GM’s Transformation Journey

Bill Bolander
Systems Solution Executive
Global Automotive Industry
The Big Challenge – the Motivation for change

Recognition that complexity is growing beyond capability to deliver

The need for smarter products drives complexity, new tools will enable engineers to spend more time being productive and less time spent managing the complexity.
Goal: Process & Tool Convergence
Converge each type of work - done one way, with the same tools globally

- Improve capability through better requirements definition, management, and traceability
- Improve electrical system design, analysis capability, and interface management
- Improve resource portability between Domains
- Improve software component sharing between Domains and suppliers with industry standard architecture – AUTOSAR
- Eliminate redundant tools and the cost/effort to maintain them
Scope of the Transformation:
Electrical, Controls and Software Domains – parts with software

Electrical, Controls, & Software Domains (5000 people)
GM Goals

Traceability across the engineering lifecycle

- Bi-directional traceability from requirements to engineering assets to components (ISO26262)
- Traceability allows upfront impact analysis before the change
  - What software or hardware might be affected by this change?
  - What test cases might be affected?

Change Management

- Global collaboration on change approval and work
- System engineering through cross functional teams
- Workload management to balance resources with change requests
- Allows automatic status of development and testing
GM Goals

Requirements Management

• Single source of the truth – reuse of requirements rather copies
• Decomposition of requirement to functional elements that can be allocated to system components with full traceability
• Highlights “suspect” links as requirements changes
• Supports validation of requirements at the lowest level possible
• Supports extraction of requirements for a specific vehicle based on it’s feature choices

Test Management

• Bi-directional traceability from requirements to test cases
• Single source of the truth – reuse of test cases rather than copies
• Highlights “suspect” links as requirements changes
• Supports validation of requirements at the lowest level possible
• Provides test planning, result capture, and defect reporting
System Design / Architecture
Model Based Systems Engineering

Requirements

Voice of the Customer
System Requirements

Business Requirements
(Constraints)

Product Requirements
Derived Requirements

Architectures or View Points

Functional Architecture

Logical Architecture

Deployment Architectures
1-n

Link to Internal Behavior
• Hand Code or
• Simulink or
• Rhapsody

Physical Architecture

Hardware, wiring, 3D math
GM Goals

Product Line Engineering

• Expand Product Line Engineering Reuse methods across the engineering lifecycle
• Model all feature variation offered by our systems
• Allow capture of vehicle specific feature choices – “Bill of Features”
• Enable vehicle specific extracts of requirements and test cases
• Enable automatic setting of configuration calibrations based on a vehicle’s feature choices
Strategy = Reuse
Method = Product Line Engineering

Product Line Engineering methods have provided gains in productivity and quality in the production of software to the point that the discipline is often referred to as Software Product Line Engineering.

It accomplishes these gains through reuse and management methods. Product Line Engineering methodologies can be applied to the development of all engineering artifacts, resulting in gains in both productivity and quality.

A Product Line is a set of systems sharing a common, managed set of features that satisfy the needs of a particular market segment or mission and that are developed from a common set of core assets in a prescribed way.
GM started a PLE approach in software engineering with astounding results:

<table>
<thead>
<tr>
<th>Application Type</th>
<th>Normalized Cost</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>1st Application of Complex Non-PLE ECU</td>
<td>25.5</td>
<td>+8%</td>
</tr>
<tr>
<td>1st Application of Complex PLE ECU</td>
<td>27.6</td>
<td></td>
</tr>
<tr>
<td>2nd Application of Complex Non-PLE ECU</td>
<td>15.8</td>
<td></td>
</tr>
<tr>
<td>2nd Application of Complex PLE ECU</td>
<td>3.2</td>
<td>-77%</td>
</tr>
</tbody>
</table>
Normalized Warranty Claims

- 70 %
Requirements from a set of vehicles are decomposed and allocated to a **common** component that is implemented and then assembled into the set of vehicles.

Rational and our Partners provide methods and capabilities to manage or configure these components!
Three Layers in Product Line Engineering

**Multi-stream/ Reuse**

- Variations of artifacts
- Configuration management approach to managing variation
- Opportunistic reuse (ad-hoc vs. pre-planned)
- Branching as primary pattern
- Re-use without “clone and own”

**Parametrics**

- Parameters specify configurations
- Simplifies complicated variations
- Creation of generic, product line-wide artifacts to increase reuse and better support platform design
- Automated derivation of artifacts

**Feature Management**

- Comprehensive business/stakeholder related feature representations
- Feature models handle complex feature combinations with multiple variation points
- Better traceability from business stakeholders to engineering
- Large variation space
Example Gear’s Feature Model

Assertion: DualZone or TriZone requires Auto
Example Gear’s Feature Profile: a program’s choices
Features come in.

Assets are configured.

A product comes out.

Just like a factory.
What were the specific tools implemented as part of this project?

Tools used across the Lifecycle:

- Rational Team Concert (Change Mgmt) \ Rational Synergy (Config Mgmt)
- BigLever Gears (Product Line Engineering)
- Method Park Stages (Process Modeling)
Why were Rational and it’s Partners chosen?

Key Decision Factors

• Open interfaces to allow selection of “best of breed”
• Linked data to provide traceability across assets in the lifecycle
• Support for global distribution of work
• Support for Product Line Engineering
• Support for PLM integration

Why IBM?

• Jazz Platform providing common services
• OSLC – open standards – best integration strategy
• Willingness to support integration of non-Rational tooling when selected
• Tool analysis and POCs showed Rational best met requirements
IBM Software Group | Rational software

Long Term Roadmap

JAZZ TEAM SERVER

Open Lifecycle Service Integrations

Best Practice Processes

Method Park Stages
Rational Team Concert (RTC)
Rational DOORS Next
Rational Quality Manager (RQM)
Rational Design Manager (RDM)
Rational Engineering Lifecycle Manager (RELM)
Rational Focal Point

Gears
Rational Publishing Engine
Rhapsody

DOORS 9.4
Rational Synergy
TcUA

VVC
Versions, Variants, Configurations

Team Awareness
Dashboards
Security
Events
Notification
Search
Query
In Context Collaboration

IBM Software Group | Rational software
Thank You

For more information visit ibm.com/rational/solutions/automotive