Capella Days

Stairway to heaven: Climbing the very first steps

Stéphane Bonnet – Systems Design Authority – Thales Avionics
November 2022

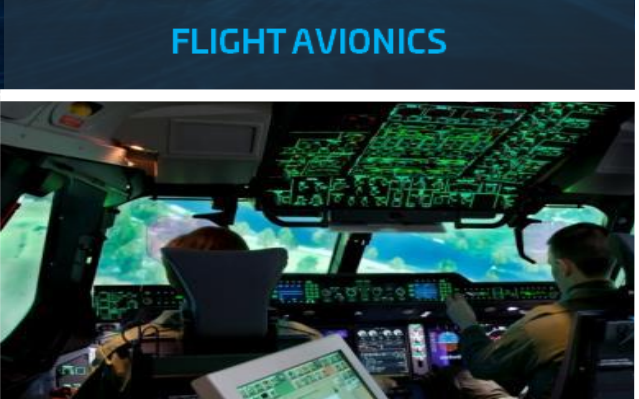
Where does the staircase start?

Strategy for climbing the very first steps

- Strengthen the basics
- Demystify both MBSE and Capella
- Promote quick wins
- Adopt a pragmatic approach

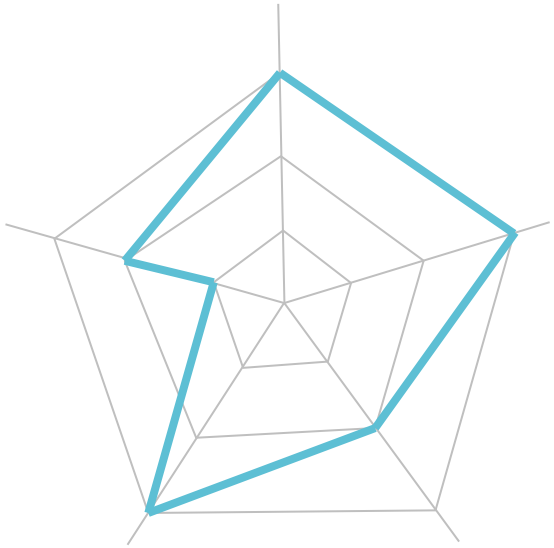
Continuous improvement is better than delayed perfection(*)

(*) Mark Twain



OPEN

THALES
Building a future we can all trust



OPEN

Initial context

- Avionics domain is changing rapidly and significantly
- Need to build lean, adjusted solutions

- Strong development assurance culture
- Strong expertise in tooling development

- Sophisticated MBSE implementation failed to inspire

Where does the staircase start?

Strategy for climbing the very first steps

Strengthen the basics

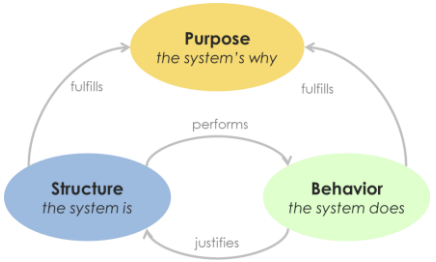
Demystify both MBSE and Capella

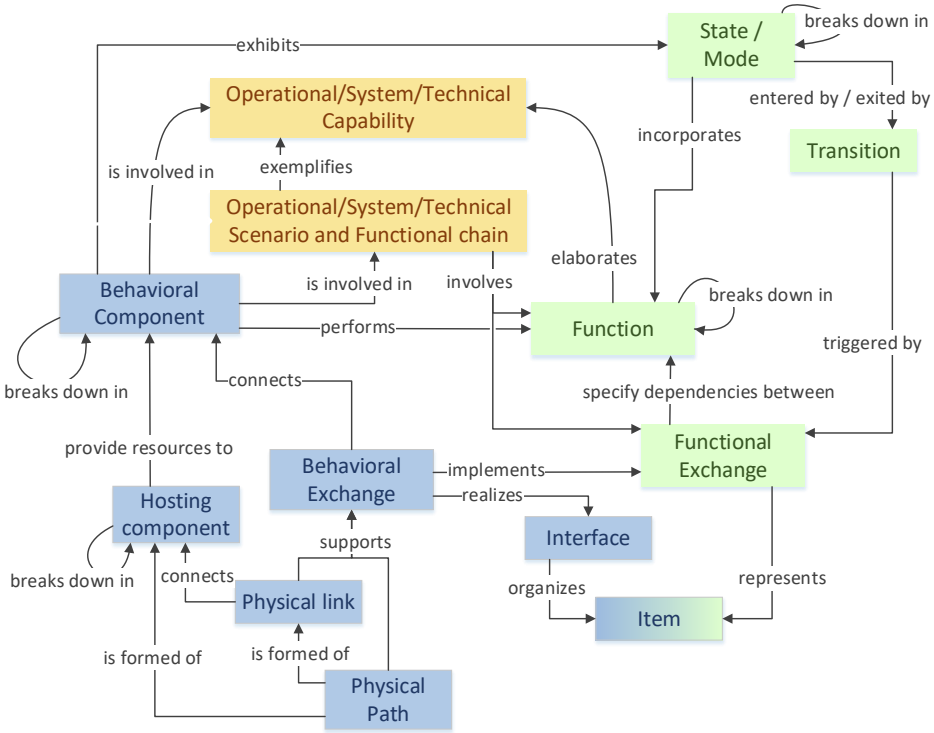
Promote quick wins

Adopt a pragmatic approach

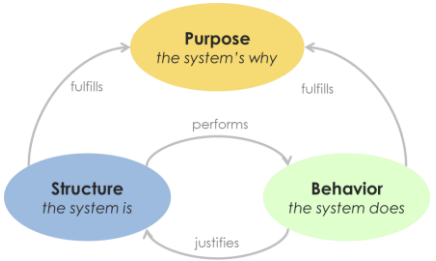
Continuous improvement is better than delayed perfection

