

Was tut ein guter Software Architekt?

Eberhard Wolff

Architecture and Technology Manager

adesso AG

- ▶ Eberhard Wolff
- ▶ Architecture & Technology Manager at adesso
- ▶ adesso is a leading IT consultancy in the German speaking region
- ▶ Speaker
- ▶ Author

- ▶ Responsible for the Architect Training Program at adesso AG

- ▶ Blog: <http://ewolff.com>
- ▶ Twitter: [@ewolff](https://twitter.com/ewolff)
- ▶ <http://www.slideshare.net/ewolff>
- ▶ eberhard.wolff@adesso.de

Software architect is a general term with many accepted definitions which refers to a broad range of roles.
Not really well defined...



The software architecture are those

decisions

that are

hard to change

Martin Fowler



Decisions: A Close Look

- ▶ Architect makes decision that are hard to change
- ▶ Architect is responsible
- ▶ Architect takes responsibility
- ▶ Is responsible for the commercial success
 - > Wrong decisions hard to revise
 - > i.e. effort and costs must be considered
- ▶ Do you know the Business Value of your project?
- ▶ Do you know the stakeholders and their agenda?
- ▶ How do your decisions match that?



Decide based on business value
Know the business value!

Technology Decision

- ▶ Which technologies and approaches will you use?
- ▶ Important part of architecture

- ▶ Need to know technologies and when to use what
- ▶ I.e. broad knowledge – don't get lost in details
- ▶ But you should know some basic technologies by heart
- ▶ It helps to have an interest in technologies

- ▶ Must not like technologies as an ends in itself
- ▶ ...but as a tool
- ▶ Does it help to solve the problem?
- ▶ Does it actually make my life easier?



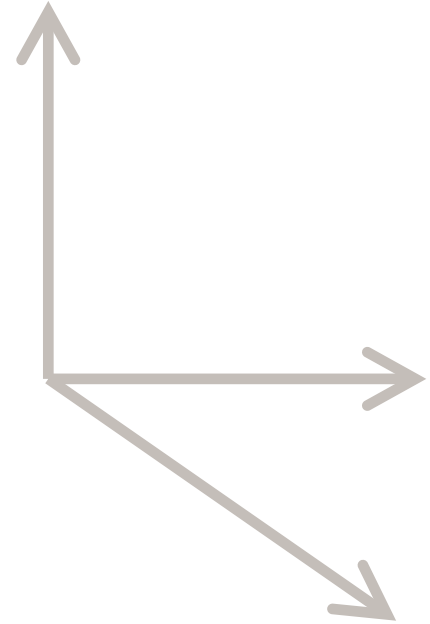
Most amazing achievement of software industry:
Continuing cancellation of the steady and staggering gains in hardware



Best F1 drivers:
Enough understanding how a machine works so they can work in harmony

Technology Decision: Multi Dimensional

- ▶ Quality of the technology itself
 - > Performance
 - > Productivity
- ▶ But the technology perspective might not be enough
 - > Skills
 - > Open Source License
 - > License costs and other costs
 - > Strategic decisions
 - > Operations
- ▶ Need to take all into account
- ▶ But: Decision can be challenged more often than you think
- ▶ Prepare your case!
- ▶ Often people are happy to get some advice



Decisions = Trade Off

- ▶ Each option has advantages and drawbacks
- ▶ ...in each dimension
- ▶ So any decision will be a trade off
- ▶ You won't find the perfect match
- ▶ So if it's not perfect – relax
- ▶ It's just too many dimensions
- ▶ Half done software might be good enough
- ▶ ...and even have a better time-to-market



Technologies are just a tool and a trade-off.
Know the technology options and the dimensions
of technology decisions!


The software architecture of a system is the set of *structures* needed to reason about it, which comprise *software elements*, *relations* among them, and *properties* of both.



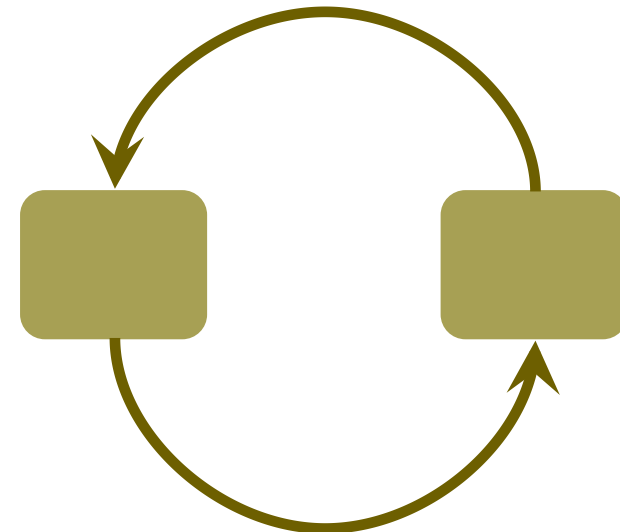
WIKIPEDIA
The Free Encyclopedia
The Free Encyclopedia
WIKIPEDIA

The software architecture of a system is the set of *structures* needed to reason about it, which comprise *software elements*, *relations* among them, and *properties* of both.

