Model Driven Requirements Development using SA, Tau and DOORS
The Goal

A Systems & Software Engineering Organization that is:

– Predictable
  • Perform on target and do not confront stakeholders with surprises

– Competitive
  • Make the right choices for your product and deliver on time

– Profitable
  • Work cost efficiently and deliver for the right price

– Compliant
  • Comply with relevant industry or government regulations
Systems Complexity Often Results in Poor Project Delivery

- Late design breakage at Systems Level
- Target Delivery Date
- Actual Delivery Date
- Concept
- Development
- Production
- Test
- Support
Ask Yourself this…

What is the cost to the organization if:

– We design and test against the wrong version of the requirement?
– Completely miss a customer need or misinterpret due to incomplete or incorrect visibility to the information hierarchy?
– Parent requirement is changed and affected organizations do not get visibility to that change?
Systems Engineering is a Club Sandwich
Models Bridge Layers of Requirements

- Needs (problem)
- Requirements (problem)
- Requirements (solution)
- Requirements (solution)

- Statement of need
- Goal / Usage modeling
- User Requirements
- Functional modeling
- System Requirements
- Architectural Modeling
- Subsystem Requirements

- Functional modeling
- Architectural Modeling
- Tau
- DOORS

© Telelogic AB

6
Let’s Drilldown into a Requirements Development Workflow…

Needs (problem)

Requirements (problem)

Requirements (solution)

Requirements (solution)

Statement of need

DOORS

Goal / Usage modeling

Functional modeling

User Requirements

System Requirements

Subsystem Requirements

DOORS

Goal / Usage modeling

Functional modeling

User Requirements

System Requirements

Subsystem Requirements

Architectural Modeling

SA

Tau
A Requirements Development Workflow Scenario

In DOORS

Requirements Engineers

Capture and evolve requirements.

In System Architect

Analysts

Goal / Usage modeling

In Tau

Functional modeling

Automation Testing

In DOORS

In System Architect

In Tau

Telelogic AB

© Telelogic AB
Two Workflows of Many

• Model-Driven Requirements
  – Model is used to discover, define, refine and understand system
  – Requirements structure created from model information
  – Predominantly trace relationships
  – Generally used earlier in the development process
  – Then moves to “flow down” stage

• Specification Flow-Down
  – Specification is supplied and pre-ordained (contractor/supplier)
  – Design is performed in modeling tool
  – Show “satisfy”, “derive”, “trace” kinds of relationships
  – Generally used later in the development process
First we must understand the problem space…

In DOORS

Requirements Engineers

Capture and evolve requirements.

In System Architect

Analysts

Goal / Usage modeling

In Tau

Functional modeling

In DOORS

In System Architect

In Tau
Problem Analysis & Planning

understand problem ... reduce risk

- Map objectives and goals
  - establish scope
- Understand the “as is” solution in terms of:
  - people
  - processes
  - applications
  - information
  - services
- Plan changes in line with objectives and goals
  - roadmaps
- Develop and maintain architecture
  - Common data dictionary
Modeling the Enterprise Architecture

enterprise blueprint... gain understanding

• Model the key domains and their relationships:
  – Problem space, Strategy, Applications, Infrastructure, Data
• Relate to the overall “as-is” infrastructure and its elements
• Establish common data dictionary
Goal / Usage modeling

Link with Requirements in problem space

- Link Model Artifacts to Requirements in DOORS
- Sync DOORS and SA

Send Model Artifacts to DOORS – Diagrams, Symbols, or Definitions
**In SA Graphically “Explore” the Links from DOORS**

