Logical Architecture

How to interpret the LA

Tim Carter, DipMgt BEng(Hons) MEng CPEng NER

24 Mar 2026

Clarity SE

Arcadia Overview

Clarity SE

What is Arcadia?

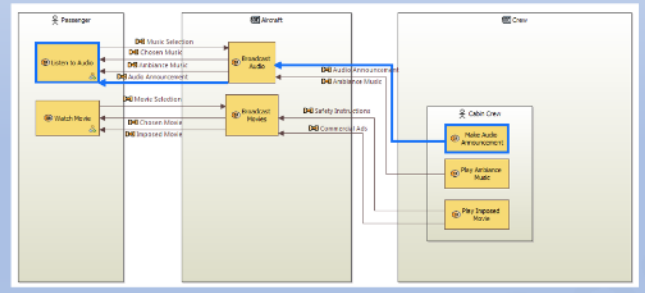
- Arcadia stands for Architecture And Design Integrated Approach
- Arcadia is a structured analysis approach to developing and decomposing a problem into a system architecture.
- It integrates with requirements analysis in an iterative fashion
  - The architectural design informs the requirements analysis and vica versa
  - Is integral to planning IVV activities
- Arcadia is designed to be flexible and scalable such that an engineer only needs to model what is important

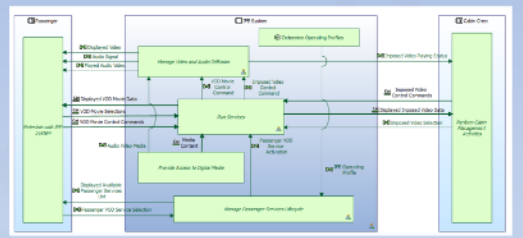
Arcadia Overview

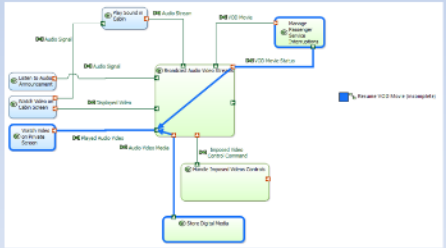
Top level concepts

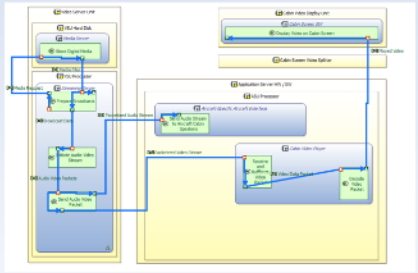
Arcadia has 4 main Architectural layers

- Operational Analysis [OA] – what the users need to do
- System Analysis [SA] – What the system needs to achieve for the users
- Logical Architecture [LA] – Today's Subject
- Physical Architecture [PA] – How the system will be built





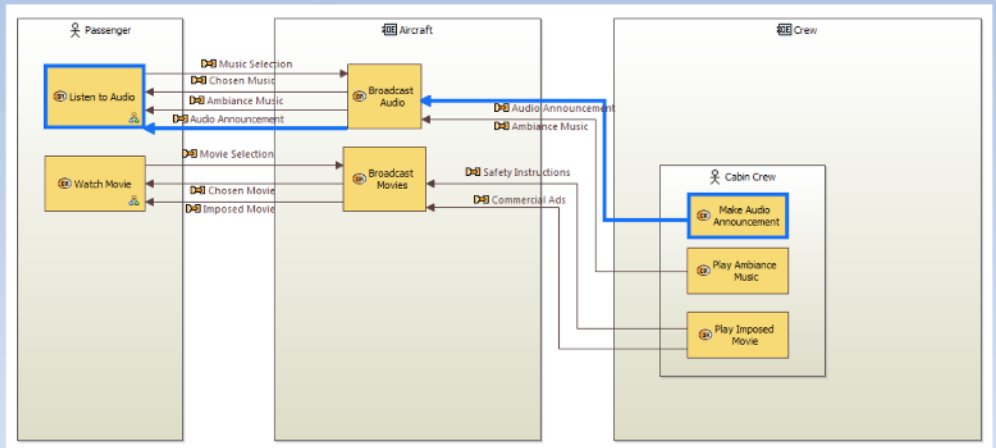




Operational Analysis [OA]

Overview

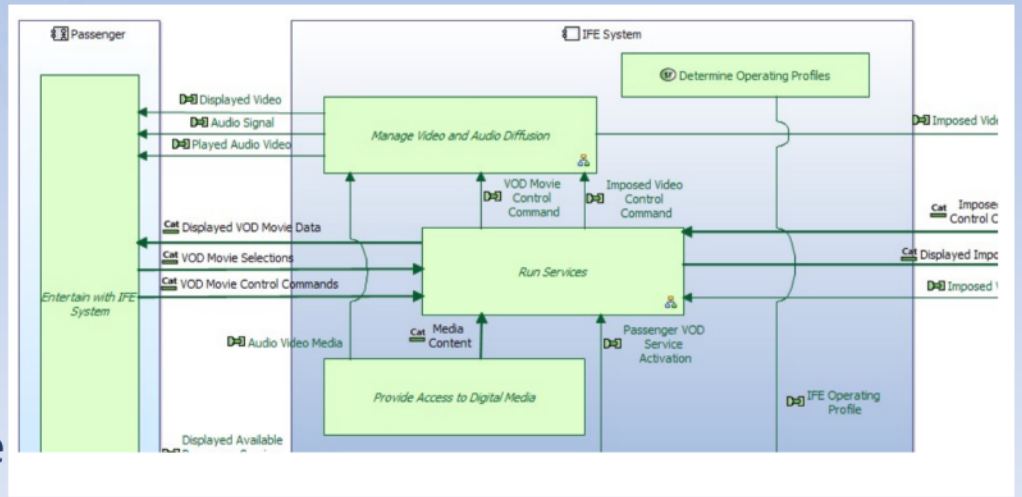
- The objective of the operational analysis is to formalise the needs, missions and activities of the end users of the system and capture them in a series of operational processes and scenarios
- The operational analysis forms the basis for a Concept Of Operations (CONOPS)
- The operational analysis ensures the system analysis is grounded in a clear understanding of the end users needs and how the system will be used



System Analysis [SA]

Overview

- The objective of the system analysis is to define the system role in satisfying the operational needs/CONOPS
- It is a black-box analysis that identifies:
  - The functions the system has to perform
  - The exchanges and interfaces between the system and external actors
  - Derives scenarios used to describe the use of the system from the functions, exchanges and interfaces of SA and the scenarios from OA
- Can be used to inform the development of customer/contract requirements and form the basis of initial cost estimates

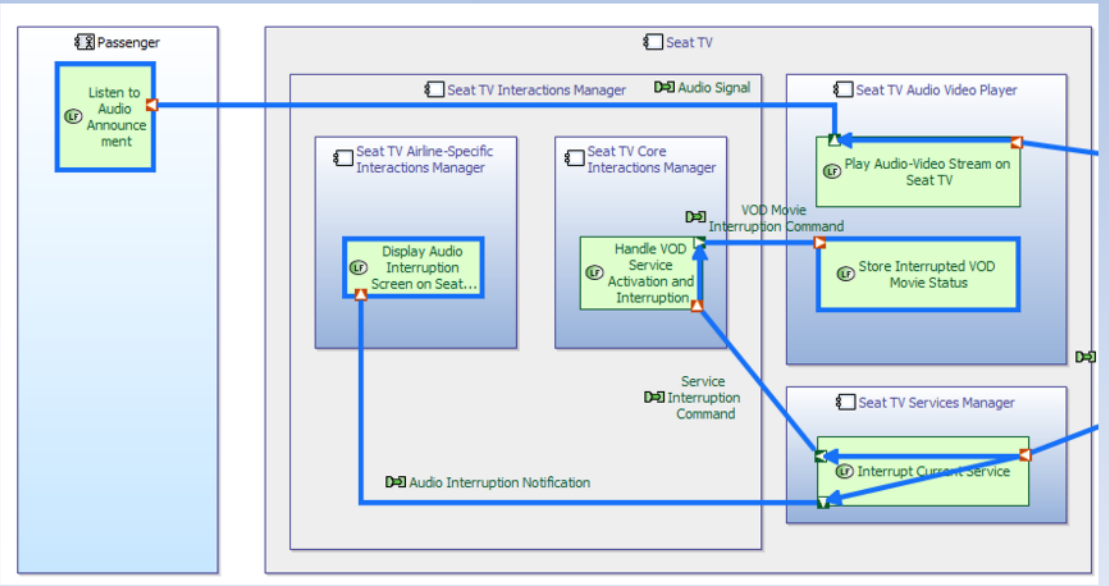


Clarity SE

Logical Architecture [LA]

