

Know Your Enemies

Probleme in Software finden –
aber richtig

Dr. Gernot Starke



Dr. Gernot Starke

innoQ Fellow



► *Softwarearchitekturen*

Entwurf, Entwicklung, Management
Evolution & Modernisierung
Training

► *Mentoring und Coaching*

Analyse und Optimierung von
Entwicklungsprozessen

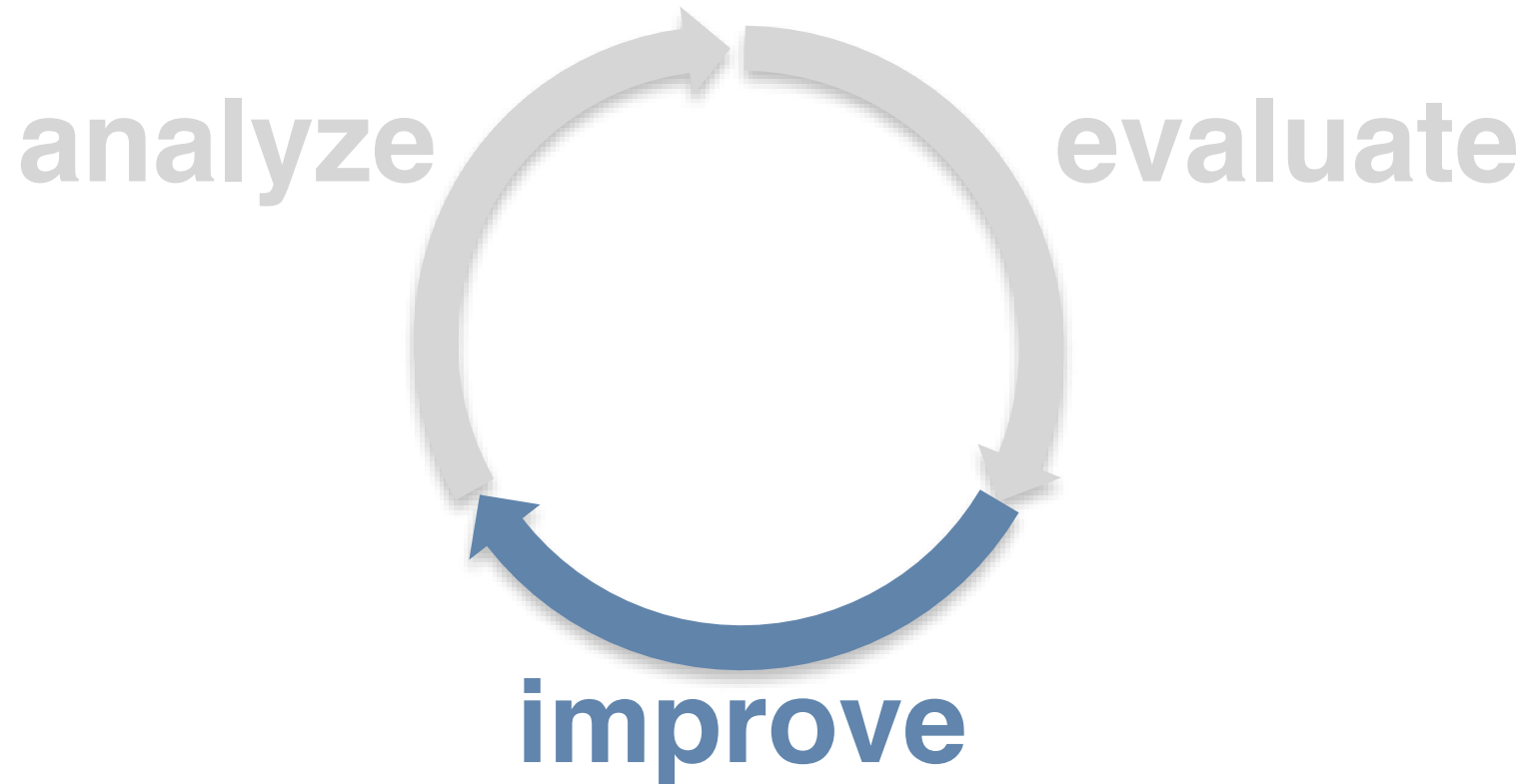
► *Reviews, Audits, Retrospektiven*

+49 177 – 728 2570