**What** is tied to Requirements?

**What** is driving Requirements?

**Where** may we have inconsistencies?

**Where** do we have redundancies?

**How** does a change in Requirements affect Enterprise Architecture?

**How** does a change in the Architecture affect Requirements?
Evolve requirements from knowledge gained…

**In DOORS**
- Requirements Engineers
- Capture and evolve requirements.

**In System Architect**
- Analysts
- Goal / Usage modeling

**In Tau**
- Functional modeling
The Glue That Brings It All Together

Requirements Management Ensure End-To-End Visibility

- Requirements persistent at all levels of decomposition
- Assess impact to reduce risk as a result of requirement change
- Adapt to change throughout all level of your traceability matrix
- Incorporate test and QA into the process early
- Find gaps in traceability
- Ensure everything is accounted for

<table>
<thead>
<tr>
<th>User Requirements</th>
<th>Technical Requirements</th>
<th>Test Requirements</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>URS1</strong></td>
<td>Users with standard access shall not be able to restrict administrator access from any project element.</td>
<td>Users with standard access shall not be able to restrict an administrator's access from any project element.</td>
</tr>
<tr>
<td><strong>URS2</strong></td>
<td>The concept of access (control) shall be changed from restricting user access to granting user access.</td>
<td>Users with standard access shall not be able to restrict an administrator's access from any project element.</td>
</tr>
<tr>
<td><strong>URS3</strong></td>
<td>Administrators shall be able to restrict who can save a major baseline of a module.</td>
<td>Administrators should be able to grant group-access rights for certain things, like saving major and/or minor baselines of a module, a group of selectable modules, an entire project, or a group of projects.</td>
</tr>
<tr>
<td><strong>URS4</strong></td>
<td>Administrators shall be able to restrict who can save a major baseline of a module.</td>
<td>Administrators should be able to</td>
</tr>
</tbody>
</table>
Traceability verification or “completeness”

Increases customer confidence

Orphan reports & traceability reports show “missing” links

Creation and deletion of links is recorded in history
Textual Requirements can be limiting…

• Too many documents (Standards, User Requirements, Functional, System, Test documents…)
• Documents reference each other causing the reader to “jump” from document to document.
• Textual requirements can be misinterpreted, or have ambiguous meaning.
• Documents have a high level of repetitiveness between them in order to provide context
  – Additional need to interpret the context and even more risk for miss interpretation and ambiguities.
• While requirements management helps focus and guide the reader and not overwhelm them with unnecessary detail, it makes it really difficult to get a high level “picture” of the system, and still easily drill down to the details when needed.
How do Models Help Clarify Requirements?

Unlike requirements

- SysML/UML allow requirements and system engineers to collaborate and produce a graphical system level specification from the textual requirements
- SysML/UML models are easily verified, communicated and updated
- Model elements inconsistencies are flagged across the different diagrams
- Model elements can be checked both statically and dynamically
- Data dictionary gets created automatically and is checked against while specifying the system
- Logic of the model can be dynamically executed
- Using a formal language helps discover inconsistencies in the description of the system because it forces systems engineers to consider many questions which are difficult to keep track of in textual requirements.
Is textual requirements obsolete?

No.

- Textual requirements and Models complement each other
- Diagrams help clarify understanding of requirements
- Modeling can help identify gaps and misunderstandings
- A formalized but flexible graphical notation enables expressive, ‘people friendly’ diagrams
Unparalleled DOORS Integrations
Align development with requirements

- Extend Requirements Engineering to development
  - Work seamlessly with requirements and models
  - Easily establish links and traceability

- Visualize how models and code relate to requirements
  - Requirements View in model

- Establish end-to-end traceability
  - Ensure regulatory compliance
  - Documented audit trail
DOORS trace view with model elements
...or view with Traceline...
Assess functions against goals and stated problem being solved…

In DOORS

Requirements Engineers

Capture and evolve requirements.