Overview

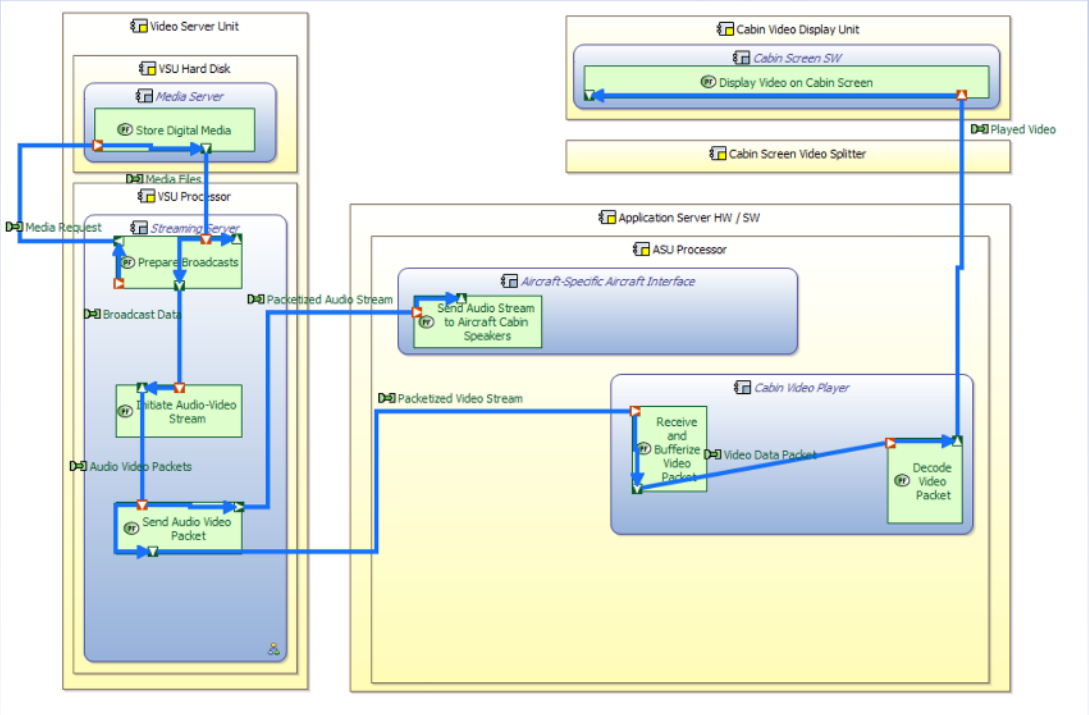
- Watch this space!



Physical Architecture [PA]

Overview

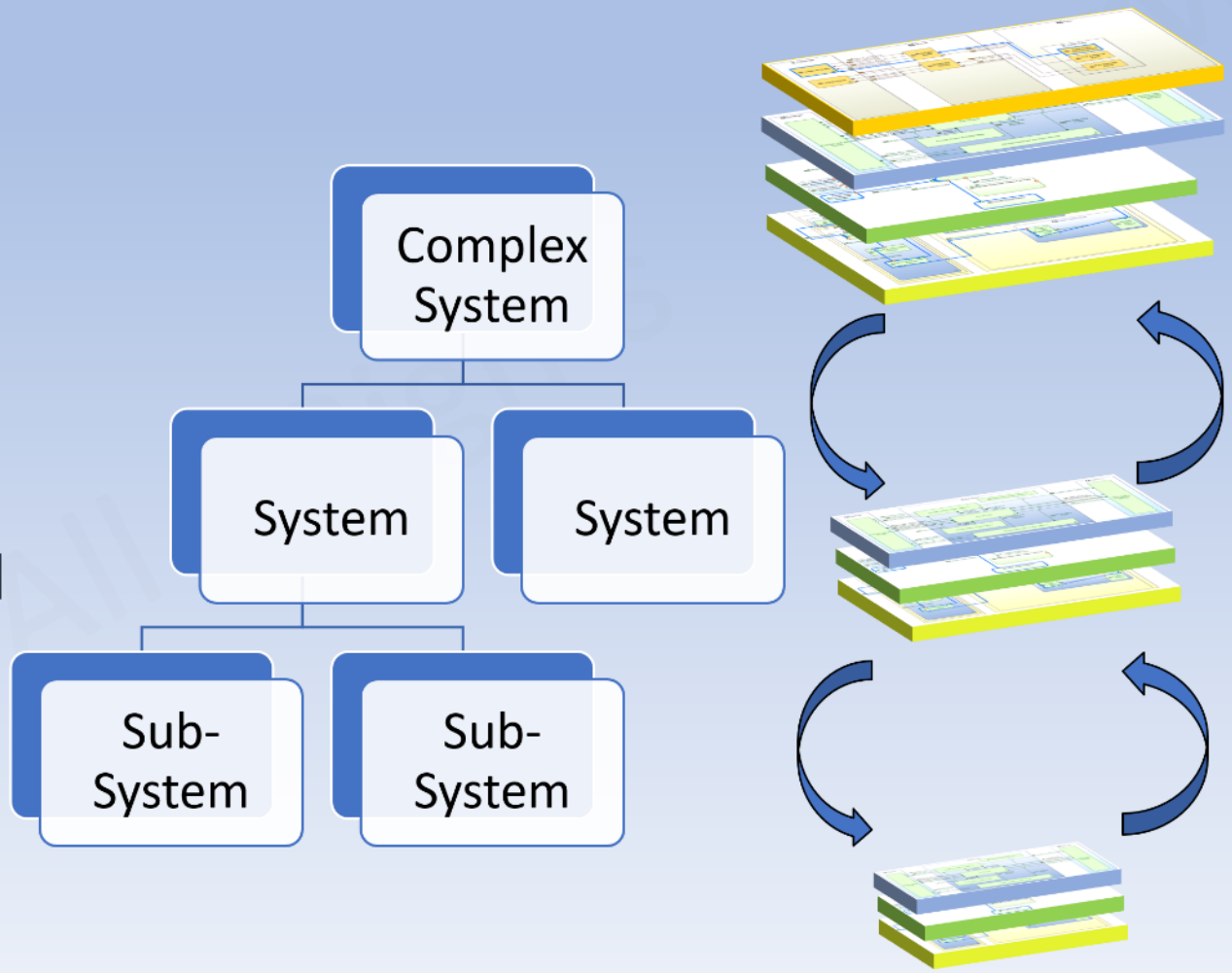
- The objective of the physical architecture is to identify all the system components their behaviours, interfaces and account for technological, deployment and system variant decisions
- The physical architecture forms the basis for a system design synthesis



Arcadia Recursively Masters Complex Systems

Main Concepts

- Arcadia uses the different architecture layers and system to sub-system relationships to master a complex system one step at a time
- Traceability and automated transitions between layers and sub-systems ensures coherence



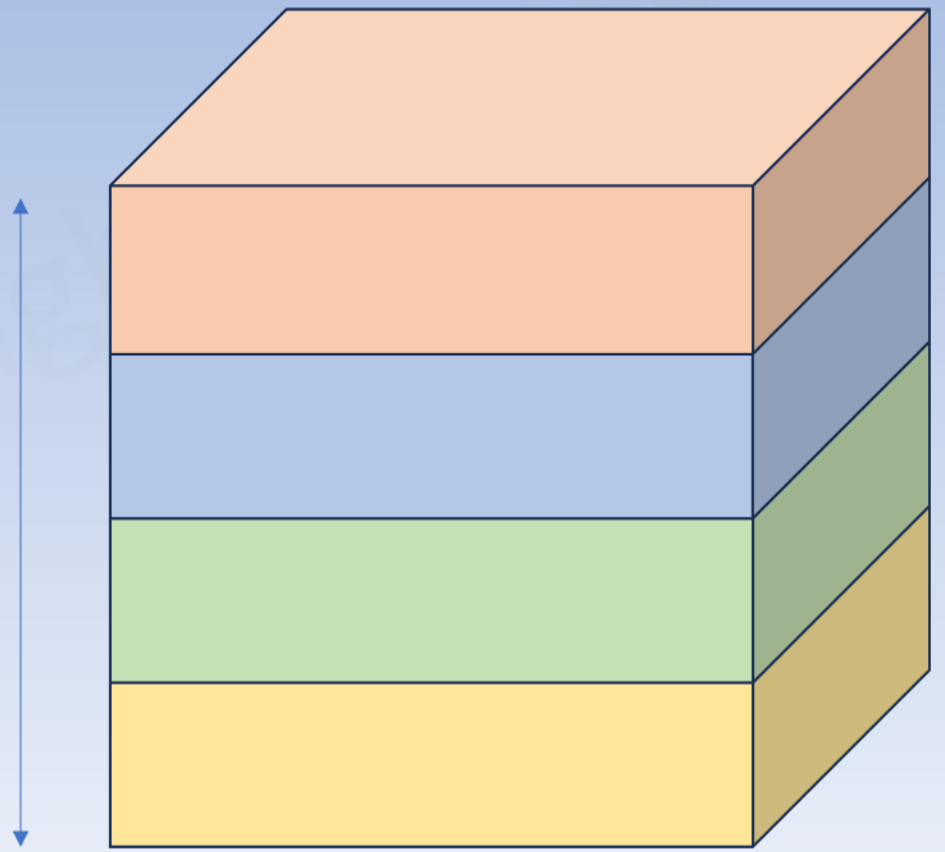
A word about Architecture/s

The fundamental concepts or properties of an entity in its environment and governing principles for the realisation and evolution of this entity and its related life cycle processes (ISO/IEC/IEEE 42020, 2019)

