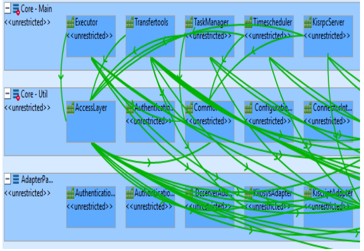
Software Construction

RWTH AACHEN UNIVERSITY

Static and Dynamic Architecture Conformance Checking: A Systematic, Case Study-Based Analysis on Tradeoffs and Synergies

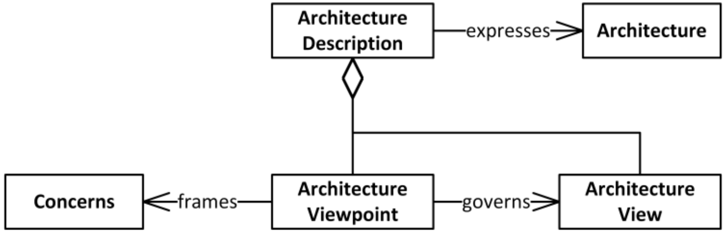
Ana Nicolaescu (Dragomir)
ana.nicolaescu@swc.rwth-aachen.de

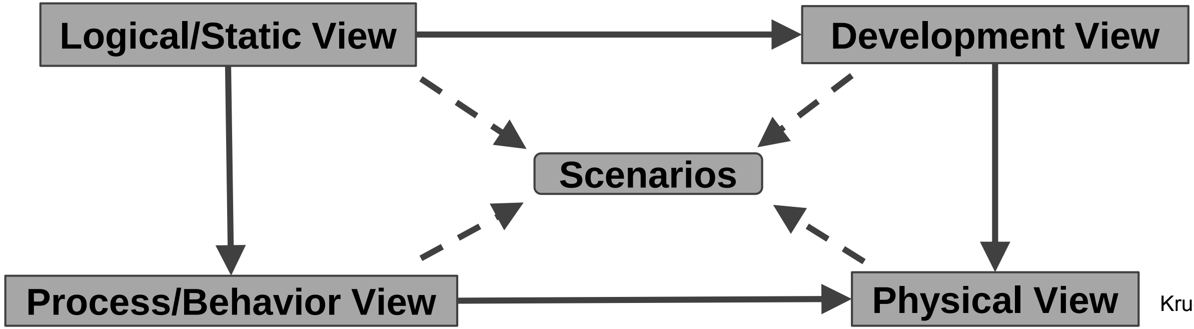
04/12/2017



What is a Software Architecture (Description)

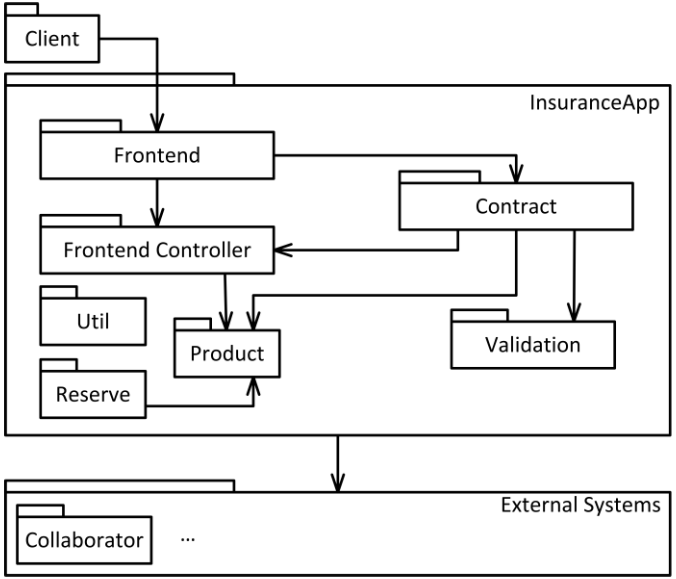
“fundamental concepts or properties of a system in its environment embodied in its elements, relationships, and in the principles of its design and evolution” (ISO/IEC/IEEE 42010)