What does that actually mean?

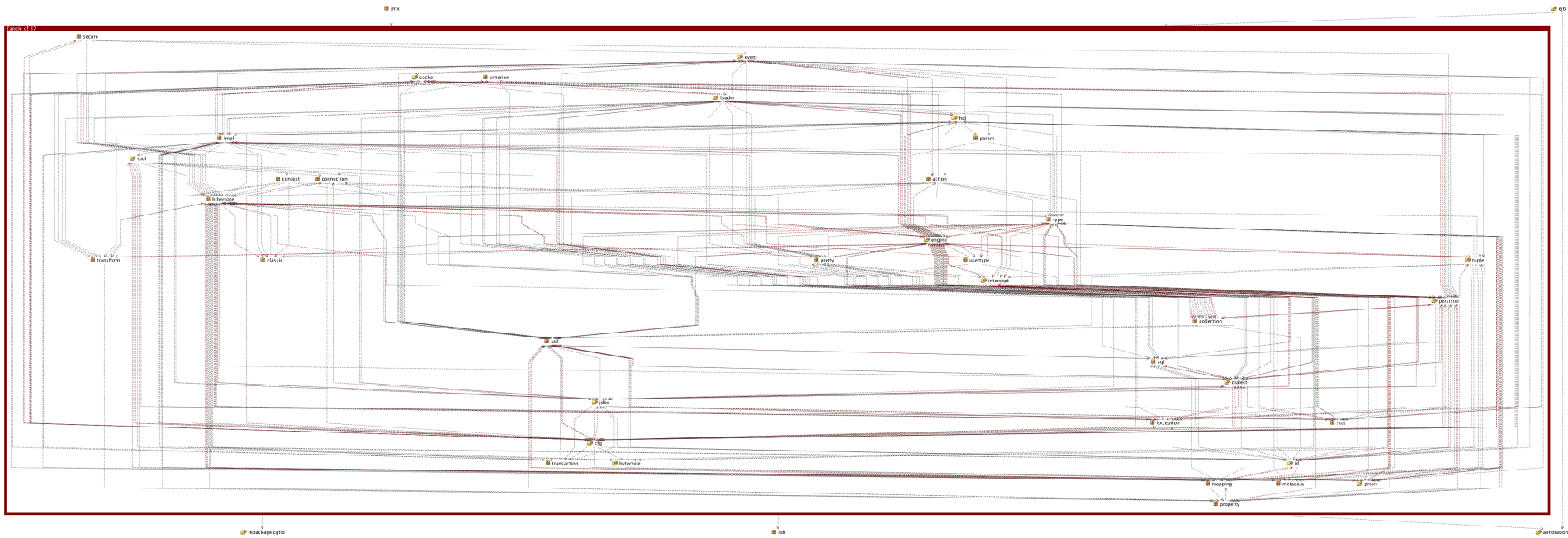


- ▶ Elements
 - > Method, classes, packages, JARs / WARs / EARs
 - > Diagrams and PowerPoint are just helpers
 - > ...and might be disconnected from reality
- ▶ Relations
 - > Dependencies i.e. usage
 - > Well ordered
 - > No excessive dependencies
- ▶ No excessively big elements
- ▶ No cyclic dependencies
 - they effectively make two elements one
- ▶ Do you manage your dependencies?
- ▶ What do you do about big elements?