In System Architect

Analysts

Goal / Usage modeling

In Tau

Functional modeling
Functional Analysis
improve accuracy ... increase quality

- Start from the problem space from System Architect
  - Capture understanding of the overall purpose for development
- Visualize the significant participants and what they need
- Determine what is involved in meeting their needs
- Maintain full traceability
- Simulate and test as a whole
Systems Modeling

Telelogic Tau

Capabilities

- Analyze, design, develop, and test systems and applications using UML 2.1, SysML, MDA, SOA, and DoDAF
- Prove systems design by validating, verifying, and testing concurrent with development
- Provide requirements-driven round-trip code support for Java, C++, C#

Benefits

- Deliver on time and on budget by finding design errors as they occur
- Satisfy regulatory requirements by maintaining complete audit and development trails
- Keep large, diverse teams in sync with powerful collaboration features

“Tau enables our architects, systems engineers, and software developers to improve the development process. We can use a single tool for analysis, design, and auto code generation.”

- Hughes Network Systems
System & Software Design

**System/Software Requirements**

- Models based on requirements
- Model simulation improves understanding & accuracy
- Common language improves communication & reuse
- Role-based usage model improves efficiency
Reqt Gaps in Tau

Functional modeling
Functional modeling

Reqts Coverage in Tau
Let’s Drilldown into the Architecture …

Needs (problem)
Modeling layer
Requirements (problem)
Modeling layer
Requirements (solution)
Modeling layer
Requirements (solution)

DOORS

DOORS

DOORS

Goal / Usage modeling

Functional modeling

Architectural Modeling

Statement of need

User Requirements

System Requirements

Subsystem Requirements

Functional modeling

Telelogic AB

An IBM Company
An Architecture Workflow Scenario

- **In DOORS**
  - Requirements Engineers
  - Capture subsystem requirements (solution).

- **In Tau**
  - Architect
  - Architecture modeling

- **In DOORS**
  - Requirements Engineers
  - Capture and evolve requirements (problem).

- **In Tau**
  - Functional modeling

Decomposition
A Common Model-Based Repository
Maintaining integrity across multiple perspectives

- All information collected in common repository
  - enables straightforward traceability between various formats and modeling languages
  - matrices, reports, and generated diagrams used to visualize and analyze information and relationships
- Information always up-to-date
- Predefined mappings and transformations between different views

- Simplify and ensure
  - consistency
  - correctness
  - coherence
Traceability Across the SDLC

User Reqs | Technical Reqs | Design | Test Cases
---|---|---|---

**3 Requirements**

This section contains the user requirements.

3.1 Capability Requirements

3.1.1 Carrying Capacity

3.1.1.1 Number of People

Four average size adults shall be able to travel in comfort for a period of 3 hours. This level of comfort is defined as being equivalent to the standard of comfort provided by the top 40% of cars produced in 1999.

The top level of comfort is those in the price range $25,000 to $40,000 at 1999 prices.

Five average size adults shall be able to travel in comfort for a period of 3 hours.

Users shall have easy entry and exit.

---

End-to-end visual validation in a single view
Total Traceability: Manage the Development Process

- Change Requests
- Requirements Analysis
- Change
- Work Orders
- Engineering Tasks
- Design & Implementation
- Configuration Management
- Synergy CM
- Tau
- FocalPoint
Demo scenario…

- Border Security (Problem space/Program) > FastPass traveler identification (SoS) > Checkpoint Kiosk (System)
- Goal/Usage captured in SA
- Operational aspects using DoDAF OV’s in SA
- System aspects with SV’s / SysML in SA/Tau
- Functions in Tau using SysML/UML
- Requirements maintained and controlled in DOORS
Summary

Standardizing on integrated tools and processes across the systems development lifecycle helps reduce risk and improve productivity.

Telelogic integrated solutions for the SDLC include:

- Telelogic System Architect
  - Support BPMN, DoDAF and IDEF notations as well as several frameworks
  - Used early in the process, can even “kick-start” requirements capture

- Telelogic DOORS
  - Tightly integrated with SA, andTau
  - Market leading RM tool

- Telelogic TAU
  - Supports UML 2.1, DoDAF and SysML notations
  - Executable models. Show “satisfies” or “implements” to requirement in DOORS.