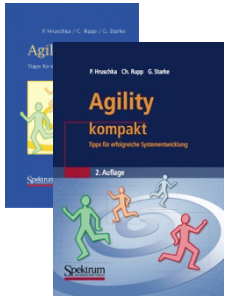
Know Your Enemies

Probleme in Software finden – aber richtig

Dr. Gernot Starke



Dr. Gernot Starke

innoQ Fellow

- ***Softwarearchitekturen***

  Entwurf, Entwicklung, Management

  Evolution & Modernisierung

  Training

- ***Mentoring und Coaching***

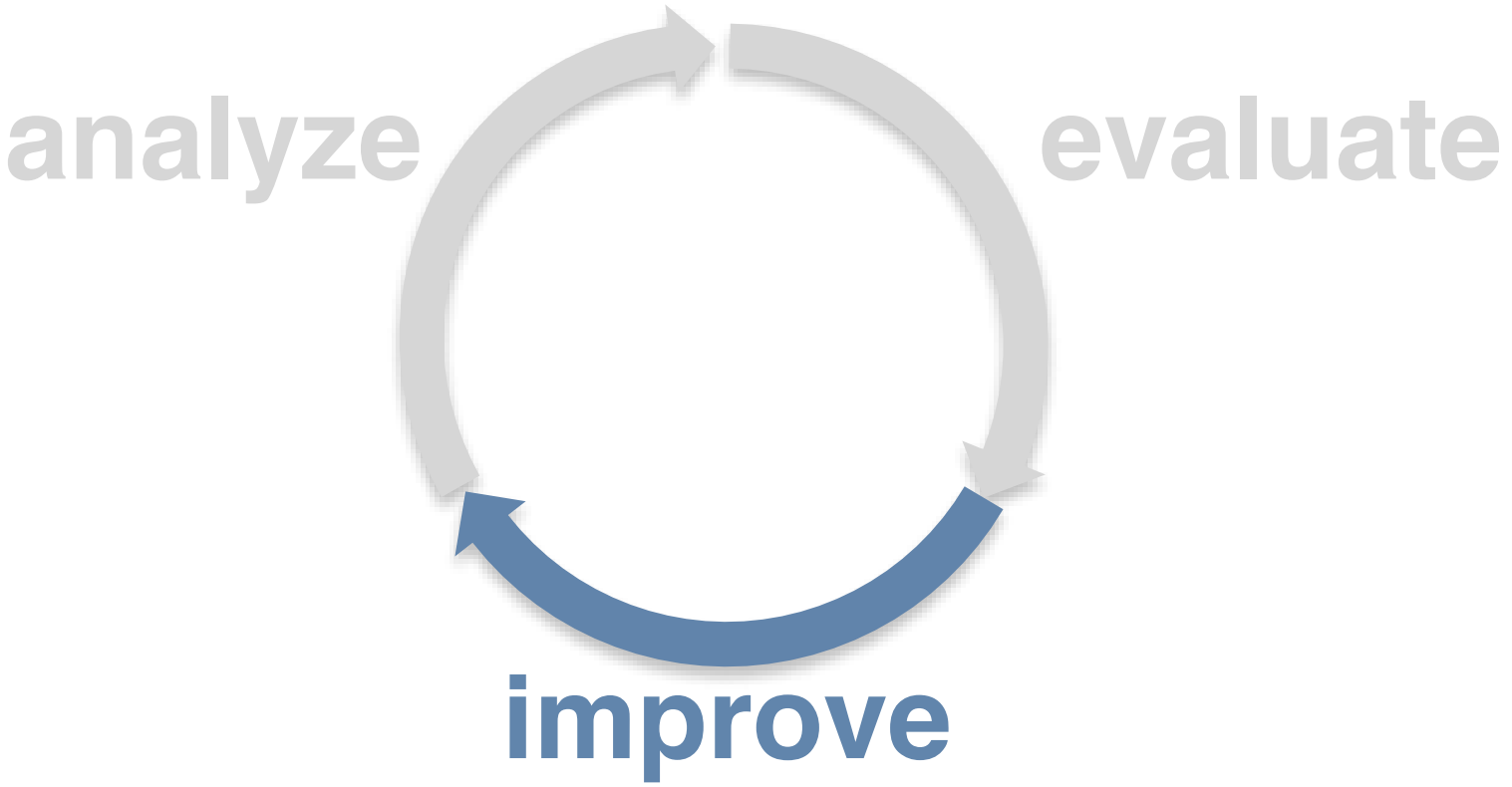
  Analyse und Optimierung von Entwicklungsprozessen