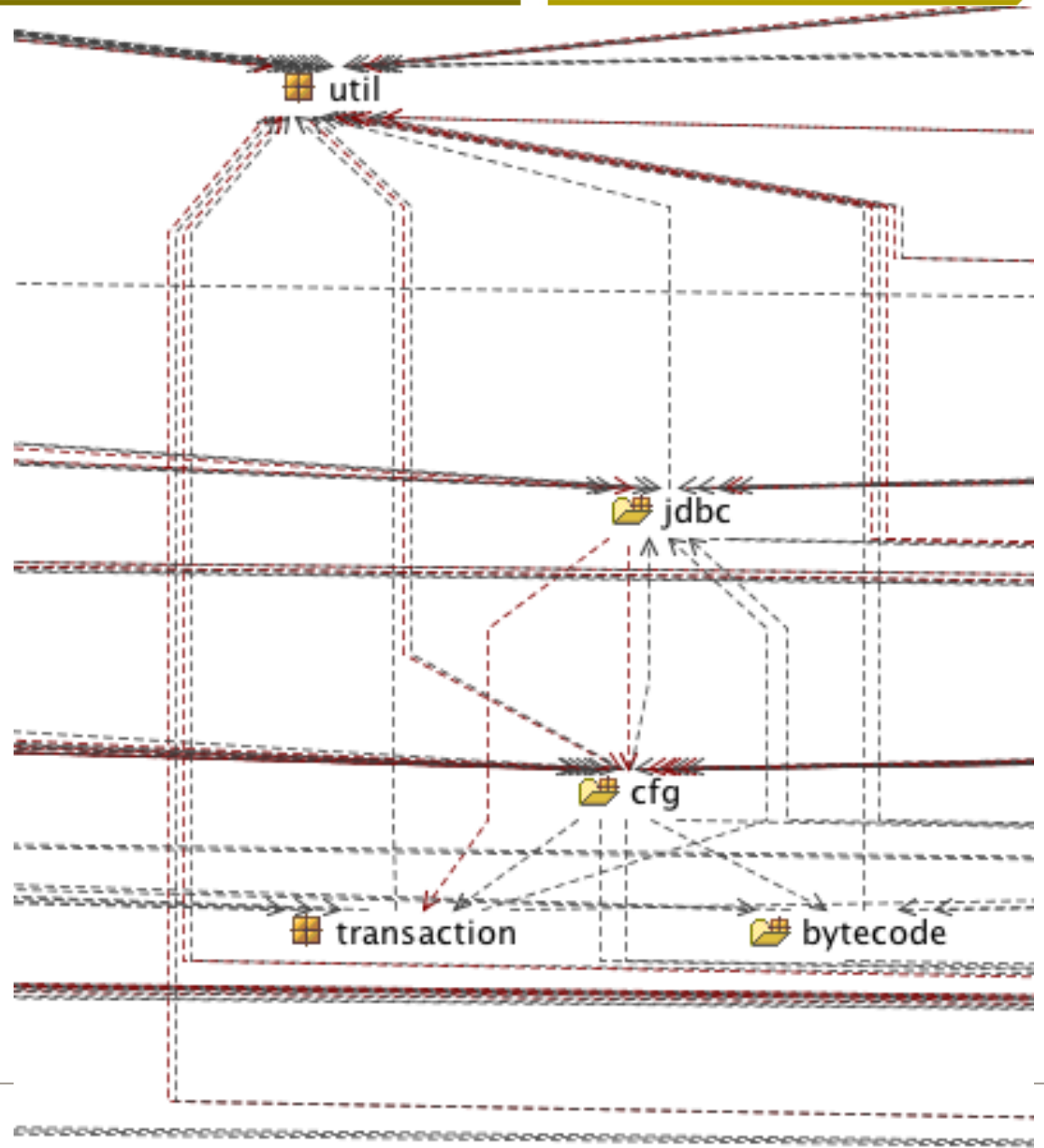


Dependency Graph

- ▶ Overview
- ▶ One large cycle



- ▶ Just a small part
- ▶ Red line show circular references



Architectural Debt

- ▶ Hard to solve if it has reached this state
- ▶ Consider managing it from the start
- ▶ ...or you are looking at a significant restructuring
- ▶ Much more than Refactoring!



Consider a tool

- ▶ Overall view on dependencies not obvious in IDE
- ▶ Without a tool structure is quite likely a mess