Clarity SE

Architecture, a 3-dimensional view classification

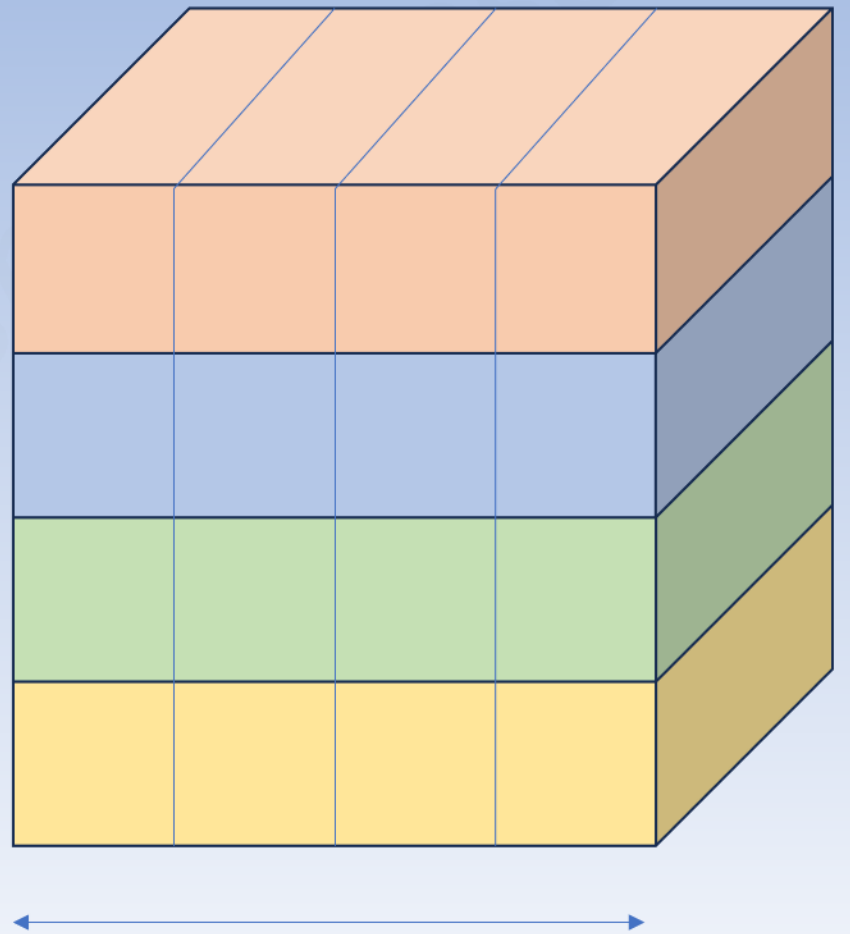
- Dimension 1 - Architecture Layer (where in the ‘V’)
  - Operational Needs
  - Scope of the project
  - Logical Architecture
  - Physical architecture (Synthesized technical solution)



Clarity SE

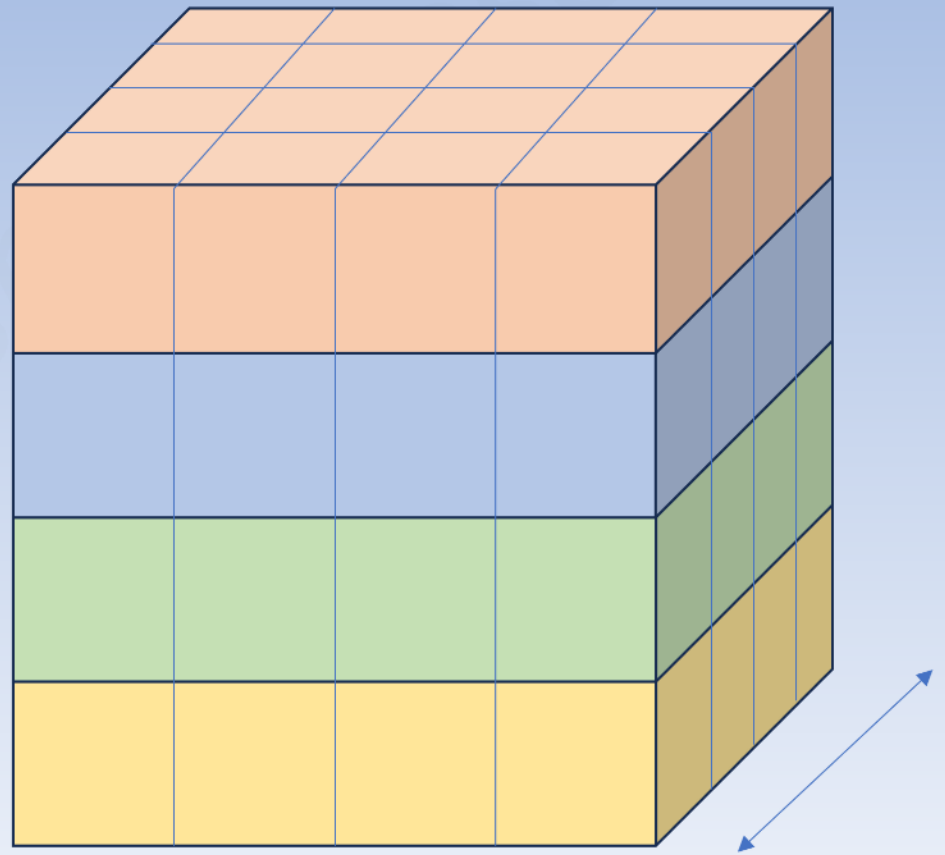
Architecture, a 3-dimensional view classification

- Dimension 1 - Architecture Layer
- Dimension 2 – Generality of domain
  - Comms solution
  - Comms for maritime
  - Comms for helicopters launched from a frigate
  - Comms for a seahawk launched for an ANZAC class frigate



Architecture, a 3-dimensional view classification

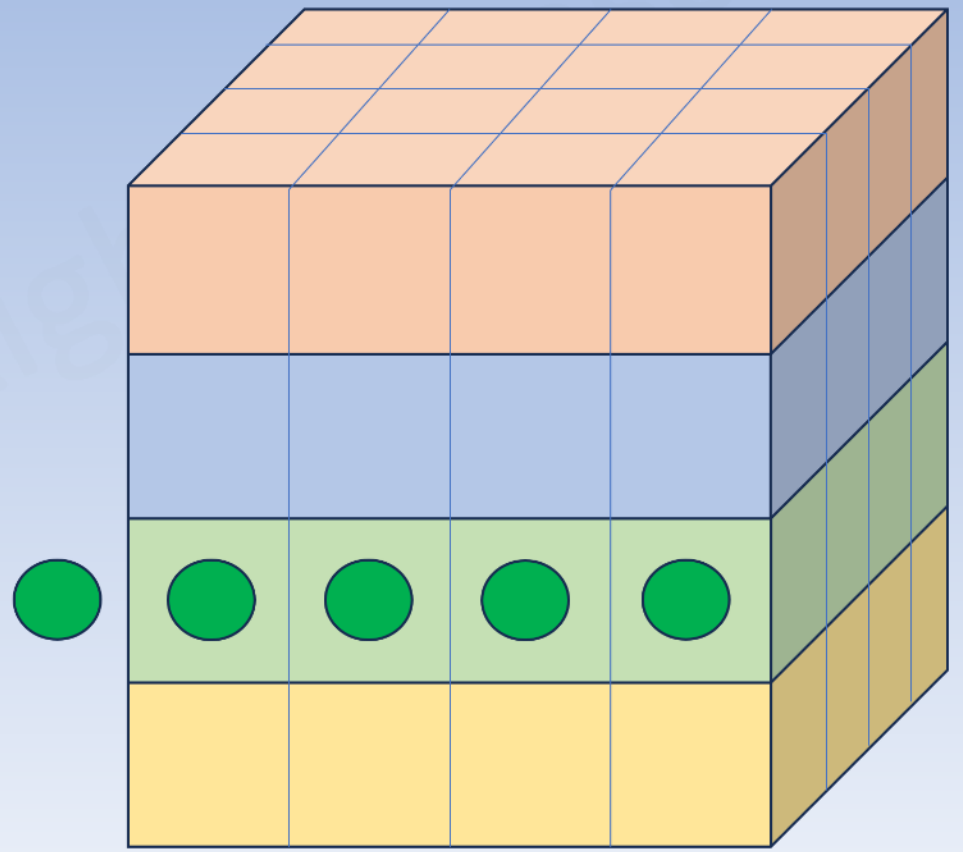
- Dimension 1 - Architecture Layer (where in the ‘V’)
- Dimension 2 – Generality of domain
- Dimension 3 – Maturity
  - RFI
  - Bid
  - Concept Design
  - Prelim Design
  - Detailed design



Architecture, a 3-dimensional view classification

- Knowing what architecture you are building to understand its utility!