- ***Reviews, Audits, Retrospektiven***

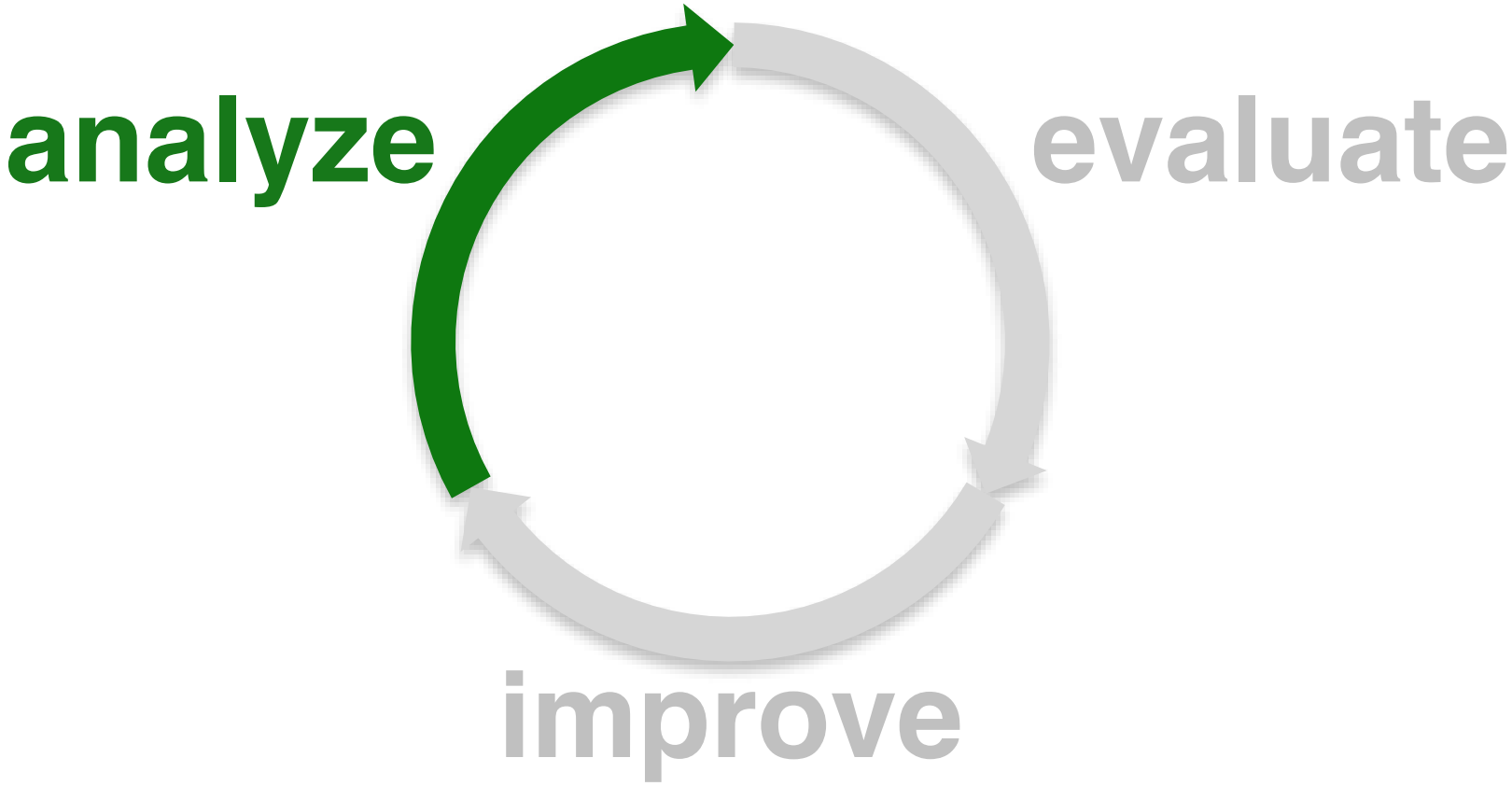
+49 177 – 728 2570
gs@gernotstarke.de

www.arc42.de

im normalen Leben...



dieser Vortrag...



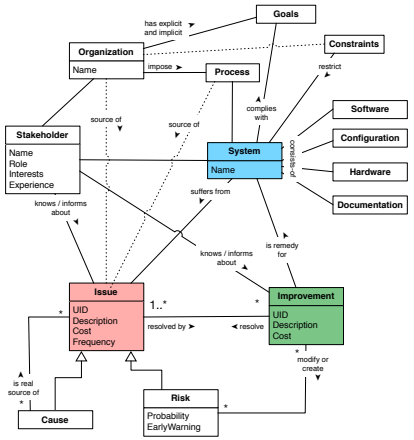
Gute Problemanalyse hilft bei...

