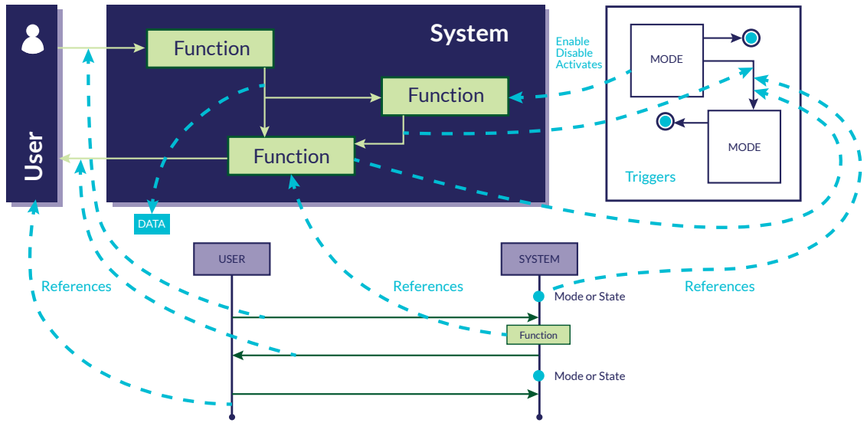
DYNAMIC BEHAVIOUR



ARCADIA

www.clarity-se.org

A TOOLED METHOD TO

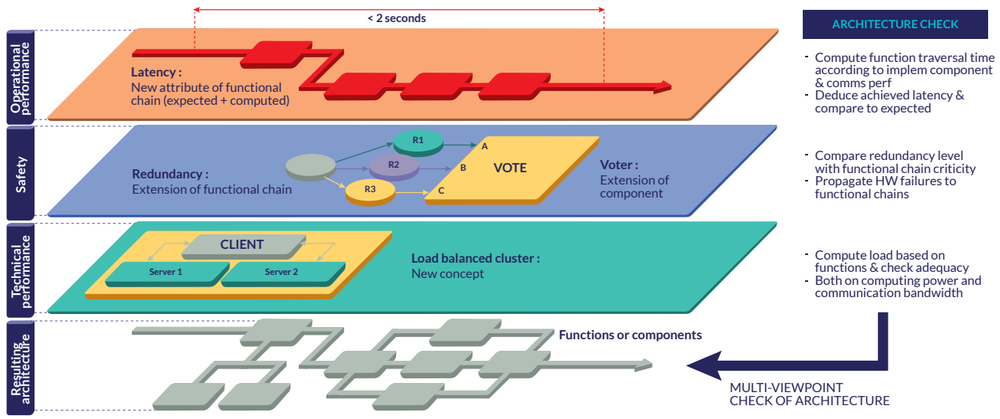
Define, Analyse, Design & Validate System, Software, Hardware Architectures

Verifying & checking solution against Non-functional & Industrial Stakes

Method layers	Performance specific data sample	Safety specific data sample
OPERATIONAL NEED ANALYSIS	Max reaction time to threat	Feared events
FUNCTIONAL/NON FUNCTIONAL NEED ANALYSIS	Functional chain (FC) to react to threat. Maximum allowed latency on FC	Critical functional chains associated to events
LOGICAL ARCHITECTURE DESIGN	Processing & exchanges complexity Functional chains allocation	Redundancy paths securing functional chains
PHYSICAL ARCHITECTURE DESIGN	Resource consumption on FC Resulting computing latency	Common failure modes Fault propagation on FC
CONTRACTS FOR DEVELOPMENT & IVVQ	Allocated resources to satisfy latency	Needed reliability level

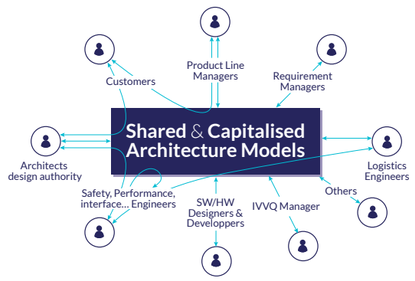
NON FUNCTIONAL VIEWPOINTS (EXAMPLES)

