Software Architecture Compliance Checking
with HUSACCT

Leo Pruijt, HU University of Applied Science, Utrecht

In cooperation with:
Raymond Slot, Wiebe Wiersema, HU, Utrecht
Christian Köppe, HAN, Arnhem
Sjaak Brinkkemper, Jan Martijn van der Werf, University Utrecht
Agenda

- Software Architecture & Quality Attributes
- Architecture Compliance Checking
- HUSACCT, a tool to provide SRMA support
- Example
- Questions
Software Architecture (SA)

- Software architecture is of major importance to achieve
  - the business goals
  - functional requirements
  - quality requirements of a system

- However …
  - Architectural models tend to be of a high-level of abstraction
  - Deviations of the software architecture arise easily during
    - the development and
    - evolution of a system

- Architectural Erosion has a negative impact on
  the system’s quality attributes
SA Example (Schiphol Group)
Common Module & Rule Types

In practice …

• Modules with different semantics are used commonly
  • Subsystems, Layers, Components, Interfaces, External Systems
• Rules of different types are used
  • Some of them specific for a certain type of module
  • E.g., ‘Back call ban’ and ‘Skip call ban’ are specific for Layers

<table>
<thead>
<tr>
<th>Type of Rule</th>
<th>Example (E)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Is not allowed to use</td>
<td>HiPanels is not allowed to use HiWS.</td>
</tr>
<tr>
<td>Back call ban</td>
<td>Service Layer is not allowed to use the Interaction Layer.</td>
</tr>
<tr>
<td>Facade convention</td>
<td>Component HiManager may be accessed only via HimInterface.</td>
</tr>
<tr>
<td>Is only allowed to use</td>
<td>HiForms is only allowed to use HiPanels.</td>
</tr>
<tr>
<td>Is the only module allowed to use</td>
<td>CorporateWebcore is the only module allowed to use Hibernate.</td>
</tr>
</tbody>
</table>
Architecture Compliance Checking (ACC)

- ACC verifies the conformance of
  - implemented program code to
  - high-level models of architectural design

- Static ACC focuses on
  the modular architecture

- Related quality attributes:
  - Accuracy
  - Maintainability
    - analyzability, changeability, testability
  - Portability
    - adaptability, replace ability
Static Architecture Compliance Checking

Currently …
- Adoption of ACC in practice is limited
- Tool support of the common sets of module & rule types is limited

Research Goals …
- Improve tool support
- Promote ACC in practice & education
ACC Process

1) Study the intended architecture (SAD/Architecture Notebook)
   1) Requirements
   2) Architectural decisions: Modules & Rules
   3) Mapping of modules to program code

2) Acquire additional or missing information

3) Enter intended architecture in ACC-tool
   1) Modules & Rules
   2) Mapping of modules to program code

4) Run conformance check

5) Study and discuss results
   1) Violations
   2) Relevance
HUSACCT: Hogeschool Utrecht
Software Architecture Compliance Checking Tool

Menu

Analyse implemented architecture

Define intended architecture

Validate conformance

Software Architect

- Analyse Application
  - Java
  - C#

- Explore Implemented Modular Architecture
  - Browse Modules & Dependencies
  - Visualize Modules & Dependencies
  - Report Modules & Dependencies

- Define Intended Modular Architecture

- Check Conformance

- Explore Results Conformance Check
  - Browse Violations
  - Visualize Violations
  - Report Violations
  - Export Violations
Outstanding Characteristics

- HUSACCT is free-to-use & open source
  - Download, video & instruction at http://husacct.github.io/HUSACCT/

- Support of rich sets of Module and Rule Types
  - 5 common Module Types with different semantics
    - Subsystem, Layer, Component, Interface, External system
  - 11 common Rule Types

- Extensive Semantic Support of the Module and Rule Types, e.g.:
  - Automatic creation of default rules, according to the Module Type
  - Type of Module determines which Rule Types are selectable

- Configurable support
  - Enable/Disable rules, Exception rules, Default rule configuration
ACC Example: Modular Architecture HUSACCT_1.0