- Priorisierung von Änderungen
- Balancieren kurz- / langfristiger Maßnahmen

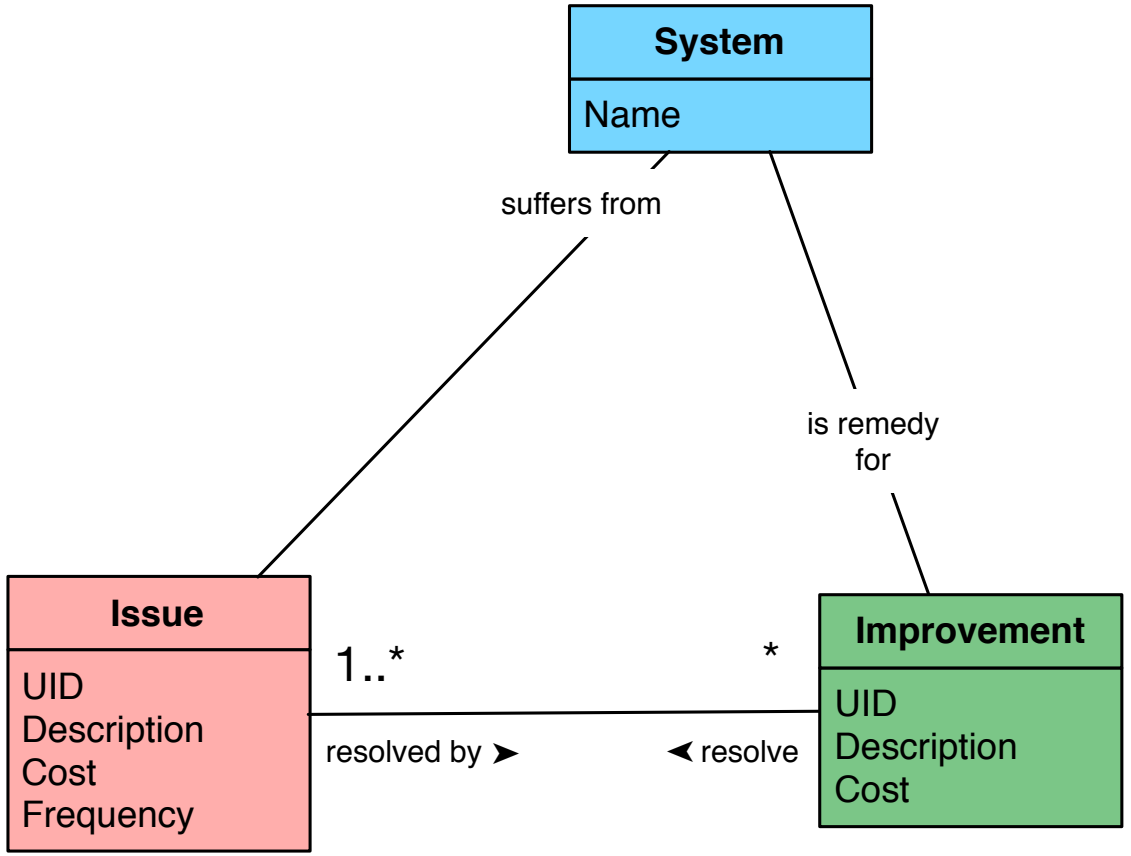
aim42

Systematisch verbessern (1)…

Trenne „Probleme“ und „Lösungsvorschläge“



aim42 domain model



warum diese Trennung?