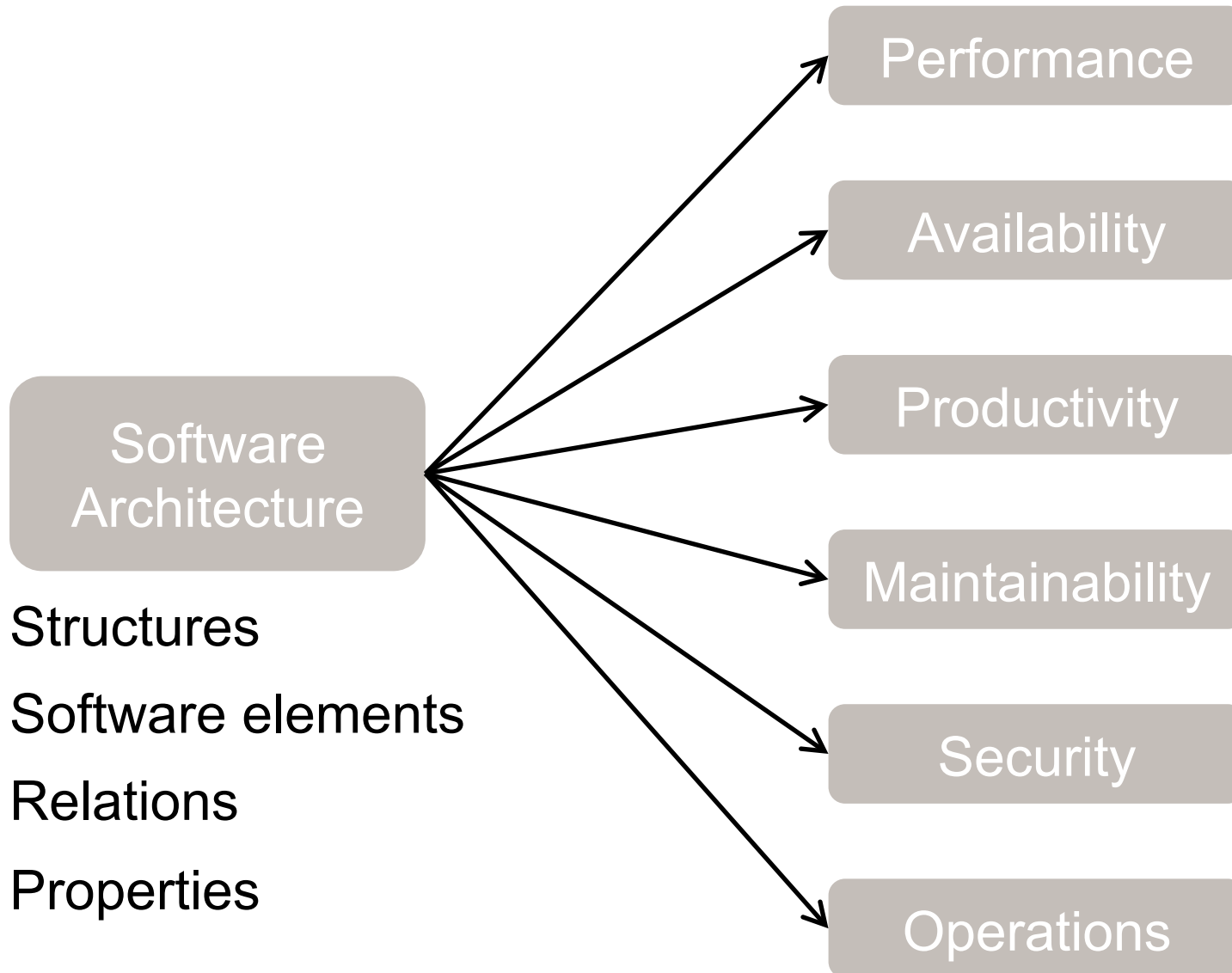
- ▶ Some tools:
 - > JDepend

 - > Structure 101
 - > Restructure 101

 - > Sonargraph

- ▶ Sonar is not enough

Manage dependencies and size of elements – ideally from the start – because that is the architecture!
You will need a tool!



Influences

non-functional

requirements &

quality

Software
Architecture

Performance

Availability

Productivity

Maintainability

Security

Compliance

Structures

Software elements

Relations

Properties

Software Architect: Traditional Role

- ▶ Manager
- ▶ Responsibility: non-functional requirements / quality
- ▶ Tool: Define and enforce architecture

- ▶ Functional requirements covered by requirements process
- ▶ Functional requirements influence the architecture

