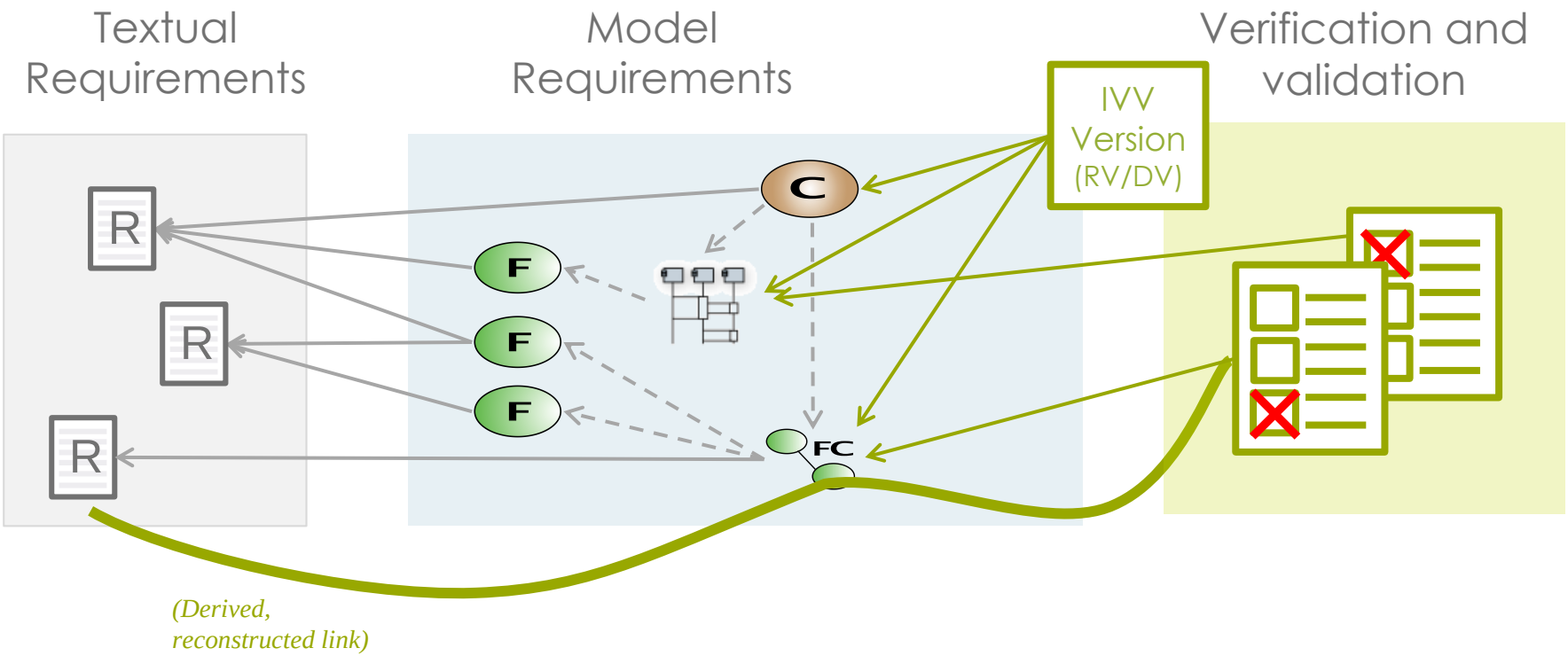
N-Level:
Tablet is a constituent of a drone-based system