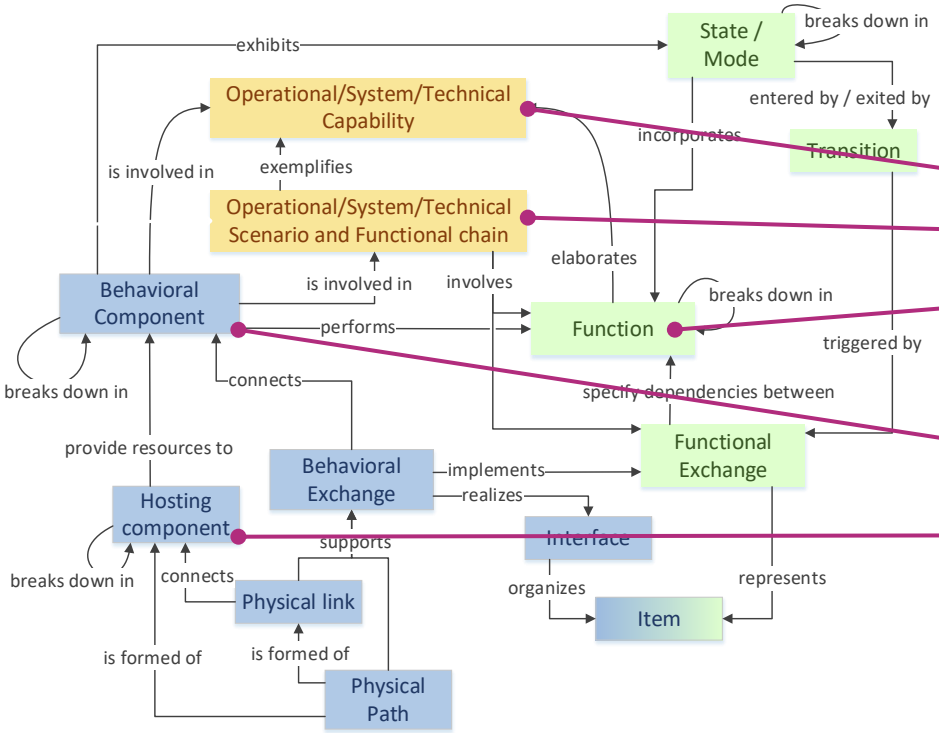
Focus on vocabulary

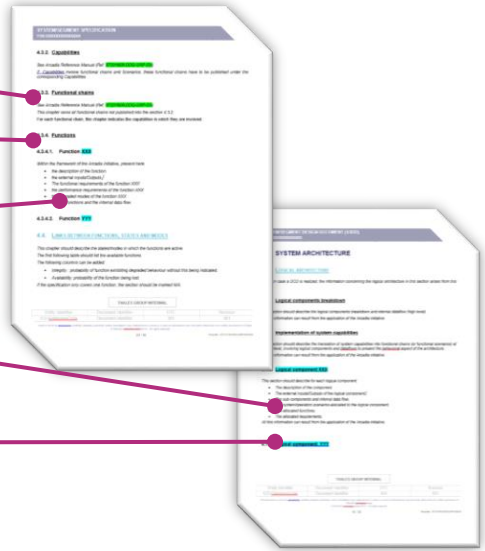




Focus on vocabulary

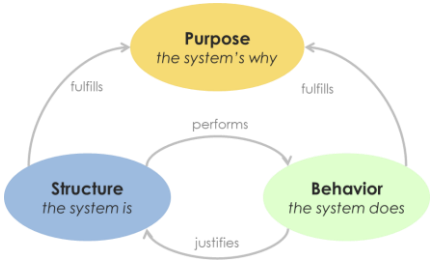






THALES
Building a future we can all trust

Focus on vocabulary



SYSTEM INCREMENT

is defined by a set of

Scope of the ontology

exhibits

State / Mode

breaks down in

entered by / exited by

Transition

incorporates

Operational/System/Technical Capability

is involved in

exemplifies

Operational/System/Technical Scenario and Functional chain

elaborates

involves

is involved in

Behavioral Component

performs

Function

breaks down in

triggered by

breaks down in

connects

specify dependencies between

Functional Exchange

provide resources to

Behavioral Exchange

implements

realizes

Hosting component

Interface

supports

breaks down in

connects

Physical link

organizes

represents

Item

is formed of

is formed of

Physical Path

SW EPIC for articulation with development

is aligned with

is aligned with

TEST PROCEDURE for articulation with V&V

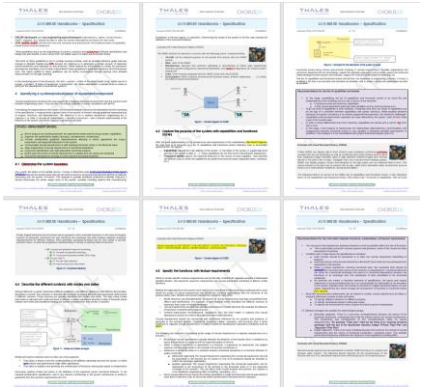
further specifies / constrains

TEXTUAL REQUIREMENT

THALES

Building a future we can all trust

Vocabulary material:
glossary, examples



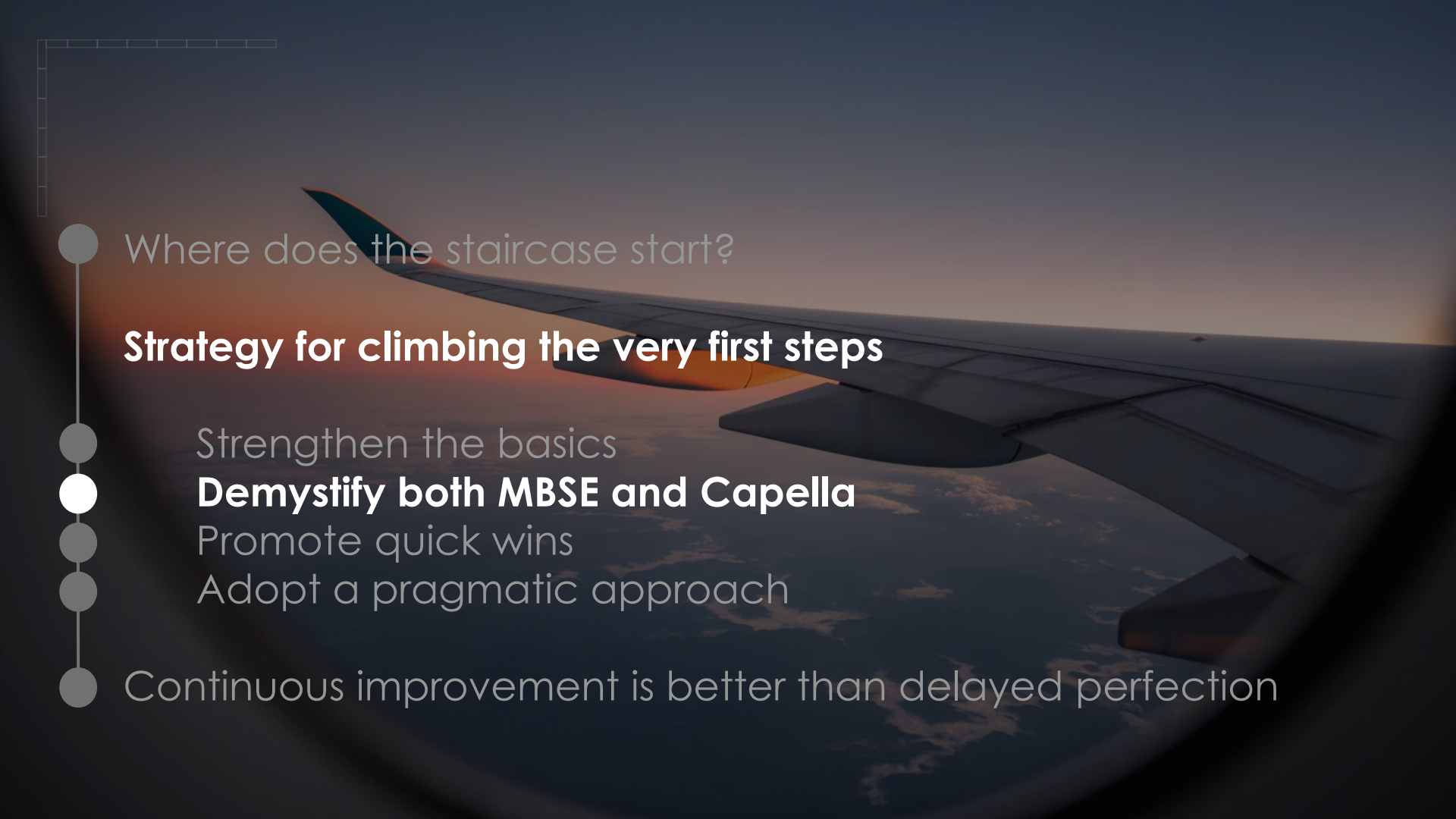
MBSE Handbooks for
Avionics GBU

Webinars
&
Mini-trainings

R&T
Acceptable
means of
compliance

OPEN

Raising
awareness



Where does the staircase start?

