



Stichting NIOC en de NIOC kennisbank

Stichting NIOC (www.nioc.nl) stelt zich conform zijn statuten tot doel: het realiseren van congressen over informatica onderwijs en voorts al hetgeen met een en ander rechtstreeks of zijdelings verband houdt of daartoe bevorderlijk kan zijn, alles in de ruimste zin des woords.

De stichting NIOC neemt de archivering van de resultaten van de congressen voor zijn rekening. De website www.nioc.nl ontsluit onder "Eerdere congressen" de gearchiveerde websites van eerdere congressen. De vele afzonderlijke congresbijdragen zijn opgenomen in een kennisbank die via dezelfde website onder "NIOC kennisbank" ontsloten wordt.

Op dit moment bevat de NIOC kennisbank alle bijdragen, incl. die van het laatste congres (NIOC2023, gehouden op donderdag 30 maart 2023 jl. en georganiseerd door NHL Stenden Hogeschool). Bij elkaar bijna 1500 bijdragen!

We roepen je op, na het lezen van het document dat door jou is gedownload, de auteur(s) feedback te geven. Dit kan door je te registreren als gebruiker van de NIOC kennisbank. Na registratie krijg je bericht hoe in te loggen op de NIOC kennisbank.

Het eerstvolgende NIOC vindt plaats op donderdag 27 maart 2025 in Zwolle en wordt dan georganiseerd door Hogeschool Windesheim. Kijk op www.nioc2025.nl voor meer informatie.

Wil je op de hoogte blijven van de ontwikkeling rond Stichting NIOC en de NIOC kennisbank, schrijf je dan in op de nieuwsbrief via

www.nioc.nl/nioc-kennisbank/aanmelden-nieuwsbrief

Reacties over de NIOC kennisbank en de inhoud daarvan kun je richten aan de beheerder:

R. Smedinga kennisbank@nioc.nl.

Vermeld bij reacties jouw naam en telefoonnummer voor nader contact.



Software Architecture Education

A Practical Approach with HUSACCT

Leo Pruijt, HU University of Applied Science, Utrecht

Agenda



- Software Architecture
- SA Education
- Architecture Reconstruction with HUSACCT
- Architecture Compliance Checking with HUSACCT
- Questions

Software Architecture (SA)



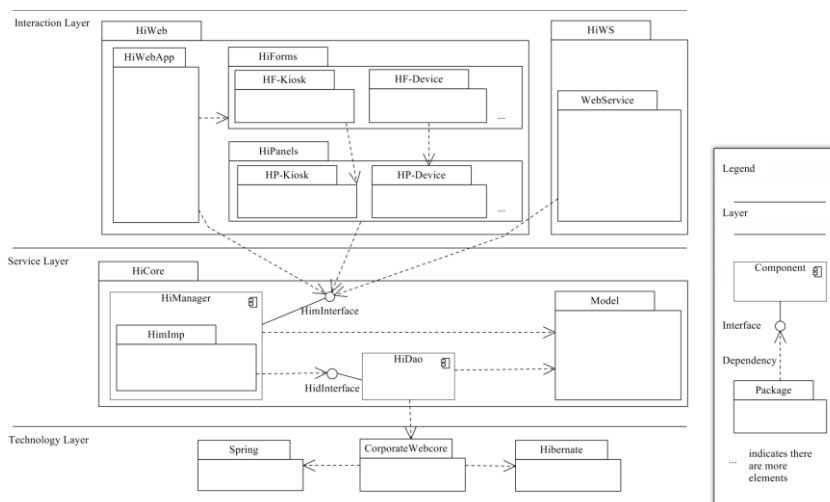
- Software architecture is of major importance to achieve
 - the business goals
 - functional and non-functional requirements
- However ...
 - Architecture models are often obsolete or incomplete
 - Architectural models tend to be of a high-level of abstraction
 - Hard to understand by students and practitioners
 - Deviations of the software architecture arise easily during the development and evolution of a system

© HU

NIOC 2015

3


SA Example (Schiphol Group)



© HU

NIOC 2015

4



Module View

Five types of modules are visible in the example


- Subsystems, Layers, Components, Interfaces, External Systems

Rules of different types are used in the example

- Some of them specific for a certain type of module
- E.g., 'Back call ban' and 'Skip call ban' are specific for Layers

Type of Rule	Example (E)
Is not allowed to use	HiPanels is not allowed to use HiWS.
Back call ban	Service Layer is not allowed to use the Interaction Layer .
Facade convention	Component HiManager may be accessed only via Him Interface .
Is only allowed to use	HiForms is only allowed to use HiPanels.
Is the only module allowed to use	CorporateWebcore is the only module allowed to use Hibernate.