- ✓ Cost & Schedule
- ✓ Interfaces
- ✓ Performance
- ✓ Maintainability
- ✓ Safety/security
- ✓ ...
- ✓ IVVQ
- ✓ Product Policy

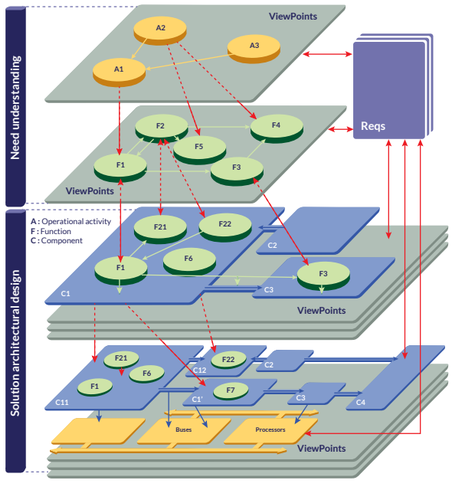


THALES

Supporting Efficient Collaboration in Engineering



Validating/Justifying solution against Operational Need Easing Impact Analysis



Operational Analysis
What the users of the system need to accomplish

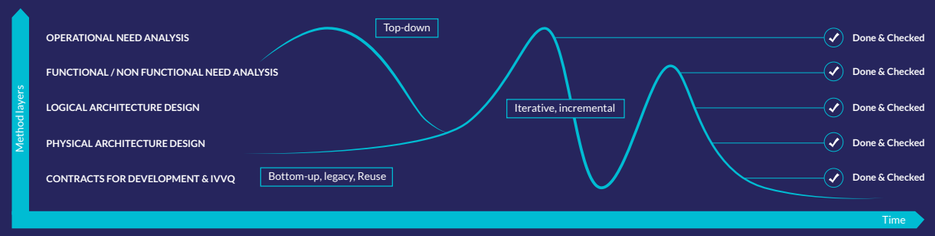
Functional & Non Functional Need
What the system has to accomplish for the users

Logical Architecture
How the system will work to fulfill expectations

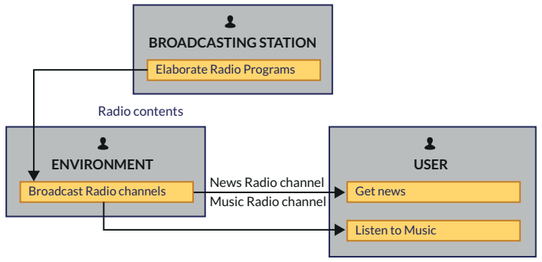
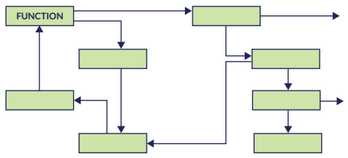
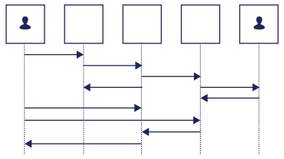
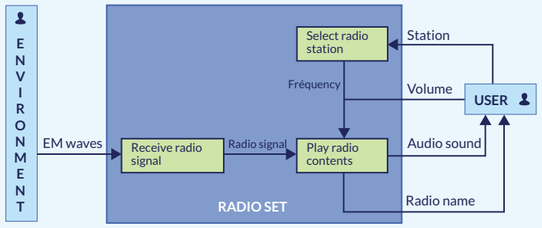
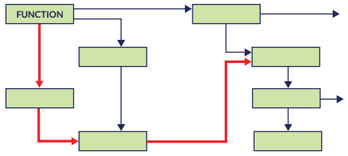
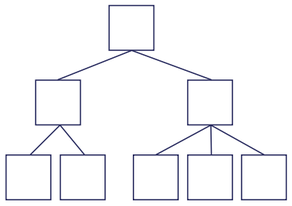
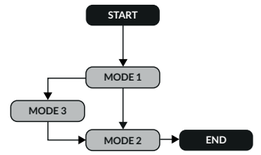
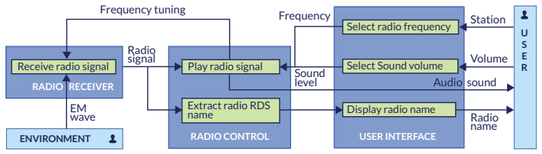
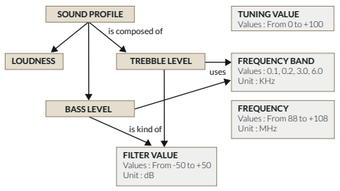
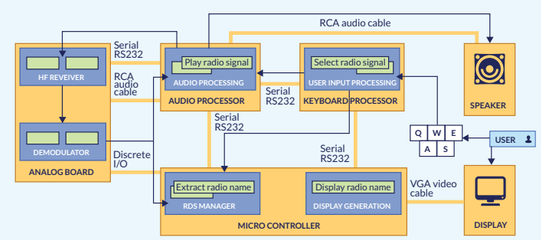
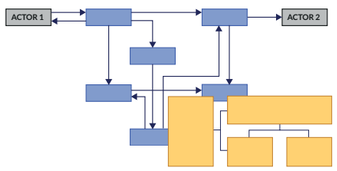
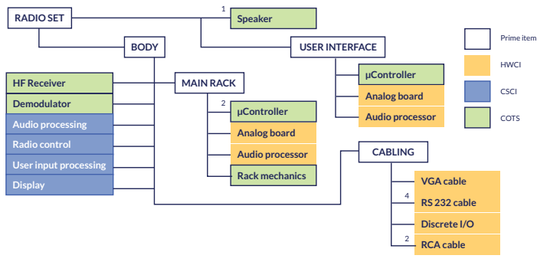
Physical Architecture
How the system will be developed and built