Today's discussion here



Clarity SE

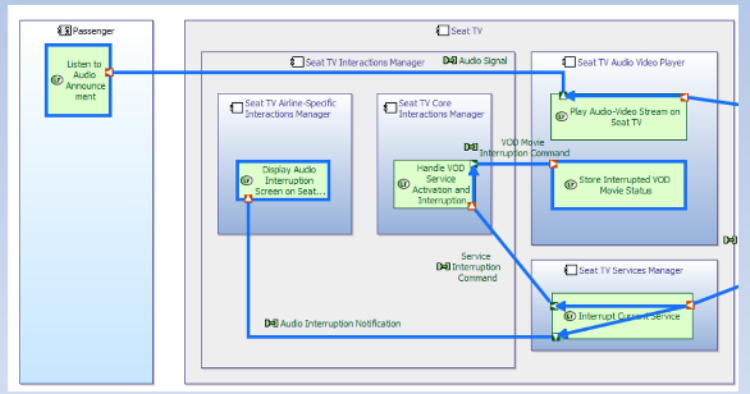
So what about logical architecture?

Clarity SE

Logical Architecture [LA]

Overview

- The objective of the logical architecture is to identify how the system will work without considering specific technology or deployment issues.
- Functions are decomposed and grouped into logical components. Interfaces and functional chain are identified
- The system can be analysed from a number of non-functional perspectives including performance, IVV, safety, etc… and appropriate compromises found
- Logical architecture can be used to
  - Focus first on how the system will function rather than how it will be built
  - Define a general architecture for a product line, system that has multiple variants or multiple deployment configurations
  - Generalise an architecture such that it is resistant to technology obsolescence
  - Manage the breakdown of scope between design teams



Clarity SE

Logical Architecture – What to put in it?

Clarity SE

Logical Architecture – What to put in it?

It Depends

Clarity SE

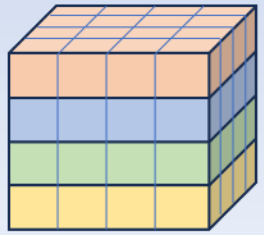
Logical Architecture – What to put in it?

It Depends

- Sure…. but what does it depend on and how do I decide?

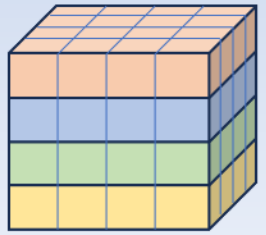
Logical Architecture – Questions to ask?

- Do I need a logical architecture? What value will it add?
- How well is the solution understood already / what is already known?
- How will it help the system through the life-cycle?
- Is the system deployed to multiple site/platforms?
- Is the scope of each sub-team team well understood?
- How much is needed to help the project?



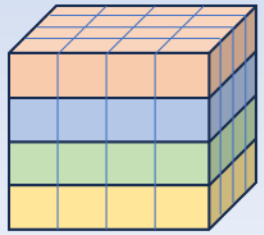
Logical Architecture – Questions to ask?

- Do I need a logical architecture? What value will it add?
  - Consider what is not known/needs management, will impact the physical architecture, and the delivery of the project. E.g.:
    - Scope breakdown
    - Solution class
    - General solution
    - Requirements
    - Interfaces between WPs or presumptive subsystems
    - Overall functional dependencies
    - Overall safety case



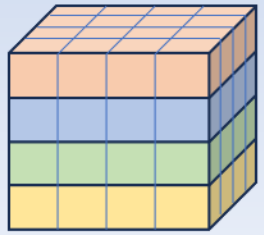
Logical Architecture – Questions to ask?

- How well is the solution understood already / what is already known?
  - If the technical solution is well understood, working through an extra layer of architecture can be expensive with low added value and cause the SE group to lose credibility with SMEs needed to complete the work.



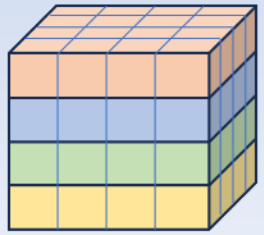
Logical Architecture – Questions to ask?

- How will it help the system through the life-cycle?
  - The architecture relates to requirements, interface documentation, Integration, Verification and Validation scenarios.
    - How much help is needed to establish the specifications
    - How many of the interfaces need to be managed between subsystem delivery teams
    - How much of the overall behaviour needs to be teased out for test scenarios



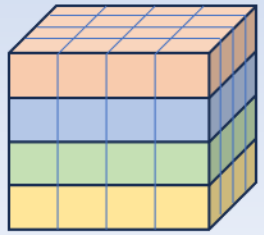
Logical Architecture – Questions to ask?

- Is the system deployed to multiple site/platforms?
  - When a system is deployed to many locations there may be:
    - Variations in the technical solution due to site specific constraints or differing performance needs
    - A variation in brownfield pre-existing systems resulting in different physical architecture
    - A customer decides to split responsibility for different deployments



Logical Architecture – Questions to ask?

- Is the scope of each design team well understood?
  - Where multiple teams are involved that cut across subsystems and life-cycle phases, there needs to be an understanding of who is responsible for what and when, or significant project risks will be realised. This cannot wait until the complete architecture is defined.

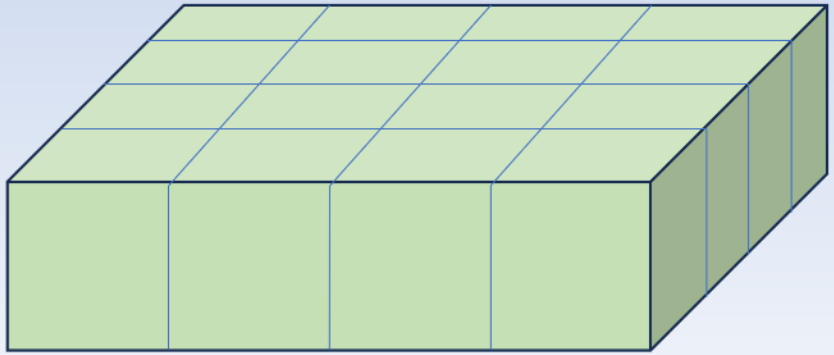


Types of Logical Architecture (LA)

Clarity SE

Types of Logical Architecture

- Scope breakdown
- Solution class / options
- Detailed technology agnostic solution
- Deployment agnostic solution
- Specific solution

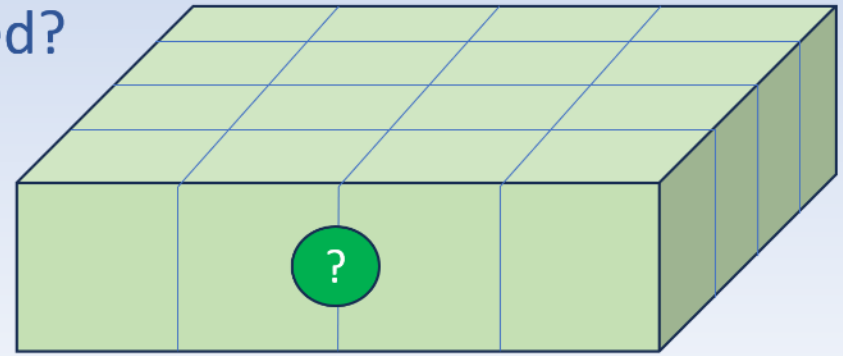


Clarity SE

Types of Logical Architecture - Main decisions to make

Overall Scope and Capabilities should already be clear from the System Analysis (SA) or the top-level project docs

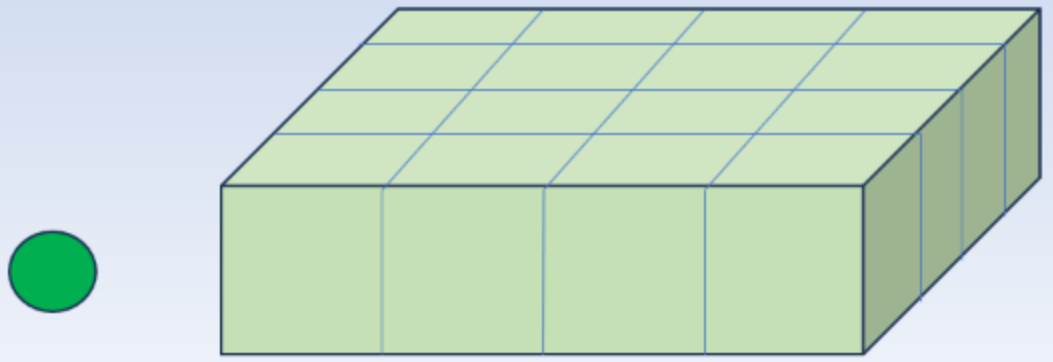
- First pick what type of Logical Architecture do you need?
- Then understand:
  - What are the Logical Components going to represent?
  - What do the Component Exchanges represent?
  - How are the Functions going to be organised?
  - How are Functional Chains broken down?



