Architecture and Quality Committee: Mission, Responsibility, Competency

2016-04-21

Dr. Martin Jung

open KONSEQUENZ

Why Committees Determine Success

- Diversity
  - Diversity determines liveliness
  - Diversity in users → broad base of requirements
  - Diversity in suppliers → broad base of knowledge and expertise
  - ► The more, the merrier!

- Conformity
  - Conformity determines success
  - Conformity to a vision → make progress, add value
  - Conformity to standards → sustain progress, keep value
  - ► Define the game and play by the rules!

Architecture Committee

- Mission
  - Define core architecture – concepts, patterns, fundamental component break-down
  - Define & evolve technological base – Framework for the projects
  - Evaluate requirements of projects, create technical requirements
  - Break down functionality and define integration for projects
  - Manage dependencies between components
- Responsibility
  - Manage technological base and core data model (CIM profile). Adapt to projects needs.
  - Support projects in usage of framework and adherence to technological base
  - Review & modify technical solution proposals; decision support for projects
  - Decide on project specific / platform solution of components (consult with SC)
  - Evaluate technical solution of project quotes and prepare PPC decision
- Competence
  - Reject / revise technical project proposals
  - Adapt project RFQs by the PPC
- Ownership (reviewed by QC and SC)
  - Architecture document
  - Core data model
  - Technology list

Quality Committee

- Mission
  - Define and enforce quality standards, guidelines, conventions
  - Provide procedure, methods and tools
  - Evaluate project results
  - Support projects in integration and verification
  - Support SC and PPC in validation
- Responsibility
  - Safeguard code base, ensure high quality solutions
  - Project support for application of procedure & methods, and quality improvement
  - Build up and maintain a reference target platform for openKonsequenz software
  - Create continuous feedback on progress & qualiy of projects
  - Review project results and prepare PPC acceptance
- Competence
  - Reject technical solutions
  - Adapt project RFQs by the PPC
- Ownership (reviewed by AC and SC)
  - Build & delivery infrastructure, staging systems
  - Guidelines and conventions
  - Acceptance criteria list

Committees: Sample Agendas

Architecture Committee Meeting

- Ag1: Projects
  - Requests from projects
  - Pending project solutions
- Ag2: Framework
  - Pending changes/pull requests
  - Technical issues
  - Qualities over all projects
- Ag3: Quotes
  - Pending quotes
  - PPC decision support (award)

Quality Committee Meeting

- Ag1: Running Projects
  - Progress & Quality
  - Results review
- Ag2: Method and Procedure
  - Quality issues
  - Project complaints
- Ag3: Validation
  - Pending project results
  - PPC decision support (accept)

- Typically: Monthly telco, meeting minutes, backlog
- Participants: AC & selected project representatives

Committees: Example Results

Architecture Committee

- Ag1: Projects
  - Requests from projects on framework
    - Prj D ok, Prj E adapt xyz
  - Pending project solutions
    - Prj B change interface, Prj C change libraries
- Ag2: Framework
  - Pending changes
    - CR748: integrate scripting framework
  - Technical issues
    - ISS1024: Migration to new version of logging
  - Qualities
    - ISS 283: Spurious crashes in App A
- Ag3: Quotes
  - Pending quotes
    - Prj F ok, small content change in course of project, Prj G nok, out of scope, key functionality not met
  - PPC decision support
    - Accept F without changes, reject G

Quality Committee

- Ag1: Running Projects
  - Progress & Quality
    - Prj B green, KPIs ok, Prj C yellow. Deliver docu and test coverage too low
- Ag2: Method and Procedure
  - Quality issues
    - ISS42: Performance drops over application run-time
  - Project complaints
    - ISS975: Code Coverage requirement too strict
- Ag3: Validation
  - Pending project results
    - Prj A project end checklist yellow, two issues (cat. nuisance) pending closure
  - PPC decision support
    - Accept Prj A