© HU
NIOC 2015
5

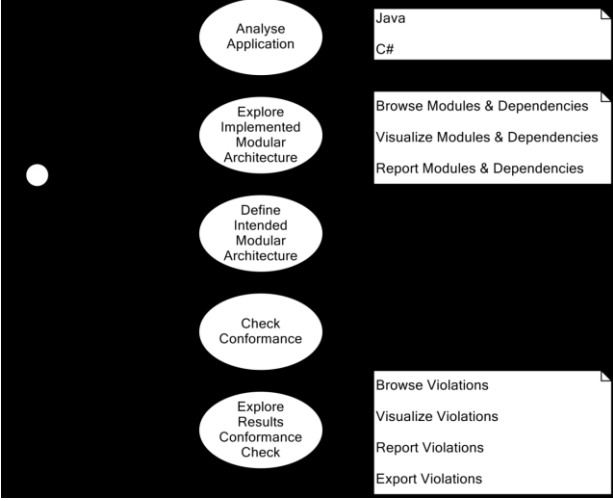



SA Education at the HU

- Base SA on foundation of Analysis & Design
 - Preceding courses (in year 1 and 2) on
 - Business processes, Use cases
 - Domain model, Object interaction, Database design
- Relate architecture models to code
 - Architecture Reconstruction with [HUSACCT](#)
 - Architecture Compliance Checking with [HUSACCT](#)
 - In year 2 (basic) and 3 (advanced)
- Apply in student projects
- Apply on practical cases (open source, visit companies)

© HU
NIOC 2015
6

HUSACCT: Hogeschool Utrecht Software Architecture Compliance Checking Tool



Menu


Analyse implemented architecture

Define intended architecture

Validate conformance

©HU NIOC 2015 7

Outstanding Characteristics



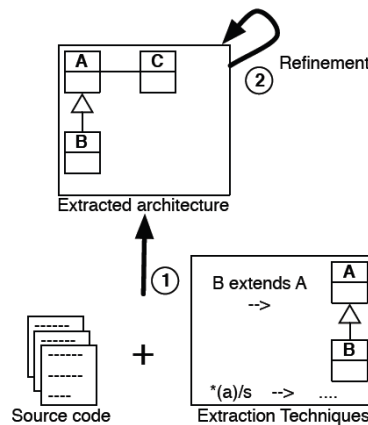
- HUSACCT is free-to-use & open source
 - Download, video & manual at <http://husacct.github.io/HUSACCT/>
- Support of rich sets of Module and Rule Types
 - 5 common Module Types with different semantics
 - Subsystem, Layer, Component, Interface, External system
 - 11 common Rule Types
- Extensive Semantic 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

©HU NIOC 2015 8

Architecture Reconstruction



Architecture reconstruction:
comprehending and
documenting an architecture
based on the source code
(and other information)



Source: Ducasse & Polle (2009)

© HU

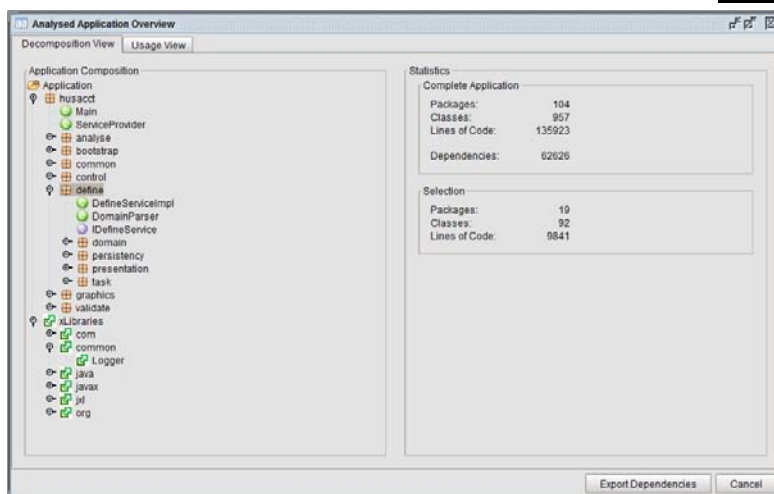
NIOC 2015

9

Analyse Implemented Architecture



Decomposition View: How is the software structured?