gs@gernotstarke.de

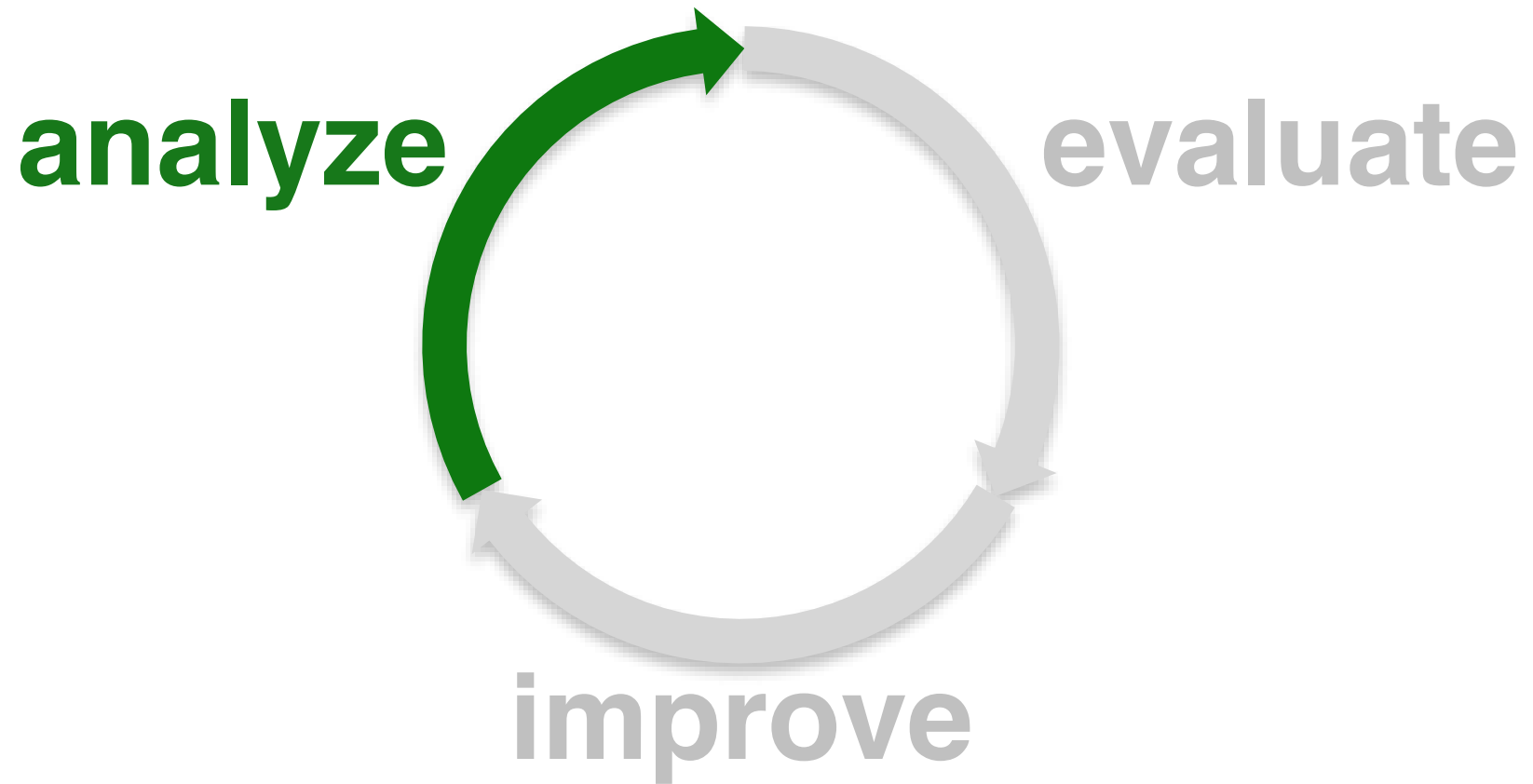
www.arc42.de



im normalen Leben...

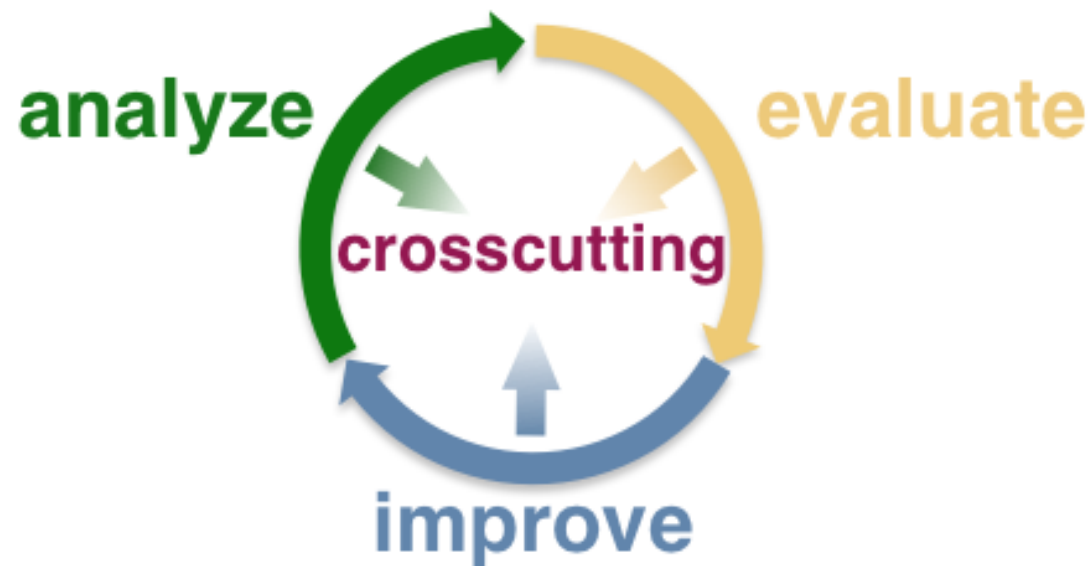


dieser Vortrag...



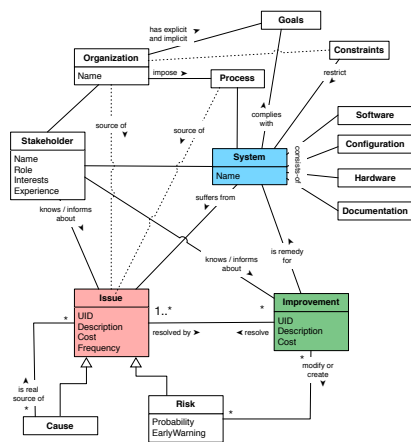
Gute Problemanalyse hilft bei...

- Priorisierung von Änderungen
- Balancieren kurz- / langfristiger Maßnahmen

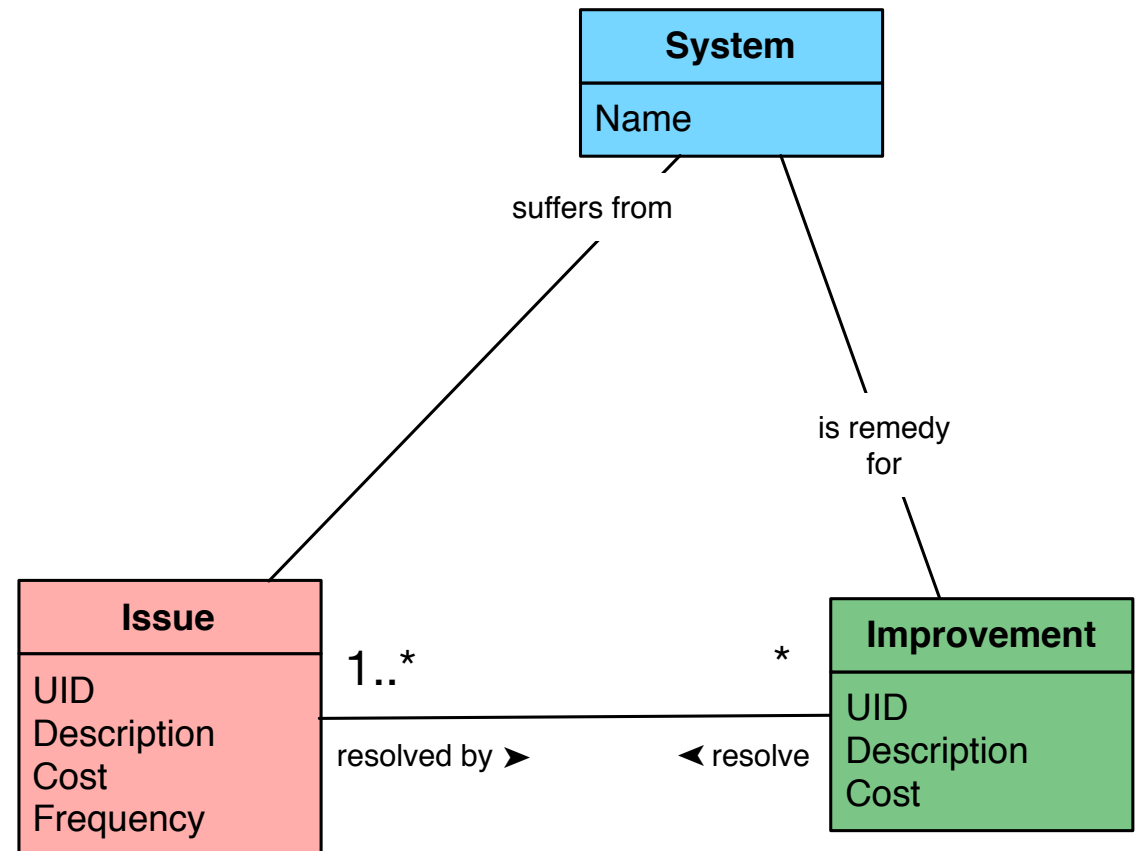


Systematisch verbessern (1)...