QUALITY IN PROJECTS

Code Quality – 1

- Automated build, package & test scripts
  - Config management conventions
- Unittests – **Automated**
  - Per Use Case (User Story) Scenario
  - Per Quality Attribute
  - Standardized test data
  - Execute in daily builds
    - Fail fast (TDD)
    - eXtreme feedback → metrics
  - Criteria: least effort < line coverage < branch coverage ☺ < data flow coverage ☺ ☺
- Code Documentation
  - HowTos (For usage and development)
  - Code documentation guidelines
- Conventions: Naming, Packaging (mapping to architecture elements)
- Reviews (according to criticality ranking)
  - Automated (stat. Analysis), Peer Walkthrough (QC determines Peer), Deep inspection (QC invites AC and peer)
- Patterns/Methods
  - Coding Guidelines, Formatters → checkstyle, PMD
  - Review Checklists

Code Quality – 2

- Config & Version Management
  - See Slides by J. Meister (Top 7.2)
  - Commits need to comply with stated rules
  - Naming conventions for tags/branches
- Automated deployment
  - See Slides by J. Meister (Top 7.4)

Design Quality

- Statement of Qualities
  - Structured according to ISO 25010 (Use as a reference, not all aspects are mandatory)
  - Criticality rating
- Documentation
  - Design decisions and alternatives → follow ArchDoc specifications
- Verification & Validation
  - Specify checks and acceptance criteria (in response to project definition)
  - Specify Test Data requirements (incl. GUI test data, e.g. click paths)
  - Specify Mocks (Data sets or Oracle, Functionality, …)
- Reviews (according to criticality rating)
  - Peer Walkthrough (QC determines Peer), Deep inspection (QC invites AC and peer)
- Document list
  - Architecture Spec incl. criticality rating
  - Module & Integration Test Spec (automated@hudson, manual@integration platform [„Test-Drehbuch“])
  - Test protocols (incl. date, executor, verdict)
  - Review protocols (participants, crit. Rating)
- Conventions: names, file format, structure (git filesystem according to maven)