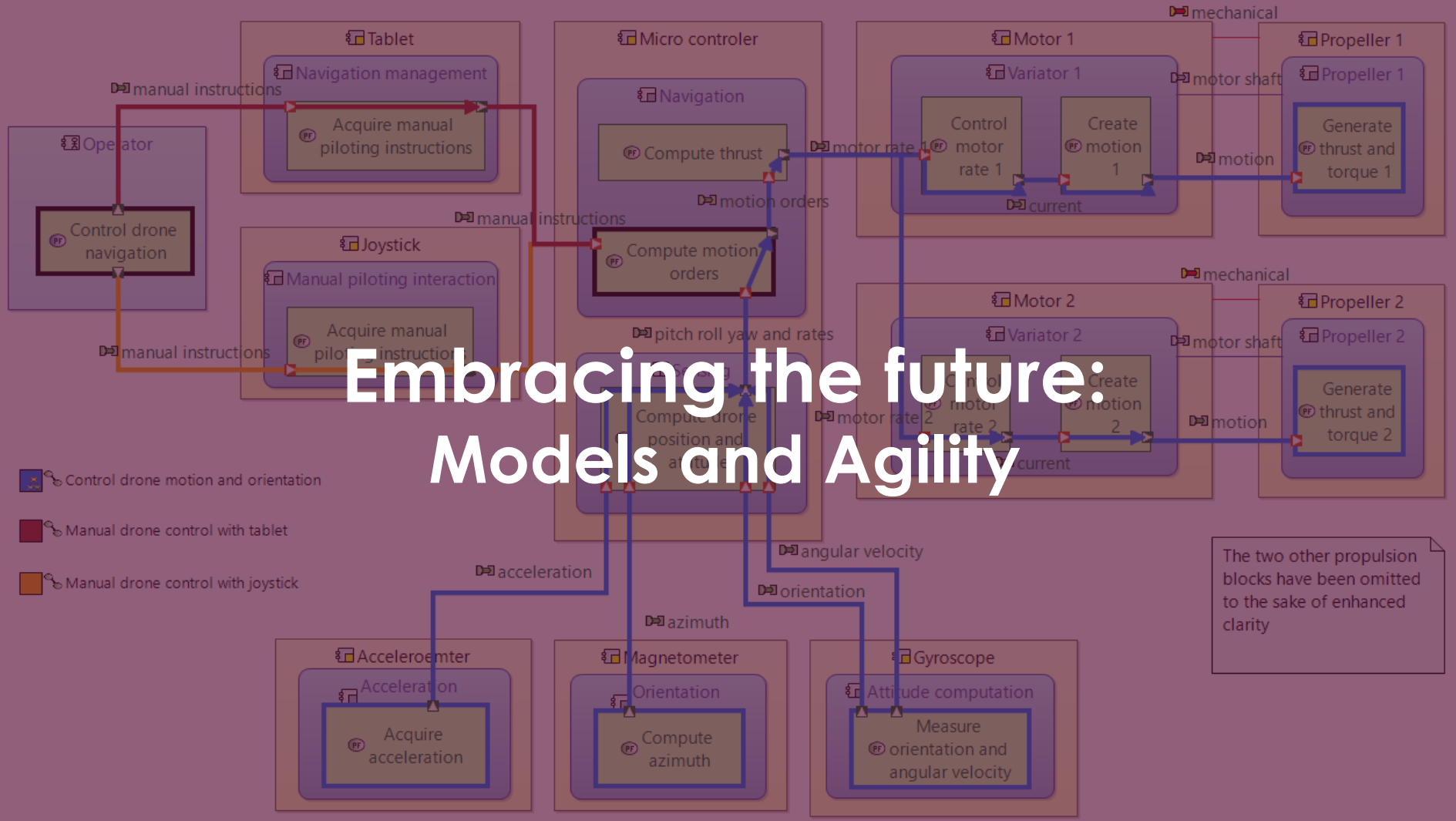
N+1 Level:
Tablet is the (sub)system of interest



Models + Textual requirements bring clarity and rigor to "contracts" between engineering levels



Models + Textual requirements enable a better coordination and planning of IVV activities