Trenne „Probleme“ und „Lösungsvorschläge“



aim42 domain model



warum diese Trennung?

- richtig:
 - **wichtige** Probleme lösen



- schlechter:
 - **einfache** Probleme lösen
- ganz schlecht:
 - **interessante Problemlösungen** umsetzen
 - **Aktionismus**

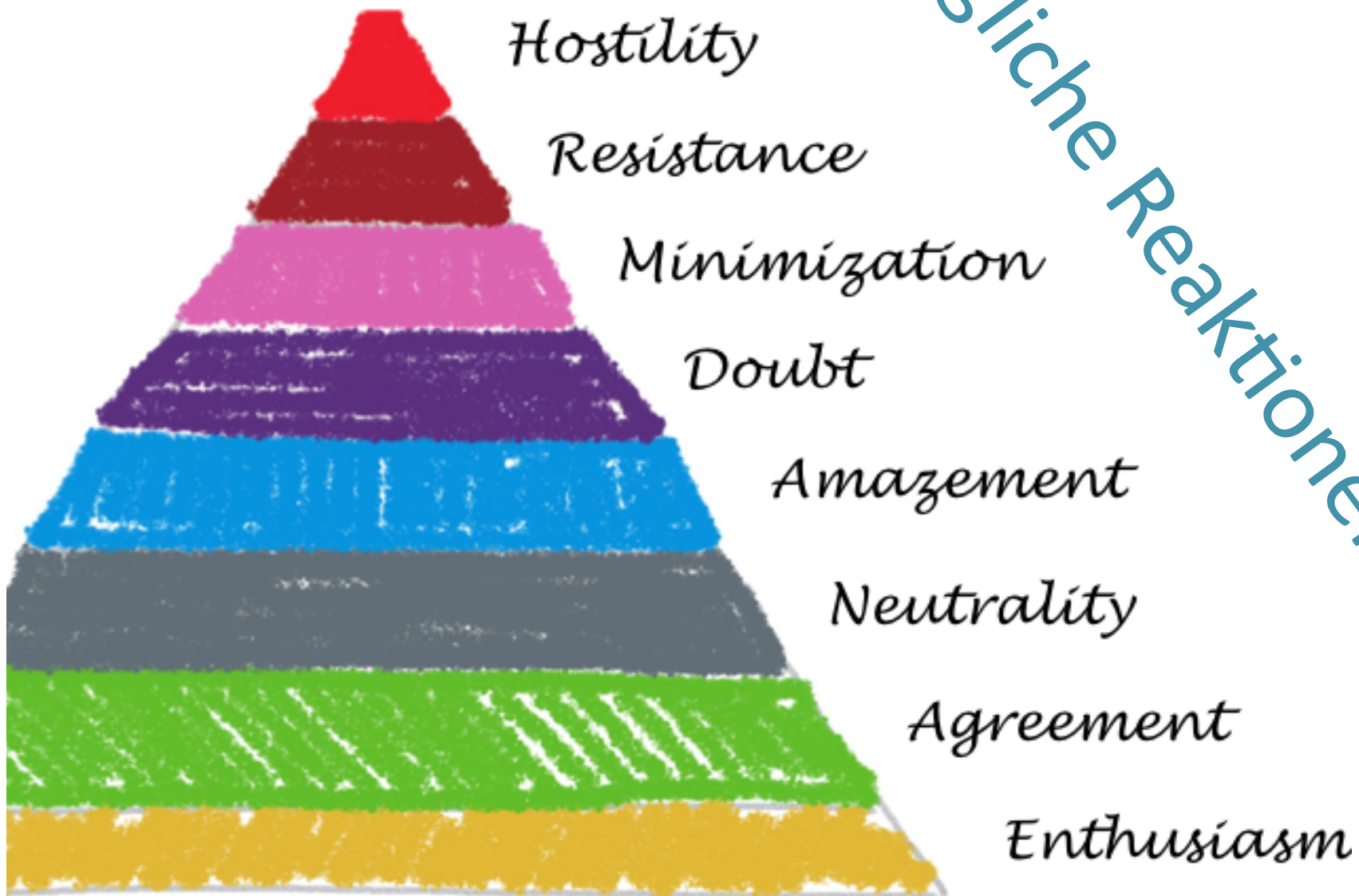


Die Relativitäts-Falle

Des Einen Problem ist
des Anderen Freund



mögliche Reaktionen...



profitieren vom Problem,
haben das Problem erschaffen,
greifen Sie und Ihre Vorschläge an

sind am Problem schuld,
stehen zu Ihnen in Konkurrenz,
greifen Ihre Kompetenz an

Hostility

Resistance

Minimization

Doubt

Amazement

leiden nicht am Problem,
leiden unter dessen Lösung
tragen (Teil-)Schuld am Problem

zweifeln Schlussfolgerungen an,
zweifeln Ihre Kompetenz an,
haben Angst vor Veränderung



Die Mikroskop-Falle

Wenn Sie NUR im Code suchen,
werden Sie NUR DORT Probleme finden...

im Code suchen ist richtig und wichtig, aber NUR dort suchen ist fatal!



