F. Deißenböck, S. Wagner, M. Pizka, S. Teuchert, J.-F. Girard

An Activity-Based Quality Model for Maintainability

October 3rd 2007
ICSM 07 Paris
Requirements: Business Processes, Functions, ...

Software System

Technology: Hardware, OS, Languages, Tools, DBs, ...

Deißenböck
An Activity-Based Quality Model for Maintainability

Maintenance

~Activities

<!-- Compile sources in source folder to build folder. Compilation is logged in $compile.log.file -->
<target name="compile" depends="mkdirs, init" description="STD-ENV: Compile Java sources.">
  <javac srcdir="${src.dir}"
    <compilerarg value="-Xlint:all" />
    <compilerarg value="-Xlint:-serial" />
    <classpath refid="classpath" />
  </javac>
</target>

private boolean testAndSplit(int refWordEnd, Object nextCharacter) {
  if (currentNode < 0) {
    // trap state is always end state
    return true;
  }
  if (refWordEnd <= refWordBegin) {
    if (nextNode.get(currentNode, nextCharacter) < 0) {
      explicitNode = currentNode;
      return false;
    }
    return true;
  }
  return true;
}
Hierarchical Quality Models

**Maintainability**
- **Modifiability**
- **Testability**
- **Understandability**

- **Augmentability**
- **Structuredness**
- **Communicativeness**
- **Accessibility**
- **Self-Descripтивness**
- **Conciseness**
- **Legibility**

**Criticism**
- Homogeneity
- Justification
- Assessment
- Operationalization

* B. W. Boehm et al., *Characteristics of Software Quality*, 1978
Separation of Activities & Facts

**Activities**
- Define costs
- Depend on maintenance task

**Facts**
- FCM-like but without activities
- Includes organizational aspects
Attributes & Impacts

Attributes

- Entities: objects we observe in the real world
- Attributes: properties that an entity possesses
- Fact: Tuple (Entity × Attribute)
- Example: [Switch Statement | COMPLETENESS]
- Types:
  - automatic: «Switch must have Default»
  - semi-automatic: Redundancy, Dead Code
  - manual: wrong Algorithm

Impacts

- Impact of a fact on a activity
- Impact: [Entity e | ATTRIBUTE A] \(\rightarrow\) [Activity a]
- Example: [Switch Statement | COMPLETENESS] \(\rightarrow\) [Modification]
Case Study I

MAN Nutzfahrzeuge Group

- Germany-based international supplier of commercial vehicles and transport systems
- > 34,000 employees, ~ 150 in electronics and software development
- Focus on embedded systems

Simulink/Stateflow

- Model-based simulation/development suite for embedded systems
- Simulink has its focus on continuous control engineering
- Stateflow is a dialect of statecharts
- C-Code generation with TargetLink by dSpace
- Combination is widely used in the automotive domain
Case Study II

- Objective: Quality model for Simulink/Stateflow development at MAN

- Sources
  - MathWorks documentation
  - MAN-internal guideline
  - MAAB guideline (MathWorks Automotive Advisory Board)
  - dSpace guideline
  - Expert know-how
  - Studies on model-based development in general

- Basis: Quality model development for business information systems

- Example

  »The current state of a Stateflow chart must be available as a measurable output.«

  [Stateflow Chart | ACCESSIBILITY] $\rightarrow$ [Debugging]
  [Stateflow Chart | ACCESSIBILITY] $\rightarrow$ [Test]
Results

- Consolidation of Terminology
  - Inconsistent naming across all sources and within MAN
  - Quality model enforces consistent naming

- Resolution of Inconsistencies
  - MAAB: »Implicit event broadcasts [...] and implicit conditions [...] make the diagram easy to read and the generated code more efficient.«
  - dSpace: »The usage of implicit events is therefore intransparent concerning potential side effects of variable assignments or the entering/exiting of states.«

- Revelation of Omissions
  - Attribute inheritance fosters completeness
  - Example: [Stateflow Variable | LOCALITY] and [Variable | LOCALITY]

- Inclusion of the model into the MAN standard development process
An Activity-Based Quality Model for Maintainability

Operationalization

Quality Engineer

Artifact

Technology Pool

QMM.editor

http://www4.cs.tum.edu/~ccsm/qmm/

http://conqat.cs.tum.edu
Conclusions & Future Work

Conclusions

- Up to now there is no sound and accepted definition for maintainbility
- Activities should be a first-class citizen in quality modeling
- Quality models need to operationalized
- Modeling quality is *modeling*

Future Work

- Application of our approach for other quality attributes
- Unification of currently isolated approaches to quality
- Tailoring/Modularization of quality models
- Goal: Truly economically justified practice of quality engineering