Compatible with most processes

top-down bottom-up, iterative, legacy-based, mixed ...



Design by **vebrand** . www.vebrand.fr . **Paris** & **Nice**

METHOD STEPS	TASKS	SAMPLE MODEL	CONCEPTS	DESCRIPTION MEANS
NEED				
Customer Operational Need Analysis What the users of the system need to accomplish	✓ Define operational capabilities ✓ Perform an operational need analysis		- Operational capabilities - Actors, operational entities - Actor activities - Interactions between activities & actors - Information used in activities & interactions - Operational processes chaining activities - Scenarios for dynamic behaviour	 **Dataflow**: functions, op. activities interactions & exchanges **Scenarios**: actors, system, components interactions & exchanges 
System/ SW/HW Need Analysis What the system has to accomplish for the Users	✓ Perform a capability trade-off analysis ✓ Perform a functional and non-functional analysis ✓ Formalise and consolidate requirements		- Actors and system, capabilities - Functions of system & actors - Dataflow exchanges between functions - Functional chains traversing dataflow - Information used in functions & exchanges, data model - Scenarios for dynamic behaviour - Modes & states	 **Functional chains, operational processes** through functions & op. activities   **Modes & states** of actors, system, components **Breakdown** of functions & components
SOLUTIONS				
Logical Architecture Design How the system will work so as to fulfil expectations	✓ Define architecture drivers and viewpoints ✓ Build candidate architectural breakdowns in components ✓ Select best compromise architecture		*SAME CONCEPTS, PLUS :* - Components - Component ports and interfaces - Exchanges between components - Function allocation to components - Component interface justification by functional exchanges allocation	 **Data model**: dataflow & scenario contents, definition & justification of interfaces 
Physical Architecture Design How the system will be developed & built	✓ Define architectural patterns ✓ Consider reuse of existing assets design a physical ✓ Design a physical reference architecture ✓ Validate and check it		*SAME CONCEPTS, PLUS :* - Behavioural components refining logical ones, and implementing functional behaviour - Implementation components supplying resources for behavioural components - Physical links between implementation components	 **Component wiring:** all kinds of components
Development Contracts What is expected from each designer/ sub-contractor	✓ Define a components IVVQ strategy ✓ Define & enforce a PBS and component integration contract		- Configuration items tree - Parts numbers, quantities - Development contract (expected behaviour, interfaces, scenarios, resource consumption, non-functional properties…)	**Allocation** of op.activities to actors, of functions to components, of behav.components to impl.components, of dataflows to interfaces, of elements to configuration items