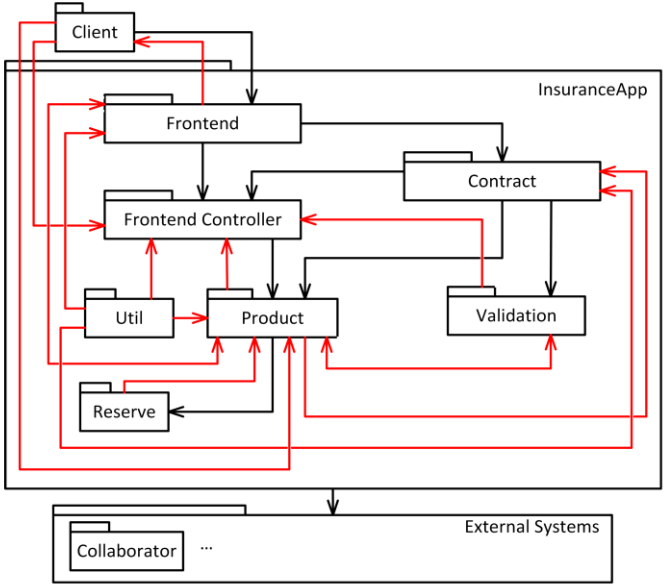
Kruchten’s 4+1

Motivation



Intended Architecture

Drift
Erosion
Gap
Chasm
…

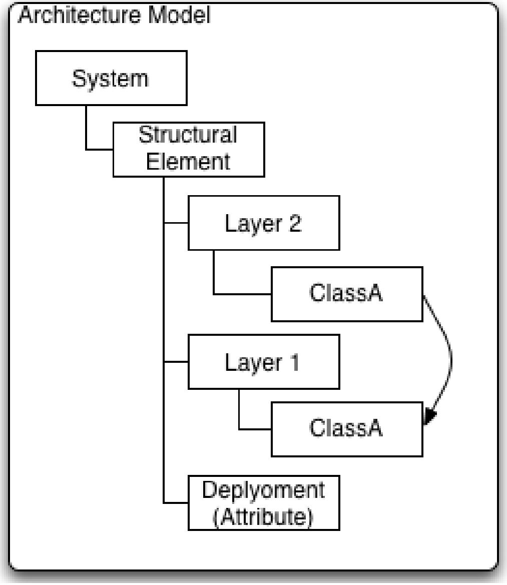


Implemented Architecture

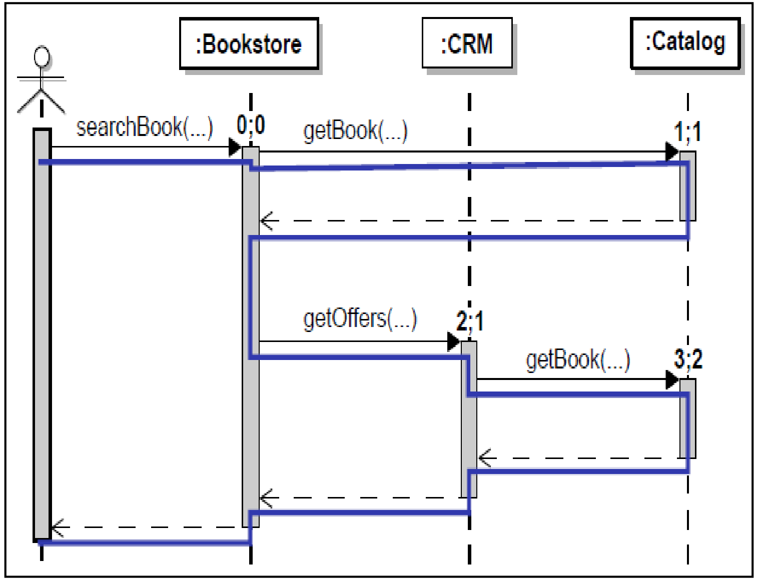
conformance checking = a process conducted to reveal the drift emerged between the intended and the implemented architecture of a system

Static vs. Dynamic Approaches

Static



Dynamic



Static vs. Behavior View

Relation	Behavior View	Static View
Method Invocation / Instantiation	Yes	Yes
Extends / Implements / Import / Variable Access	No	Yes
Remoting (REST, Queue, ...) / Dynamic Usage (Reflection, DI, ...)	Yes	No
Execution Frequency / Time / Order	Yes	No

