Supporting Efficient Collaboration in Engineering

Validating/Justifying solution against Operational Need Easing Impact Analysis

Compatible with most processes top-down bottom-up, iterative, legacy-based, mixed...
**Customer Operational Need Analysis**

- Define operational capabilities
- Perform an operational need analysis

**System/ SW/HW Need Analysis**

- Perform a capability trade-off analysis
- Perform a functional and non-functional analysis
- Formalise and consolidate requirements

**Logical Architecture Design**

- Define architecture drivers and viewpoints
- Build candidate architectural breakdowns in components
- Select best compromise architecture

**Physical Architecture Design**

- Define architectural patterns
- Consider reuse of existing assets design a physical
- Design a physical reference architecture
- Validate and check it

**Development Contracts**

- Define a components IVVQ strategy
- Define & enforce a PBS and component integration contract
CONCEPTS

- Operational capabilities
  - Actors, operational entities
  - Actor activities
  - Interactions between activities & actors
  - Information used in activities & interactions
  - Operational processes chaining activities
  - Scenarios for dynamic behaviour

- 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

SAME CONCEPTS, PLUS:

- Components
- Component ports and interfaces
- Exchanges between components
- Function allocation to components
- Component interface justification by functional exchanges allocation

SAME CONCEPTS, PLUS:

- Behavioural components refining logical ones, and implementing functional behaviour
- Implementation components supplying resources for behavioural components
- Physical links between implementation components

- Configuration items tree
  - Parts numbers, quantities
  - Development contract (expected behaviour, interfaces, scenarios, resource consumption, non-functional properties...)

DESCRIPTION MEANS

Dataflow: functions, op. activities interactions & exchanges

Scenarios: actors, system, components interactions & exchanges

Functional chains, operational processes through functions & op. activities

Breakdown of functions & components

Data model: dataflow & scenario contents, definition & justification of interfaces

Component wiring: all kinds of components

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
**Verifying & checking solution against Non-functional & Industrial Stakes**

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

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

**ARCHITECTURE CHECK**
- Compute function traversal time according to implemented component & communications performance
- Deduce achieved latency and compare to expected
- Compare redundancy level with functional chain criticality
- Propagate HW failures to functional chains
- Compute load based on functions & check adequacy
- Both on computing power and communication bandwidth

**MULTI-VIEWPOINT CHECK OF ARCHITECTURE**