Clarity SE

Types of Logical Architecture

- Scope Breakdown based LA
  - Logical components are used to define the scope of different organisations scope
  - Interfaces represent contractual IRSs
  - Functions map to contractual functional requirements



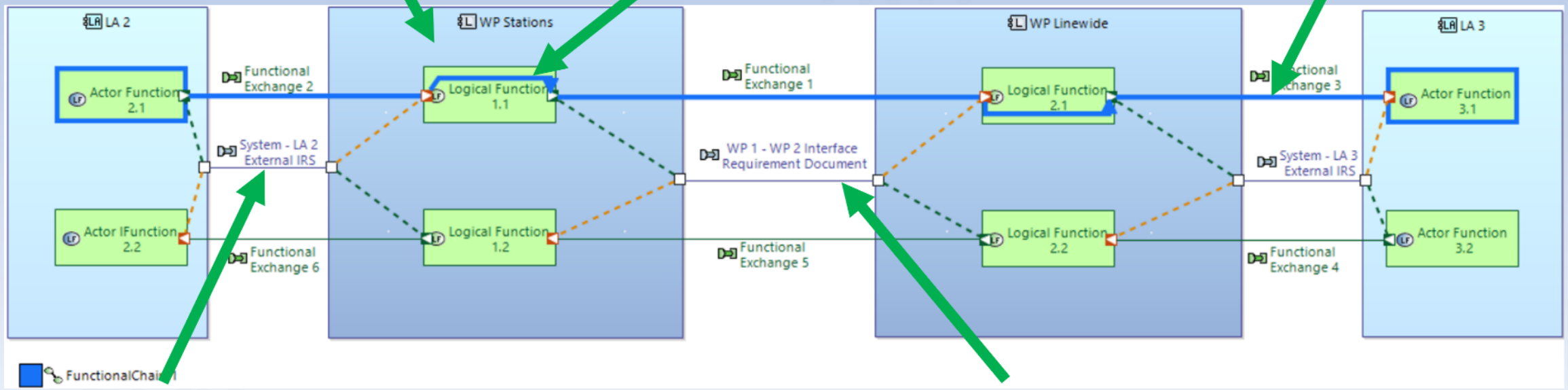
Types of Logical Architecture

- **Scope Breakdown** based LA

Logical Components represent WPs

Functions map to functional requirements for a WP team

Maps to interface requirements



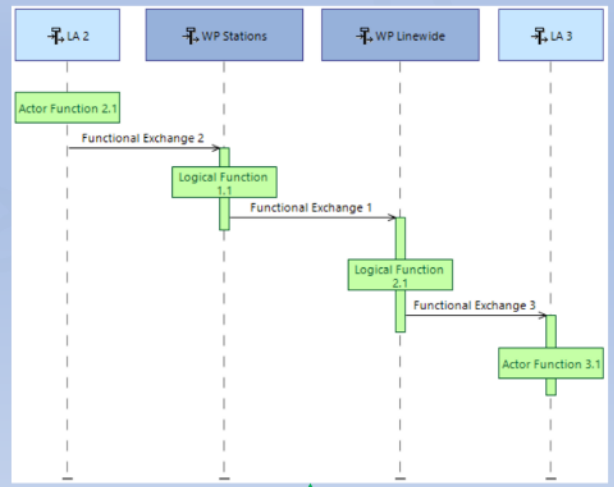
Agreement with external stakeholders

The basis of agreement between different WP teams

Clarity SE

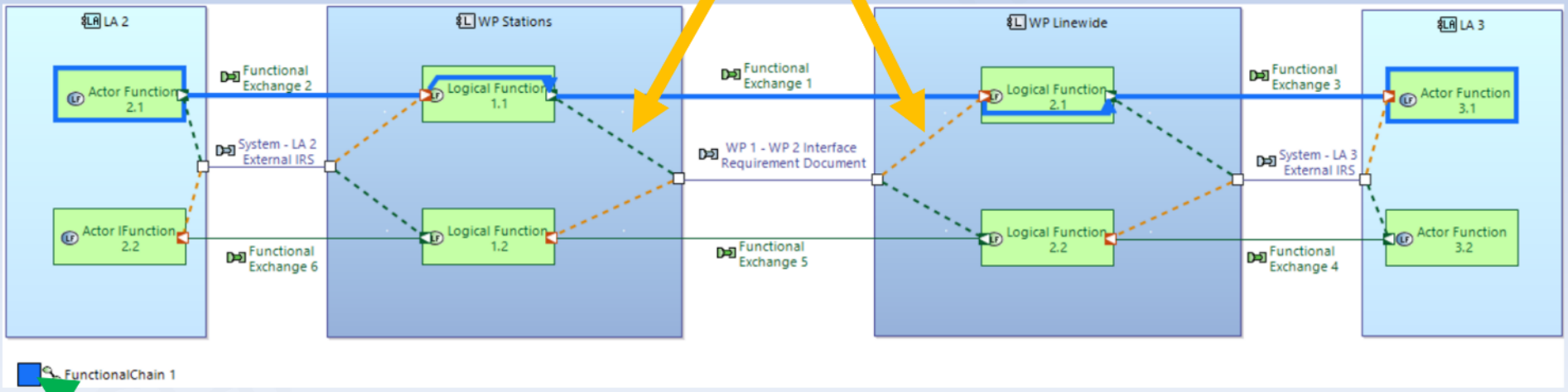
Types of Logical Architecture

- Scope Breakdown based LA



Map to Capability Realisations. Input to scope & requirement workshops

Low Added Value

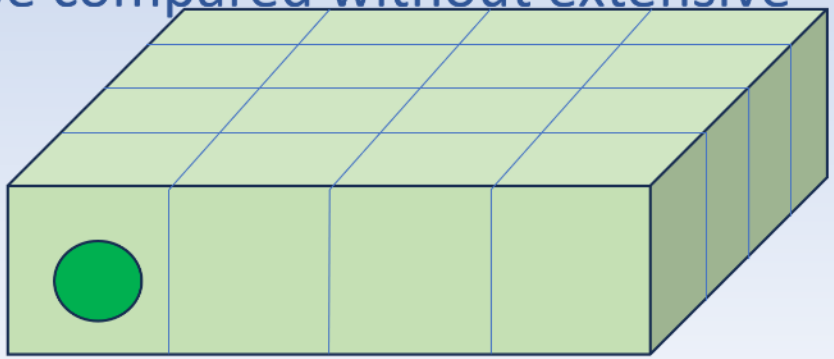


Map to Capability Realisations. Input to scope & requirement workshops

Clarity SE

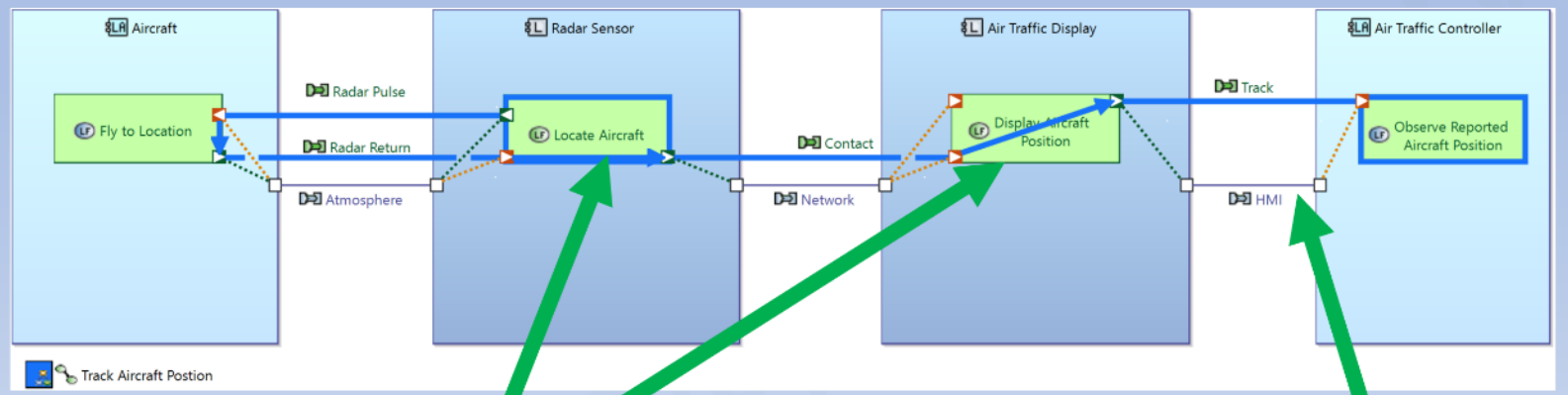
Types of Logical Architecture

- Solution class / options
  - Modelled to match a typical technical solution to a problem without worrying about project specifics
    - E.g. Heavy rail vs Metro vs Tram
  - Used to help identify the general functional requirements for major subsystems, define interfaces that will need to be managed, etc.. Without selecting a specific project or vendor solution
  - Very useful when putting out a tender for contracts to respond to
  - Also useful when solution options need to be compared without extensive design being done.