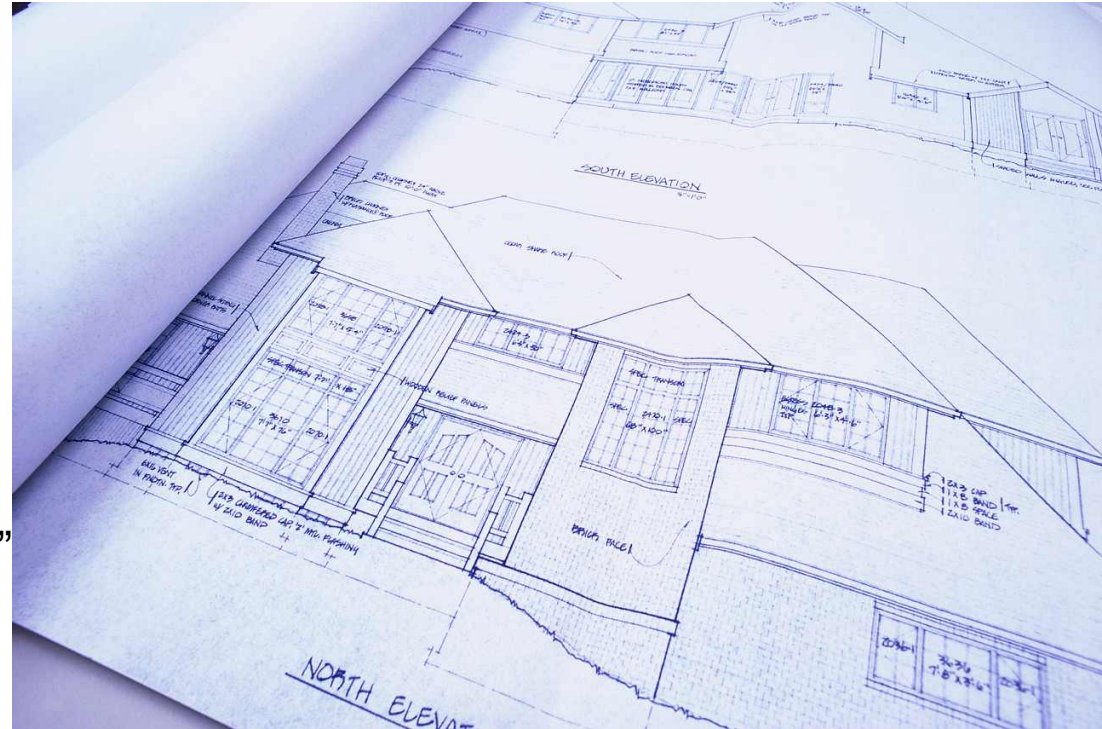


Traditional View on Architect's Responsibilities

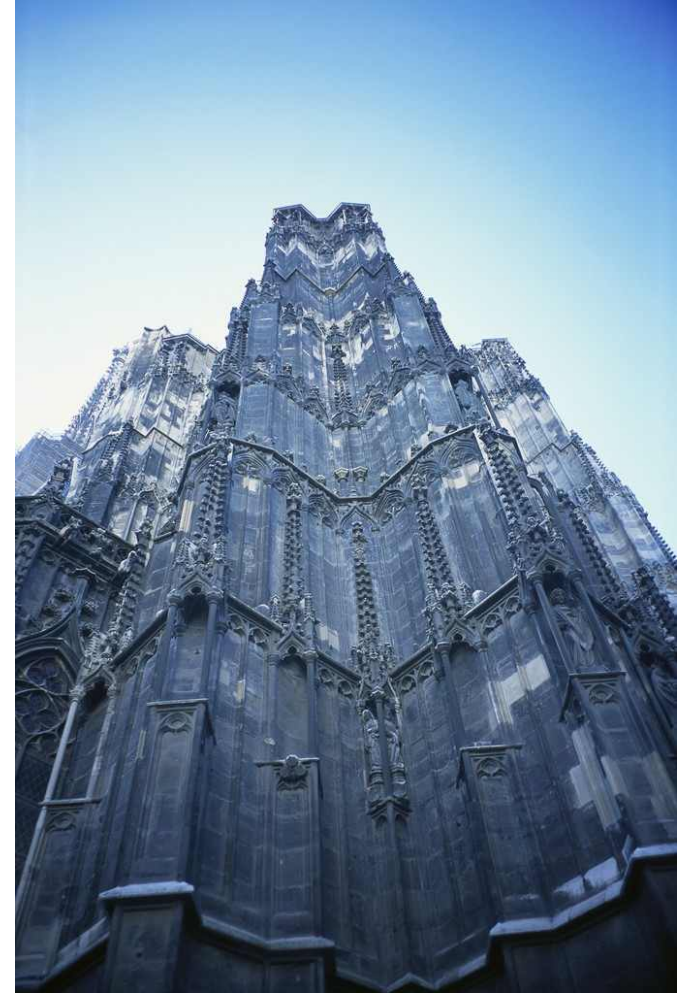
- ▶ Define the architecture
- ▶ Enforce the architecture
- ▶ i.e. create frameworks
- ▶ Not necessarily any coding
- ▶ Code reviews (maybe)

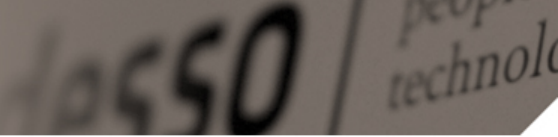
- ▶ Assumptions
 - > Separation of labor
 - > Developers must be “controlled”

- ▶ Does that still work in today's world?



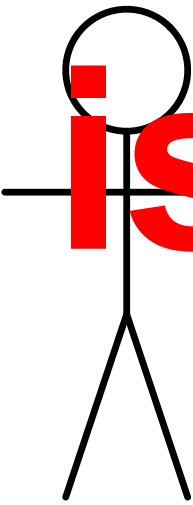
- ▶ Ivory tower architecture
- ▶ Architecture does not fit the domain
- ▶ Architecture is not in the code
- ▶ The documented architecture is different from the real architecture
- ▶ Developers don't feel their feedback is listened to
- ▶ Either architecture is ignored
- ▶ ...or project results in a failure