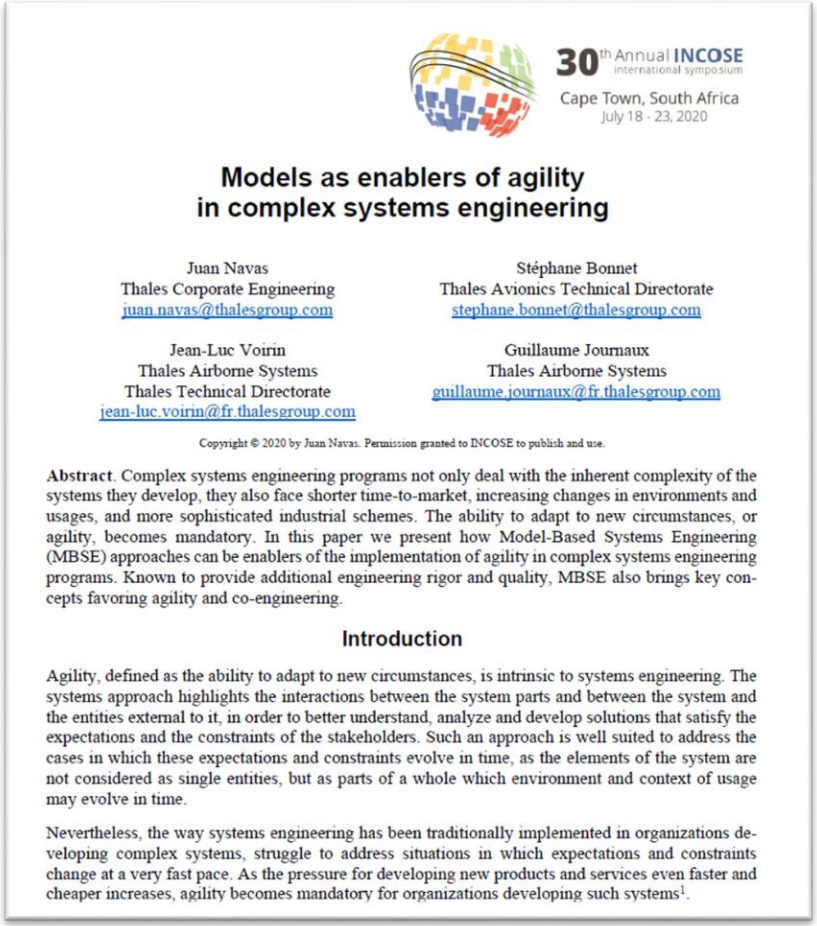


Embracing the future:
Models and Agility

Models add rigor to needs expression / solution description

Agility on systems engineering is required to cope with customers' expectations

… Why not implementing Model-Based Agility?



30th Annual INCOSE international symposium
Cape Town, South Africa
July 18 - 23, 2020

Models as enablers of agility in complex systems engineering

Juan Navas
Thales Corporate Engineering
juan.navas@thalesgroup.com

Stéphane Bonnet
Thales Avionics Technical Directorate
stephane.bonnet@thalesgroup.com

Jean-Luc Voirin
Thales Airborne Systems
Thales Technical Directorate
jean-luc.voirin@fr.thalesgroup.com

Guillaume Journaux
Thales Airborne Systems
guillaume.journaux@fr.thalesgroup.com

Abstract. Complex systems engineering programs not only deal with the inherent complexity of the systems they develop, they also face shorter time-to-market, increasing changes in environments and usages, and more sophisticated industrial schemes. The ability to adapt to new circumstances, or agility, becomes mandatory. In this paper we present how Model-Based Systems Engineering (MBSE) approaches can be enablers of the implementation of agility in complex systems engineering programs. Known to provide additional engineering rigor and quality, MBSE also brings key concepts favoring agility and co-engineering.