© HU

NIOC 2015

10

Analyse Implemented Architecture



Usage View: Which dependencies are present?

The dialog box shows the 'Usage View' of dependencies between selected modules. It includes two tree views for 'From Module' and 'To Module', both showing a hierarchy starting with 'Application' and 'Main'. Below these are checkboxes for 'Include direct dependencies' and 'Include indirect dependencies'. A table lists the dependencies between modules.

From	To	Type	Line	Direct
husacct.define.presentation.Application.InternalFrame	husacct.control.ILocaleChangeListener	Import	3	Direct
husacct.define.presentation.Application.InternalFrame	husacct.control.ILocaleChangeListener	Inheritance	15	Direct
husacct.define.presentation.dialog.AddModuleValuesJDialog	husacct.control.ControlServiceImpl	Import	5	Direct
husacct.define.presentation.dialog.AddModuleValuesJDialog	husacct.control.ILocaleChangeListener	Import	6	Direct
husacct.define.presentation.dialog.AddModuleValuesJDialog	husacct.control.ILocaleChangeListener	Inheritance	28	Direct
husacct.define.presentation.dialog.AddModuleValuesJDialog	husacct.control.ControlServiceImpl	Declaration	44	Direct
husacct.define.presentation.dialog.AddModuleValuesJDialog	husacct.control.ControlService	Call	44	Indirect
husacct.define.presentation.dialog.AppliedRuleJDialog	husacct.control.ControlServiceImpl	Import	5	Direct
husacct.define.presentation.dialog.AppliedRuleJDialog	husacct.control.presentation.util.DialogUtils	Import	6	Direct
husacct.define.presentation.dialog.AppliedRuleJDialog	husacct.control.ControlServiceImpl	Declaration	58	Direct

© HU

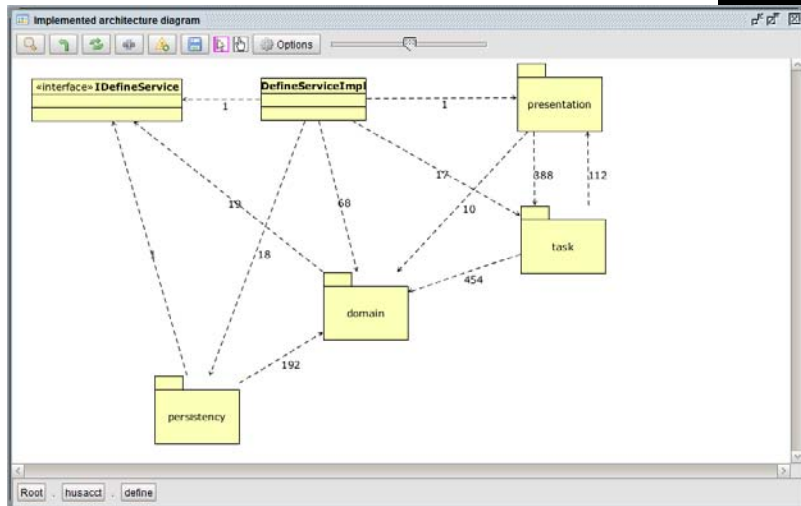
NIOC 2015

11

Analyse Implemented Architecture



Implemented Architecture Diagram



© HU

NIOC 2015

12

Architecture Compliance Checking (ACC)

- ACC verifies the conformance of
 - Implemented program code to
 - High-level models of architectural design

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

```

            graph TD
            Architect((Architect)) --> IA[Intended Architecture]
            IA --> CC((Conformance Check))
            PC[Program Code] --> SA((Static Analysis))
            SA --> CC
            CC -- Violations --> Architect
            IA --> M[Mapping]
            M --> CC
            
```

Intended Architecture
Layers: L1, L2, L3

Mapping
L1 → P1, ...

Rules
L3 is not allowed to use L2 (Back call), ...

Program Code
Package P1
Class C1, Class C2 ...
Package P2
Class C3 ...