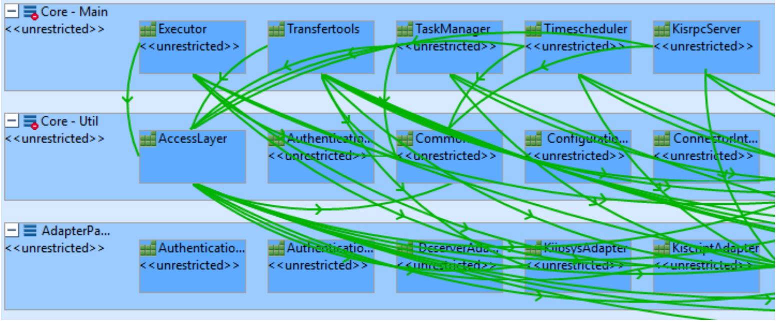
1999, Thomas Ball: behavior-based solutions to software architecture understanding have a more adequate scope and better precision

2014, Nenad Medvidovic (ECSA Keynote); conformance checking should go beyond structure!

Sonargraph Architect

- Architecture conformance checking based on static source-code analysis

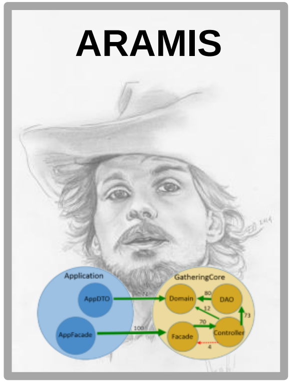
- Detect violations against specified architecture rules



