Knowing When You're Done with Your Requirements

Mike Bartley,
Test and Verification Solutions
Background

- **NMI verification roadmapping 2009 - 2010**
  - Identified 13 different hardware verification bottlenecks
  - The management of verification data ranked very high

- **DVClue identified numerous internal solutions**

- **TVS collaborated with XMOS internal tool**
  - Most mature
  - Used for both hardware and software signoff
  - Track requirements to signoff with many-many mapping
The need for requirements tracing

"the ability to follow the life of a requirement, in both a backward and forward direction"

Makes good commercial sense

Important in building rigorous software systems

- U.S. Food and Drug Administration (FDA)
- Mandatory for CMMI level 2 and above
- Certification in Aeronautics (DO-178B, DO-254), Railway (EN-5012x), Automotive (ISO26262, IEC 61508), Medical Systems (FDA 21 CFR),..
• Good tool support for requirements tracing
  – Doors, Reqtify, Enterprise Architect, Jira, …
• But customers report limited support for testing

• What do they want?
  – Capture the mapping of requirements to tests
  – Automate the recording of tests results
  – Automate reporting of requirements test status
  – Document the sign off of the requirements
When does requirements signoff start?

- **Not just at the final stage**

- **Early Stage**
  - Feature extraction

- **Planning Stage**
  - Define the signoff activities based on the features
  - Capture the mapping of features to those signoff activities

- **Implementation Stage**
  - Writing tests, coverage, properties, etc

- **Execution**
  - Tracking progress of those tests etc

Tests, coverage, formal, bug stats, code churn, mutation analysis, ..
What to track?

- Start tracking testing from the start of the project
  - Are tests defined, written, executing, passing?
The solution is based on a database approach.
Automating the test result collection and analysis

- **Result collection**
  - Link running of tests to the database
    - Test started
    - Test result
  - Automatically record the source code version
    - By connecting to the source control system

- **Result analysis**
  - Identify orphan tests
  - %’s of tests defined, written, executing, passing
  - Which requirements aren’t being tested?
  - Which tests have started failing? Why?
  - Automatic linking to a bug database

- **Map tests to coverage**
  - What do the tests cover? Are there test overlaps?
Summary

• Map requirements to signoff activities
• Database to record mappings and results
• Store results from test automation
  – Record %’s of tests defined, written, executing, passing
• Advantages
  – Track the status of the whole verification effort
  – Use historical perspective for more accurate predictions
  – Better reporting of requirements status
  – Better analysis of test results, code versions and bug database
  – Support for risk-based testing
  – Support for regulatory-based requirements signoff