Types of Logical Architecture

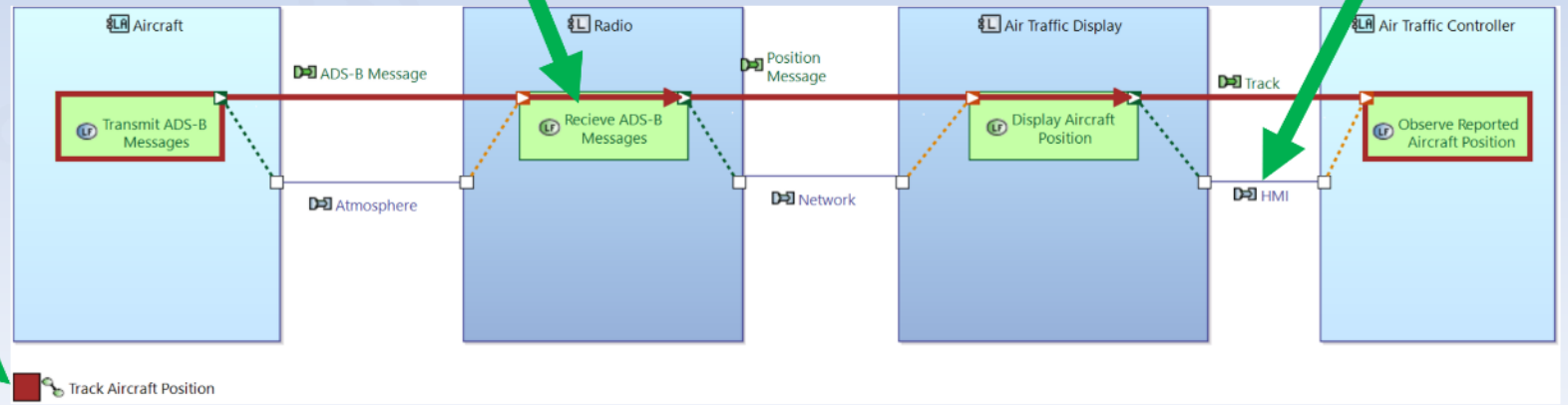
- Solution Class / options based LA



The basis of Top-Level Technical Spec.

Identifies External IRSs

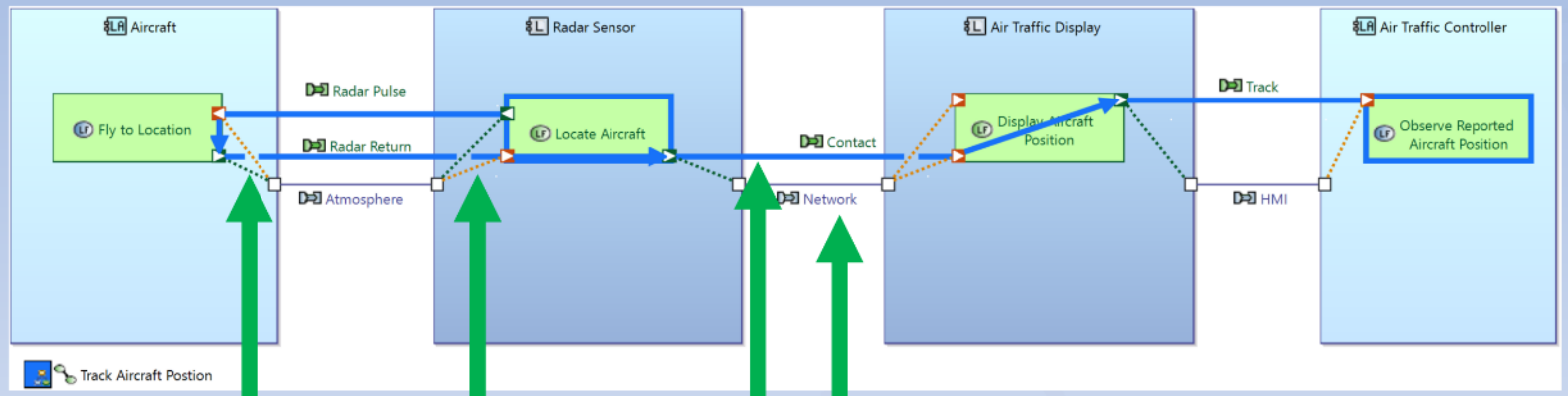
Helps with Top-level technical requirements Analysis, map to system level FCs



Clarity SE

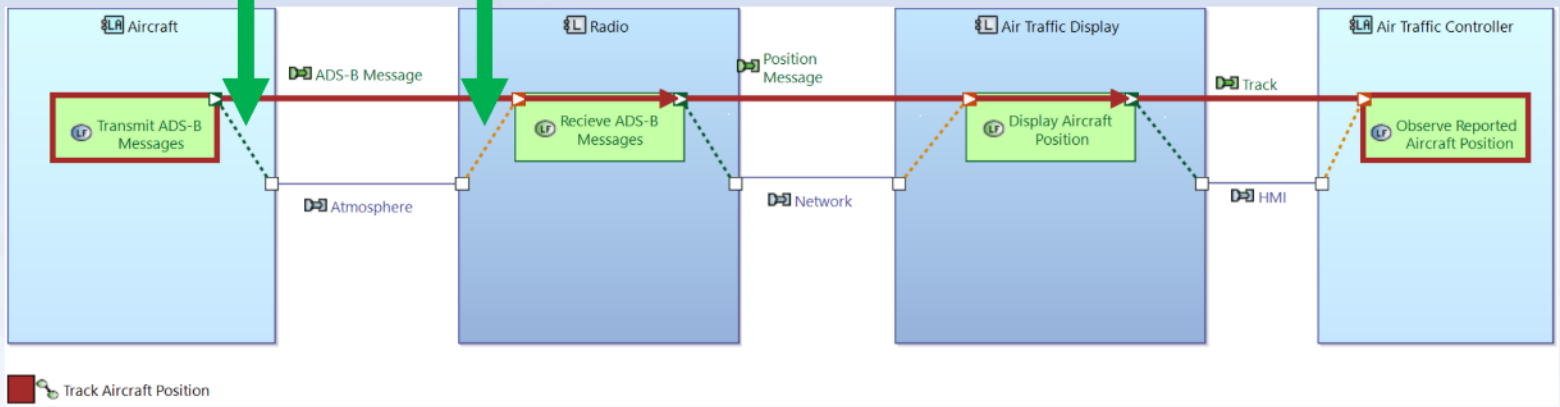
Types of Logical Architecture

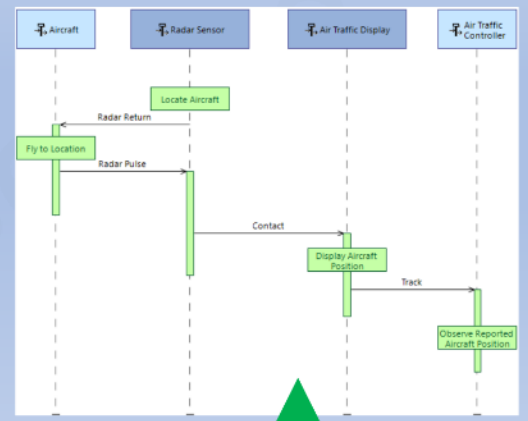
- Solution Class / options based LA



Defines the interface scope

The basis of an internal IRS



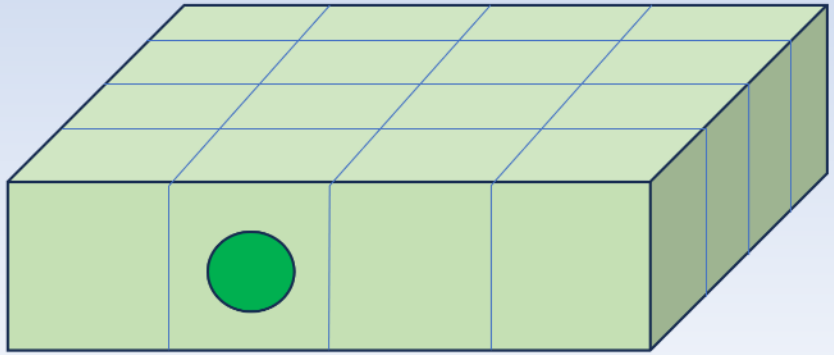


Helps with requirements analysis and evaluating system option performance

Clarity SE

Types of Logical Architecture

- Detailed technology agnostic solution
  - Used when there is a need to capture the programme solution, but in a way that is agnostic to a specific technology.
    - E.g. when you have many ships that are built in different years, are upgraded over time using different technology for the same capability, such as different radars from different suppliers
  - Allows for a common architecture across deployments
  - Leaves the specifics to the physical architectures for each deployment
  - Is resilient to changes in technology, adaptable to generational solution changes