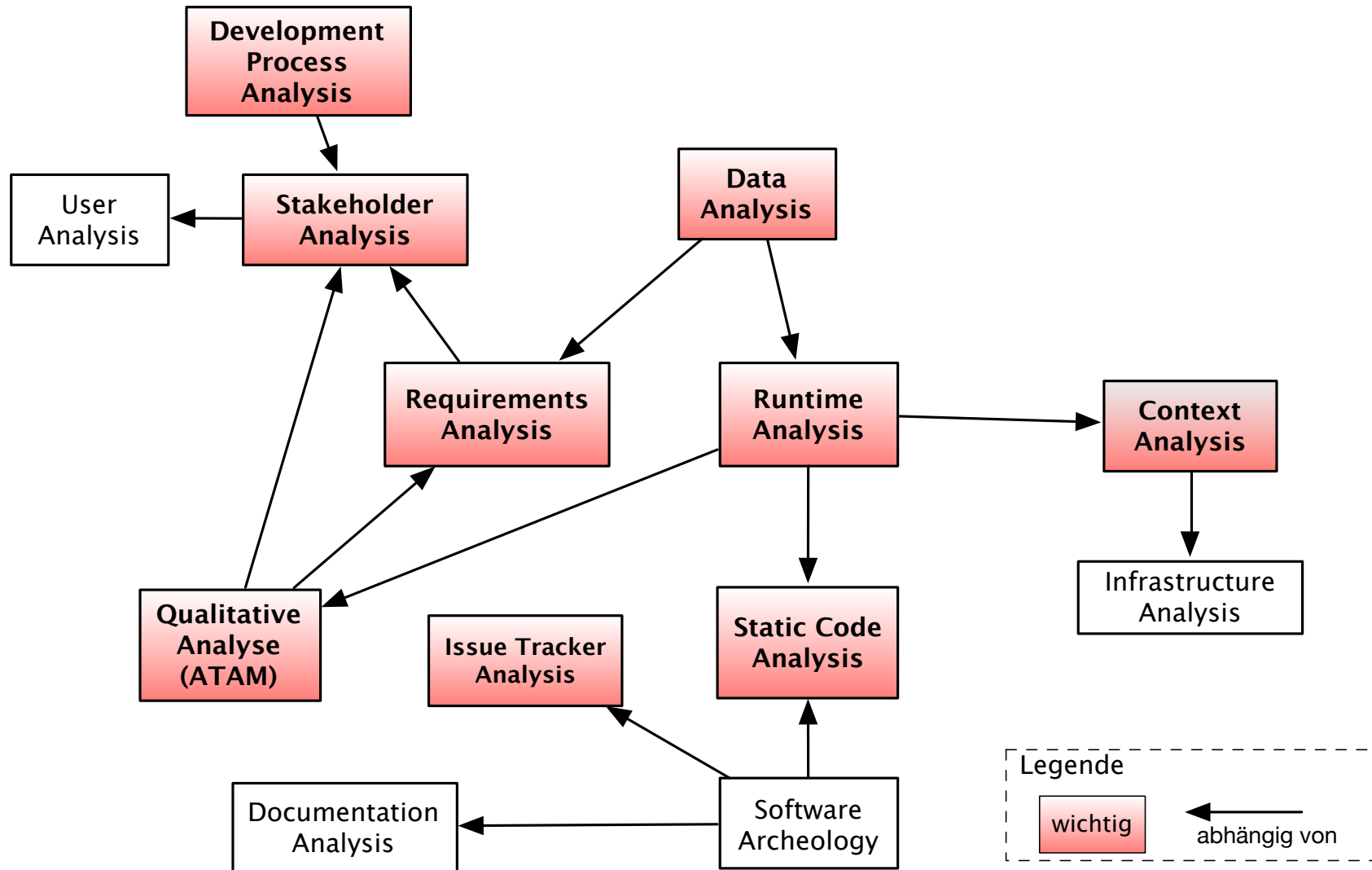


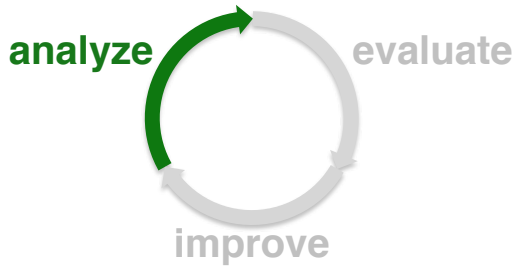
<https://www.flickr.com/photos/jonasb/24341322>



Bauch – Beine – Po:

Wo liegen die IT-Problemzonen?





Identifiziere Probleme

- Konkret / spezifisch
- Wiederholbar
- Was ist das Problem?
- {Wobei | Wo | Wann} tritt es auf?
- Reproduzierbar?
- Auswirkungen? (Verursachter Schaden?)



Probleme sammeln!!

ID	Beschreibung	Häufigkeit	Wert (min)	Wert (max)	Abhilfen
PD-1	Falsche Berechnung Artikelpreis bei Kombination aus Rabattkarte, Einzelkunde und yxz-Konfiguration	zZt 3-5x pro Woche	110€	550€	V-1 + V-2
PZ-1	Zu lange Warteschlangen (queues) in Entwicklungsprozess: waiting-time für neue Anforderungen >6W	30x / Woche	300€	15.000 €	V-3
C-1	Zeit zur Identifikation + Behebung von Laufzeitfehlern zu lang (bis zu 5 T bei kritischen Fehlern)	1x / Woche			V-4 V-3
C-2	Zeit für kompletten Test der Anwendung > 10 T	4 x / Jahr			V-4

Werkzeuge:

- (gut) Issue-Tracker (Voraussetzung: stabile URL's)
- (ok) Excel, Karteikarten, Flipchart
- (schlecht) Kopf



Stakeholderanalyse

Welche Personen
oder Rollen?

- Stakeholder kennen Probleme
 - nehmen Probleme aus ihrer Perspektive wahr (subjektiv)
 - nennen oftmals **Symptome**, keine Ursachen
 - äußern **Vermutungen**
- Vorsicht:
 - Gewöhnungseffekt
 - Angst vor Änderung

Anwender, Entwickler, Support,
Fachleute, BackOffice, Architekten,
Betreiber, Auftraggeber, Tester, Admins,
Projekt-/Linienverantwortliche,
Controller, CEO, COO, CFO ...

Stakeholder-Interviews (1)

- Breitensuche: Unterschiedliche Stakeholder
 - Mindestens: Anwender, Fachexperten, Entwickler, Support-Mitarbeiter, Tester, Betreiber, Manager
- Ungestörtes Umfeld
 - insbesondere: ohne Vorgesetzte
 - garantierte Vertraulichkeit
- Keine unhaltbaren Versprechungen geben!!



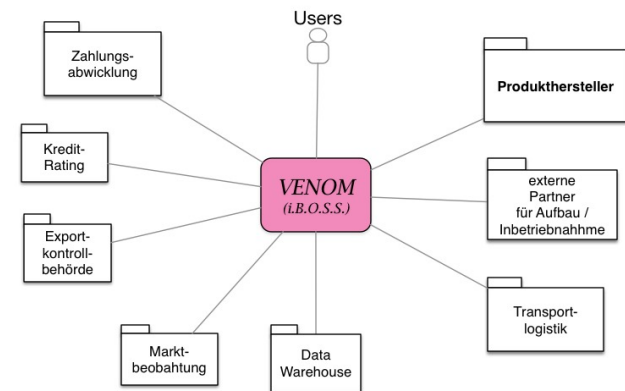
Stakeholder-Interviews (2)

- Vorbereitung: individueller Vorab-Fragebogen
 - Zeigt Hochachtung vor Gesprächspartner
 - Siehe: <http://aim42.github.io/#Stakeholder-Interview>
- Fragen nach:
 - Problemen + Lösungsmöglichkeiten bezüglich
 - System,
 - Prozesse,
 - Organisation



Kontextanalyse (1)