Introduction

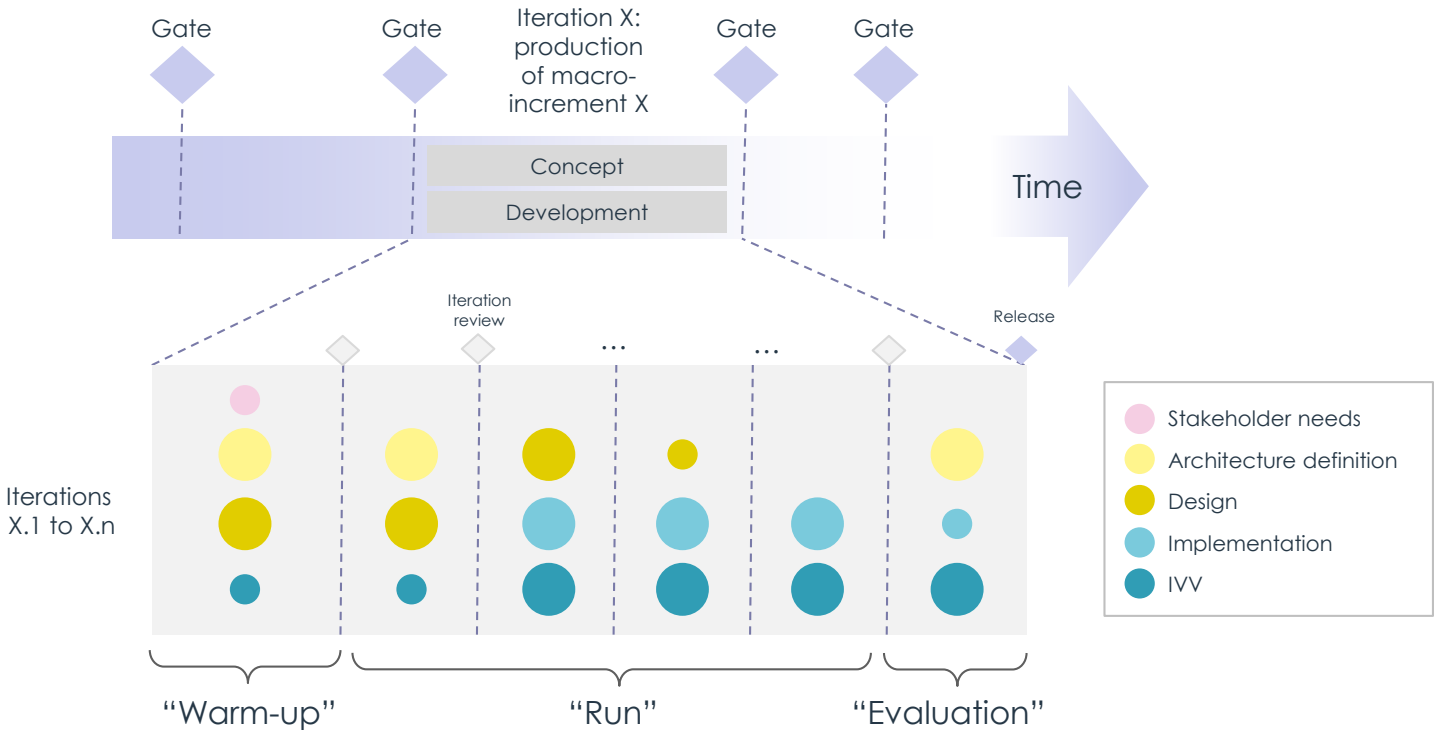
Agility, defined as the ability to adapt to new circumstances, is intrinsic to systems engineering. The systems approach highlights the interactions between the system parts and between the system and the entities external to it, in order to better understand, analyze and develop solutions that satisfy the expectations and the constraints of the stakeholders. Such an approach is well suited to address the cases in which these expectations and constraints evolve in time, as the elements of the system are not considered as single entities, but as parts of a whole which environment and context of usage may evolve in time.

Nevertheless, the way systems engineering has been traditionally implemented in organizations developing complex systems, struggle to address situations in which expectations and constraints change at a very fast pace. As the pressure for developing new products and services even faster and cheaper increases, agility becomes mandatory for organizations developing such systems[1].

Model-based practices are effective enablers of systems engineering agility

Build the solution in an incremental way based on value creation, using system-level **Capabilities** and end-to-end **Functional Chains** and **Scenarios**

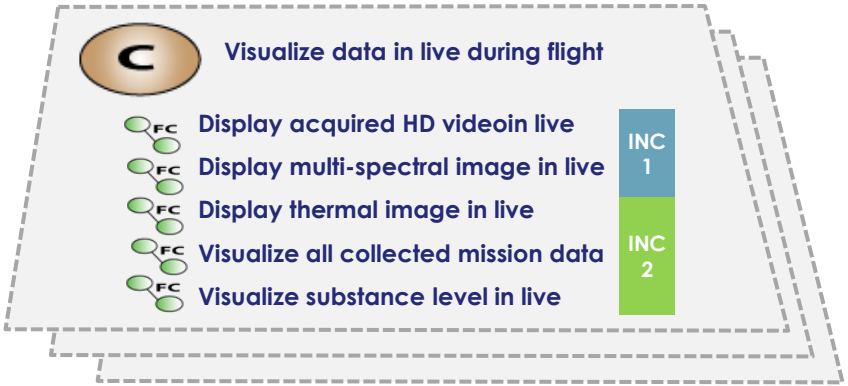
Between 2 Gates, teams go through phases that can be iterated: increments at the team level