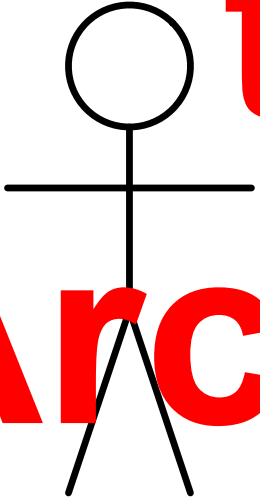
Where is

Scrum Master
Removes obstacles
enforces rules



the

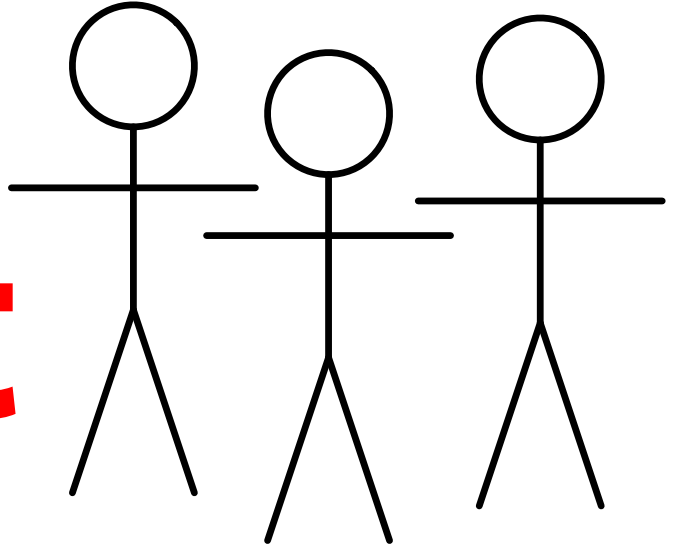
Architect



Product Owner
Creates stories



?



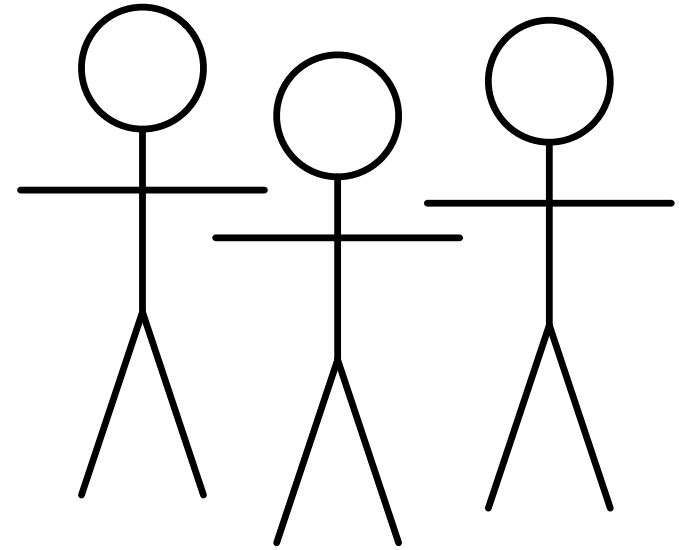
Team
Self-organizing
Implements stories

- ▶ Is self organizing
- ▶ An architect might / will emerge
- ▶ ...but is not planned for

- ▶ Benefit:
 - > Responsibility is shared
 - > i.e. not just the architect cares

- ▶ If the architecture / architect is not helpful, it / he will be ignored
- ▶ Less damage in the end

- ▶ Architect will see his ideas directly in action
- ▶ Better feedback
- ▶ Needs trust and collaboration



Architect as a Manager

- ▶ Limited tools to influence team
- ▶ Actually that is very common for managers
- ▶ A team cannot be lead against its will

- ▶ Need to listen to other team members
- ▶ ... and stake holders





Role

- ▶ Needs to collaborate with other team members
- ▶ ...and make himself useful

- ▶ Supports and trusts other team members

- ▶ Leads by experience
- ▶ ...not by title

Creating an Architecture

- ▶ Stories defined during the project
- ▶ Not all requirements known at the start
- ▶ No Big Design Upfront possible

- ▶ Architecture needs to emerge
- ▶ Architecture must be constantly redefined

- ▶ More focus on code
- ▶ Code is the reliable source for the current architecture and state of the project

Architect is a team member – need to collaborate and listen!

- ▶ Not all parts of a system will have the same quality
- ▶ Not all team members are equally skilled
- ▶ The compromise on the quality can happen “by chance”
- ▶ ...or you can steer it
- ▶ Identify core domains
 - > The ones that add the most value
 - > i.e. have a good business reason
- ▶ Might want to isolate those
- ▶ ...and focus on them



