Increasing productivity with requirements reuse and variant management with DOORS Next Generation

1.2 Continuous engineering
Han van Gerwen
han.vangerwen@nl.ibm.com

Innovate 2014
The IBM Technical Summit

© 2014 IBM Corporation
Please note

IBM’s statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM’s sole discretion.

Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.

The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract. The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.

Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user’s job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

Plans are based on best information available and may change in future.
Outline

• Motivations & Definitions
• Patterns
• Enabling capabilities
• Demo
• Summary
Doing more with less in a customizing world

- Trend toward mass customization and shorter product lifecycles
- More embedded software; more complex connected products
- Need to adhere to safety standards, compliance and regulations

Source: http://commons.wikimedia.org/wiki/File:ITPB_health_Club.jpg
Some reuse scenarios…

• Managing requirements for a product family e.g.,
  – A vehicle platform
  – A set of insurance claim systems
• Shared requirements across different programs for different customers
• Parallel development of multi-year programs
• Handling requirements for a trade-study prototypes
GM started a reuse approach (PLE) in software engineering with impressive results:

- First Application of Complex Non-PLE ECU: 25.5
- First Application of Complex PLE ECU: 27.6
  - 38% reduction
- Second Application of Complex Non-PLE ECU: 15.8
- Second Application of Complex PLE ECU: 3.7
  - 87% reduction
GM’s recent publication of strategic reuse metrics are clear validation of the automotive industry’s direction towards common EE architecture.

- GM’s results are part of their comprehensive engineering strategy for a global common architecture and single SW build across all valid variants in their portfolio.

- Results show 3x-6x improvement in engineering costs for subsequent ECU development. This does not include the recent efforts to greatly increase the automation in this space with additional Rational and partner solutions.
Outline

• Motivations & Definitions

• **Patterns**
  – Strategic Reuse
  – Parallel development

• Enabling capabilities

• Demo

• Summary
Patterns of reuse

• “Branching” from closest product / component
• A more tactical reuse approach…
Patterns of reuse

• Core assets pattern
Strategic reuse: the conceptual scenario…

- Management of core platform engineering (“base”)
- Enable parallel engineering of platform variants
- Enable controlled reuse and change propagation downstream and upstream
Parallel development for multi-year programs

Workspace V 2012

Module 3 V1

Accept changes

Workspace V 2013

Module 3 V3

Deliver changes

V4 = merge (v2,v3)

Deliver parallel development

Engineer A

Engineer B
Outline

- Motivations & Definitions
- Patterns
- Enabling capabilities
  - Configurations
  - Streams and Baselines
  - Change Sets
- Demo
- Summary
Concept of requirement configurations

- Artifacts have versions
- A configuration specifies the included artifacts and their versions
- Configurations can share common artifacts and manage variability of other artifacts
- Configurations can be mutable (workspace) or immutable (baseline)
Realizing variant management with streams

- Each product variant is a branch of evolving artifacts
  - A stream or “workspace” – A mutable configuration
  - Streams are associated with baselines – Immutable configurations
- Common artifacts are shared across branches
- On one branch, evolution is a sequence of baselines
- New variants can be branched from existing variants or from the base
  - Can evolve in parallel
- Branches can update other branches using workspace delivery
Realizing parallel development with streams

- Streams are mapped to **workspaces** in the various domain tools
- Changes can be delivered across streams
- Deliveries may result in conflict detection that leads to a **merge**

![Diagram showing parallel development with streams](image-url)
Some essentials for reuse scenarios

- Rebasing (base workspace to other baseline)
- Compare
- Merge
- Changesets (group of related changes)

- Key reuse patterns:
  - Creating a new variant – create a child workspace (stream)
  - Updating common requirements from the base stream to a variant
    – Rebasing
  - Updating the base with changes already in the variant → deliver changes
  - Looking at a difference between two variants → compare streams
  - Handling “conflicting” changes -> merge
The bigger picture – global configurations

- How do we configure *requirements* along with the respective *tests*, *architecture*, and *code*?
- Global configurations (GCs) create compositions of configurations into multi-domain composite configurations
- GCs are part of OSLC configuration management
- GCs can be hierarchical
- GCs can also be mutable (global streams) or immutable (global baselines)
Outline

- Motivations & Definitions
- Patterns
- Enabling capabilities
- Demo
- Summary
Demonstration scenario - 1

1. **Initial setup: AMR Base, AMR Europe, AMR Asia**

2. Add new Requirements in the AMR Base Configuration. Create an explicit baseline.
   - Compare the new baseline with the old one

3. In AMR Europe, rebase configuration on the new baseline from step 2
   - Show that changes are present in AMR Europe, not present in AMR Asia.

4. Derive a new configuration - AMR South America using the baseline from step 2
   - Show that AMR Europe and AMR South America are identical

5. In AMR Europe, make changes to a common requirement using a change set linked to a work item and deliver change to AMR Base.
1. Initial setup: AMR Base, AMR Europe, AMR Asia

2. Add new Requirements in the AMR Base Configuration.
   - Create an explicit baseline.
   - Compare the new baseline with the old one

3. In AMR Europe, rebase configuration on the new baseline from step 2
   - Show that changes are present in AMR Europe, not present in AMR Asia.