- richtig:
  - **wichtige** Probleme lösen

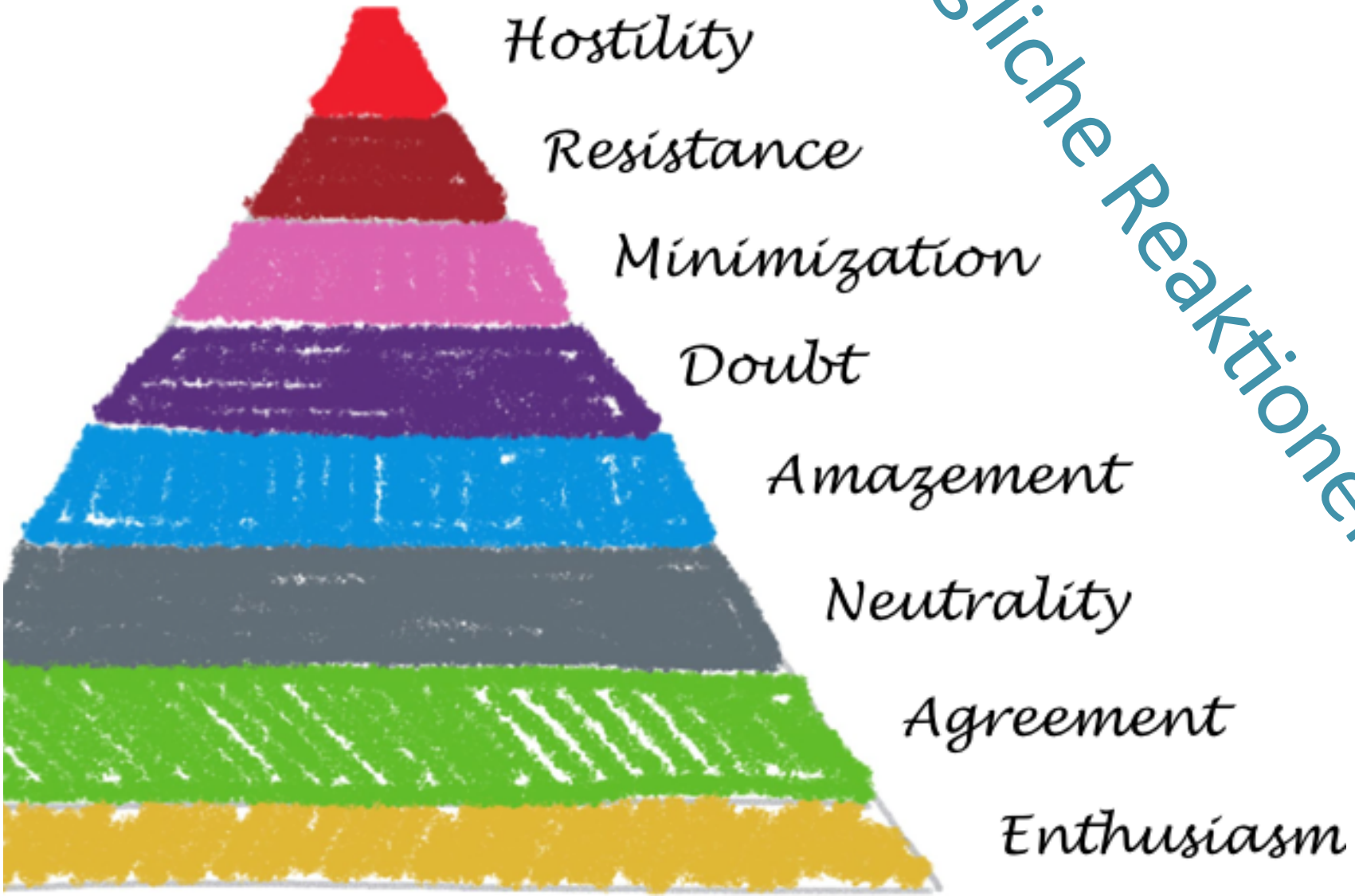
- schlechter:
  - einfache Probleme lösen

- 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

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...