©HU
NIOC 2015
13

ACC Example: Modular Architecture HUSACCT_1.0

Top-level Components

Rules

From-Module	Constraint	To-Module
Analyse	is not allowed to use	Define
Analyse	is not allowed to use	Validate
General GUI & Control	Is the only module allowed to use	Graphics
All five components	Facade convention	

©HU
NIOC 2015
14

Define Intended Architecture



Define intended architecture

Module Hierarchy

- SoftwareArchitecture
 - General GUI & Control
 - Analyse
 - Facade<Analyse>
 - Define
 - Validate
 - Graphics
 - Common
 - ExternalSystems

Module Properties

Module name: Analyse
 Description:
 Module Type: Component

Assigned Software Units

Software unit name	Type
husacct.analyse	PACKAGE

Rules

Rule type	To module	Enabled	Exceptions
Facade convention	On	2	
Is not allowed to use	Validate	On	0
Is not allowed to use	Define	On	0

© HU

NIOC 2015

15

Validate Conformance

Results of the Conformance Check



Validate conformance

Violations Per Rule | All Violations

Rules with Number of Violations

Id	Logical module from	Rule type	Logical module to	Violations
1	Analyse	Is not allowed to use	Define	1
2	Define.Presentation	Is not allowed to skip call		10
3	Define.Task	Is not allowed to back call		113
4	Define.Task	Is not allowed to skip call		18
5	General GUI & Control	Facade convention		51

Violations

From	To	Rule type	Dep.type	Direct	Line
husacct.analyse.presentation.ExportDependencie	husacct.control.task.threading.ThreadWithLoader	Facade convention	Import	Direct	5
husacct.analyse.presentation.ExportDependencie	husacct.control.task.threading.ThreadWithLoader	Facade convention	Access	Indirect	74
husacct.analyse.presentation.ExportDependencie	husacct.control.task.threading.ThreadWithLoader	Facade convention	Declaration	Direct	74
husacct.analyse.presentation.ExportDependencie	husacct.control.task.threading.ThreadWithLoader	Facade convention	Call	Direct	75
husacct.validate.presentation.BrowseViolations	husacct.control.task.threading.ThreadWithLoader	Facade convention	Access	Indirect	201
husacct.validate.presentation.BrowseViolations	husacct.control.task.threading.ThreadWithLoader	Facade convention	Declaration	Direct	201
husacct.validate.presentation.BrowseViolations	husacct.control.task.threading.ThreadWithLoader	Facade convention	Call	Direct	202
husacct.validate.presentation.BrowseViolations	husacct.control.task.threading.ThreadWithLoader	Facade convention	Access	Indirect	393
husacct.validate.presentation.BrowseViolations	husacct.control.task.threading.ThreadWithLoader	Facade convention	Declaration	Direct	393
husacct.validate.presentation.BrowseViolations	husacct.control.task.threading.ThreadWithLoader	Facade convention	Call	Direct	394
husacct.validate.presentation.BrowseViolations	husacct.control.task.threading.ThreadWithLoader	Facade convention	Access	Indirect	398
husacct.validate.presentation.BrowseViolations	husacct.control.task.threading.ThreadWithLoader	Facade convention	Declaration	Direct	398
husacct.validate.presentation.BrowseViolations	husacct.control.task.threading.ThreadWithLoader	Facade convention	Call	Direct	399
husacct.validate.presentation.BrowseViolations	husacct.control.task.threading.ThreadWithLoader	Facade convention	Import	Direct	399