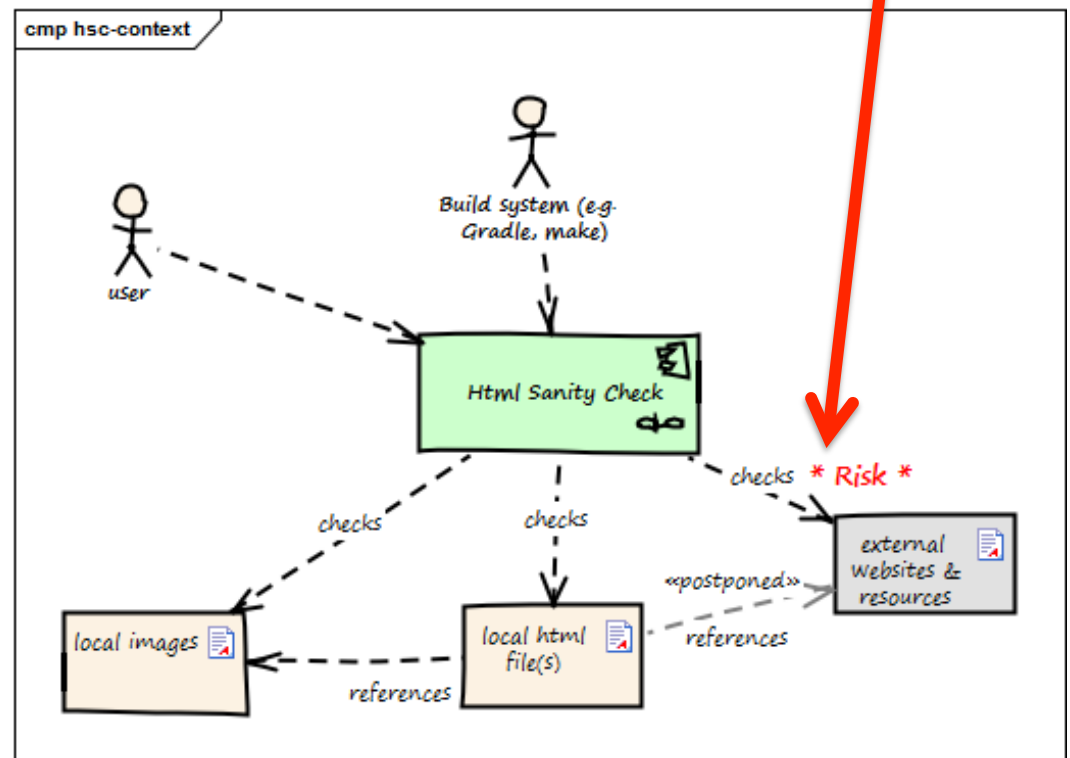
- **Fachlicher Kontext**
 - System-Scope
 - Externe Schnittstellen
 - Daten-Eingang
 - Daten-Ausgang
 - UI
 - Events
- **Technischer Kontext**
 - Kanäle
 - Protokolle



Kontextanalyse (2)

Mögliche Risiken im Kontext:

- Abhängigkeiten bzgl. Qualitätszielen
 - Verfügbarkeit, Robustheit,
 - Sicherheit
 - Kosten
 - Performance
- Fehlende Schnittstellen



Kontextanalyse (3)

Mögliche Risiken im Kontext:

- Volatile Nachbarsysteme

Risiko / Problem bzgl.
Stabilität der
Schnittstelle



„Improve Analyzability“

Verbessern Sie die Möglichkeiten,
Probleme zu finden!

- zur „Root-Cause-Analyse“
- zum Beweis von Vermutungen
- Instrumentierung, z.B.
 - (fachliches/technisches) Logging
 - Tracking von Qualitätseigenschaften

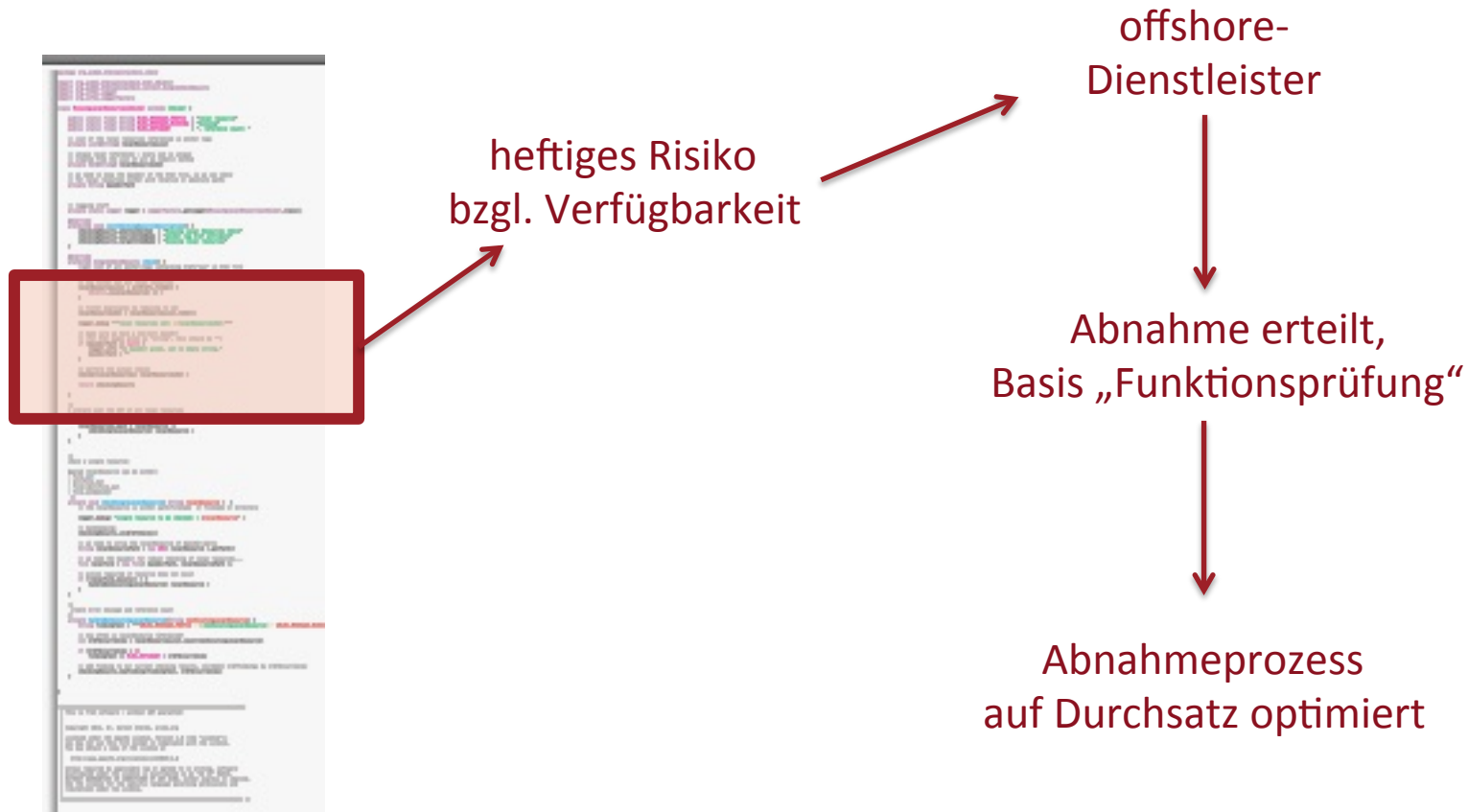


Prozessanalyse

- Anforderungsprozesse
 - erheben, klären, managen
- Entwicklungs- /Entwurfsprozesse
 - Architektur, Implementierung, Dokumentation
- Betrieb
 - Deployment, Rollout, Administration, Monitoring
- Management
 - Team- und Taskmanagement, Risikomanagement



warum, warum, warum...