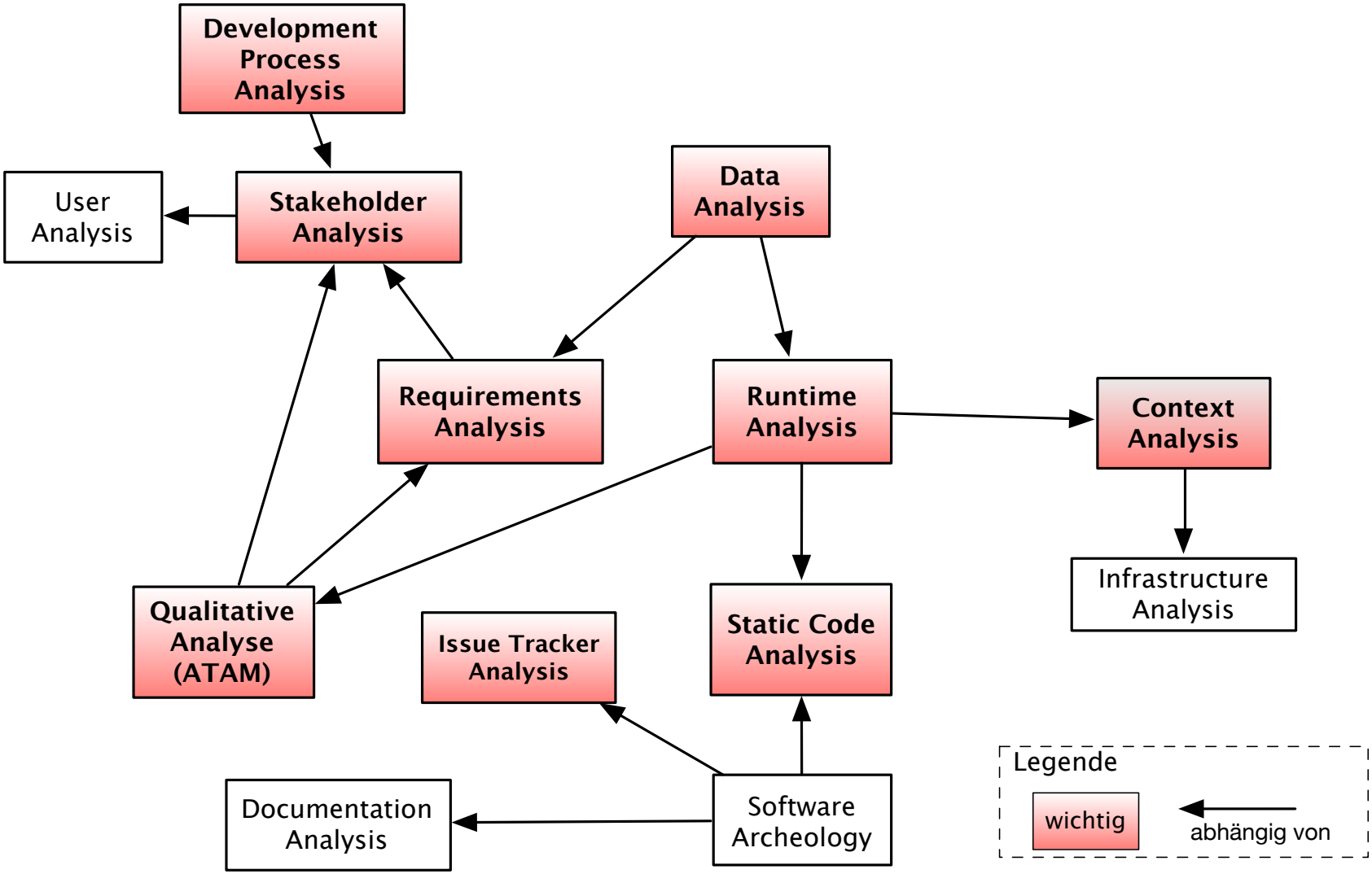
im Code suchen ist richtig und wichtig, aber NUR dort suchen ist fatal!

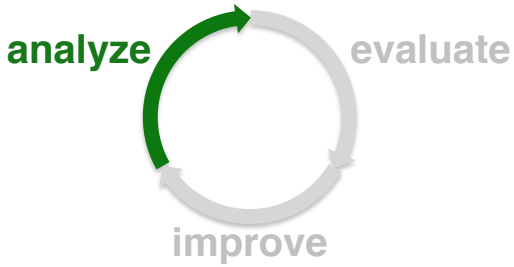

https://www.flickr.com/photos/jonasb/24341322

https://www.flickr.com/photos/emiliano-iko/5354414276

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 Entwicklungs-prozess: 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

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

- Vorsicht:
  - Gewöhnungseffekt
  - Angst vor Änderung

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!!

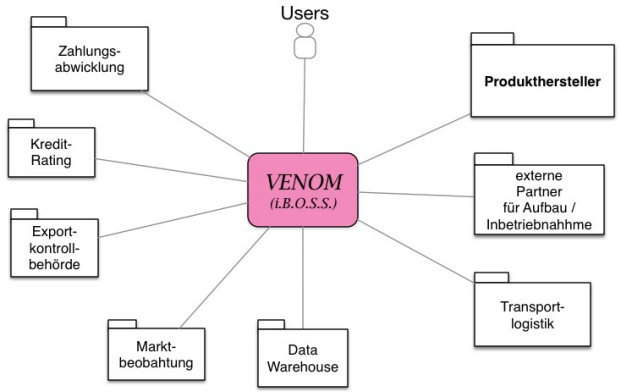
Stakeholder-Interviews (2)

- 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)