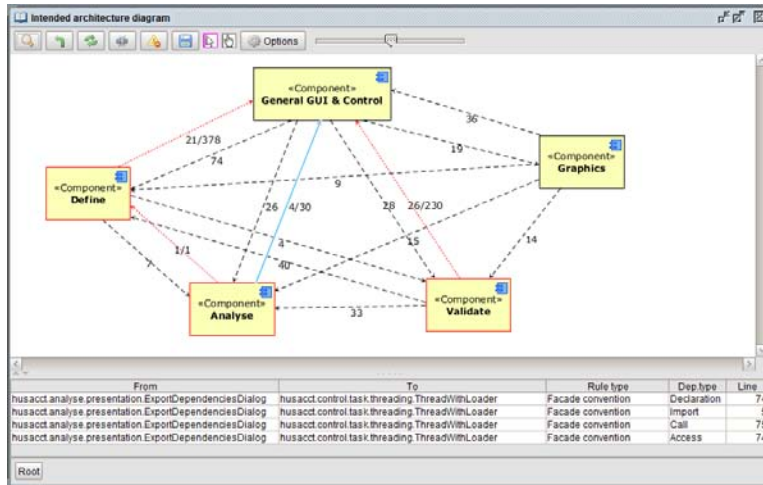
© HU

NIOC 2015

16

Intended Architecture Diagram

Top-level components, with dependencies and **violations**



© HU

NIOC 2015

17

Code Viewer Double-click on dependency



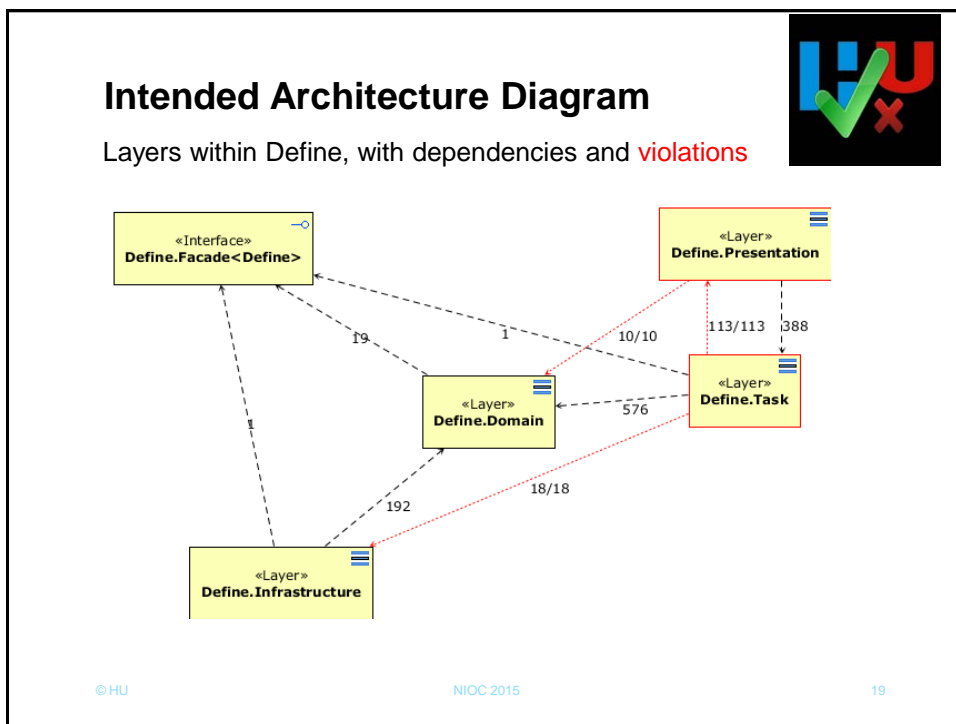
```

ExportDependenciesDialog.java
52  exportButton.setEnabled(false);
53  pathField.setEnabled(false);
54
55  getRootPane().setDefaultButton(exportButton);
56
57  add(pathLabel);
58  add(pathField);
59  add(browseButton);
60  add(exportButton);
61
62
63  private void setListeners(){
64      browseButton.addActionListener(new ActionListener() {
65          public void actionPerformed(ActionEvent arg0) {
66              showFileDialog();
67          }
68      });
69      exportButton.addActionListener(new ActionListener() {
70          public void actionPerformed(ActionEvent arg0) {
71              if(validateData()) {
72                  dispose();
73                  ThreadedDependency/Export dependency/Export = new ThreadedDe
74 selectedFile.getAbsolutePath());
75                  ThreadWithLoader analyseExportThread =
76 ServiceProvider.getInstance().getControlService().getThreadWithLoader(ServiceProvider.getInstance().getLocaleService().getTranslate
77 "ExportingDependencies", dependency/Export);
78                  analyseExportThread.run();
79              }
80          }
81      });
82  private void showFileDialog() {
83      FileNameExtensionFilter filter = new FileNameExtensionFilter("xls", "xls");
    
```

© HU

NIOC 2015

18



Finally

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

- 1) HUSACCT: Architecture Compliance Checking with Rich Sets of Module and Rule Types. *2014 IEEE/ACM Int. Conf. on Automated Software Engineering, ASE 2014-09, Vasteras, Sweden*
- 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*
- 3) A Metamodel for the Support of Semantically Rich Modular Architectures in the Context of Architecture Compliance Checking. *SAEroCon workshop, WICSA 2014-04, Sydney, Australia*
- 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*

©HU NIOC 2015 20