Statische Codeanalyse

- Kopplung
- Komplexität
- Kohäsion (inhaltlicher Zusammenhang)
- Konsistenz (identische Probleme ähnlich gelöst)
- Duplizierter Code
- Verletzung von Idiomen (Style-Checking)



Korrelierte Codeanalyse

Abgleich unterschiedlicher
Beobachtungen/Messungen

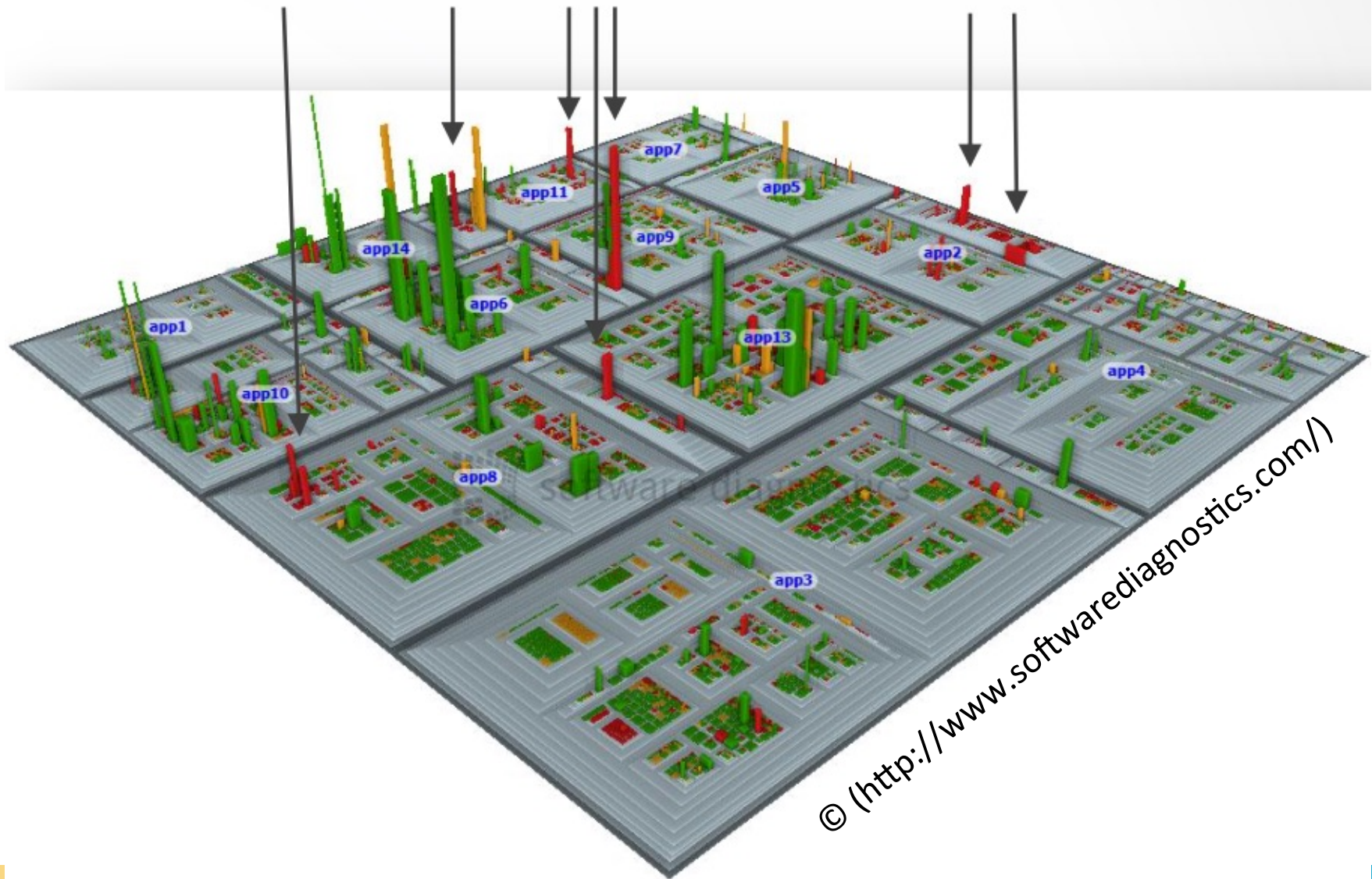
Beispiele:

- Fehler pro Komponente / Subsystem
- Benötigte Zeit pro Bugfix pro Komponente
- Benötigter Aufwand pro Feature pro Komponente



Korrelierte Codeanalyse (2)

Risky situation: Very complex code and only a single developer knows it. Costly effort-bombs will explode when the knowledge-having developer leaves the team.



Datenanalyse (1)

- Struktur  Struktur / Typen ungeeignet für das Problem
- Typen
- Zugriffe
– Read / write  Wonach ist Persistenz optimiert, read oder write?
- Volumen
– auch von Query-Results + Indizes  Haben wir besonders viel / groß von etwas?
- Inhalte
– Korrektheit  Haben wir falsche Daten?
– Schutzbedarf
- Durchsatz -> Laufzeitanalyse

Laufzeitanalyse

- Debugger
 - Logfile-Analyse
 - Performance-Analyse, Profiling
 - Ressourcenverbrauch zur Laufzeit
- Analyse von Benutzerverhalten
 - Analyse fachlicher Abläufe