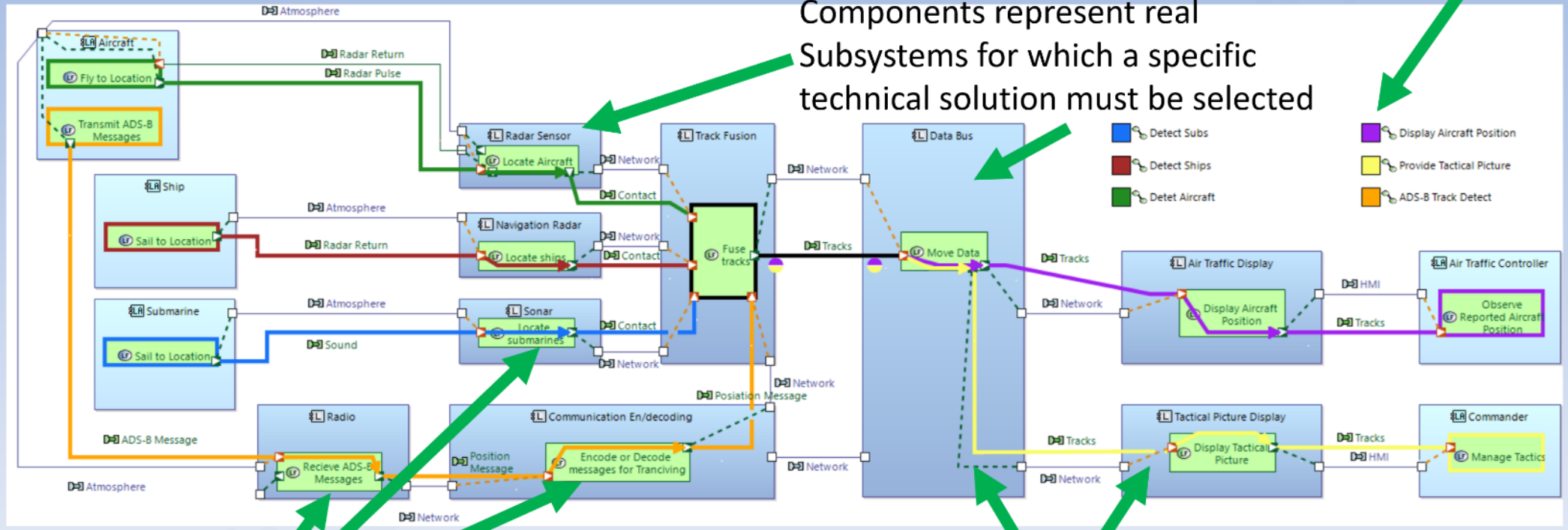


Types of Logical Architecture

- Detailed technology agnostic solution

Functional Chains map to system functions, can included in composite FCs to cover system level FCs

Components represent real Subsystems for which a specific technical solution must be selected



Functions help Allocated Baseline requirements analysis

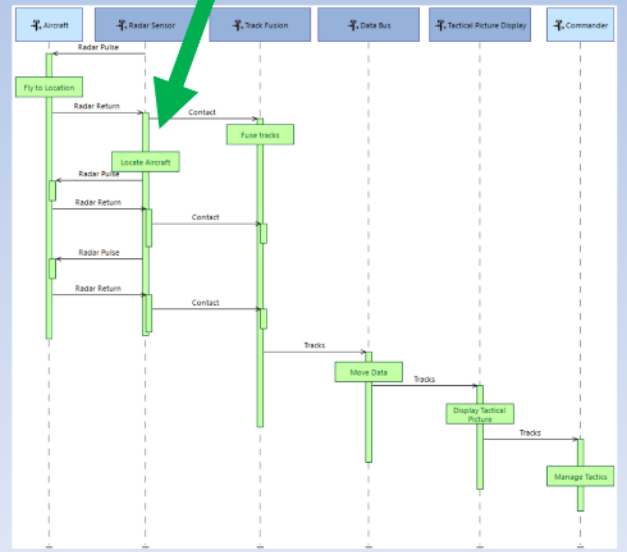
Mapping FEs to Component Exchanges is essential to define interfaces

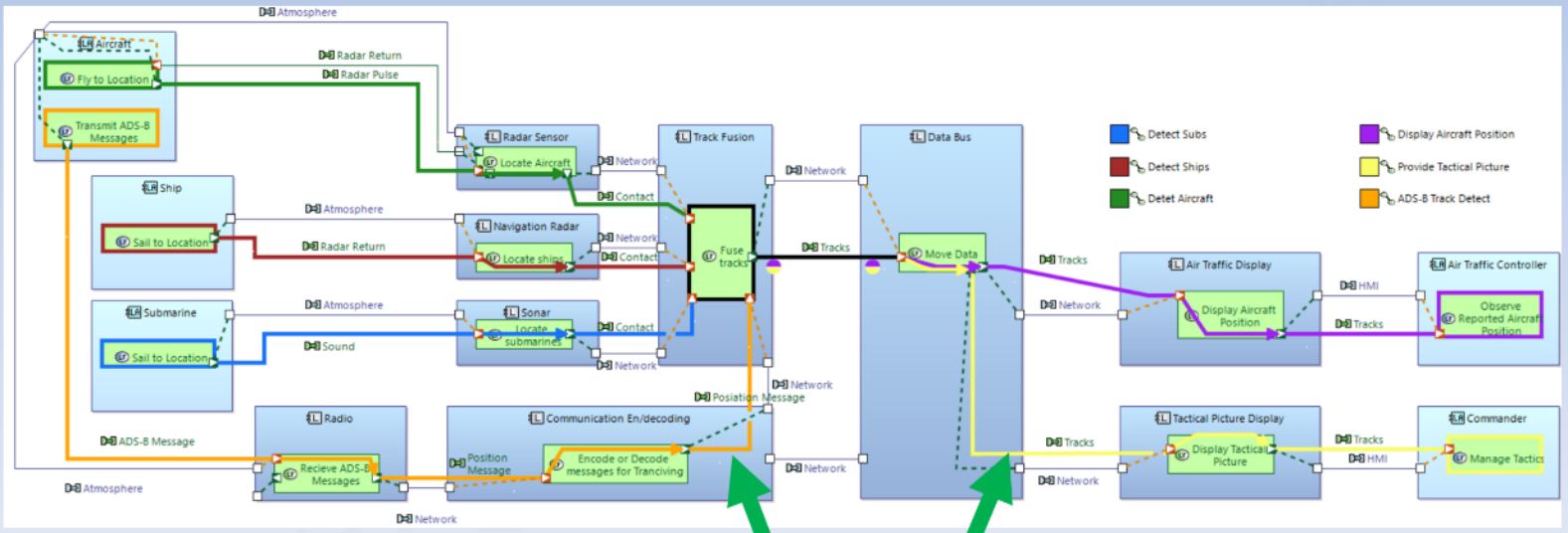
Clarity SE

Types of Logical Architecture

- Detailed technology agnostic solution

Scenarios show the expected dynamic behaviour. They can assist in selecting technologies and in establishing verification cases