Broken Windows Theory

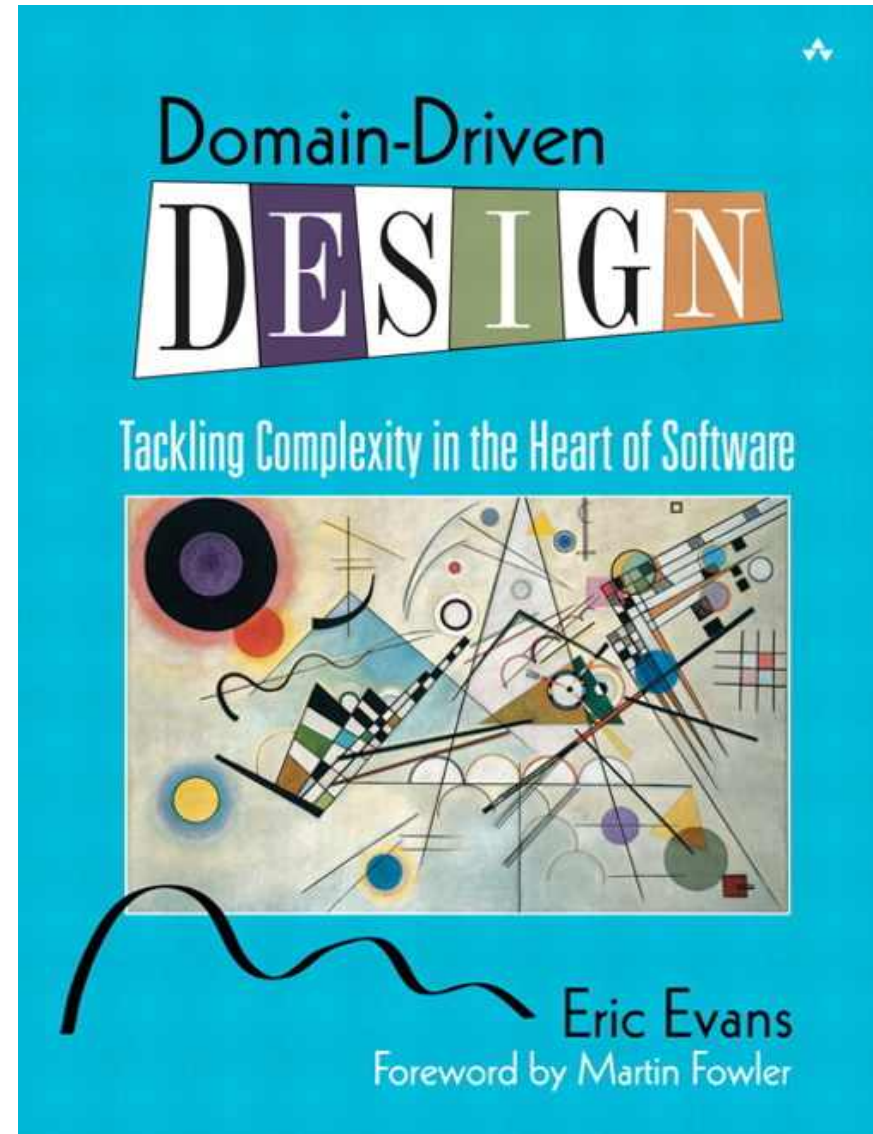
- ▶ Once windows are not repaired...
- ▶ ...vandals will break more
- ▶ ...break into the building
- ▶ ...

- ▶ Accepting compromises on quality is risky
- ▶ ...but if you strive for ultimate quality everywhere, you will fail

