ISE Paper Talk:

“Scenario-Based Analysis of Software Architectures”

R.Kazman, G. Abowd, L. Bass, P. Clements

Speaker: Peter Knauber

November 12, 1997

Filename: 1997-11-12.PaperTalk@IESE.fm

Fraunhofer Einrichtung
Experimentelles
Software Engineering

Outline

- Motivation
- Software Architecture Analysis Method (SAAM)
- Case Study: WRCS
- Conclusion / Experiences
- Open Questions

Fraunhofer Einrichtung
Experimentelles
Software Engineering

Motivation

Advances of architecture analysis:

Definition: Software architectures describe a high-level configuration of components that compose the system and the connections that coordinate the activities of those components

Why analyze architectures? Significance of software architectures:

- manifestation of early design decisions
- key artifact in successful product line engineering
- means of communication between developers and managers

Quality Attributes

The architecture of a software system is responsible for non-functional qualities of the system

Goal: Evaluate software architectures with respect to certain quality attributes

Problem: Quality attributes are too abstract. Example:

usability: by novice / by expert

Other quality attributes:

- modifiability
- reliability
- reusability

Scenarios

Quality attributes do not exist in isolation, they have meaning within a context:

- usable (or not)
  with respect to certain user groups
- modifiable (or not)
  with respect to certain classes of changes

Scenarios are used to realize quality attributes in a sufficiently concrete way for meaningful analysis

Scenarios have to be weighted according to their relative significance (consider future uses, consider all roles relevant to a system)

Fraunhofer Einrichtung
Experimentelles
Software Engineering

Software Architecture Analysis Method (SAAM)

Definition:

SAAM is a method for analyzing software architecture descriptions with respect to the requirements of some roles related to the system in order to determine how non-functional software qualities are supported.

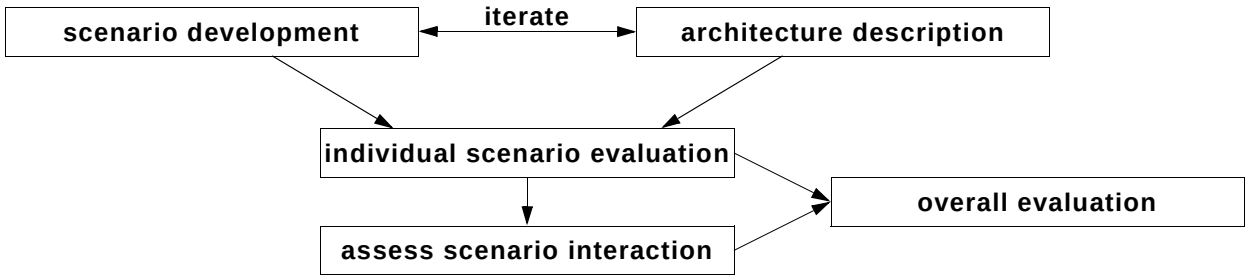
Areas of Application:

Two possible areas of application:

- comparison of different system architectures
- evaluation of a single system architecture to discover possible weak points

Fraunhofer Einrichtung
Experimentelles
Software Engineering

SAAM Overview



Passes 1 and 2 are highly interdependent:

- kind of scenario
  → level of granularity for an architecture description
- architectural changes
  → scenario generation

IESE
Fraunhofer Einrichtung
Experimentelles
Software Engineering

Case Study: WRCS

WRCS: WRCS is a commercially-available revision control system

Functionality:

- create archives
- compare files
- check files in and out
- create releases
- back up to old versions
- …

IESE
Fraunhofer Einrichtung
Experimentelles
Software Engineering

Applying SAAM to WRCS

Available information:

- interviews with development team members
- documentation
- the WRCS system

→ no source code, no requirements!

