Solving the System Software Development Challenge by Integrating Teamcenter Systems Engineering with UML-based Model-driven Development

Jim McElroy
I-Logix, Inc.
Jimm@ilogix.com
978-645-3026
Global System Development Challenges

- Complexity
- Communication and Collaboration
- Productivity
- Time-to-market
- Quality
- Safety, Security, Reliability
Technical System Development Evolution

Hardware
  software
  Hardware
  software
  Hardware
  Software
  Hardware
  Software
  Hardware
  Software
  Hardware
  Software
  Hardware
  Software
  Hardware
  Software
  Hardware
  Software
Some Strategies to Solve These Issues

- Requirements Management Tools
- Model-driven Development
  - Abstraction
  - Automation
  - Verification
- Product Life-cycle Management Tools
Our Goal

Present a practical cohesive methodology and infrastructure for both requirements analysis and development and system software development
Who is Likely to Care and Why?

- Program Management
- System Engineers and Architects
- System and Software Developer
- Quality Assurance and Testing

Why…

- Better communication and collaboration
- Better estimation of requirements changes and affects
- Increased developer productivity
- Improved software quality
The Solution – Teamcenter Systems Engineering Tightly Coupled with MDD

- Requirements analysis and design
- System architectural definition
- Traceability to design, implementation, and test
- Impact analysis
- Coverage analysis
- Requirements documentation
Model-driven Development

A design and development methodology which uses models as the basis for analyzing requirements, developing the design, implementing, testing, and deploying the application.
Modeling Requires a Language

- UML 2.0 – Standards Compliance
  - Architecture
  - Behavior
  - Collaboration
MDD by Example
MDD – Functional Requirements View
Subsystem view showing realization of flows between blocks through links and port.
MDD – Architecture Level 2
This is a Subsystem level scenario which describes how the subsystem interact with a successful request to place a call.
MDD – Example Capabilities

- Modeling Systems in UML 2.0
- Modeling Behavior and Collaboration
- Design-level debugging
- Dynamic Model/Code Associativity
- Documentation Generation
Benefits of Modeling

- Modeling gives us the ability to visualize the system clearly
- Simplify the problem through abstraction
- Executable models further enhance the visualization, understanding, clarification of intended functionality and behavior
Key Differentiators for MDD

- **Modeling (UML 2.0 PLUS)**
  - Benefit: Designing systems at a higher level of abstraction in order to easily deal with complexity

- **Code Generation**
  - Benefit: Get to the final product quicker. Enables designers to work at a higher level of abstraction to deal with system complexity

- **Model/Code Associativity**
  - Benefit: Freedom to work at the model level or source code level, and ensure views of the system are always synchronized.

- **Real Time Framework**
  - Benefit: Allows you to create a deployable application, with the generated code, onto a real time operating system

- **Reverse Engineering**
  - Benefit: Allows you to reuse your IP, as well as coexist with ongoing hand coding activities

- **Design for Testability**
  - Benefit: Automate the testing process through using the *design requirements* to validate and to completely cover all system scenarios
In an ideal world …
UML Models within TcSE
UGS Teamcenter SE and Rhapsody

Create Rhapsody UML Model Elements directly within Teamcenter Systems Engineering
UGS Teamcenter SE and Rhapsody

Create traceability links between Rhapsody UML Model Elements and Teamcenter Systems Engineering
For a given breakdown of Rhapsody UML Elements created in Teamcenter Systems Engineering, create a Rhapsody Diagram using “Rhapsody Live”
UGS Teamcenter SE and Rhapsody

Based on the type of Rhapsody UML Elements selected in Teamcenter Systems Engineering browser, Rhapsody launches from Teamcenter, and the diagram is auto populated with the correct Elements.

Hitting Save in Rhapsody, automatically saves the diagram within Teamcenter Systems Engineering.
Here we see the saved Rhapsody Use case Diagram now in Teamcenter Systems Engineering!
The Integrated TcSE and Rhapsody Solution

- Optimal workflow for systems and software engineers increasing overall team productivity
- Dramatically improves communication throughout the entire team
- Facilitates domain specific modeling
- Ensures requirements are addressed in the design
- Quickly illuminates requirement issues or design flaws for rapid product development