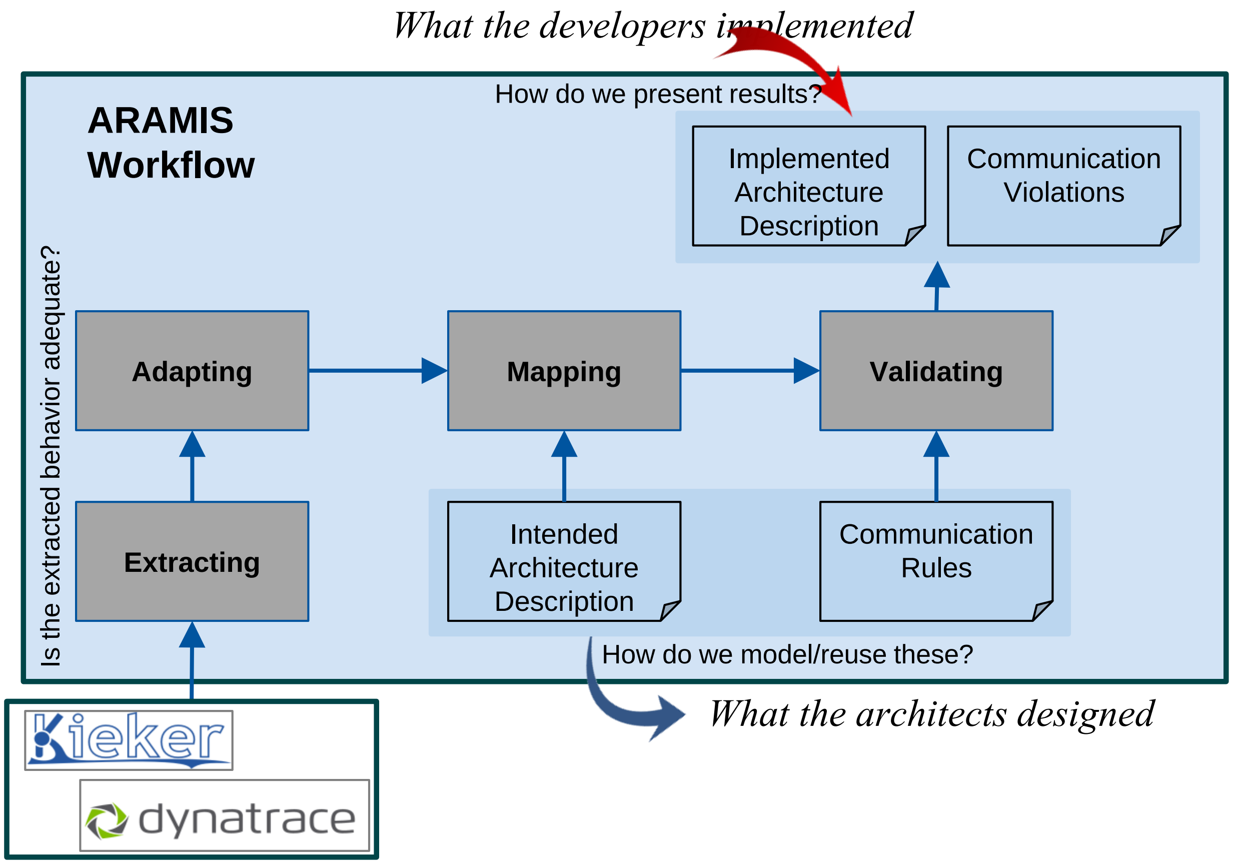
ARAMIS

- **AR**chitectural **A**nalysis and **M**onitoring **I**nfra**S**tructure

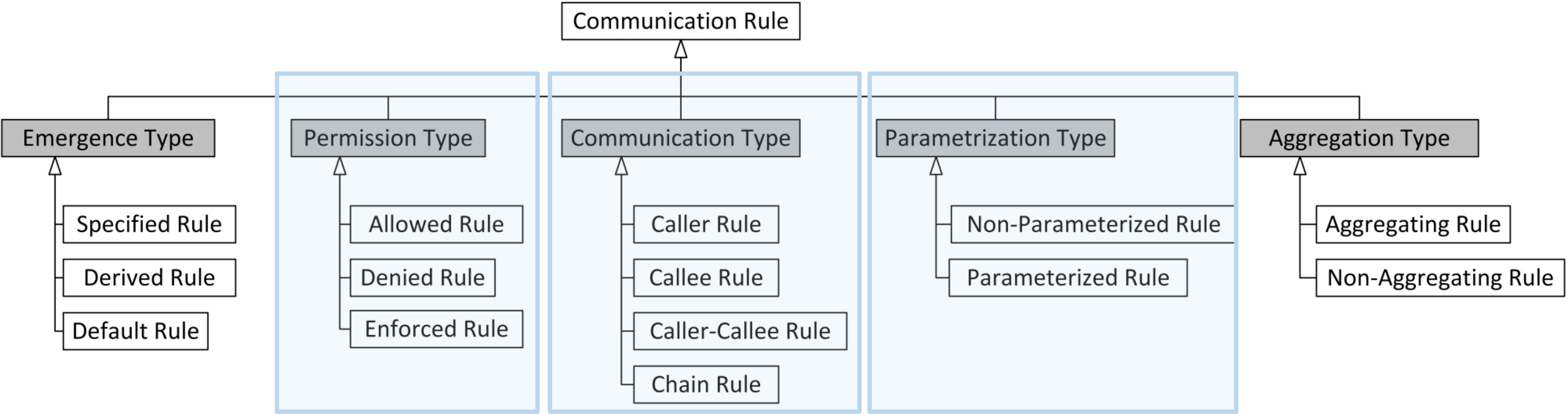
- Behavior-based architecture conformance checking
- Monitor & analyze communication
- Detect violations against specified architecture rules



The Architectural Analysis and Monitoring Infrastructure


What the developers implemented
How do we present results?
ARAMIS Workflow
Implemented Architecture Description
Communication Violations
Is the extracted behavior adequate?
Adapting
Mapping
Validating
Extracting
Intended Architecture Description
Communication Rules
How do we model/reuse these?
What the architects designed
Kieker
dynatrace

Communication Rules in ARAMIS



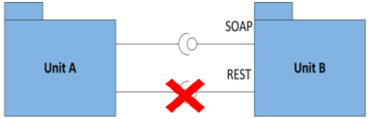
Communication Rules in ARAMIS

From **Simple Rules**

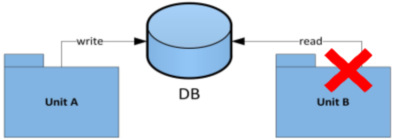
One architecture unit is allowed/denied to (transitively) access another one

To **Complex Rules**

Communication-protocol based rules



Indirect coupling rules



…

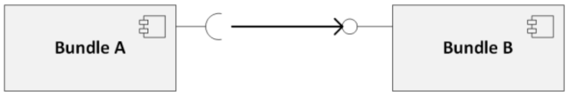
ARAMIS Case Studies – TADD (Task Automation and Data Distribution System)

LOC	~125000
Number of Processes	5
Number of OSGI Bundles	30
Number of Source Files	879

Statement Coverage	33% / 17%
Adequate	?