Warm-up - collaborative definition of the detailed scope, goals an schedule of the increment and of the necessary resources

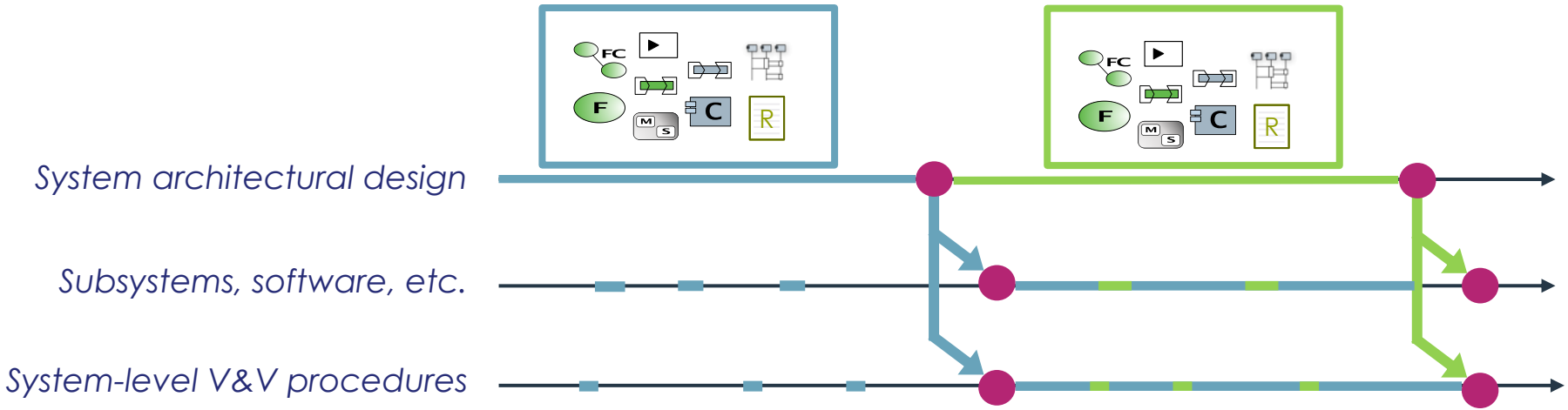
Run - iterative effort punctuated by iteration reviews

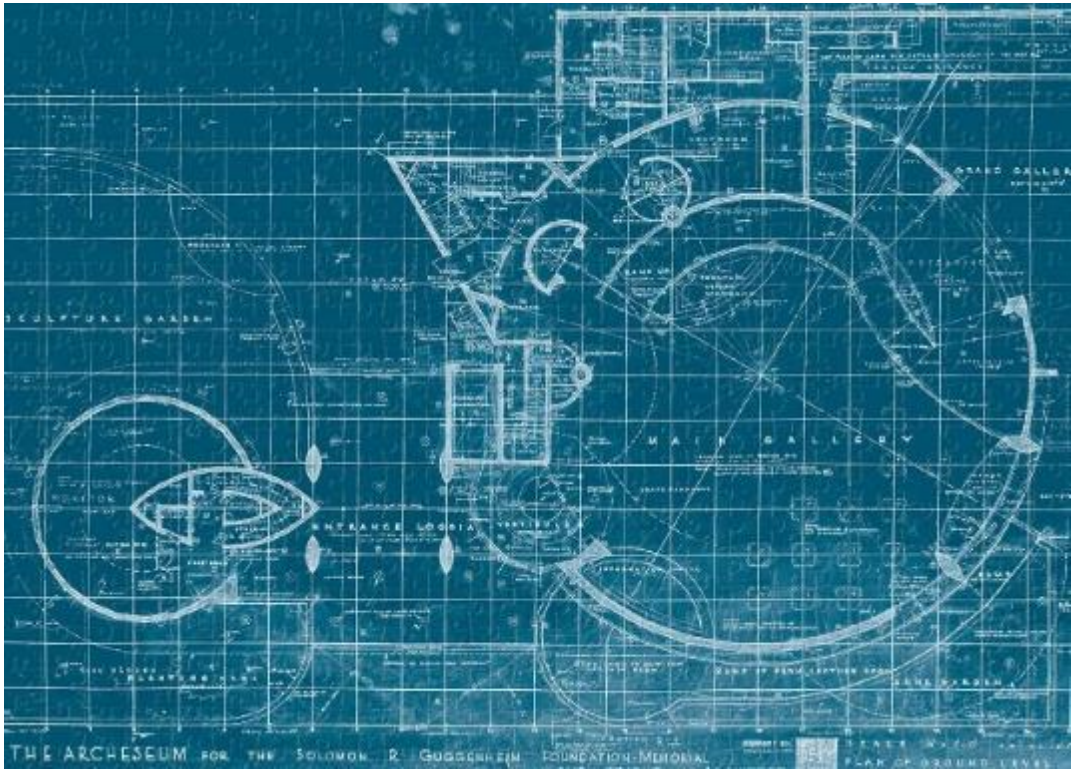
Evaluate - assess how the engineering was performed, that the expected outcomes are there and that conditions for pursuing are met