Software-Archäologie

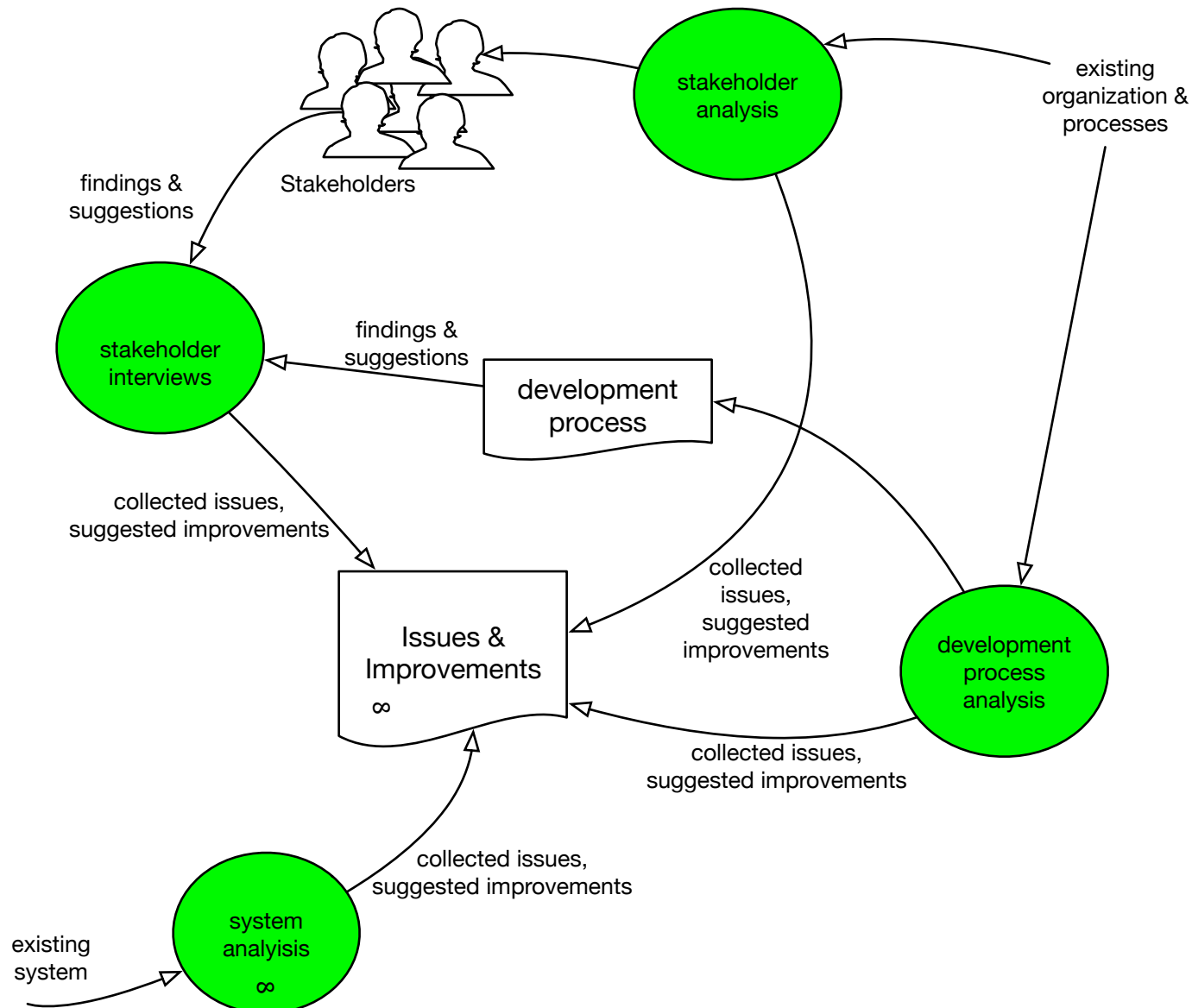
Historie von Code untersuchen

– benötigt Versionsverwaltung

- Gründe für Entscheidungen suchen
 - etwa: für „seltsamen Code“
- Zeitpunkte von Verschlechterungen erkennen



Fazit: Analyse in Vogelperspektive



Fazit (2)

Führe

- Problem-Liste und
- Improvement-Backlog



Fazit (3)

Vorsicht: Mikroskop-Falle



Fazit (4)

Vorsicht: neue Feinde...





Dr. Gernot Starke

gs@gernotstarke.de

<http://gernotstarke.de>

<http://innoq.com>

Projekt



Praktisch eingesetzt...

- Automotive:
 - „Multimedia-Framework“
 - Rail-Service „Infrastruktur“
 - Mobilfunk
 - „Billing“
 - Airport-Operations „Luggage Handling“
 - Systemsoftware für Maschinenbau / Lebensmittelindustrie
- ▶ 2014:
 - Logistik-Konzern (Audit Online-Sales System, >1MLOC)
 - ERP-Hersteller (Audit und Rebuild Kernsystem, 2 MLOC)
 - Finanzdienstleister, Modernisierung Core, 500kLOC



Website



The screenshot shows the homepage of the aim42 website. At the top, there is a dark header with the text "software architecture improvement" and a navigation menu with links: HOME, APPLY, READ, FAQ, and CONTRIBUTE. Below the header is a light gray banner with the aim42 logo (a blue arrow pointing up and to the right with "aim" and "42" in a red circle) and the tagline "software evolution and optimization - done right". A central blue box contains the text "aim42 - systematic software evolution and improvement" followed by a list of bullet points: "Optimize your software and reduce maintenance cost", "Control risks, issues and technical debt", "Organizes patterns and practices in three phases", "Free and open-source", and "Grounded in practice, proven approaches, backed by serious research". Below this, there are three columns: "analyze", "evaluate", and "improve", each with a brief description and a "more ..." link. At the bottom, there is a "Download Whitepaper" button and a circular diagram with the words "analyze", "evaluate", and "improve" around it. A "news" section on the right lists recent events: "Whitepaper available", "April 2014: aim42 experience", "Feb. 2014: Online version of method guide", and "Feb. 2014: Presentation on a OOP conference (Stefan Tilke)".

software architecture improvement

HOME APPLY READ FAQ CONTRIBUTE

aim42 software evolution and optimization - done right

aim42 - systematic software evolution and improvement

- Optimize your software and reduce maintenance cost
- Control risks, issues and technical debt
- Organizes patterns and practices in three phases
- Free and open-source
- Grounded in practice, proven approaches, backed by serious research

➤ See the online aim42 method guide for details

analyze

ms, risks, deficiencies and
ebt within your system and your
nt process.

root-causes of problems.

evaluate

estimate „value“ of problems, issues and
their remedies, prioritize.

more ...

improve

systematically improve code an
reduce technical debt, remove
optimize.

more ...

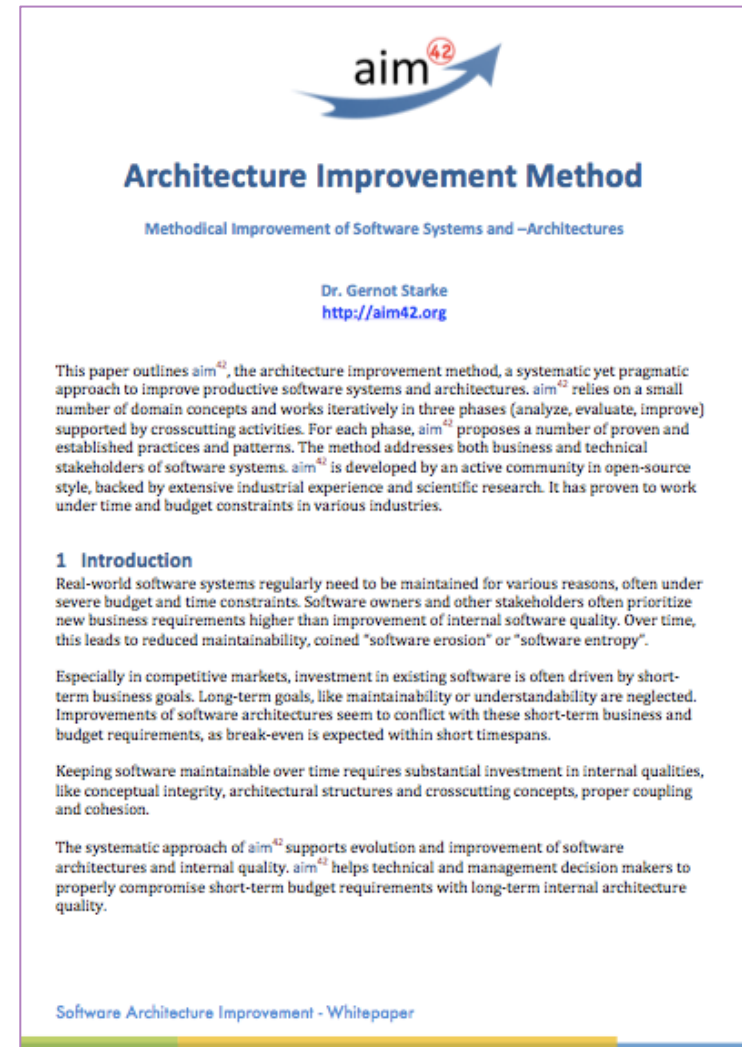
Download Whitepaper

analyze evaluate improve

news

- Whitepaper available
- April 2014: aim42 experience
- Feb. 2014: Online version of method guide
- Feb. 2014: Presentation on a OOP conference (Stefan Tilke)