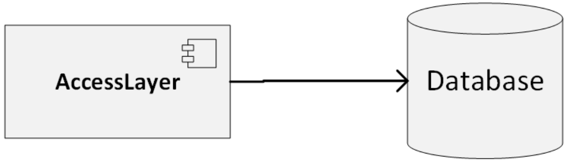
140 Squish Tests
6h execution time

TADD - Some Rules



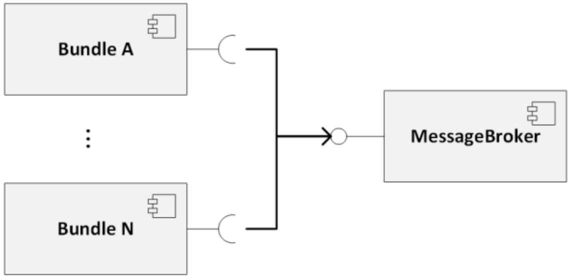
Bundle A

Bundle B



AccessLayer

Database

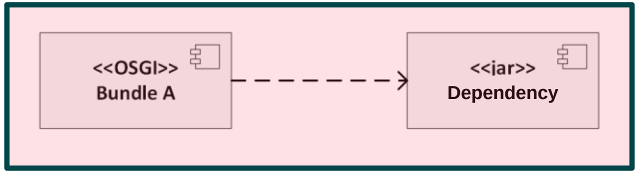


Bundle A

⋮

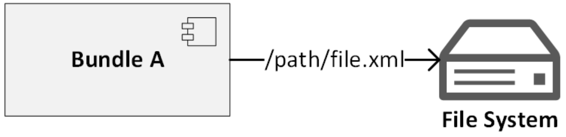
Bundle N

MessageBroker



<<OSGI>>
Bundle A

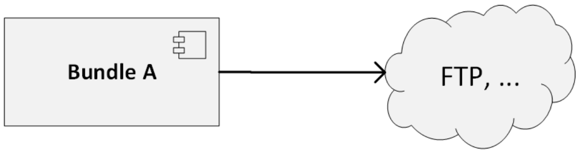
<<jar>>
Dependency



Bundle A

/path/file.xml

File System



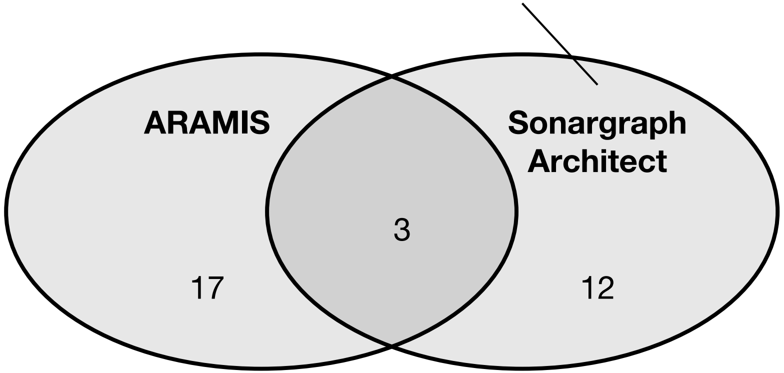
Bundle A

FTP, ...

Evaluation - Violations

4 are in the scope of ARAMIS

file-systems violations
performance issues



use of external dependencies

ARAMIS vs. Sonargraph - number of violation types

False positives: 2 polymorphism anomalies
4 partial trace anomalies

False positives: 6 split package anomalies

Comparison - 1

Dimension	Behavioral Approach	Static Approach
Inputs	● Architecture documentation ● Source code ● Episode selection ● Instrumentation configuration	● Architecture documentation ● Source code
Analysis Scope	● Systems: heterogeneous ● Relations: includes dynamic usage and remoting ● Rules: direct and indirect usage	● Systems: homogeneous ● Relations: static / source ● Rules: direct usage
Completeness	● Depends on selected episodes ● Approximately measured by Coverage metrics	● Complete

Comparison - 2

Dimension	Behavioral Approach	Static Approach
Causes for false positives	● Split packages ● Polymorphism and partial traces anomaly	● Split packages
Eval. Performance	● 13 to 33 hours ● Long running process	● Less than a minute ● Instant feedback

Conslusions

- **Dynamic Approaches**
  - Much effort and resources needed
  - Valuable insights into run-time
  - Suitable for detailed on demand analyses
- **Static Appoaches**
  - Less effort needed
  - Suitable for continuous analysis
- Both have strengths, weaknesses
- Approaches can not substitute each other
- Proposal: if possible, combine both for broader analysis scope

Thank You!