Arcadia and Capella on the Field: Real-World MBSE Examples

MBSE Symposium, Systems Engineering Society of Australia, Canberra
October 27th, 2014

Stéphane Bonnet, Fabrice Lestideau, Jean-Luc Voirin
Current Systems Engineering Practices

ENGINEERING KEY QUESTIONS

- How is the customer need received? How are its **consistence and feasibility checked**?

- Which are the **engineering phases in the solution elaboration**, how are they related?

- **How is complexity managed?**

- How are different alternatives evaluated, **how do the specialists collaborate**?

- **How is the solution justified** w.r.t. the need and the different constraints?

FACTS

- Needs and solutions are more complex, more stakeholders, more constraints, less time

- The approach Doors / Word / Visio / Excel reaches limits

- Manual processes are not compatible with agility and short loops
Model-Based Engineering Method for Architectural Design

Graphical Modelling Workbench supporting Arcadia
Arcadia: Model-Based Method for Architectural Design

**NEED UNDERSTANDING**

- **What the users of the system need to accomplish**
- **What the system has to accomplish for the users**
- **How the system will work to fulfill expectations**

**SOLUTION ARCHITECTURAL DESIGN**

- **How the system will be developed and built**

Diagram showing relationships between different components and processes, with labels for ViewPoints, Processors, Buses, and Reqs.
Arcadia: Multi-Viewpoint Trade-Off Analysis

functions

viewpoints

solution architecture

architect

specialty engineering: safety, perf, security, ...

evaluation rules

product line manager, etc.

sub-contractors

This document is not to be reproduced, modified, adapted, published, translated in any material form in whole or in part nor disclosed to any third party without the prior written permission of Thales. © THALES 2014 – All rights reserved.
Guidance
[Embedded methodological browser]

Complexity management
[Abstraction via computed information]

Productivity tools
[Automated transitions and diagram creation accelerators]

Model Analysis & Navigation
[Model validation, semantic browser]

Multi-criteria analysis
[Viewpoints and management framework]

First operational deployments in 2009
Used on most major engineering projects
Open sourced in 2014
Capella: Embedded Methodological Guidance

Overview of SAR

Define Stakeholder Needs and Environment
Capture and consolidate operational needs from stakeholders. Define what the users of the system have to accomplish. Identify entities, actors, roles, capabilities, activities, concepts.

System Analysis
Formalize System Requirements
Identify the boundary of the system, consolidate requirements. Define what the system has to accomplish for the users. Model functional dataflows and dynamic behaviour.

Logical Architecture
Develop System Logical Architecture
See the system as a white box: define how the system will work so as to fulfll expectations. Perform a first trade-off analysis.

Physical Architecture
Develop System Physical Architecture
How the system will be developed and built. Software vs. hardware allocation, specification of interfaces, deployment configurations, trade-off analysis.

Formalize Component Requirements
Manage industrial criteria and integration strategy. What is expected from each designer/ subcontractor. Specify requirements and interfaces of all configuration items.

Physical Architecture

Logical Architecture

EPBS

Transition from Logical Functions

Perform an automated derivation of Logical Functions

Create Traceability Matrix

Refine Physical Functions, describe functional exchanges

EPBS: Create a new Functional Block diagram

EPBS: Create a new Functional Diagram (Blank diagram)

EPBS: Create a new Functional Scenario

Define Physical Components and Actors, Manage deployments

Perform an automated derivation of Logical Actors

Perform an automated derivation of Logical Components

EPBS: Create a new Physical Component breakdown diagram

EPBS: Create a new Physical Architecture diagram

Create a new Physical Component (Logical Component Matrix)

Allocate Physical Functions to Physical Components

Delegate Logical Interfaces and create Physical Interfaces

Enrich Physical Scenarios

Communicate with Ships

Determine Position and steer Aircraft

Route Aircraft towards Defender Ship

Manage Risky Events

Drop time top

Acquire radars track

Acquire BLR image

Allocate Physical Functions

Delegated Functional Interface Creation

Physical Interface Creation

Acquire BLR image

Allocate Physical Functions

Delegated Functional Interface Creation

Physical Interface Creation

Acquire BLR image
Use Case 1:

Managing System Design Complexity
Managing System Design Complexity

Context

- Issues in the latest phases of operational validation
- Very good design documents, but in silos