Strategy for climbing the very first steps

- Strengthen the basics
- **Demystify both MBSE and Capella**
- Promote quick wins
- Adopt a pragmatic approach

Continuous improvement is better than delayed perfection

Changing the narrative

#1 – **Implement and respect the vocabulary** for system purpose and architecture

#2 – Guarantee **consistency** across diagrams

#3 – **Automatically produce** diagrams

#4 – Automatically produce and maintain **simplified views**

#5 – **Validate** the model

#6 – Properly **characterize** elements

#7 – Write and **execute queries, extract tables**

#8 – Master **variability**

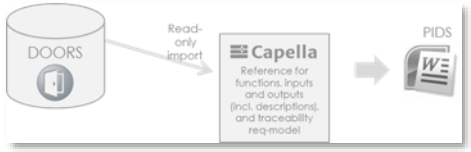
#9 – **Generate** Word documents and HTML websites

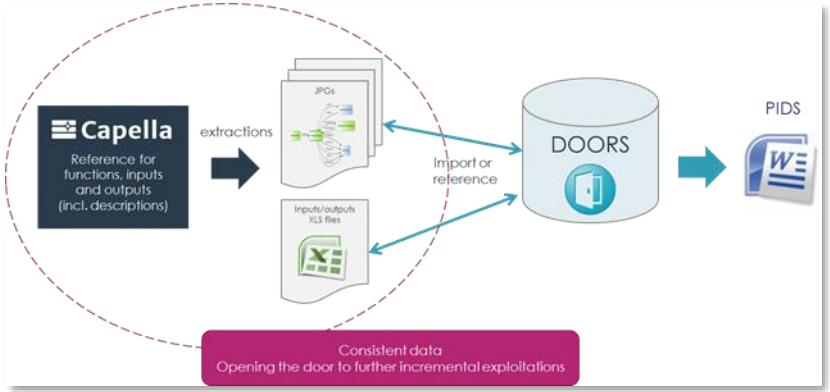
#10 – Work **collaboratively** with tracking of history

Changing the narrative

Current approach

- Textual requirements in Doors
- Inputs/outputs interface tables in Doors
- VISIO diagrams referenced from DOORS with OLE links
- Documentation production from DOORS
- Reviews in Word? In Doors?





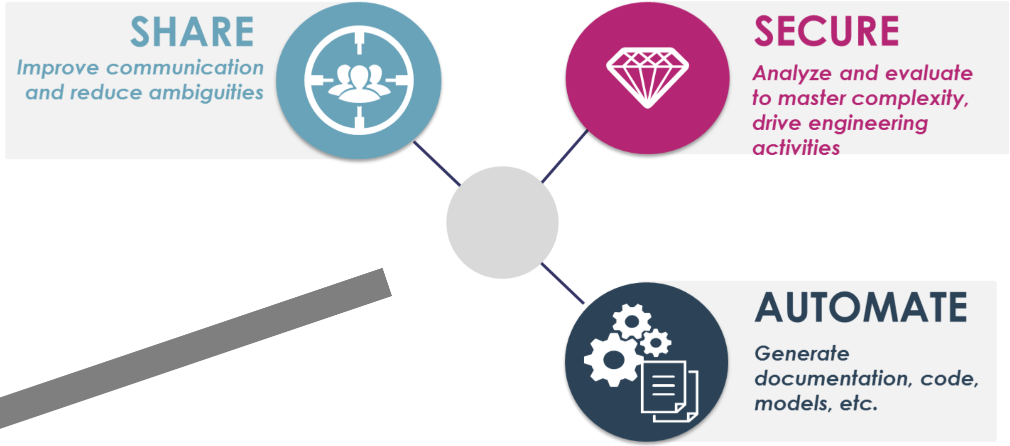
Layout irrelevant

1. Create a new function, describe it
2. Create exchanges to/from existing functions, give label names
3. Create or reuse item flows, describe them in a table for example)
4. Associate exchanges to items flows
5. Visualize generated diagrams
6. Run queries to produce interface tables

A 2-3 hours tool training of the different contributors would be sufficient for this very concrete objective

The collaborative Capella VM is already available for use

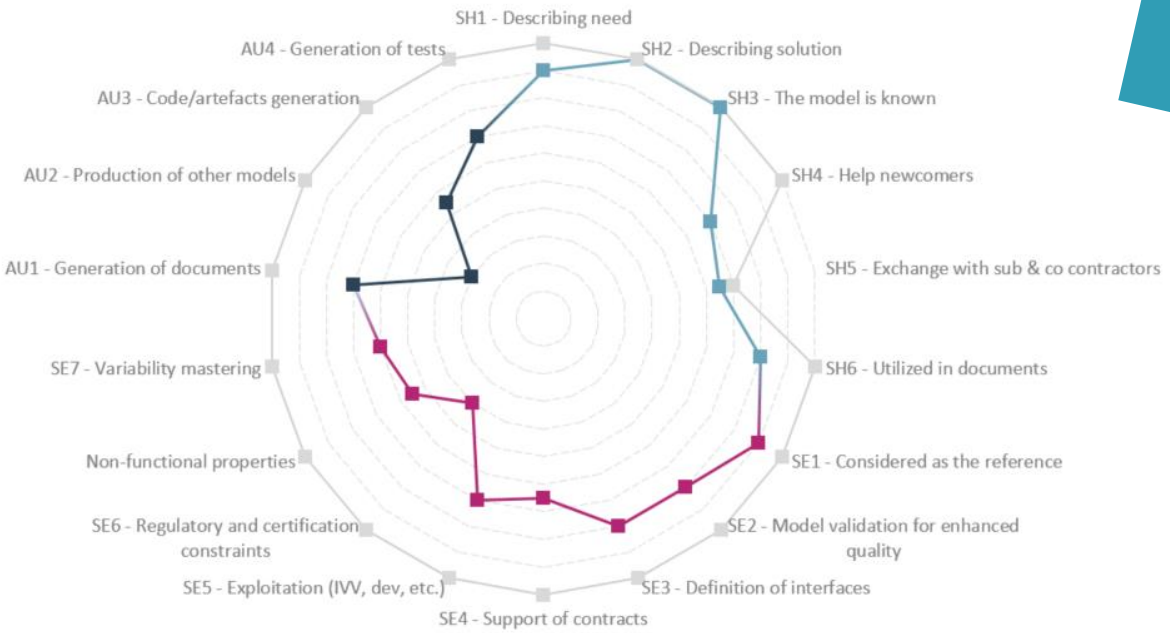
Framework for model-based engineering objectives