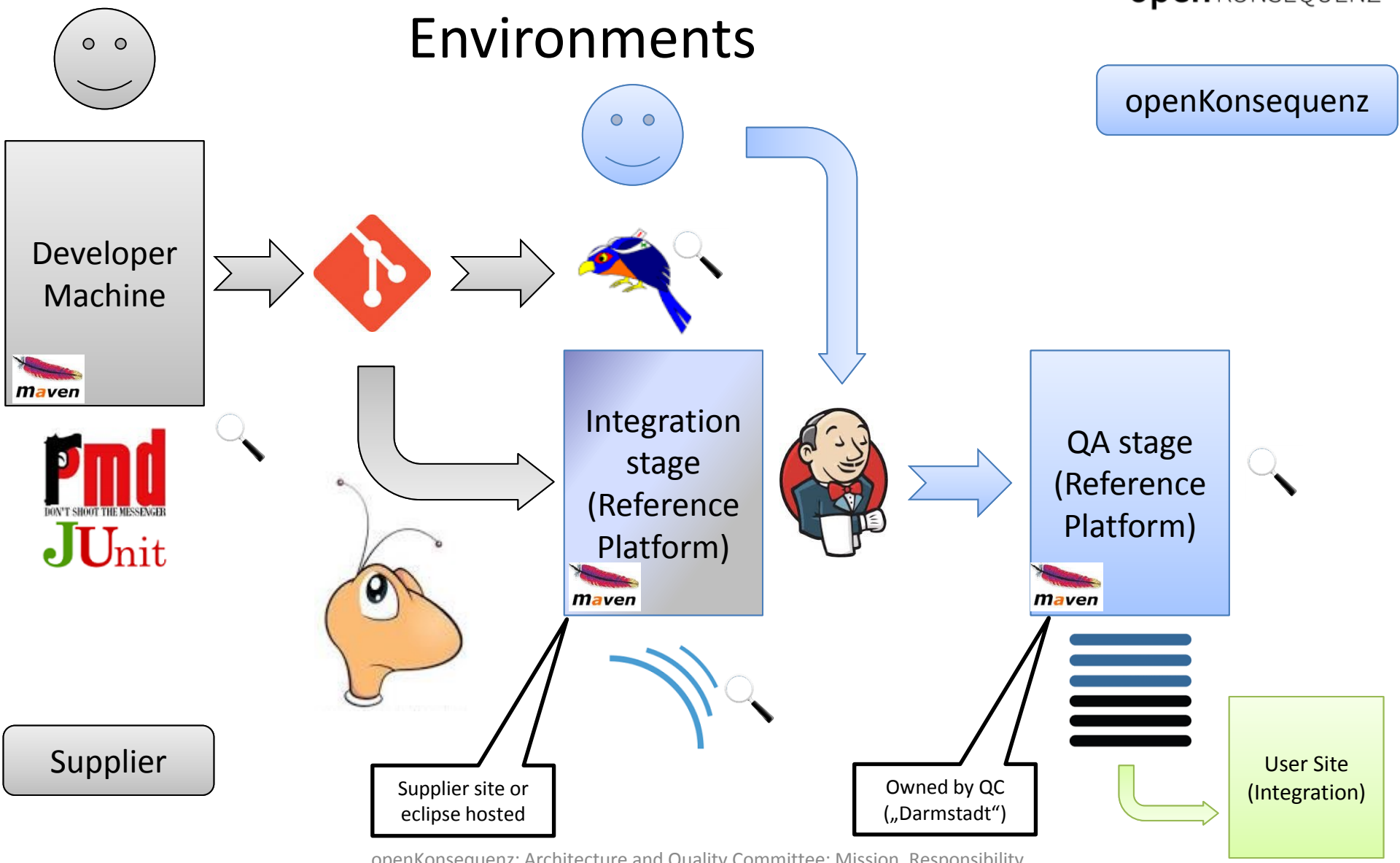
Product Quality

- Automated deploy, integrate, publish & test scripts
- Continuous Integration – **Automated**
  - Integration test cases („Smoke test suite“)
  - Metrics, eXtreme Feedback
  - Static & dynamic analysis
- Continuous Deployment
  - Staging and Quality
  - Automated software/system test on QA stage incl. autom. UI-Test – auto/manual
- (Continuous Delivery)
- Reviews (according to criticality rating)
  - 4 eyes, Walkthrough, Deep inspection
- Document list:
  - Software Test Spec (automated@hudson, manual@integration platform [„Test-Drehbuch“])
  - Test protocols (incl. date, executor, verdict)
  - Review protocols (participants, crit. Rating)
- Formats and conventions: publish name conventions, gui styleguide, etc.

Process Quality

- (not presented on workshop)
  - Documentation of Product & Sprint backlog
  - User stories & Tracing to code/commits
  - Maintenance of Burndown chart
  - Tracking of done criteria
  - Details of done criteria (3 months test run is inadequate)

Development and Staging Environments



develop group
Connect for Competence!

Dr. Martin Jung
Software Engineering Consulting
develop group Basys GmbH
martin.jung@develop-group.de

UML Tool selection

- Commercial (ranging from 809 to 1100 €; license pro/floating)
  1. MagicDraw: Best productivity
  2. Enterprise Architect: Best value
- Open Source
  - **Modelio**: Usable tool, roughly comparable to commercial tools
  - Papyrus: Integration in eclipse ecosystem, but „Don‘t try this at home“
  - PlantUML: Model in code, „Ultra-lightweight“

Module Document naming

- Maven anchor: src/site/resources/howto
  - Build.txt (Asciidoc) or Build.md (Markdown): How to build the module
  - Run.txt or Run.md: How to run the module
  - Code.txt or Code.md: How to modify code
  - Test.txt or Test.md: How to set up & perform tests
- Maven anchor: src/site/resources/arch
  - Criticality.txt or Criticality.md
  - Architecture.txt Architecture.md
  - reviews/YYYYMMDD-scope-type-author.txt or *.md
    - scope~=ControllerComponent; type=[Peer|Inspection]; author ~= DagobertDuck
    - Protocol contains participants
- Maven anchor: src/site/resources/test
  - Moduletest.txt or Moduletest.md
  - Integrationtest.txt or Integrationtest.md
  - protocols/YYYYMMDD-type-tester.txt or *.md
    - type=[module|integration]; tester ~=DonaldDuck