Whitepaper



The image shows the cover of the "Architecture Improvement Method" whitepaper. At the top right is the aim42 logo. The title "Architecture Improvement Method" is in a large, bold, blue font. Below it, the subtitle "Methodical Improvement of Software Systems and –Architectures" is in a smaller, blue font. The author's name, "Dr. Gernot Starke", and his website, "http://aim42.org", are listed. The main text describes the aim42 method as a systematic yet pragmatic approach to improve productive software systems and architectures. It mentions that the method is based on a small number of domain concepts and works iteratively in three phases: analyze, evaluate, and improve, supported by crosscutting activities. The paper is developed by an active community in open-source style, backed by extensive industrial experience and scientific research. The "1 Introduction" section starts by stating that real-world software systems regularly need to be maintained for various reasons, often under severe budget and time constraints. It notes that software owners often prioritize new business requirements higher than improvement of internal software quality, which leads to reduced maintainability, coined "software erosion" or "software entropy". It also mentions that keeping software maintainable over time requires substantial investment in internal qualities, like conceptual integrity, architectural structures and crosscutting concepts, proper coupling and cohesion. Finally, it states that the systematic approach of aim42 supports evolution and improvement of software architectures and internal quality, helping technical and management decision makers to properly compromise short-term budget requirements with long-term internal architecture quality. At the bottom, it says "Software Architecture Improvement - Whitepaper".

aim42

Architecture Improvement Method

Methodical Improvement of Software Systems and –Architectures

Dr. Gernot Starke
<http://aim42.org>

This paper outlines aim⁴², the architecture improvement method, a systematic yet pragmatic approach to improve productive software systems and architectures. aim⁴² relies on a small number of domain concepts and works iteratively in three phases (analyze, evaluate, improve) supported by crosscutting activities. For each phase, aim⁴² proposes a number of proven and established practices and patterns. The method addresses both business and technical stakeholders of software systems. aim⁴² is developed by an active community in open-source style, backed by extensive industrial experience and scientific research. It has proven to work under time and budget constraints in various industries.

1 Introduction

Real-world software systems regularly need to be maintained for various reasons, often under severe budget and time constraints. Software owners and other stakeholders often prioritize new business requirements higher than improvement of internal software quality. Over time, this leads to reduced maintainability, coined "software erosion" or "software entropy".

Especially in competitive markets, investment in existing software is often driven by short-term business goals. Long-term goals, like maintainability or understandability are neglected. Improvements of software architectures seem to conflict with these short-term business and budget requirements, as break-even is expected within short timespans.

Keeping software maintainable over time requires substantial investment in internal qualities, like conceptual integrity, architectural structures and crosscutting concepts, proper coupling and cohesion.

The systematic approach of aim⁴² supports evolution and improvement of software architectures and internal quality. aim⁴² helps technical and management decision makers to properly compromise short-term budget requirements with long-term internal architecture quality.

Software Architecture Improvement - Whitepaper

Code: github

public repo for aim42, especially the "aim42 method guide" — Edit

170 commits

2 branches

0 releases

6 contributors

Your recently pushed branches:

atam (6 minutes ago)

Compare & pull request

branch: master aim42 / +

Merge branch 'atam' ...



gernotstarke authored 11 hours ago

latest commit b2383582b5

graphics	added image for "view-based-understanding"	6 days ago
guide	fixing list	21 hours ago
whitepaper	improved layout of whitepaper	3 days ago
.gitignore	Merge branch 'atam'	11 hours ago
.travis.yml	Added encrypted key for repository aim42/aim42	2 months ago
readme.md	Improved image links in Readme.	2 months ago

readme.md



Architecture Improvement Method

Code

Issues 25

Pull Requests 0

Wiki

Pulse

Graphs

Network

Settings

HTTPS clone URL

https://github.com/

You can clone with [HTTPS](#), [SSH](#), or [Subversion](#).

Clone in Desktop

Download ZIP

Issues: viele...

Assigned to you9

Created by you15

Mentioning you1

No milestone selected

Labels

Fundamental4

enhancement6

pattern10

practice10

question3

website1

bug0

duplicate0

wontfix0

Manage labels

New label

New label name

Close

Label

Assignee

Milestone

Convert OOP-2014 talk on business-relation of architecture into aim42 pattern

enhancementpattern

Opened by gernotstarke 6 days ago

#53

Write pattern "shepharding the architecture"

patternpractice

Opened by gernotstarke 6 days ago

#52

Estimate-in-Intervalls (new practice for evaluation phase)

practice

Opened by gernotstarke 8 days ago 3 comments

#51

Agree on a name (symbol) for the "system under improvement"

Fundamentalquestion

Opened by MichaelMahlberg 24 days ago 1 comment

#50

expand "code review" or "code reading" practice

practice

Opened by gernotstarke a month ago

#48

enhance build to generate pdf version

enhancement

Opened by gernotstarke a month ago

#46

Cross-Check our patterns/practices with "OORP" from Nierstrasz et al

enhancementFundamental

Opened by gernotstarke a month ago 1 comment

#44

Include "architectural feature" from SAFE-framework

enhancementpattern

Opened by gernotstarke a month ago

#43

Add pattern for better logging and runtime metrics under "Improve"

Opened by vanto 2 months ago

#38

What is technical debt? (Warning: Meta Question)

Fundamentalquestion

#36

Graph

Members

Contributors...

Members of the aim42 Network

aim42 created aim42 and everyone else forked it. This is the family tree.

- 
- aim42 / aim42**
-  **aheusingfeld / aim42**
 -  **BoronNitride / aim42**
 -  **ck-innoq / aim42**
 -  **feststelltaste / aim42**
 -  **gbeine / aim42**
 -  **httpPrincess / aim42**
 -  **MichaelMahlberg / aim42**
 -  **otigges / aim42**
 -  **rschimmack / aim42**
 -  **vanto / aim42**

Beiträge
willkommen!

Contributions welcome

- Method guide: **<http://aim42.github.io>**
 - Source: <https://github.com/aim42/aim42>
- <https://github.com/aim42/aim42/issues>
- Twitter: @arc_improve42
- Mailing list: aim42@lists.innoq.com