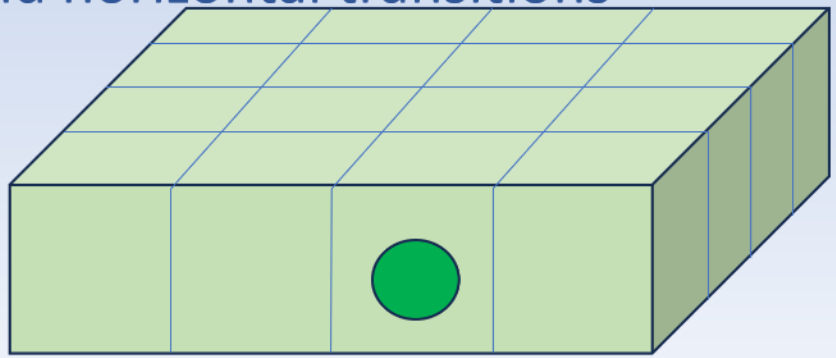


Functional Exchanges map to Allocated Baseline Interface requirements

Clarity SE

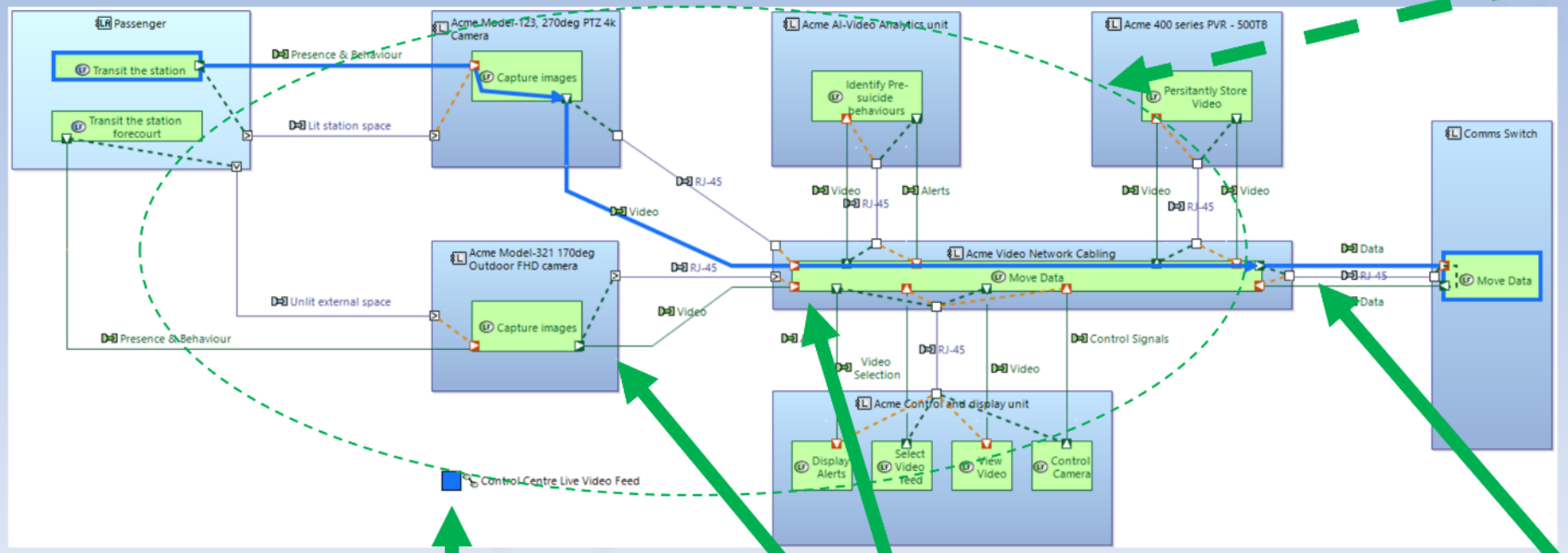
Types of Logical Architecture

- Deployment agnostic solution
  - Used when there is a need to capture the specific project solution, but in a way that is agnostic to a specific deployment.
    - E.g. when you have many train stations, each a needing to fit a slightly different physical site, with a different access, egress, different platforms configuration, etc…
  - Allows for a common architecture across deployments
  - Leaves the specifics to the physical architectures for each deployment
  - Helps set up integration sequences through the functional chains
  - Consider also using the physical architecture and horizontal transitions



Types of Logical Architecture

- Deployment agnostic solution
  - Similar to the technology agnostic solution, with the following delta:



These details are a single subsystem and are well understood already by a single design team.

Functional chains can be useful for integration sequencing

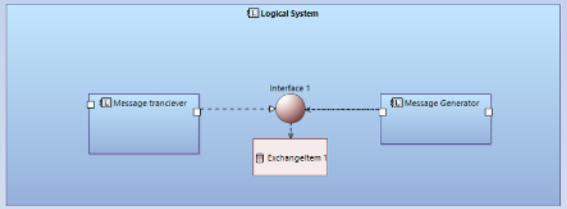
Components represent what components need to be deployed, just not how many

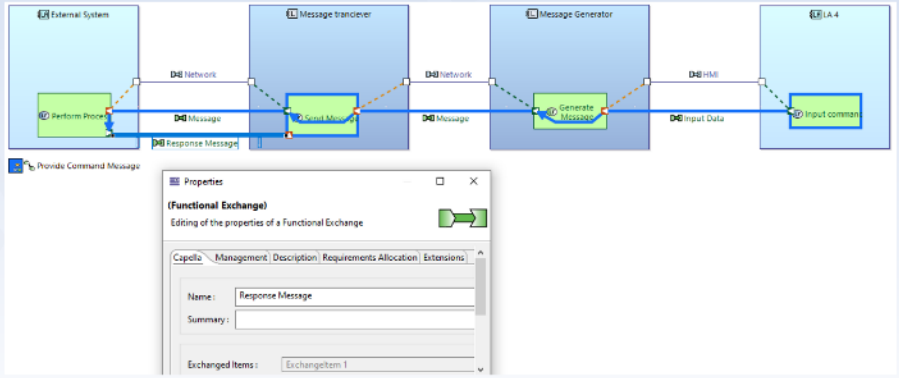
Only some of the interfaces aid management. Others are managed by a single design team. Mapping FEs to Component Exchanges Helps define IDD scope

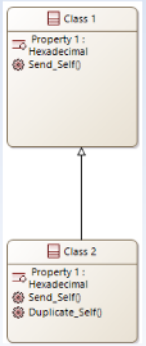
Clarity SE

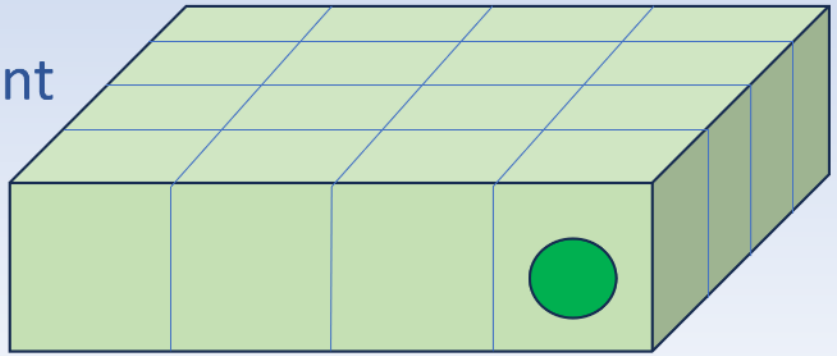
Types of Logical Architecture

- Specific solution LA
  - Used where there is strong alignment between potential logical and physical architectures, such as when the vast majority of components are software.
    - E.g. when the vast majority of components are software and the supporting computer hardware is also well understood
  - Removes the need for a physical architecture
  - should include detailed interfaces using exchange items, many scenarios
  - May extend to state diagrams, class diagrams
  - Consider using a UML tool, audience dependant







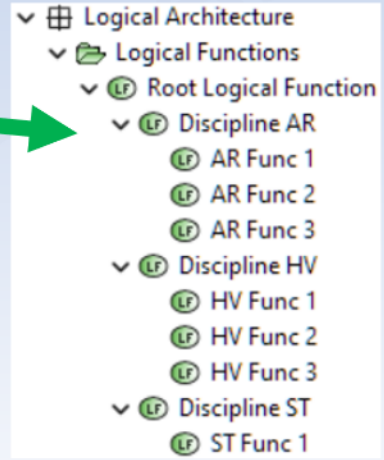


Clarity SE

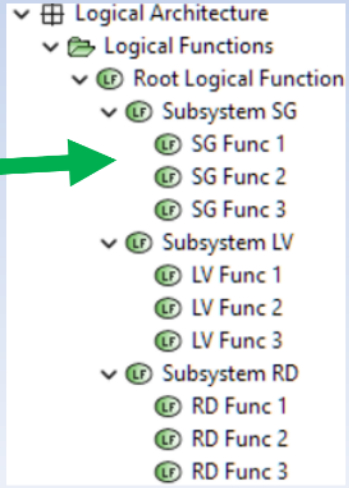
How to Organise Functions

- Function hierarchy
  - Functions need to be easy to find, intuitive for SEs
  - Facilitate efficient conversation with SMEs
  - Avoid risk of duplication, or overlap
  - Appropriate for the LA layer, not tied to organisation in other layers

Organise by discipline where the technical sln is less understood, but the design teams are

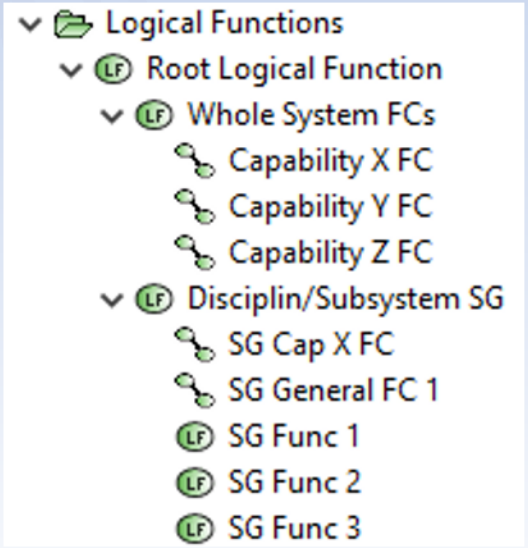


Organise by subsystem where the technical sln is understood



How to Organise Functional Chains

- Where to put them in the Function hierarchy, what has worked for me:
  - Functional chains need to be close to the functions they use
  - Functional chains need to be close to the capabilities they describe
  - Break up FCs realising an SA FC within major subsystems or disciplines as appropriate
  - Create dummy abstract Functions to organise composite FCs



Final questions to ask yourself

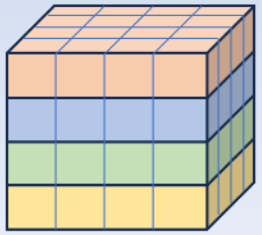
- Still not sure, but tasked with building one?
  - Consider a hybrid or what has been discussed today
  - Use your engineering judgement to determine what you would like the components, functions, Function Chains, etc… to represent and give it go
  - A good test is to ask yourself is:

    "If I use the script interface to generate a spreadsheet from the model, will I get out of it the reports my stakeholders need?"

  - *Remember,*

    "All models are wrong, some are useful"

    "Know what the problem you're trying to solve,
    Know how the model contributes,
    Know when to stop!"



Clarity SE

Thank You.

And please talk to your Systems Engineering professionals!

Clarity SE