MBSE usage

- 1 man month to reverse a first level of detail in a model, based on existing documents
- **First time overall views have been available**
  - Good support for discussion
  - Visualization of transverse functional chains
Managing System Design Complexity

275 Functions
(230 Leaves)

578 Functional
Exchanges between leaf functions

5 levels of decomposition
Contextual Diagrams: Low-level internals, high-level neighborhood
Managing System Design Complexity

Challenge: Build and maintain simplified views

How to analyze transverse topics?
How to get transverse overviews?
Computed Diagrams: High-level Functions, Low-level Exchanges
Children of F21 and F22 not displayed

Ports on F21 and F22 are graphically computed (they actually belong to the children of F21 and F22)
Children of F2 not displayed

Ports on F2 are graphically computed (they actually belong to the children of F21 and F22)
Children of F1 and F2 not displayed

Ports on F1 and F2 are graphically computed (they actually belong to the children of F21 and F22)
Tag-based simplification mechanism: each exchange can be marked with several «grouping» tags

Computed graphical simplifications free engineers from tedious and error-prone maintenance of abstraction levels
End-to-end visualization of Functional Chains
Use Case 2:

MBSE-based Change Management
Use Case 2: Change Management

Context

- Maritime Patrol Program delivered to the Customer
- New functionalities asked by the Customer

MBSE usage

- Up-to-date model of the delivered System available
- Modification of the model in order to:
  - Estimate feasibility, cost and risks
  - Drive developments and IVVQ
- Product line management
A regular layout / reading pattern across all diagrams

- Multiple contributors modelling the same way
- Facilitates first access to diagrams
- Eases diagram review
- Allows quick inconsistency detection
Need Representation based on Delivered Solution

SSS: Need

PIDS: Reverse Engineering from Software Specification
Managing Change: Feasibility and Risks

New Customer needs

Existing Functional Chains

Impacted Functions

New Functional Chains
**COST Analysis Viewpoint**

Elementary work decomposition, and estimation of an average development cost for each category of function

- Panels
- External / Internal Interfaces
- Data Memorisation
- Processing Complexity

### Computed Data: Estimated Cost

**Capella Outputs (model export)**
Use Case 3:

Multi-Level Engineering
Use Case 3: Multi-Level Engineering and Automated Transitions

Context

- Complex systems with full Thales responsibility (from Mission System to SW Component)

MBSE usage

- Setup a global, multi-level engineering approach
  - Joint effort with Thales Airborne Systems / Thales Corporate to specify and develop an automated, iterative transition
  - Incubation on two projects
  - Now integrated in the product and used in other contexts
System Physical Architecture

Subsystem Need Analysis

Subsystem Logical Architecture

Subsystem Physical Architecture

CO-ENGINEERING

CO-ENGINEERING

SUBSYSTEM CONTRACT

SUBSYSTEM ARCHITECTURAL DESIGN INITIALIZED
Multi-Level Engineering and Automated Transitions

System Physical Architecture

Subsystem Need Analysis

Subsystem Logical Architecture

Subsystem Physical Architecture

Multi-Level Engineering and Automated Transitions

LINKER

Computed system - subsystem traceability

SUBSYSTEM CONTRACT

SUBSYSTEM ARCHITECTURAL DESIGN INITIALIZED

Co-EN
Use Case 4:

Model-driven IVV
Requirements are clarified with Functional Chains

Test Procedures are linked to Functional Chains
Define operational content expected for each project milestone

Deduce functional content and components to be delivered

Define components versions and content

Based on an (non open source) IV&V viewpoint on top of Capella
Based on an (non open source) IV&V viewpoint on top of Capella
Red: Delayed, missing
Grey: expected in further version

Based on an (non open source) IV&V viewpoint on top of Capella
Based on an (non open source) IV&V viewpoint on top of Capella

Release management viewpoint:
Automated visualization of versions based on activation of diagram layers
Based on an (non open source) IV&V viewpoint on top of Capella

Developed Version 1
Available elements in BLUE
Based on an (non open source) IV&V viewpoint on top of Capella

Developed Version 2
Available elements in CYAN
Developed Versions 1 & 2
Common available elements in GREY

Based on an (non open source) IV&V viewpoint on top of Capella
**Compare Planned vs Developed versions**

---

**Based on an (non open source) IV&V viewpoint on top of Capella**

**IVV in Progress: Ups and Downs**
Lots of Other Different Use Cases

And more to come!
Thank you for your attention!

Any Questions?