	SHARE	SECURE	AUTOMATE
1. UNDERSTAND NEED AND CONTEXT	Architecture Framework / Arcadia OA/SA models describe (some of) the **expectations and needs of stakeholders**. They are known and accepted by all engineers and are used in contractual documentation on ad hoc basis.	Architecture Framework / Arcadia OA/SA models **cover all stakeholder expectations and needs** of a given scope. They are possibly related to value analysis. **Completeness and consistence** of the need understanding is enforced by **model analysis and validation**.	**OCD are (at least partially) generated** from models. Models from the customer are possibly imported.
2. TRANSLATE NEEDS INTO REQUIREMENTS	Arcadia SA model **describes** the solution **needs/constraints** and **boundaries/interfaces**. It is known by most engineers and guarantees a **common understanding**. It is used in contractual documentation on ad hoc basis.	Arcadia SA model covering **needs/constraints** and **boundaries/interfaces** is part of the solution requirements baseline. It is rigorously **traced to OA model and textual requirements**. **Completeness and consistence** of the need formalization is enforced by **model analysis and validation**.	**SSS/interface documents is (at least partially) generated** from the Arcadia SA models.
3. DESIGN THE ARCHITECTURE	Arcadia LA/PA models describe the architectural design: **functional expectations** from each solution constituent, interfaces, functional chains. They provide a **common understanding of the chosen solution**, are known by all stakeholders, and are used in documentation on ad hoc basis.	Arcadia LA/PA models describing the architecture are exploited to **guarantee the consistence and completeness of the design**. They are used to consider **alternatives** and support **early evaluation** (sizing assumptions, performance, cost, etc.). Model and textual requirements are rigorously articulated (derivation, justification, etc.).	**SSDD/ICD are (at least partially) generated** from the Arcadia LA/PA models. Engineering data is extracted from the model to feed engineering specialties models and vice versa. Early design validation is performed with **simulation techniques**.
4. DEFINE THE APPROPRIATE REUSE	Models of reusable assets and of product architecture exist and are known. They are mainly **used for documentation** purpose. Feature models describe the product variability and standard configurations	**Models of reusable assets are rigorously governed** and managed in configuration. They are assembled in solution architecture models. **Architecture models are mapped to feature models**, they help **capture, strengthen, and optimize** product variability and configurations.	Project models are **automatically initialized or derived** from feature models and architecture models
5. MAKE YOUR PRODUCT MAINTAINABLE AND MANUFACTURABLE	Solution architecture/design models are challenged by multiple stakeholders with focuses on different topics (i.e. RAMT, among others)	RAMT analysis dedicated models are up-to date and fully consistent with solution architecture/design models.	RAMT analysis dedicated models are **initialized from** and **kept consistent** with solution architecture/design models.
6. ANTICIPATE AND PERFORM IVV	Functional chains described in Arcadia SA/LA/PA models are used as **inspiration** for **test procedures** and IVV **releases content**	**IVV strategy is built consistently with model capabilities**. Test procedures are aligned on functional chains and scenarios. Models are exploited to **build integration trees** and **define integration configurations**.	Models are used to **generate solution test elements**: test scenarios, pieces of code, specification of test means, etc.
7. SAFE AND CYBER SECURED SOLUTION	**Models are used during technical discussions** between system/SW/HW engineers and Safety and Cybersecurity specialists	Models **are the reference for systems/SW/HW and safety/cybersecurity co-engineering**, used to identify and evaluate safety-critical functional chains, threats and assets. The consistency of models guarantee the integration of safety and cybersecurity concerns "by design"	Safety and Cybersecurity-dedicated models are **initialized from** and **kept consistent** with solution architecture models.
8. MASTER HW DEVELOPEMENT	2D/3D views coming from the CAD tools are used to **document the HW design**. 2D synoptics block diagrams are used to **describe HW structures and physical flows**.	1D/2D/3D design models are used to create simulation models that are used to perform multiphysics simulations in order to **guarantee the compliance of the HW design** with functional requirements and applied standards.	2D/3D design models are used to **produce** parts of the datapackage. Results of multiphysics **simulations** are part of the DJF.
9. MASTER SW DEVELOPMENT	Arcadia/UML/CCM/Simulink models are used to **document the SW design**	Arcadia/UML/CCM/Simulink models (functional analysis, state machines, algorithms, component interfaces and data, etc.) are rigorously built and maintained. Consistency of models and SW implementation can be demonstrated.	Design SW models are **simulated**, SW artefacts are **generated** (documentation, interfaces, skeletons with variable content, configuration and deployment files, database schemas, etc.)
10. BUILD THE DEVELOPMENT STRATEGY	Models help share a **common understanding of the PBS** (functional expectations from each constituent, interfaces, etc.)	Models **support the definition of the release plan and secure its execution** (impact analysis, effort sizing, maturity evaluation, etc.). Models play a **structuring role in the development lifecycle**, with capabilities, scenarios, and functional chains enabling an **optimum coordination between team deliveries**	NA
13. MASTER CO- AND SUB-CONTRACTING	**Architecture models at solution level are known from co- and sub-contractors**. They are used in contractual documents on ad hoc basis.	Architecture models at solution level are the **reference for co-engineering**. Their consistency guarantees the **successful integration of solution constituents**.	Architecture models or design models of subsystems, SW or HW components are **initialized from** and **kept consistent** with solution architecture models.

Relate MBSE deployment to more global engineering improvement actions

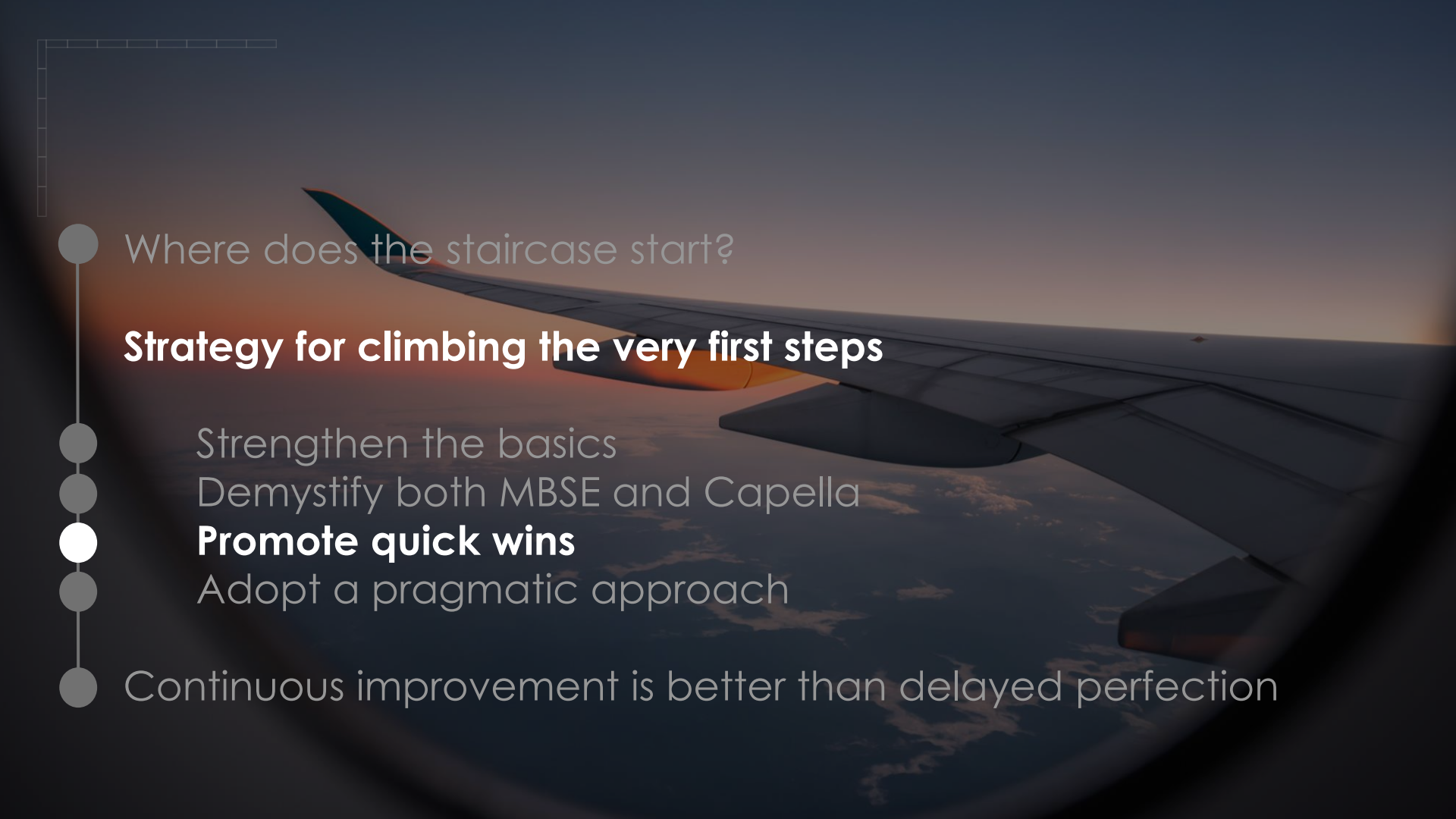
Framing the exploitation of models



Values have been changed, they do not represent the actual MBSE footprint of Thales Avionics

Exploitation of models: Typical patterns

Formalizing the descriptions of the system	Model as a driver of (rigorous) engineering activities	Model-centric, automation, variability mastering



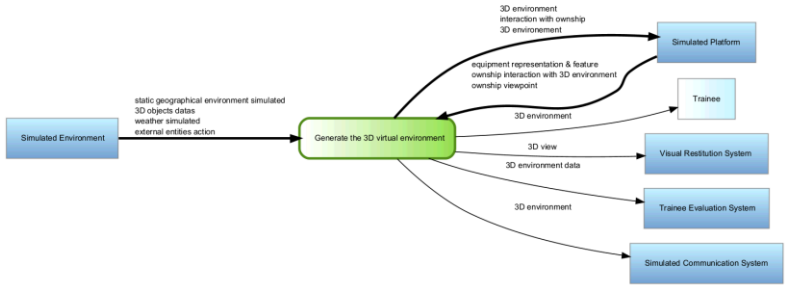
Where does the staircase start?

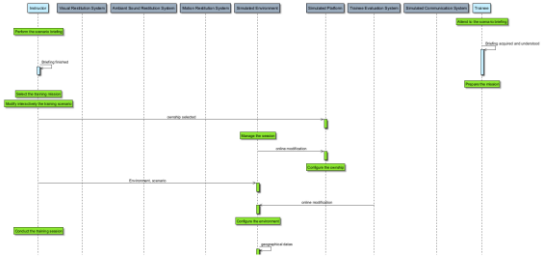
Strategy for climbing the very first steps

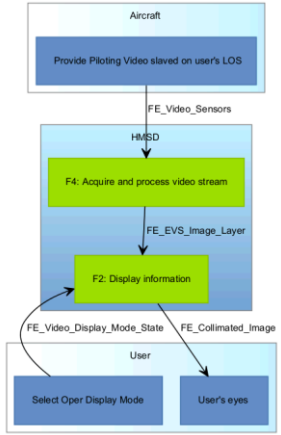
Strengthen the basics
Demystify both MBSE and Capella
Promote quick wins
Adopt a pragmatic approach

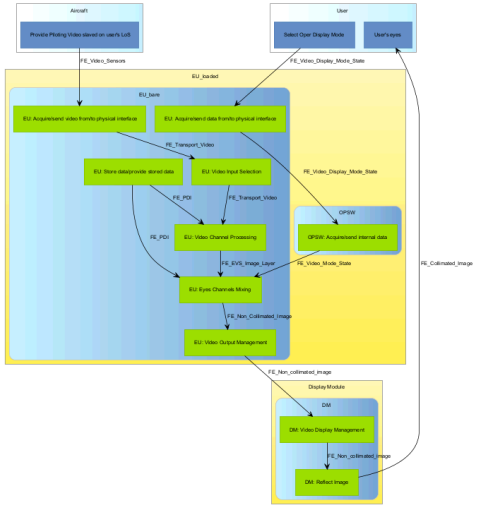
Continuous improvement is better than delayed perfection

1. **DDV – Dynamic Diagram Viewer**
2. Requirements
3. Documentation generation
4. PVMT – Annotating models
5. Queries - Extracting data
6. Articulation with software



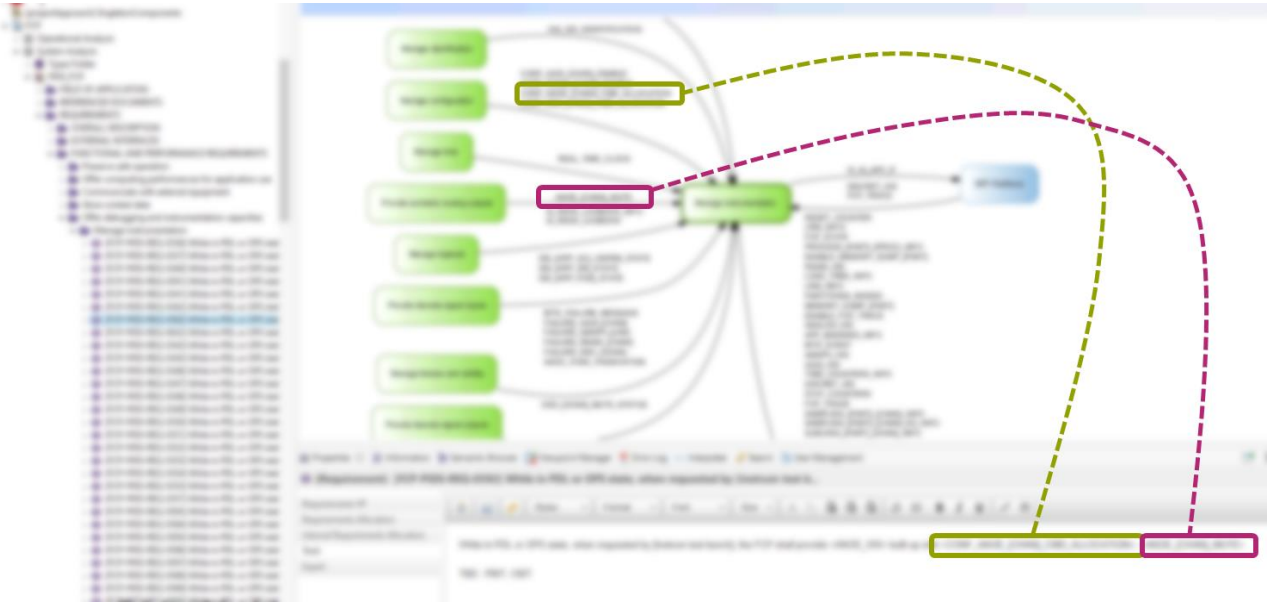






OPEN

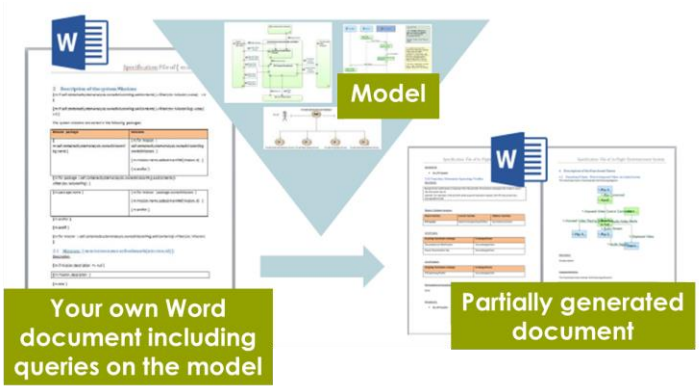
1. DDV – Dynamic Diagram Viewer
2. **Requirements**
3. Documentation generation
4. PVMT – Annotating models
5. Queries - Extracting data
6. Articulation with software



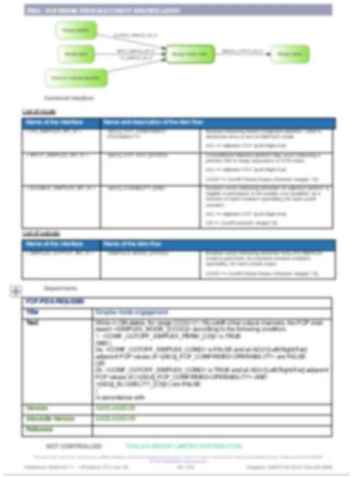
OPEN

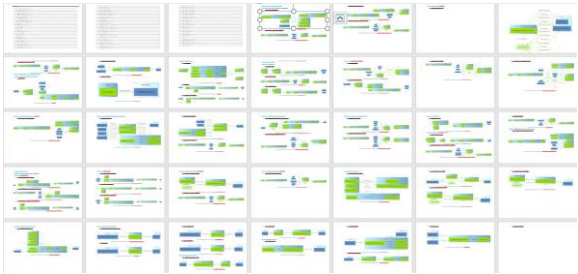
THALES
Building a future we can all trust

1. DDV – Dynamic Diagram Viewer
2. Requirements
3. **Documentation generation**
4. PVMT – Annotating models
5. Queries - Extracting data
6. Articulation with software



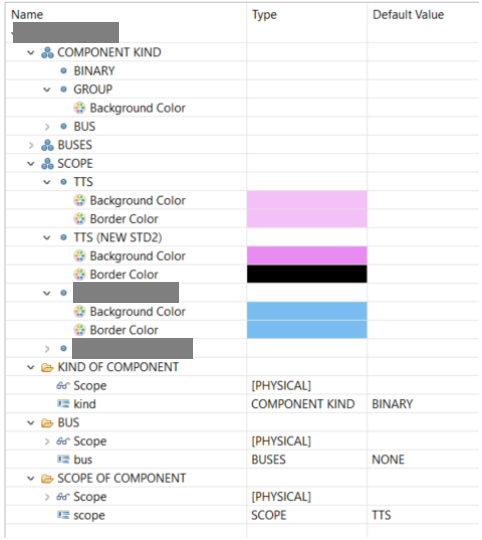
M2Doc

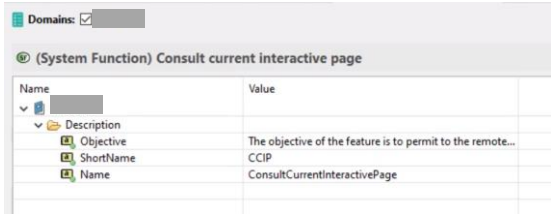


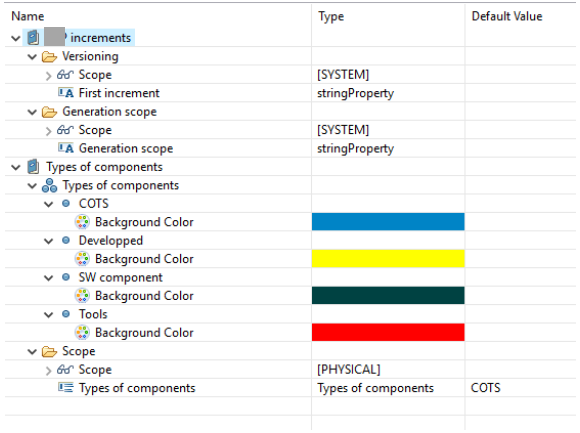


THALES
Building a future we can all trust

1. DDV – Dynamic Diagram Viewer
2. Requirements
3. Documentation generation
4. **PVMT – Annotating models**
5. Queries - Extracting data
6. Articulation with software







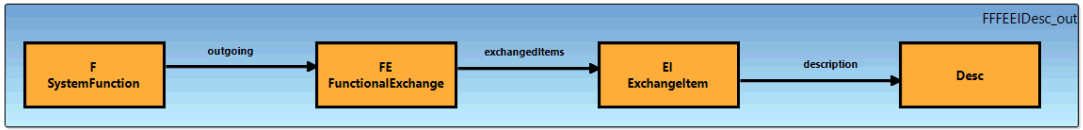
Integration of "versioning" in doc generation

```
{ m:if SF.childrenSystemFunctions->size() = 0 and [redacted]:appliedPropertyValueGroups.eContents()->filter(capellacore::StringPropertyValue)->first().value.contains(SF.appliedPropertyValueGroups.eContents()->filter(capellacore::StringPropertyValue)->first().value)}
```

OPEN

THALES
Building a future we can all trust

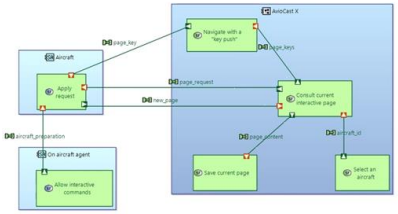
1. DDV – Dynamic Diagram Viewer
2. Requirements
3. Documentation generation
4. PVMT – Annotating models
5. **Queries - Extracting data**
6. Articulation with software



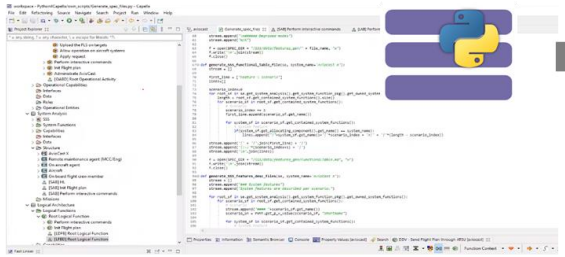
OPEN

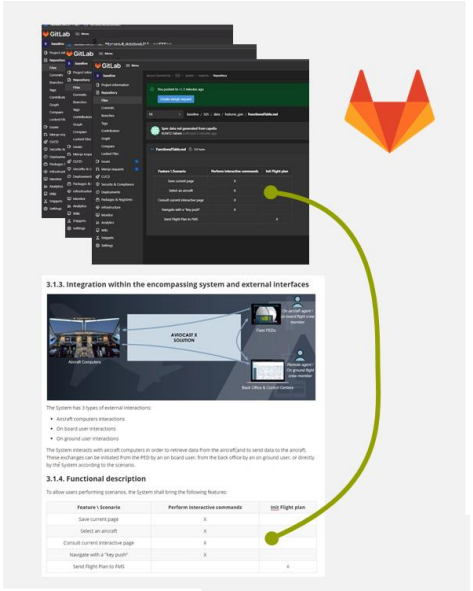
THALES
Building a future we can all trust

1. DDV – Dynamic Diagram Viewer
2. Requirements
3. Documentation generation
4. PVMT – Annotating models
5. Queries - Extracting data
6. **Articulation with software**



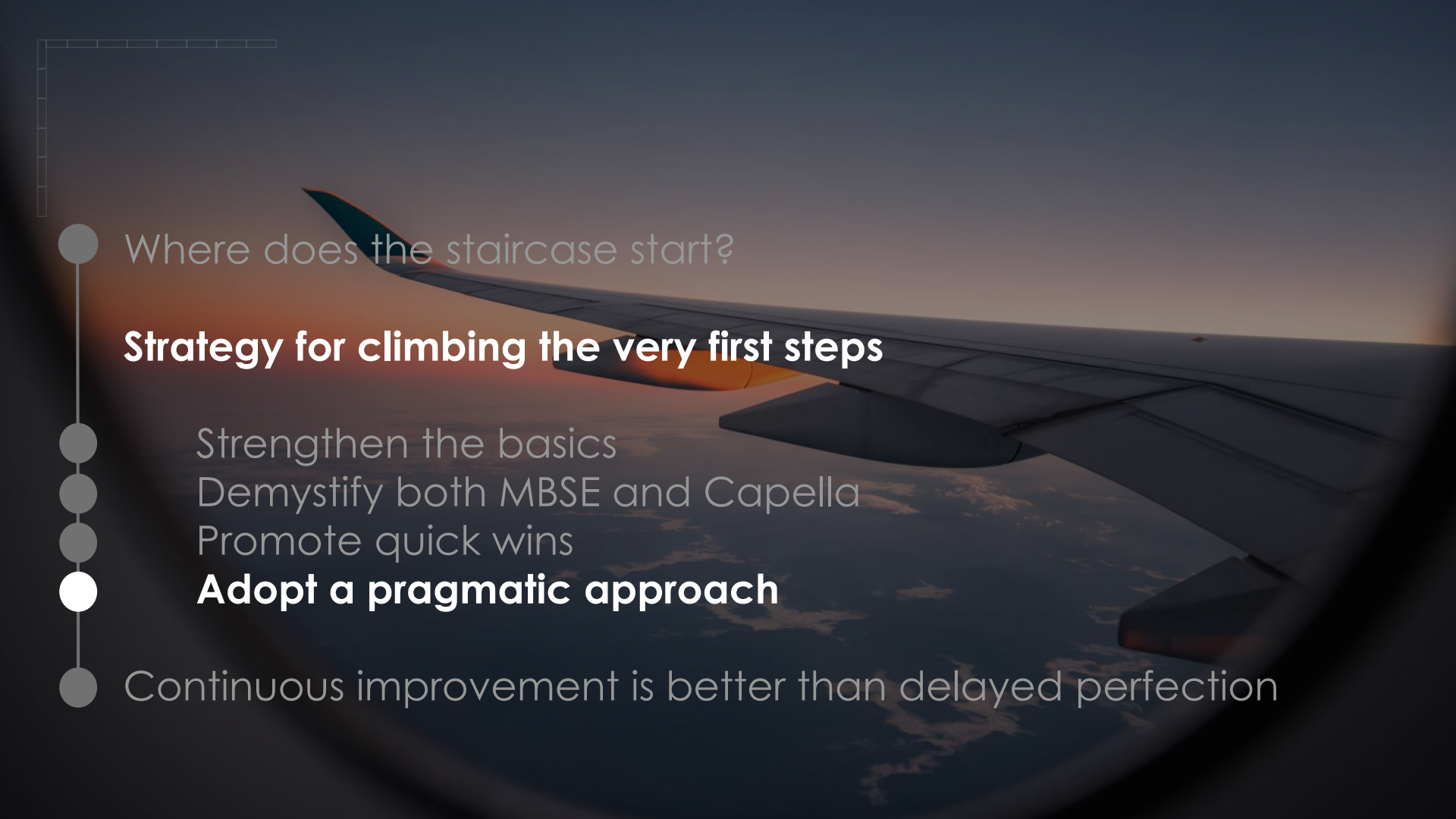
Python 4 Capella





OPEN

THALES
Building a future we can all trust



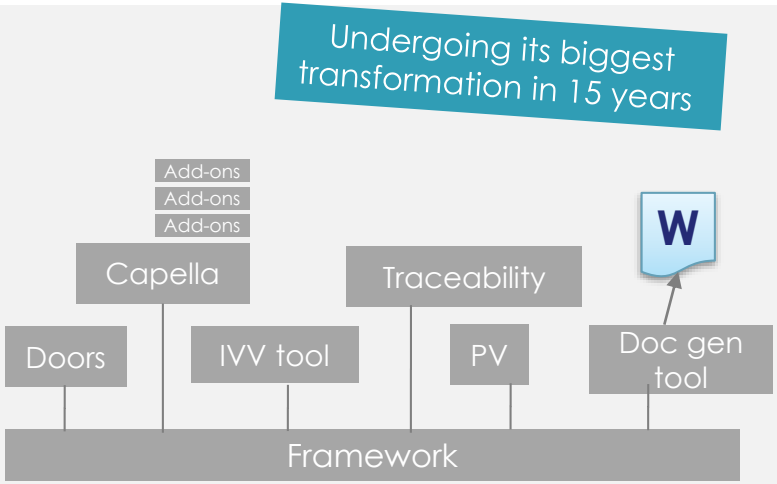
Where does the staircase start?

Strategy for climbing the very first steps

Strengthen the basics
Demystify both MBSE and Capella
Promote quick wins
Adopt a pragmatic approach

Continuous improvement is better than delayed perfection

Official Thales workbench



- Group official workbench offer
- Particularly interesting when connecting multiples kinds of artefacts (requirements, models, tests) and when product variability is applied to all artefacts

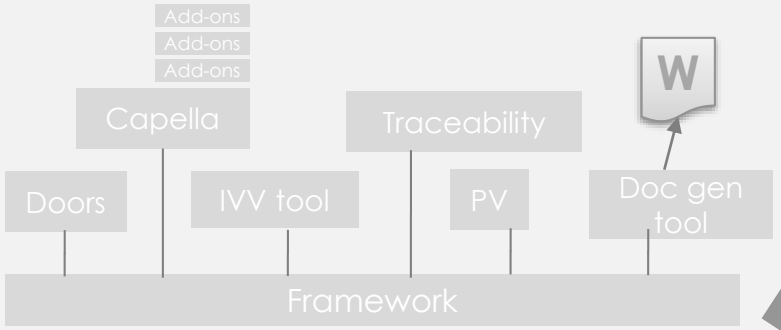
- Requires a lot of local support
- Version upgrades is complicated

3 options

- Wait for the new ultimate solution
- Keep on trying to force the deployment of the standard tooling
- Acknowledge the difficulty, and imagine a workaround

Official Thales workbench

Undergoing its biggest transformation in 15 years

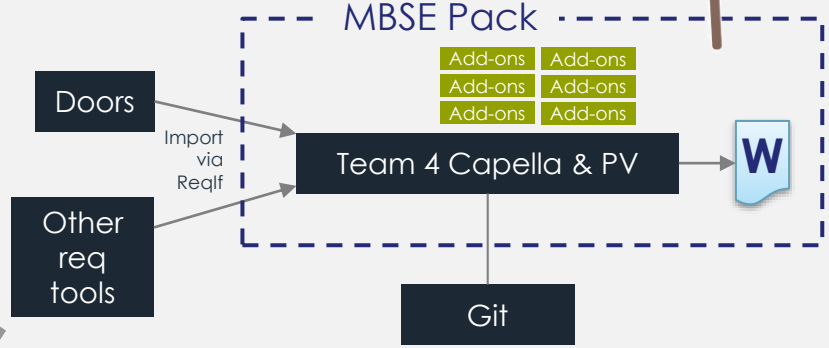


- ⊕ Group official workbench offer
- ⊕ Particularly interesting when connecting multiples kinds of artefacts (requirements, models, tests) and when product variability is applied to all artefacts

- ⊖ Requires a lot of local support
- ⊖ Version upgrades complicated

Possible interfacing

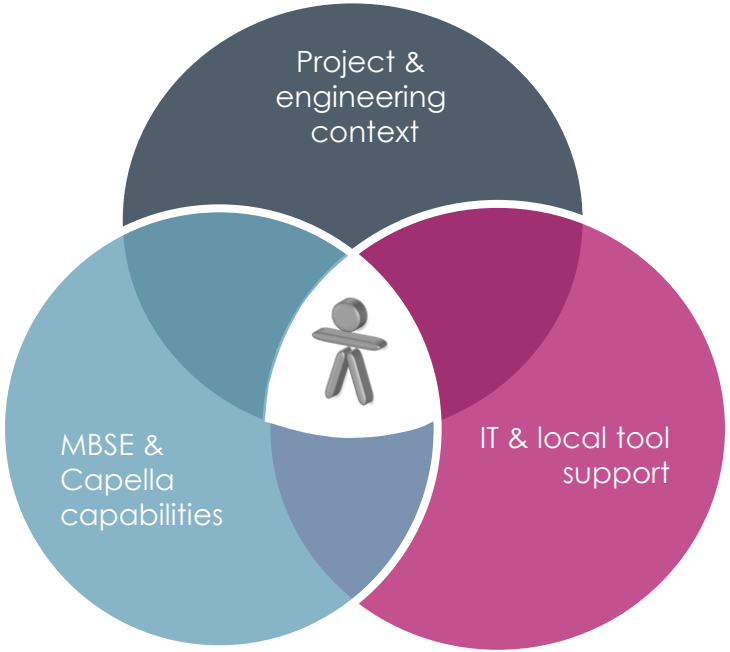
MBSE Pack

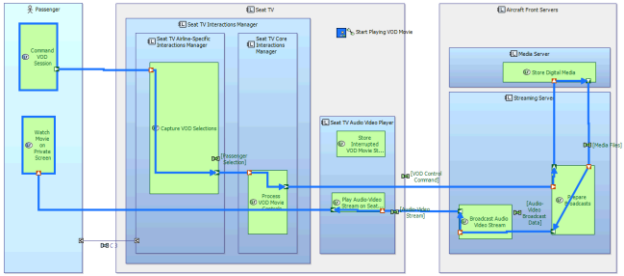


- ⊕ Same tools than Group workbench, but connected differently. Supported by AVS DSI & workbench teams
- ⊕ Very few tooling configuration, multi-user by default. Intended to be a minimalist MBSE environment
- ⊕ Easy to adopt recent versions of tools
- ⊕ Freedom for each user to install add-ons locally

- ⊖ Only suitable for approaches where models play a central role. No built-in integration of IVV tool

Who can make the whole thing happen?





Manuel extraction and update

	[LAB][CTX] Start Playing VOD Movie FC
External Functions	Watch Movie on Private Screen
	Command VOD Session
Internal Functions	Store Interrupted VOD Movie Status
	Play Audio-Video Stream on Seat TV
	Process VOD Movie Controls
	Capture VOD Selections
	Broadcast Audio Video Stream
	Prepare Broadcasts
	Store Digital Media
Actors	Passenger
System/System Components	Seat TV
	Seat TV Audio Video Player
	Seat TV Interactions Manager
	Seat TV Core Interactions Manager
	Seat TV Airline-Specific Interactions Manager
	Aircraft Front Servers
	Streaming Server
	Media Server
Interfaces	C 3

OPEN

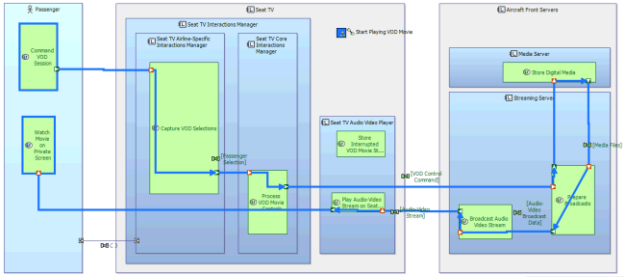
Example

Project practice: One manually maintained LAB per capability. No usage of functional chains. Manually produce tables reflecting the LAB content.

Recommended practice: Create functional chains to illustrate capabilities, make the LAB unsynchronized and contextual to all functional chains. Write queries exploiting functional chains to produce tables.

Pragmatic updated practice: LAB still maintained manually, but Python script to extract Capability content.

Example



Automated extraction and update

Python 4 Capella

```
139 # Writing excel file headers
140 worksheet1 = workbook.active
141 worksheet1.title = 'Diagram Content'
142
143 i = 1
144 worksheet1.cell(row = i, column = 1).value = myDiagram.
145 worksheet1.merge_cells(start_row=1, start_column=1, en
146
147
148 index_first_external_functions = i + 1
149 for e_f in external_functions:
150     i=i+1
151     worksheet1.cell(row = i, column = 1).value = 'Exte
152     worksheet1.cell(row = i, column = 2).value = e_f.g
153
154 worksheet1.merge_cells(start_row=index_first_external_
155
156 index_first_internal_functions = i+1
157 for i_f in internal_functions:
158     i=i+1
159     worksheet1.cell(row = i, column = 1).value = 'Internal Functions'
160     worksheet1.cell(row = i, column = 2).value = i_f.get_name()
161
162 worksheet1.merge_cells(start_row=index_first_internal_functions, start_column=1, end_row =i, end_column=1)
163
164 index_first_actors = i + 1
165 for act in actors:
166     i=i+1
167     worksheet1.cell(row = i, column = 1).value = 'Actors'
168     worksheet1.cell(row = i, column = 2).value = act.get_name()
169
170 worksheet1.merge_cells(start_row=index_first_actors, start_column=1, end_row =i, end_column=1)
171
172 index_first_system_components = i + 1
173 for s_c in system_components:
174     i=i+1
175     worksheet1.cell(row = i, column = 1).value = 'System/System Components'
```

	[LAB][CTX] Start Playing VOD Movie FC
External Functions	Watch Movie on Private Screen
	Command VOD Session
Internal Functions	Store Interrupted VOD Movie Status
	Play Audio-Video Stream on Seat TV
	Process VOD Movie Controls
	Capture VOD Selections
	Broadcast Audio Video Stream
	Prepare Broadcasts
	Store Digital Media
Actors	Passenger
System/System Components	Seat TV
	Seat TV Audio Video Player
	Seat TV Interactions Manager
	Seat TV Core Interactions Manager
	Seat TV Airline-Specific Interactions Manager
	Aircraft Front Servers
	Streaming Server
	Media Server
Interfaces	C 3

OPEN

Project practice: One manually maintained LAB per capability. No usage of functional chains. Manually produce tables reflecting the LAB content.

Recommended practice: Create functional chains to illustrate capabilities, make the LAB unsynchronized and contextual to all functional chains. Write queries exploiting functional chains to produce tables.

Pragmatic updated practice: LAB still maintained manually, but Python script to extract Capability content.

Where does the staircase start?

Strategy for climbing the very first steps

- Strengthen the basics
- Demystify both MBSE and Capella
- Promote quick wins
- Adopt a pragmatic approach

Continuous improvement is better than delayed perfection

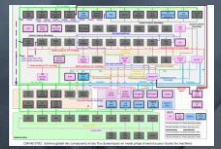
Every single step counts

MBSE as support of more efficient engineering practices on project 2

Capability-based approach, functional chains, part of engineering workflows

Generated design document

Ad-hoc model, 2 add-ons, 2 days



10+ Visio diagrams for modes, busses, etc.

MBSE quick win in a critical moment of project 1

Every single step counts

Questions?

stephane.bonnet@thalesgroup.com