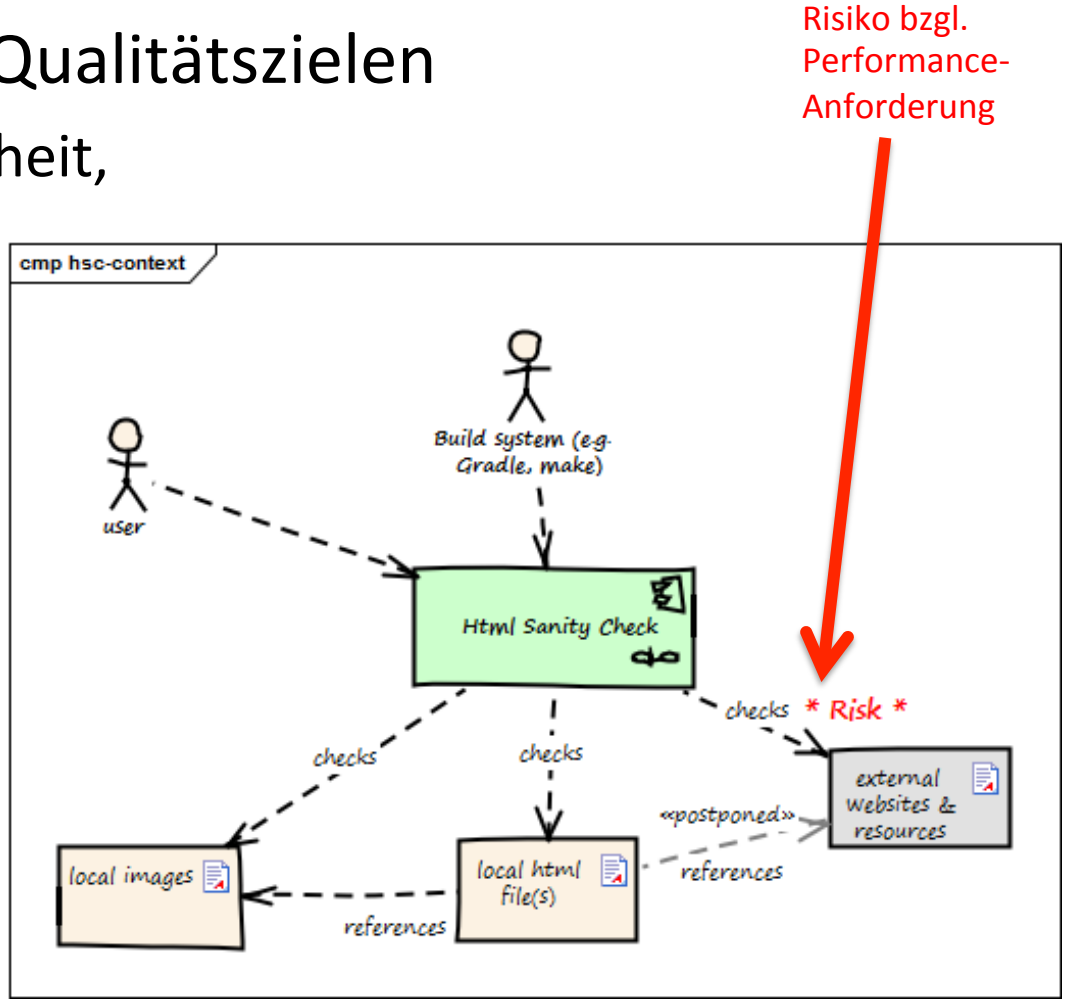
- Fachlicher Kontext
  - System-Scope
  - Externe Schnittstellen
    - Daten-Eingang
    - Daten-Ausgang
    - UI
    - Events
- Technischer Kontext
  - Kanäle
  - Protokolle



Kontextanalyse (2)

Mögliche Risiken im Kontext:

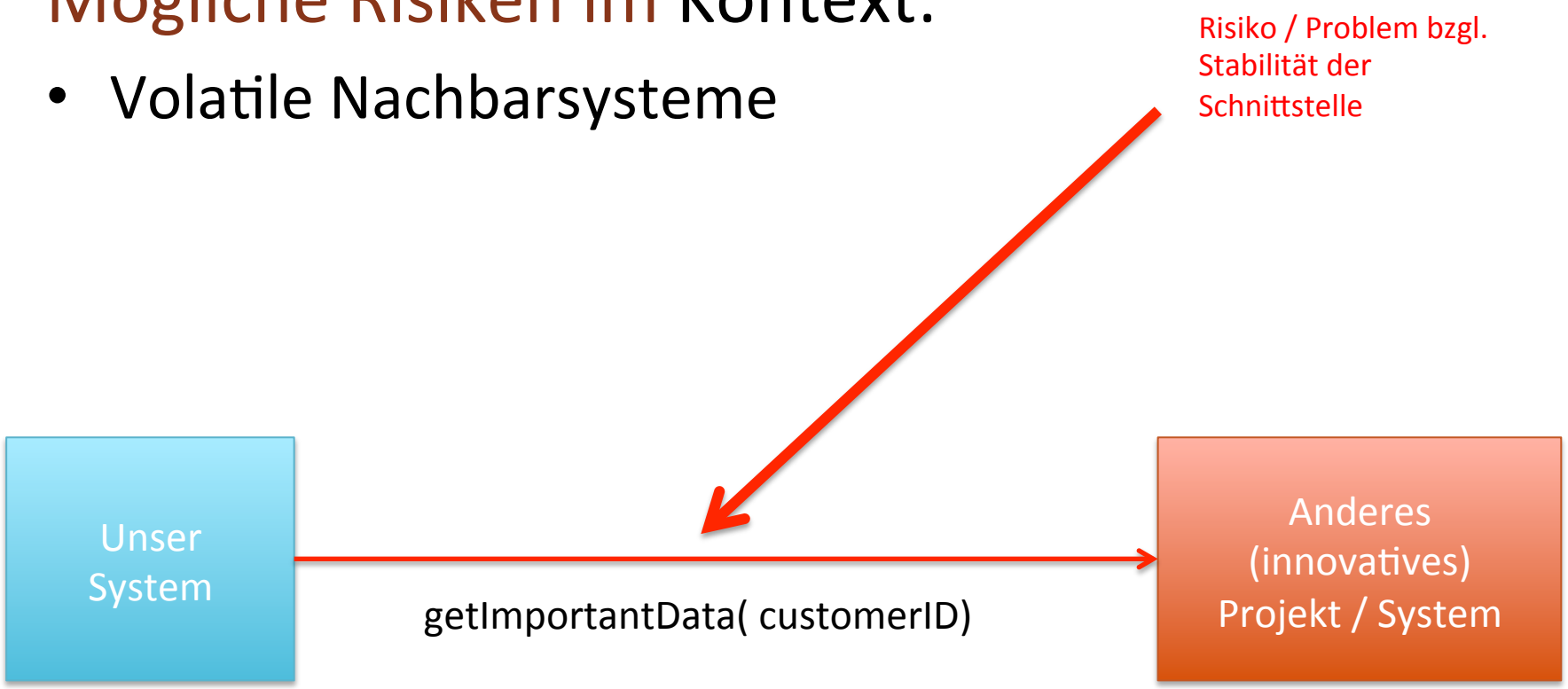
- Abhängigkeiten bzgl. Qualitätszielen
  - Verfügbarkeit, Robustheit,
  - Sicherheit
  - Kosten
  - Performance
- Fehlende Schnittstellen



Kontextanalyse (3)

Mögliche Risiken im Kontext:

- Volatile Nachbarsysteme



„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...