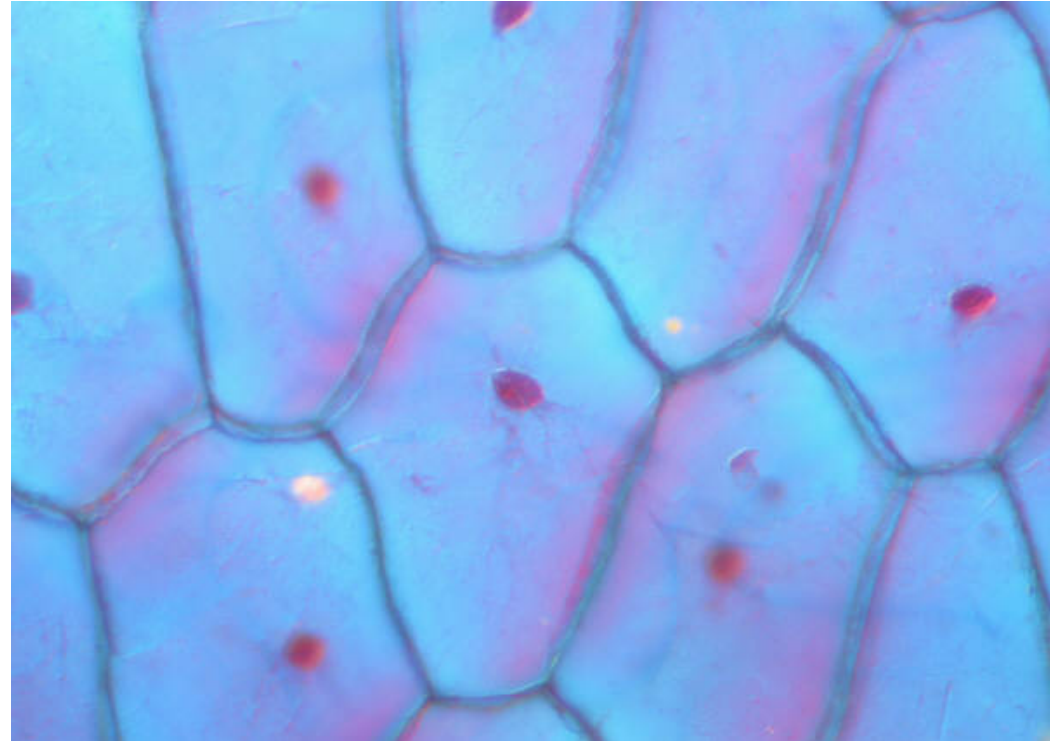
- ▶ In particular with legacy software



- ▶ “Tackling Complexity in the Heart of Software”
- ▶ E.g. Ubiquitous Language for Code, Developers and Customers

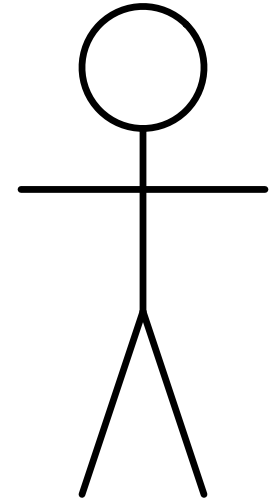


- ▶ **Bounded Context:**
Model used only in a specific part of the system
- ▶ **Context Map:**
Translate models from different parts of the system
- ▶ **Anti-Corruption Layer:**
Make sure the core domain is not corrupted



More Responsibilities for Architects

- ▶ Define the Core Domain
- ▶ Ensure that the Core Domain will be implemented properly
- ▶ ...and won't be compromised
- ▶ I.e. manage the overall quality
- ▶ Needs detailed domain knowledge
- ▶ Need to understand business case



Quality will differ in the individual parts –
Your choice is only to manage it or let it happen!



▶ <http://lemmings.mytrash.tv/>

You Must Not Be a Lemming!

- ▶ No matter what others say
- ▶ Take it as an advice
- ▶ Come to your own conclusion
- ▶ ...and work on them.

- ▶ It is you project
- ▶ It is your decision
- ▶ It is your responsibility



- ▶ If you can't come to your own conclusion you are probably not a good architect

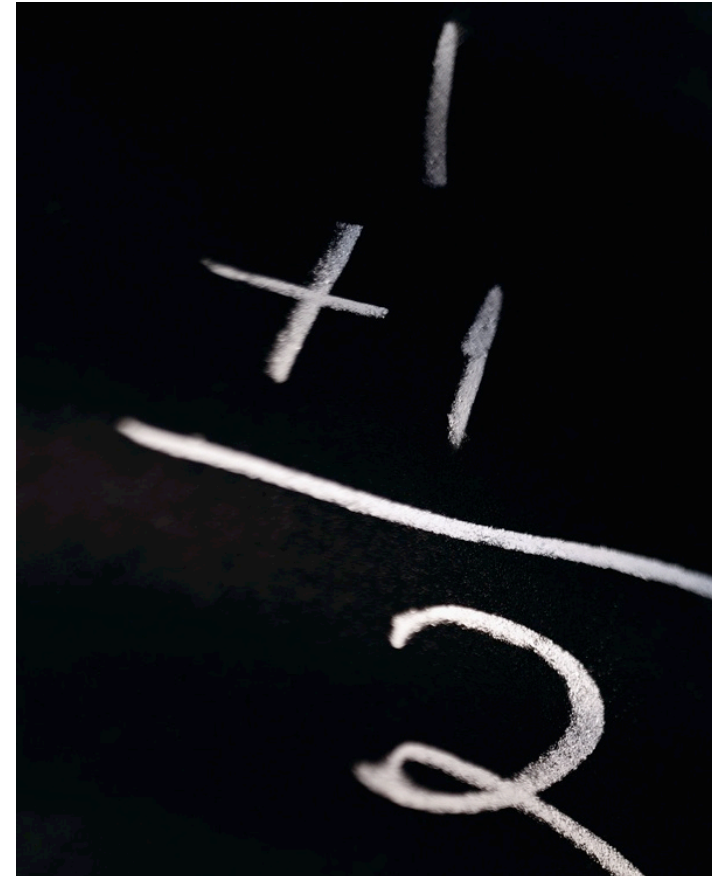
- ▶ Decide based on business value!
- ▶ Know the business value!

- ▶ Technologies: just a tool and trade-off
- ▶ Know technology options and the dimensions of technology decisions!

- ▶ Architect is a team member – need to collaborate and listen!

- ▶ Manage architecture = dependencies and size of elements

- ▶ Quality will differ in the individual parts – Your choice is only to manage it or let it happen!





Wir suchen Sie als

- Software-Architekt (m/w)
- Projektleiter (m/w)
- Senior Software Engineer (m/w)



jobs@adesso.de
www.AAAjobs.de