Start with modules with their relationships (instead of an architectural representation)

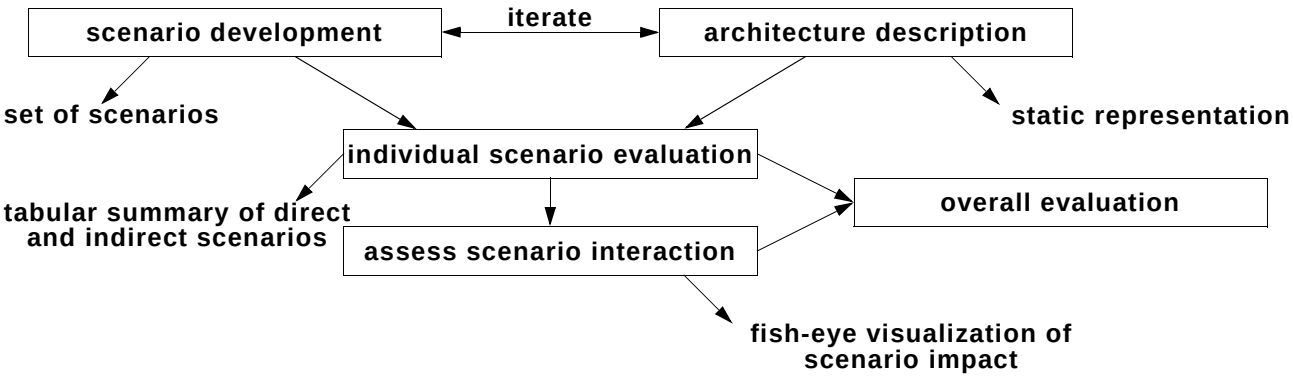
Iteration:

- Questions (Scenarios)
- Complete description

Scenarios

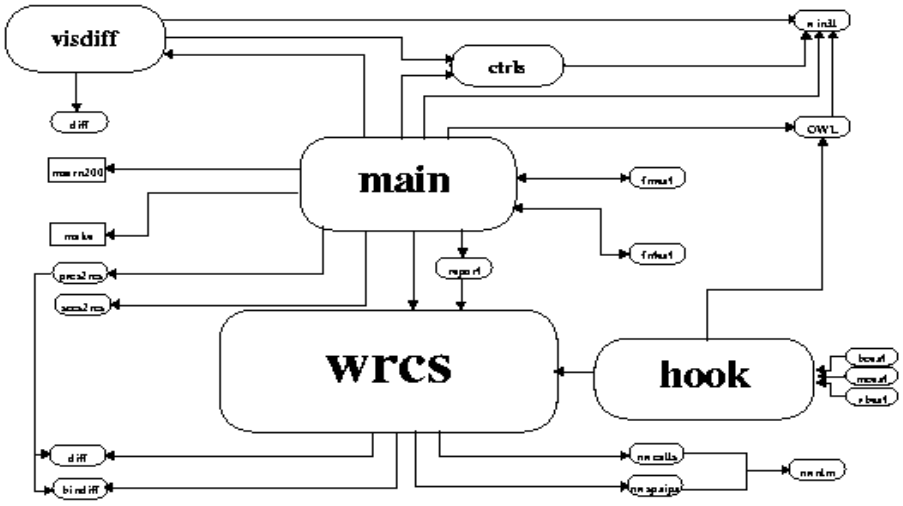
role	scenario description	scenario classification	changes
user	compare new binary file representations	indirect	…
	configure the product's toolbar	direct	
maintainer	port to another operating system	indirect	…
	make minor modifications to the user interface	indirect	…
administrator	change access permissions	direct	
	integrate with a new development environment	indirect	…

Products of the SAAM Process



IESE
Fraunhofer Einrichtung
Experimentelles
Software Engineering

Result: Fish-Eye Representation of WRCS Architecture



IESE
Fraunhofer Einrichtung
Experimentelles
Software Engineering

Conclusion

Experiences

Mixed acceptance:

- senior developers, managers:
  - very important tool in architecture analysis
  - identifies many potential problems early in the software life cycle (reduce costs!)
- WRCS development team:
  - just another academic exercise

Limitations of SAAM:

- Possible usage of tools is not considered
- Possible existance of domain-specific languages is not considered
- Performance, architectural elements are represented in domain-specific ways

Open Questions

- How to find out
  - all roles that are important for a system?
  - all scenarios that have an impact on the architecture?
- How to consider tools, domain-specific languages?
- How to deal with, e.g., performance requirements?
- Where are the limits of architectural impact on non-functional quality attributes?

IESE
Fraunhofer Einrichtung Experimentelles Software Engineering