4. Derive a new configuration - AMR South America using the baseline from step 2
   - Show that AMR Europe and AMR South America are identical

5. In AMR Europe, make changes to a common requirement using a change set linked to a work item and deliver change to AMR Base.
1. Initial setup: AMR Base, AMR Europe, AMR Asia
2. Add new Requirements in the AMR Base Configuration. Create an explicit baseline.
   - Compare the new baseline with the old one
3. **In AMR Europe, rebase configuration on the new baseline from step 2**
   - Show that changes are present in AMR Europe, not present in AMR Asia.
4. Derive a new configuration - AMR South America using the baseline from step 2
   - Show that AMR Europe and AMR South America are identical
5. In AMR Europe, make changes to a common requirement using a change set linked to a work item and deliver change to AMR Base.
Demonstration scenario - 4

1. Initial setup: AMR Base, AMR Europe, AMR Asia
2. Add new Requirements in the AMR Base Configuration. Create an explicit baseline.
   - Compare the new baseline with the old one
3. In AMR Europe, rebase configuration on the new baseline from step 2
   - Show that changes are present in AMR Europe, not present in AMR Asia.
4. Derive a new configuration - AMR South America using the baseline from step 2
   - Show that AMR Europe and AMR South America are identical
5. In AMR Europe, make changes to a common requirement using a change set linked to a work item and deliver change to AMR Base.
1. Initial setup: AMR Base, AMR Europe, AMR Asia
2. Add new Requirements in the AMR Base Configuration. Create an explicit baseline.
   - Compare the new baseline with the old one
3. In AMR Europe, rebase configuration on the new baseline from step 2
   - Show that changes are present in AMR Europe, not present in AMR Asia.
4. Derive a new configuration - AMR South America using the baseline from step 2
   - Show that AMR Europe and AMR South America are identical
5. In AMR Europe, make changes to a common requirement using a change set linked to a work item and deliver change to AMR South America.
Outline

• Motivations & Definitions
• Patterns
• Enabling capabilities
• Demo
• Summary
Summary

• Configuration Management benefits
  – Isolated changes with controlled propagation
  – Reuse without copying

• Streams (workspaces) can express product versions and variants

• Propagate changes
  – In the common requirements by delivering them to the variant configurations
  – In a variant and deliver to the common stream

• Global configurations to manage versions and variants over the full lifecycle.
  – Interface defined in OSLC, will be supported in Rational ALM suite

• Start exploring now
  – Download the DOORS NG with CM open beta from jazz.net
Further references

**DOORS Next Gen with CM on Jazz.net**

Rational DOORS Next Generation with Configuration Management 5.0 beta 1 task guide

Following are some simple tasks to get you started with the next generation (RDNG) with Configuration Management and versions as well.

If you encounter product defects, you can use the wiki to post a question or discuss some aspect of the product.

These tasks assume you have

You should also be familiar with the overall configuration management process.

**Rational DOORS Next Generation**

1. Create and set up a project
2. Work with baselines
3. Compare a workspace
4. Track changes with a workspace
5. Create a workspace

**Rational DOORS Next Generation with Configuration Management 5.0 beta 1**

- Early Access Milestone: June 2, 2014
- Important notes about using this build

**Getting Started**

- New & Noteworthy
- Release Notes
- All Downloads

In this beta, Rational DOORS Next Generation adds support for requirements versioning and the configuration management of requirements. This support provides many benefits that software configuration management tools provide for file-based information:

- Re-create an engineer's workspace based on a stream or baseline
- Develop in parallel or develop multiple variants
Thank You

Innovate 2014

Innovate @ SPEED
# Acknowledgements and Disclaimers

**Availability.** References in this presentation to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates.

The workshops, sessions and materials have been prepared by IBM or the session speakers and reflect their own views. They are provided for informational purposes only, and are neither intended to, nor shall have the effect of being, legal or other guidance or advice to any participant. While efforts were made to verify the completeness and accuracy of the information contained in this presentation, it is provided AS-IS without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, this presentation or any other materials. Nothing contained in this presentation is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software.

All customer examples described are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics may vary by customer. Nothing contained in these materials is intended to, nor shall have the effect of, stating or implying that any activities undertaken by you will result in any specific sales, revenue growth or other results.

© Copyright IBM Corporation 2014. All rights reserved.

- U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

IBM, the IBM logo, ibm.com, are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. If these and other IBM trademarked terms are marked on their first occurrence in this information with a trademark symbol (® or ™), these symbols indicate U.S. registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at “Copyright and trademark information” at [www.ibm.com/legal/copytrade.shtml](http://www.ibm.com/legal/copytrade.shtml)

Other company, product, or service names may be trademarks or service marks of others.
Handling multiple components - component dependencies

- To enable higher reuse, it is useful to organize requirements in multiple components
- Initially, DNG uses projects as component boundaries
  - To be refined in 2015
- Component configurations are linked using dependencies
  - Essentially imply a hierarchical structure
- A “component” can be part of multiple products
  - At same or different baseline
Example scenario: supplier communications

ReqIF with requirements configuration management

Central DOORS NG Database

Main Development Stream

Specifications V1
Specifications V2
Specifications V2 + a2
Specifications V2 + a2 + b2

Supplier ‘a’ Stream

Specifications V1 + a1
Specifications V1 + a2

Supplier ‘b’ Stream

Specifications V1 + b1
Specifications V1 + b2

Supplier ‘b’ DB

Supplier ‘a’ DB

Round trip ReqIF to get supplier feedback

All updates reflected back in central database but in private ‘streams’
Supplier updates protected until they are approved

IBM confidential – shared under NDA