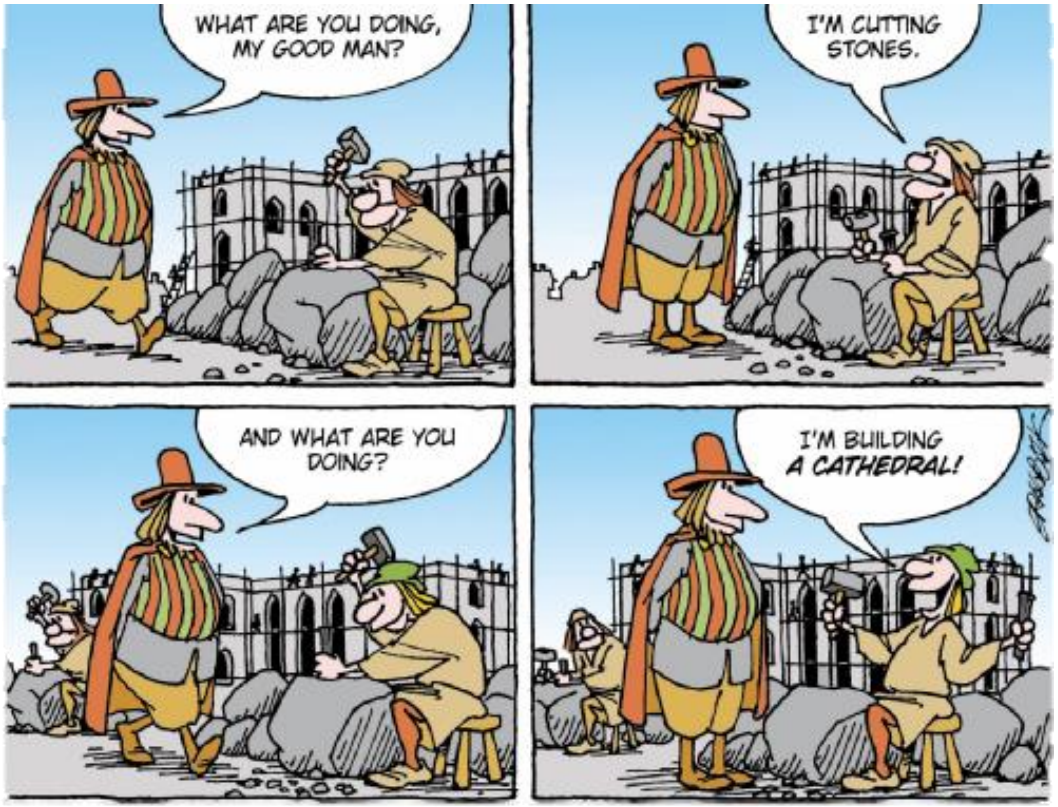
Definition of increments with expected Functional Chains

Vertical slices of architectural design across need and solution models





Better collaboration, organization and progress monitoring by using architectural models as the common “blueprint”



Model elements such as Capabilities and Functional Chains provide meaning to what SW developers are doing


The road to Digital Engineering is in front of us

Tomorrow, Tuesday October 13th, 4:45 pm CET: