MBSE2021
SPECIFICATION AND ARCHITECTURE OF THE SYSTEM FACTORY (SASyF)

September 29nd/30th, 2021

Speakers
Elena Alaña (ealana@gmv.com)
Tiago Jorge (tiago.jorge@gmv.com)

PROJECT CONTEXT

PROJECT CONTEXT

SASyF Project

- **Objective**
  - To define the specification and architecture of a Model-Based System Engineering infrastructure for Space System Engineering, the so-called, *System Factory*, covering all phases of a space system development, by applying the Arcadia method.

Capella used to model a **reference MBSE development/engineering system** that allows to better develop **the space (mission) systems**

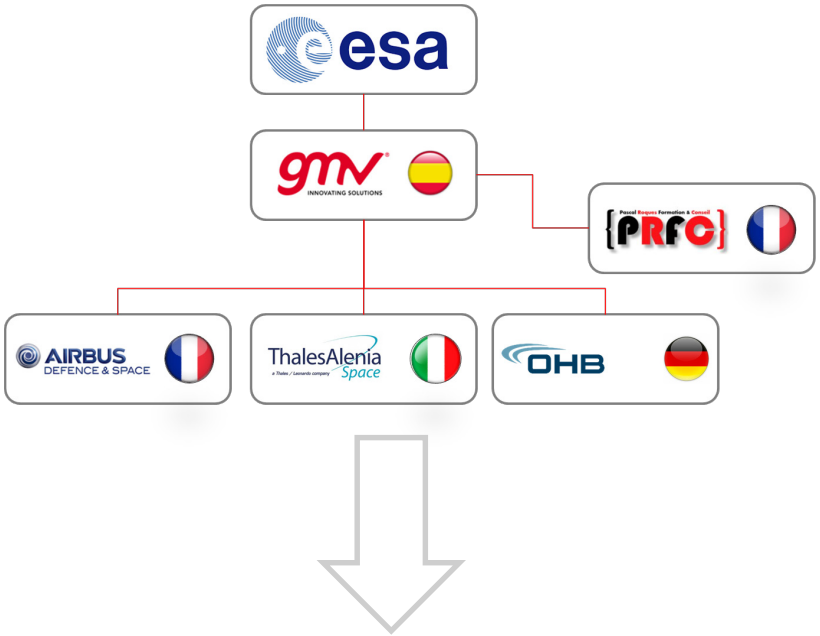
  - To model how a MBSE-based *System Factory* supports the Systems Engineers in executing the tasks described in the standard ECSS-E-ST-10 (not at individual discipline level).

- **Scope**
  - The scope is *System Factory* local to a LSI. It could be tailored to the one of an Agency or a LSI's subcontractor.
  - Special focus on the information exchanged (delivered/received) from the different stakeholders and internal interactions.

PROJECT CONTEXT

SASyF Team

- **Consortium**
  - Technical Officer: Andreas Jung
  - GMV – Main Contractor
  - Airbus DS, Thales Alenia Space – Subcontractors
  - PRFC (Pascal Roques) – External Consultant
- **Schedule**
  - Started on January 15th, 2020
  - Expected completion in October 2021



https://mb4se.esa.int/

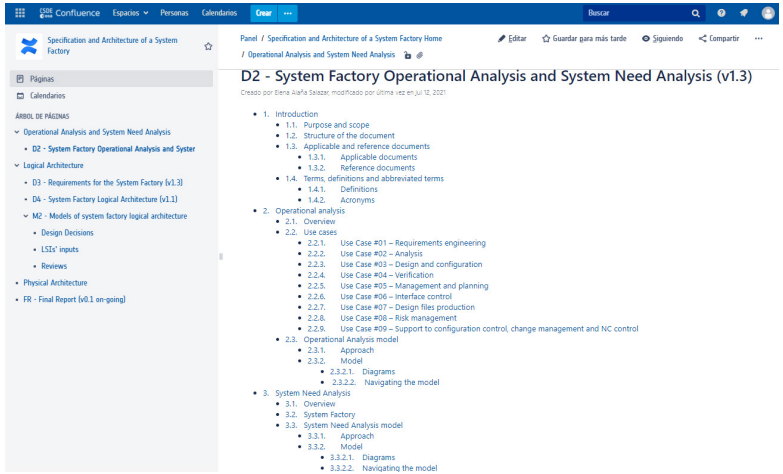
SPECIFICATION AND ARCHITECTURE OF THE SYSTEM FACTORY (SASyF)
APPROACH

APPROACH
Working Method

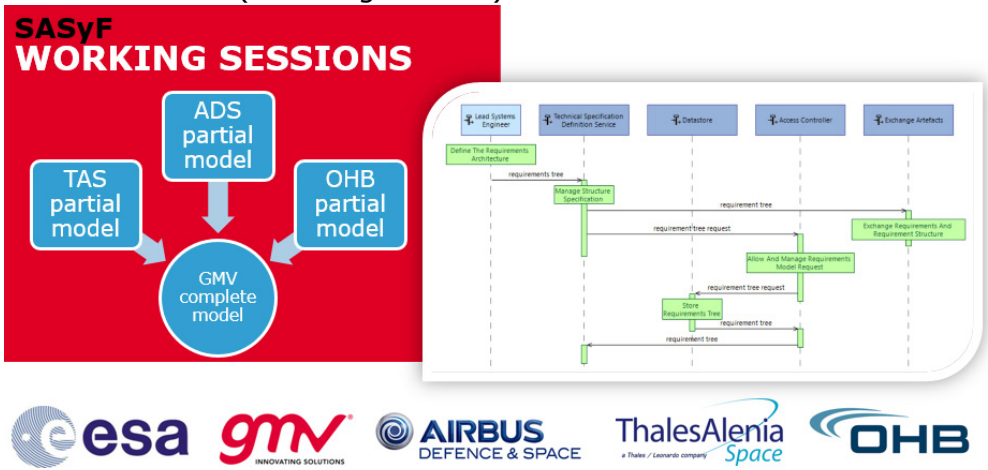
- The architecture of the System Factory is the result of a **collaborative work** among *SASyF's partners* and reviewed by *MB4SE Advisory group*.

https://mb4se.esa.int/

- Documentation:



- Model (including reviews):



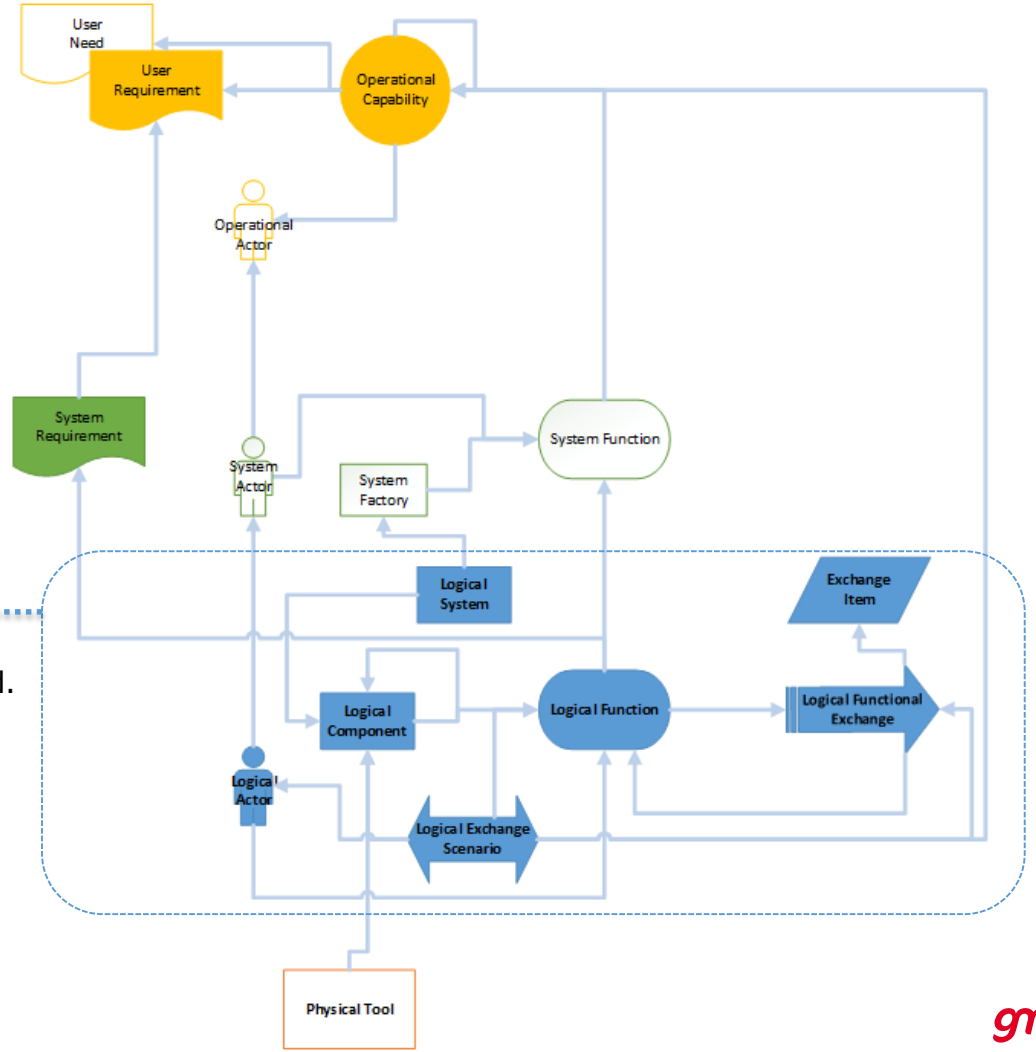
SPECIFICATION AND ARCHITECTURE OF THE SYSTEM FACTORY (SASyF)
RESULTS

RESULTS
System Factory

The main results of the SASyF project are:

1. Use Cases (Operational Capabilities)
2. User Requirements
3. System Requirements
4. Logical Architecture

They are obtained following the Arcadia method.



RESULTS

Use cases of the System Factory

- The use cases compile the **main exchange scenarios of a space system development process among stakeholders and project phases**.
- The use cases were provided by the LSIs based on their experience on building space systems.
- The use cases have been organised according to the space System Engineering activities when a model-based approach is adopted: one use case per System Engineering activity which integrate several sub use cases.
    - Use Case #01 – *Requirements engineering*
    - Use Case #02 – *Analysis*
    - Use Case #03 – *Design and configuration*
    - Use Case #04 – *Verification*
    - Use Case #05 – *Management and planning*
    - Use Case #06 – *Interface control*
    - Use Case #07 – *Design files production*
    - Use Case #08 – *Risk management*
    - Use Case #09 – *Support to configuration control, change management and NC control*

RESULTS

Use case structure (e.g. Requirements Engineering)

- Sub case: *Customer requirements analysis* (phases A/B).

USE CASE ID	UC#01		
USE CASE TITLE	Requirements Engineering	**PHASES**	Mostly (but not limited to) early phases (A/B).
DESCRIPTION	Main sub-use cases: Requirements engineering is not a single pass and it is performed by iterations with the maturity increase from both *Customer* and *Supplier* sides. Thus the need to have a tool to share easily both customer requirements and technical specifications with history and impact tracking is needed. 1.1. Customer requirements analysis (phases A/B) As a *Customer* I need to deliver the specifications to my supplier, and being able to receive comments, compliance status and change proposals, and to perform co-engineering to support the improvement of the requirements specification during early project phases and to evaluate the impact of any proposed changes. As a *Supplier*, I need to receive the customer specification and related ancillary specifications (Level 0, L-0 spec: SRD, IRD, etc.) in a format compatible with my requirements management system and being able to separate internal comments with customer-related comments, and being able to iterate with the customer so to improve the common understanding in its requirements specification. As a *Project Manager* or *Systems Engineer* of the *Customer* or the *Supplier* I need to keep control of such flow of information and approve such exchanges. (Applicable to all sub use cases). *Exchanges: Specifications from customer (MRD, SRD, IRD, etc.), supplier comments and change proposals, compliance matrices*		

...

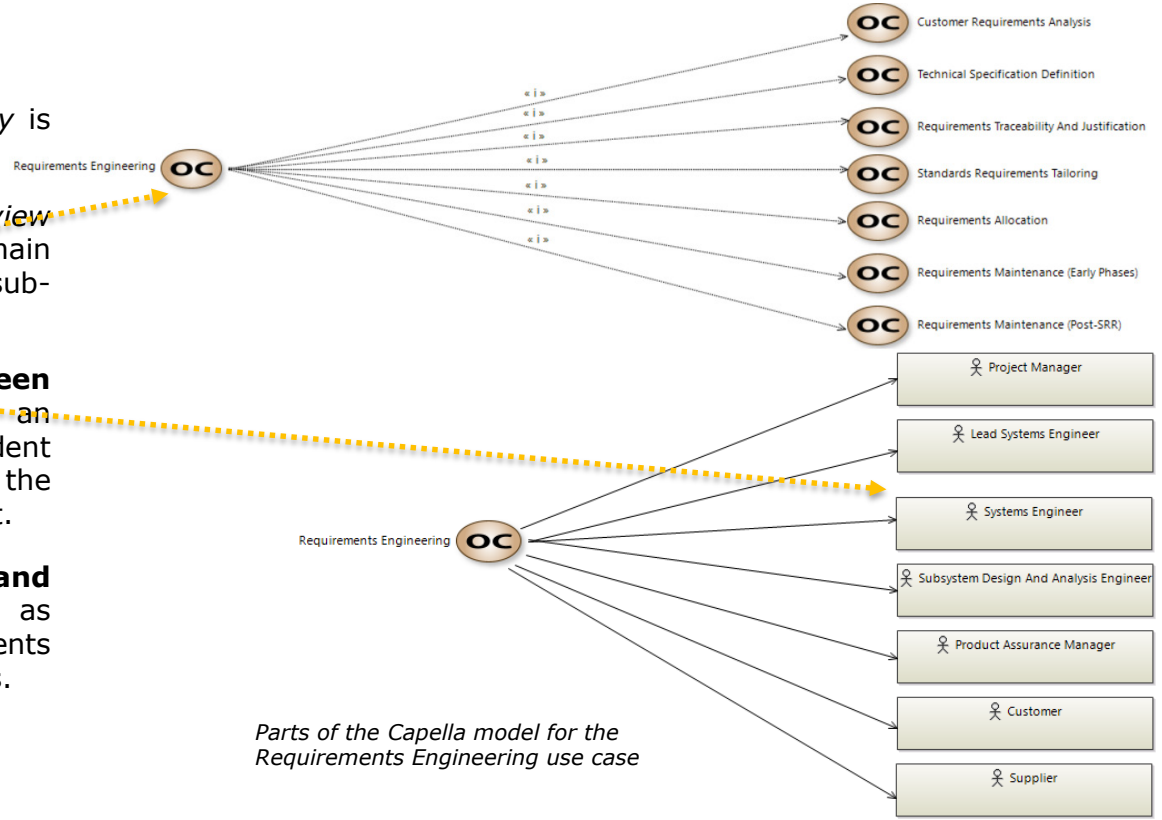
ROLES	*Customer/Supplier* involving: • *Project Manager* • *Lead Systems Engineer* • *Systems Engineer* • *Subsystem Design and Analysis Engineer* • *Product Assurance Manager*
SEMANTICS	Requirement • Including traceability to upper and lower level requirements • Including justification and allocation to lower level components • Associated verification method, level, stage, model Specification • Including link with product tree, specification tree and company

RESULTS

Operational Analysis of the System Factory

The Operational Analysis of the *System Factory* is characterised by the following features:

- It is *simple* and represents a *high abstract view* of the user needs, identifying the main **Operational Capabilities** (use cases and sub-use cases) and Actors (roles).
- The **Actors have been simplified/abstracted** to provide an Operational Analysis which is not dependent from the type of discipline involved in the project, simplifying also the modelling effort.
- The **User Requirements are modelled and traced to the sub-use cases** (modelled as refined capabilities). The User Requirements are also traced to the System Requirements.



Parts of the Capella model for the Requirements Engineering use case

RESULTS

User Requirements

- The User Requirements represent the **user needs for a typical space system development process from different users' perspectives**. Therefore, they are user-oriented and are derived from the Operational Analysis **Sub-Use Cases**.

- Example for the *Requirements Engineering* use case (The example does not include all the requirements).

Requirement Id	Requirement Text	Trace to Sub-Use Cases
URD-REQ-1010	The customer shall be able to deliver its specifications to the supplier.	UC 1.1
URD-REQ-1020	The customer specifications shall be delivered in a format which can be imported by the supplier.	UC 1.1
URD-REQ-1030	It shall be possible to exchange comments and related answers on customer specifications between customer and supplier.	UC 1.1
URD-REQ-1040	The supplier shall be able to provide to the customer the state of compliance w.r.t. to the provided specifications and applicable documents in matrix form or via compliance statements.	UC 1.1 UC 1.2
URD-REQ-1050	It shall be possible to plan, execute and trace co-engineering sessions between customer and supplier, to improve understanding and formulation of customer specification or of supplier specifications.	UC 1.2
URD-REQ-1060	*DELETED*	-
URD-REQ-1070	It shall be possible to exchange the structure of the technical requirements of the lower level suppliers and related support specifications.	UC 1.2 UC 1.3
URD-REQ-1080	The supplier shall be able to deliver its solution technical specifications (including ancillary specifications) to the customer.	UC 1.2
URD-REQ-1090	It shall be possible to exchange traceability information between the customer and lower level specifications.	UC 1.3
URD-REQ-1100	It shall be possible to provide a dashboard providing a synthesis of the traceability between different specifications, such as number and percentage of requirements traced on lower level specifications or traced requirements per each type of specifications.	UC 1.3

RESULTS

System Need Analysis of the System Factory

- The System Need Analysis level derives **concrete information from the Operational Analysis level, detailing the scope of the *System Factory*, including the interfaces with actors**.
- The architecture of the *System Factory* is defined as a "black box".
- The System Need Analysis of the *System Factory* is characterised by the following features:
    - The ***System Factory's*** **boundary** is identified together with the **System Functions** handled by the *System Factory* and by the Actors, as well as the functional breakdown.
    - The criteria to determine if a functionality is performed or not by the *System Factory* is limited by the fact that the proposed ***System Factory*** **is defined at company level**. Therefore, the interaction with other similar infrastructures, e.g. in the *Customer* side, are later (logical level) represented as exchanges with the corresponding Actors.

RESULTS

System Need Analysis of the System Factory

The System Needs Analysis of the *System Factory* includes:

- **Functions allocation** to the Actors or the system.
- **Functional breakdown.**
- The **System Requirements** modelled and traced up to User Requirements (Operational Analysis) and down to Logical Functions (Logical Architecture).

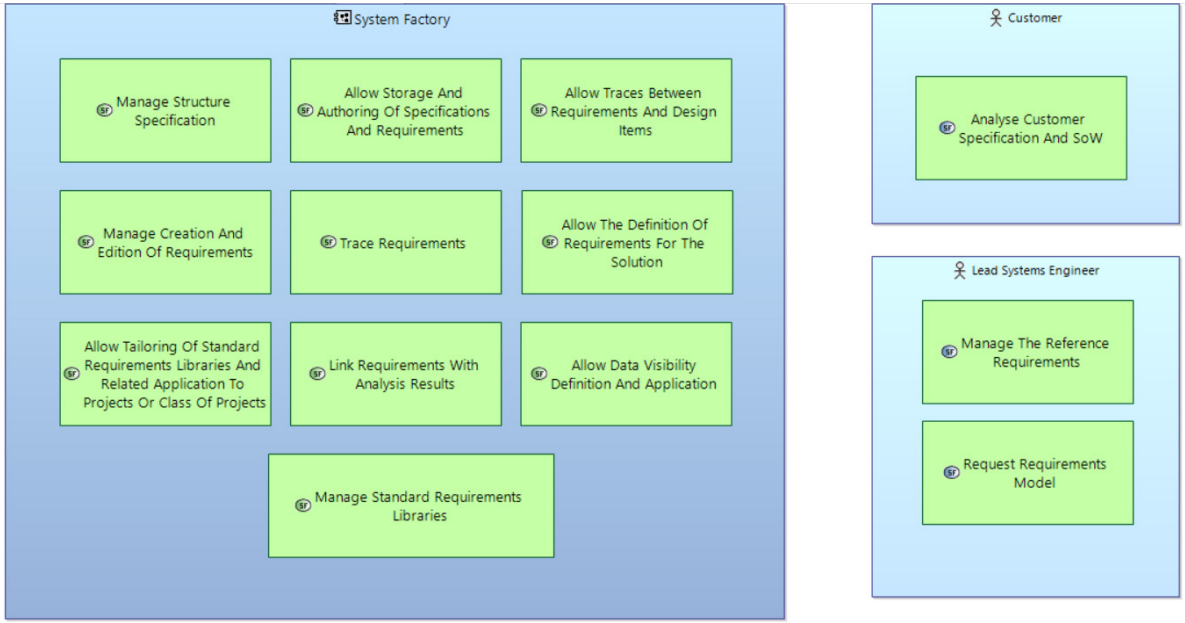


Diagram for the Requirements Engineering use case

RESULTS

System Requirements

- The **System Requirements** are derived from the User Requirements included considering the scope of the System Factory.
- Therefore, they specify if the user needs are satisfied by the System Factory itself.
- System Requirements include both **functional and non-functional** requirements but the main focus is on functional ones.

Requirement Id	Requirement Text
SYS-REQ-FUN-0010	The *system factory* shall allow delivering customer specifications to different entities depending on the industrial set-up.
SYS-REQ-FUN-0020	The *system factory* shall allow receiving supplier specifications from a non-model-based approach. *NOTE: The required flexibility of the system factory to import different types of format and the ability to provide a framework to develop import scripts will be developed in the lower-levels.*
SYS-REQ-FUN-0030	The *system factory* shall allow importing customer specifications and integrate it as requirements into the model. *NOTE: Customer requirements are not only mission-level requirements, but can also refer to system/sub-system requirements or via an applicability matrix (e.g. environment requirements).*
SYS-REQ-FUN-0040	The *system factory* shall allow storage and editing of specifications and requirements.
SYS-REQ-FUN-0050	The *system factory* shall manage structure specifications.
SYS-REQ-FUN-0060	The *system factory* shall allow allocation of specification sections /requirements.
SYS-REQ-FUN-0070	The *system factory* shall store and provide reference requirements.
SYS-REQ-FUN-0080	The *system factory* shall allow tailoring of standard requirements libraries and related application to projects or class of projects.
SYS-REQ-FUN-0090	The *system factory* shall provide requirement allocation capability.
SYS-REQ-FUN-0100	The *system factory* shall allow alternative requirements tree for change impact and trade-off analysis.
SYS-REQ-FUN-0110	The *system factory* shall allow definition of requirements metrics (criticality, completeness).
SYS-REQ-FUN-0111	The *system factory* shall allow linking requirements to the product tree elements.
SYS-REQ-FUN-0112	The *system factory* shall allow defining, maintaining and exchanging parametrized requirements that are machine readable and not only textual.

Part of Requirements specification

RESULTS

Logical Architecture - Characteristics

- The Logical Architecture presents **how** the system works to fulfill expectations.
- The level of Logical Architecture aims to identify **Logical Components** inside the System ("how the system will work to fulfill expectations"), their **breakdown**, their **relations** and their **Logical Functions**, *independently of any considerations of technology or implementation*.
- Characteristics:
  - There is not a unique logical solution.
  - SASyF Logical Architecture shall be a reference point for all companies to implement their Physical Architectures and it represents one feasible alternative already agreed by Airbus, TAS and OHB.
  - Focus on what is exchanged between stakeholders (Actors) and components.
  - High-level abstract architecture.
  - It should evolve to be more precise when new digital engineering practices are clarified.

RESULTS

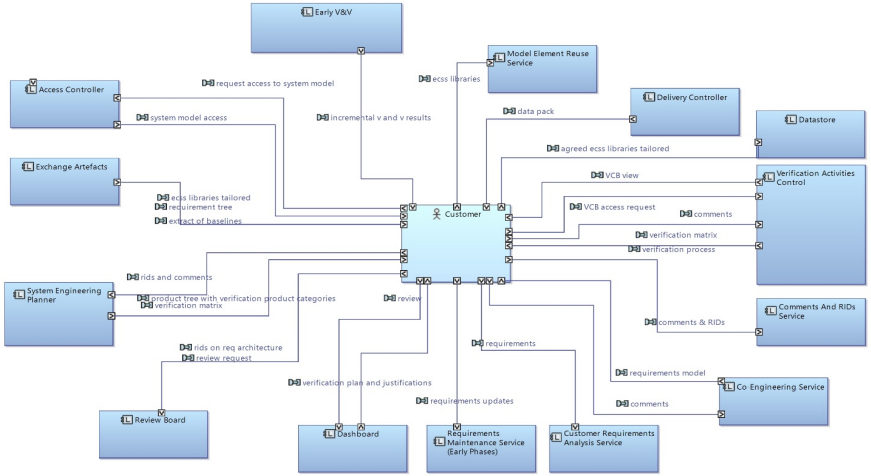
Logical Architecture - Usage

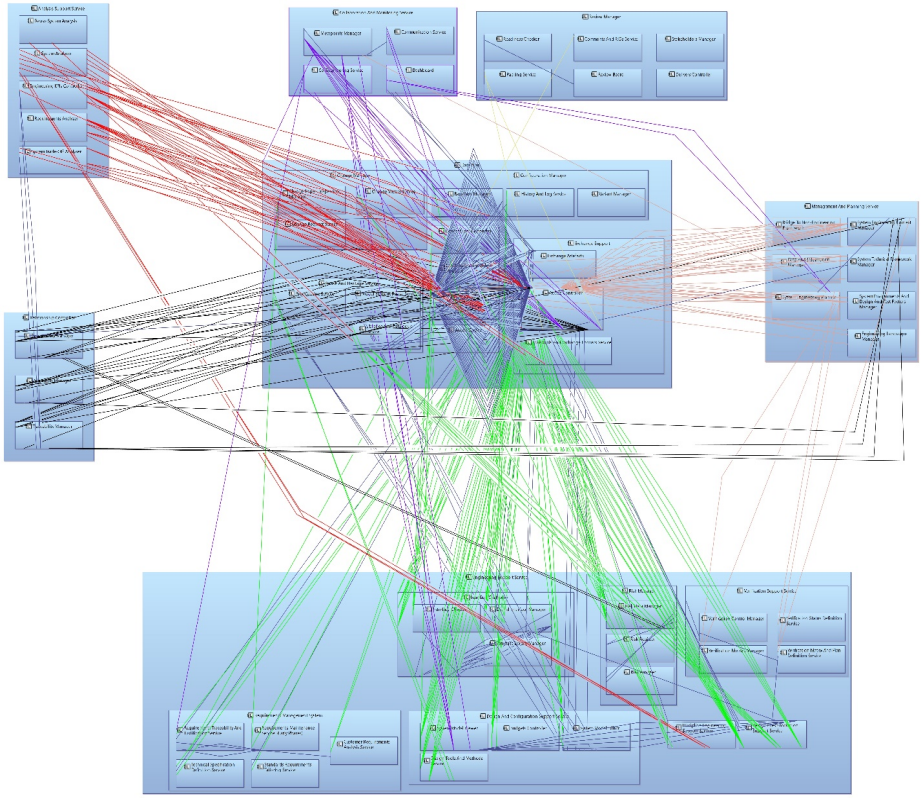
It ensures *completeness* of **Functions** and **Exchange Items** necessary to establish interoperability across industry and agencies.

Together with the *Space System Ontology* will facilitate the *interoperability* due to the common interfaces and common semantics.

Logical Architecture will mainly used by Primes. ESA will use it to interface with Primes in order to have smoother interactions. It allows to have a common way to map their own architectures and define standard interfaces.

Specific views according to the Actor/Role.

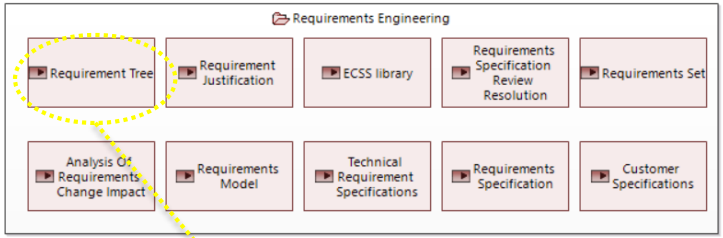




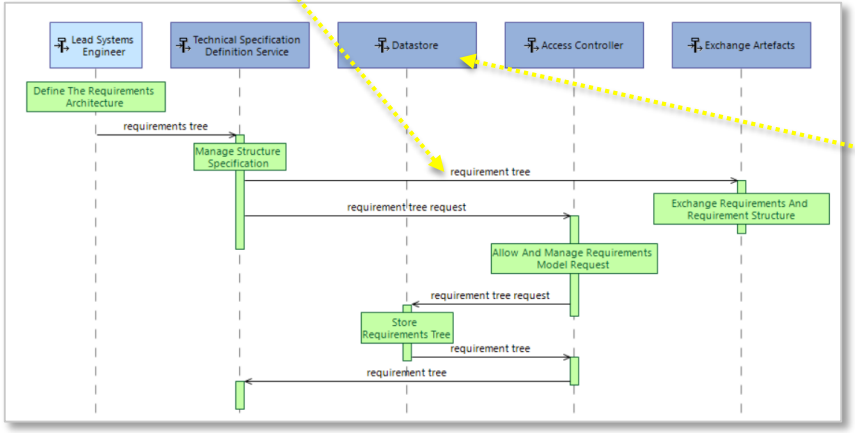
RESULTS

Logical Architecture - Representations

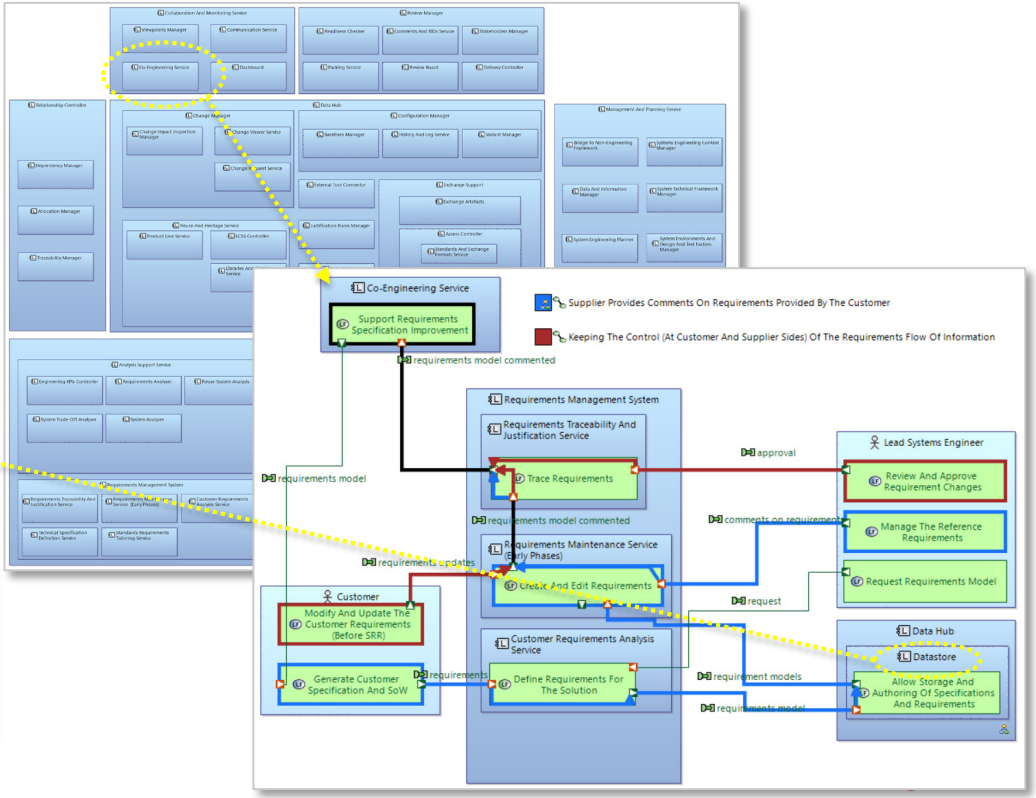
Exchange Items



Scenarios



Architecture & Functional chains



RESULTS

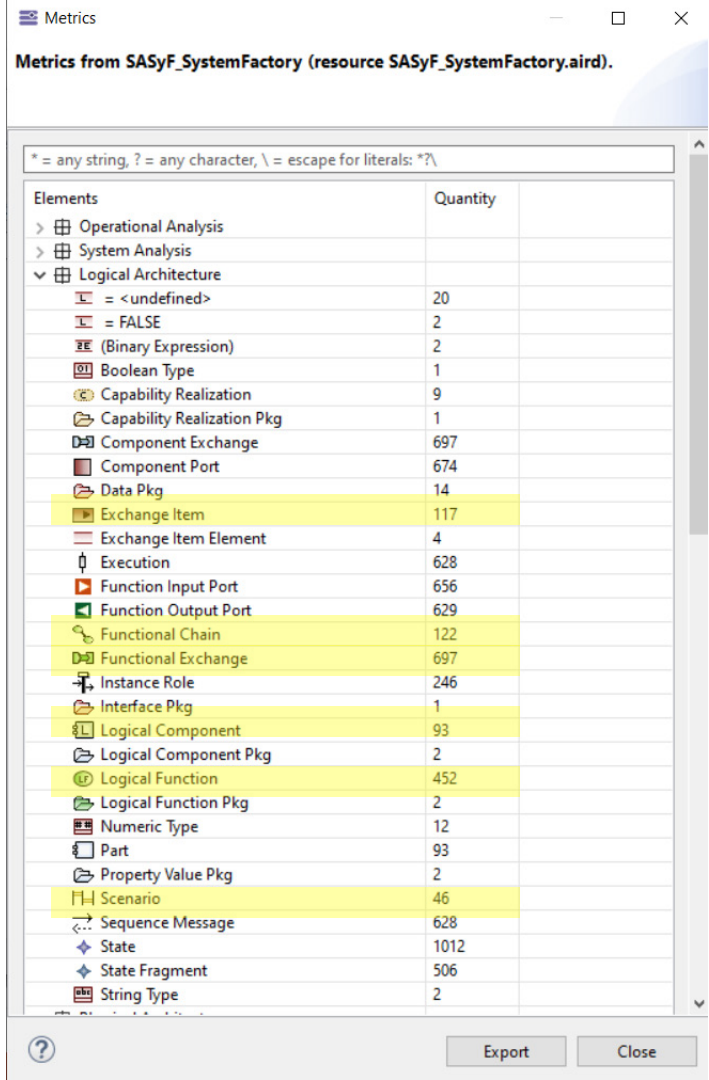
Metrics of the LA

Main Metrics:

- **Logical Components: 93**
- **Logical Functions.: 452**
- **Exchange Items: 117**
- **Functional Exchanges: 697**
- **Scenario: 46**
- **Functional Chains: 122**

Additional information:

- **22 Working sessions** (2 hours each session every 2 weeks)
- **27 contributions** (Capella model provided by the LSIs)
- **6 organisations involved** (ESA, GMV, TAS, ADS, OHB, PRFC)
- **~3-4 people involved per organisation**
- **495 commits**



RESULTS
Physical Architecture

- Concrete implementation of the Logical Architecture.
- Different Physical Architectures will be implemented (one per LSI).
- Characteristics:
  - Provide **information on how each leaf logical component is realized**: by what tool, to what extend (completely, partially or barely),...
  - It is expected to **evolve** as new digital engineering practices and needs arise. Current defined Physical Architecture represents the state-of-the-art tooling capabilities and implementations of each of the LSIs organisations.
  - They are a **reference point for the conduction of the gap analysis** that enables the identification of the three LSIs tooling limitations and shortages, the tools that are closely connected and so shall be interoperable.

CONCLUSIONS

Conclusions

- The Logical Architecture represents a **common vision and concrete architecture for the *System Factory***.
- This convergence is **challenging**, mainly due to diverse background and communication challenges, requiring close coordination and review iterations.
- The **model size** and the need to **work concurrently** impacts the modelling and review effort, a strategy being required.
- This architecture will contribute to enable **interoperability** together with the Space System Ontology and the Data Hub.
- This architecture shall be a **reference point** and evolve according to the digital engineering practices.

Thank you

SASyF team

Elena Alaña (ealana@gmv.com)
Tiago Jorge (tiago.jorge@gmv.com)

INNOVATING SOLUTIONS