WP3: Lifecycle Traceability for Static Code Analysis

ICF Transition Workshop
May 9 2017

Frédéric Loiret (KTH)
Stefan Otten (FZI)
WP3: Lifecycle Traceability for Static Analysis - Our Vision

- Enable Static Code Analysis Tools for improved lifecycle traceability and interoperability (ISO26262,...)
- Integration of different code analysis tools
- Towards a Static Analysis Platform (SAP)
WP3: Lifecycle Traceability for Static Code Analysis

Requirements
- REQ/CF
- Integrity
- IBM

Design
- Model
- Enterprise Architect
- MATLAB

Implementation
- SW-File
- Git
- Visual Studio
- Eclipse

Analysis
- OSLC Adaptor

WP2
- Exchange Format (WP2)
- Data, Control, Config
- *xml

Frontend/Navigation/Visualization

OSLC, Linked Data

Static Code Analysis
- AbsInt
- KIT

Static Code Analysis
WP3: Lifecycle Traceability Concept and Implementation

- Implementation based on OSLC Quality Management Specification

- First scenario / use case:
  - The Developer edits C/C++ Files from the IDE and commits changes to a repository
  - The Analysis Expert (or Developer) manually executes a new analysis with the latest revision of the software, generating Analysis Reports
  - The Analysis Expert (or Developer) checks in the Analysis Report containing check results of the Static Code Analysis
  - The OSLC Adaptor links the check result to the locations in the C/C++ Files for review by the Developer
WP3: Lifecycle Traceability
Status and Further Topics

Actual Status:
• Initial Adaptors for Static Code Analysis Tool QPR implemented and tested
• First scenario finalized in 06/2017

Further Topics:
• Establish link and traceability to further design and requirement artifacts/tools (IBM Doors, Matlab,...)
• Automation of analysis tool execution via OSLC automation
• Engagement of more interested partners of ASSUME
• Integration of other static code analysis tools
• Embedding into ASSUME use case