Top-level Components

Rules

<table>
<thead>
<tr>
<th>From-Module</th>
<th>Constraint</th>
<th>To-Module</th>
</tr>
</thead>
<tbody>
<tr>
<td>Analyse</td>
<td>is not allowed to use</td>
<td>Define</td>
</tr>
<tr>
<td>Analyse</td>
<td>is not allowed to use</td>
<td>Validate</td>
</tr>
<tr>
<td>General GUI &amp; Control</td>
<td>Is the only module allowed to use</td>
<td>Graphics</td>
</tr>
<tr>
<td>All five components</td>
<td>Facade convention</td>
<td></td>
</tr>
</tbody>
</table>
Define Intended Architecture
Analyse Implemented Architecture

Needed to Assign Software Units and to Check Conformance
Validate Conformance

Results of the Conformance Check

<table>
<thead>
<tr>
<th>Id</th>
<th>Logical module from</th>
<th>Rule type</th>
<th>Logical module to</th>
<th>Violations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Analyse</td>
<td>Is not allowed to use</td>
<td>Define</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>General GUI &amp; Control</td>
<td>Facade convention</td>
<td></td>
<td>38</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>From</th>
<th>To</th>
<th>Rule type</th>
<th>Dep. type</th>
<th>Direct</th>
<th>Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>husacct.define.presentation.jdialog.AppliedRuleJDialog</td>
<td>husacct.control.ControlServiceImpl</td>
<td>Facade convention</td>
<td>Declaration</td>
<td>Direct</td>
<td>59</td>
</tr>
<tr>
<td>husacct.define.presentation.jdialog.ViolationTypesJDialog</td>
<td>husacct.control.ControlServiceImpl</td>
<td>Facade convention</td>
<td>Import</td>
<td>Direct</td>
<td>6</td>
</tr>
<tr>
<td>husacct.define.presentation.jdialog.ViolationTypesJDialog</td>
<td>husacct.control.ControlServiceImpl</td>
<td>Facade convention</td>
<td>Declaration</td>
<td>Direct</td>
<td>32</td>
</tr>
<tr>
<td>husacct.define.presentation.jdialog.AddModuleValuesJDialog</td>
<td>husacct.control.ControlServiceImpl</td>
<td>Facade convention</td>
<td>Declaration</td>
<td>Direct</td>
<td>44</td>
</tr>
<tr>
<td>husacct.define.presentation.jdialog.AddModuleValuesJDialog</td>
<td>husacct.control.ControlServiceImpl</td>
<td>Facade convention</td>
<td>Import</td>
<td>Direct</td>
<td>5</td>
</tr>
<tr>
<td>husacct.define.presentation.jdialog.AddModuleValuesJDialog</td>
<td>husacct.control.ControlServiceImpl</td>
<td>Facade convention</td>
<td>Import</td>
<td>Direct</td>
<td>5</td>
</tr>
<tr>
<td>husacct.define.presentation.jdialog.SoftwareUnitJDialog</td>
<td>husacct.control.ControlServiceImpl</td>
<td>Facade convention</td>
<td>Declaration</td>
<td>Direct</td>
<td>42</td>
</tr>
</tbody>
</table>
Intended Architecture Diagram

With the intended top-level components, their dependencies and violations
Intended Architecture Diagram

With the intended layers within Define, their dependencies and violations
Status and Outlook

- **HUSACCT_3.4** is fit for practical use
  - Accurate, fast, easy in use

- Intended use:
  - Introduction of ACC within organizations
  - Software architecture education
    - Relate abstract models to code
  - Stimulation of tool vendors to provide support for **Semantically Rich Modular Architectures**

- Future work:
  - Case studies: *Are you interested in a free ACC?*
  - Extension of functionality
  - Metrics
Finally

- More information:
  - Ask me: Leo Pruijt, leo.pruijt@hu.nl
  - Watch the video at http://husacct.github.io/HUSACCT/
  - Read the published papers
- Thank you for your attention!
- Questions?

2) Architecture Compliance Checking of Semantically Rich Modular Architectures: A Comparative Study of Tool Support. *Int. Conf. on Software Maintenance, ICSM 2013-09, Eindhoven, NL*
4) On the Accuracy of Architecture Compliance Checking Support: Accuracy of Dependency Analysis and Violation Reporting. *Int. Conf. on Program Comprehension, ICPC 2013-05, San Francisco, USA*