heftiges Risiko
bzgl. Verfügbarkeit

offshore-
Dienstleister

Abnahme erteilt,
Basis „Funktionsprüfung“

Abnahmeprozess
auf Durchsatz optimiert

Statische Codeanalyse

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

Korrelierte Codeanalyse

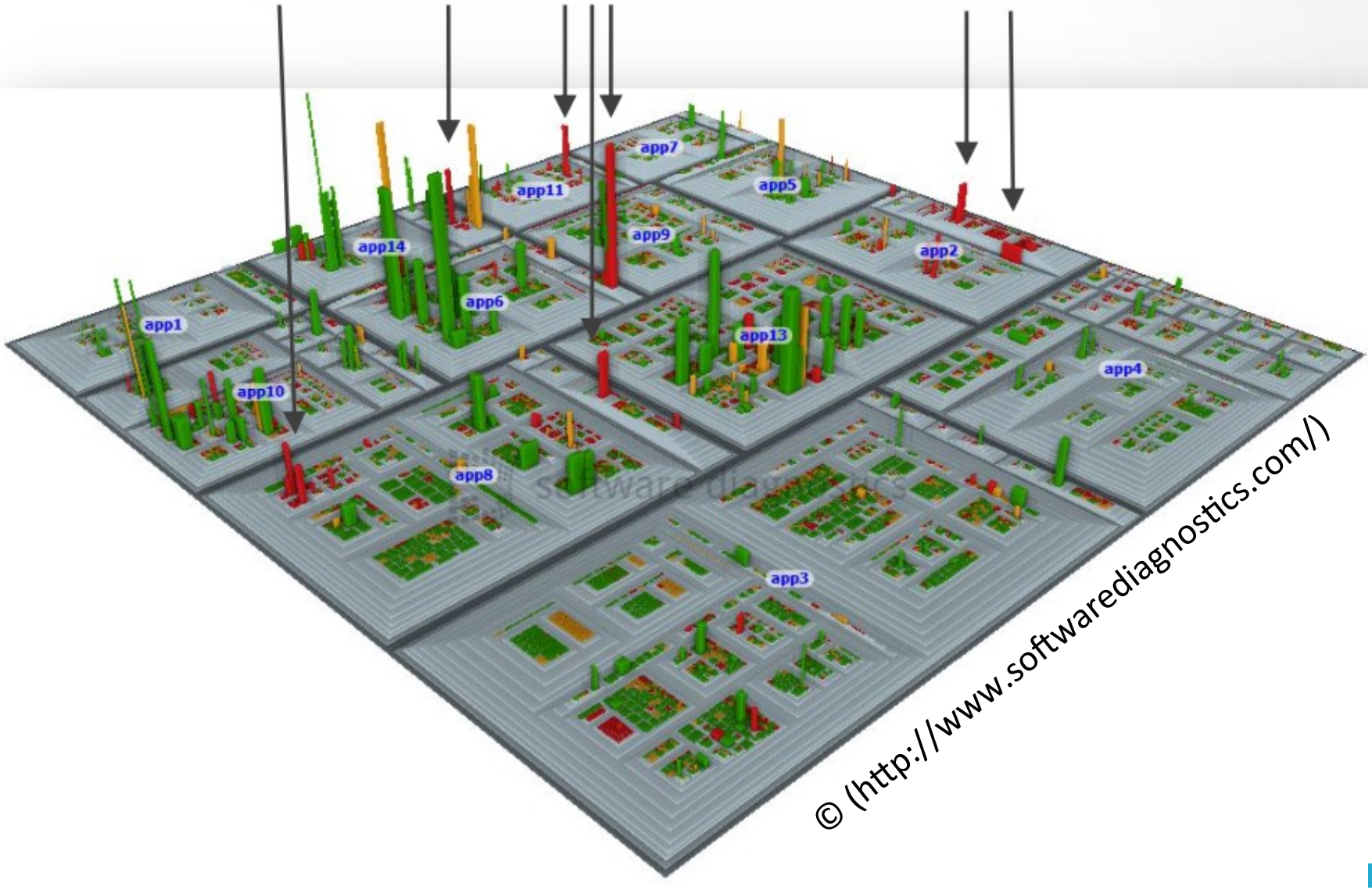
Abgleich unterschiedlicher Beobachtungen/Messungen

Beispiele:

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

Korrelierte Codeanalyse (2)

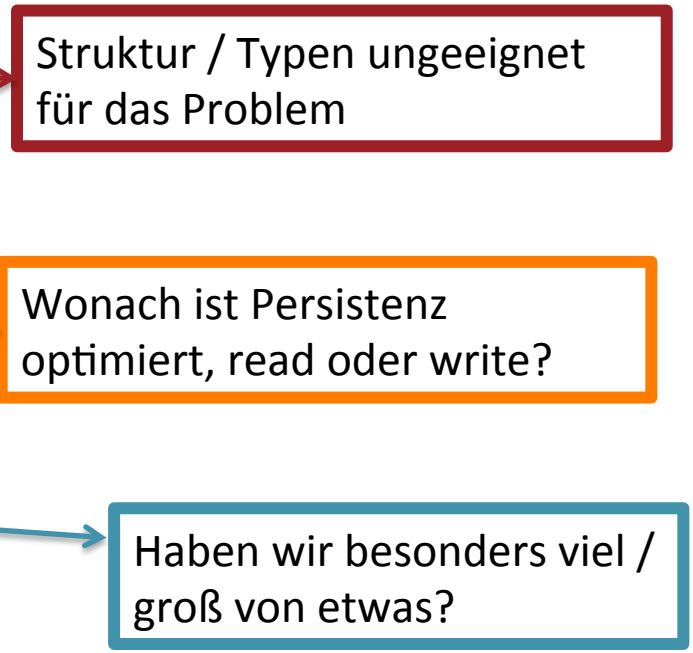
Risky situation: Very complex code and only a single developer knows it. Costly effort-bombs will explode when the knowledge-having developer leaves the team.



© (http://www.softwarediagnostics.com/)

Datenanalyse (1)

- Struktur
- Typen
- Zugriffe
  - Read / write
- Volumen
  - auch von Query-Results + Indizes
- Inhalte
  - Korrektheit
  - Schutzbedarf
- Durchsatz -> Laufzeitanalyse



Laufzeitanalyse

- Debugger
- Logfile-Analyse

- Performance-Analyse, Profiling
- Ressourcenverbrauch zur Laufzeit

- Analyse von Benutzerverhalten
- Analyse fachlicher Abläufe

Software-Archäologie

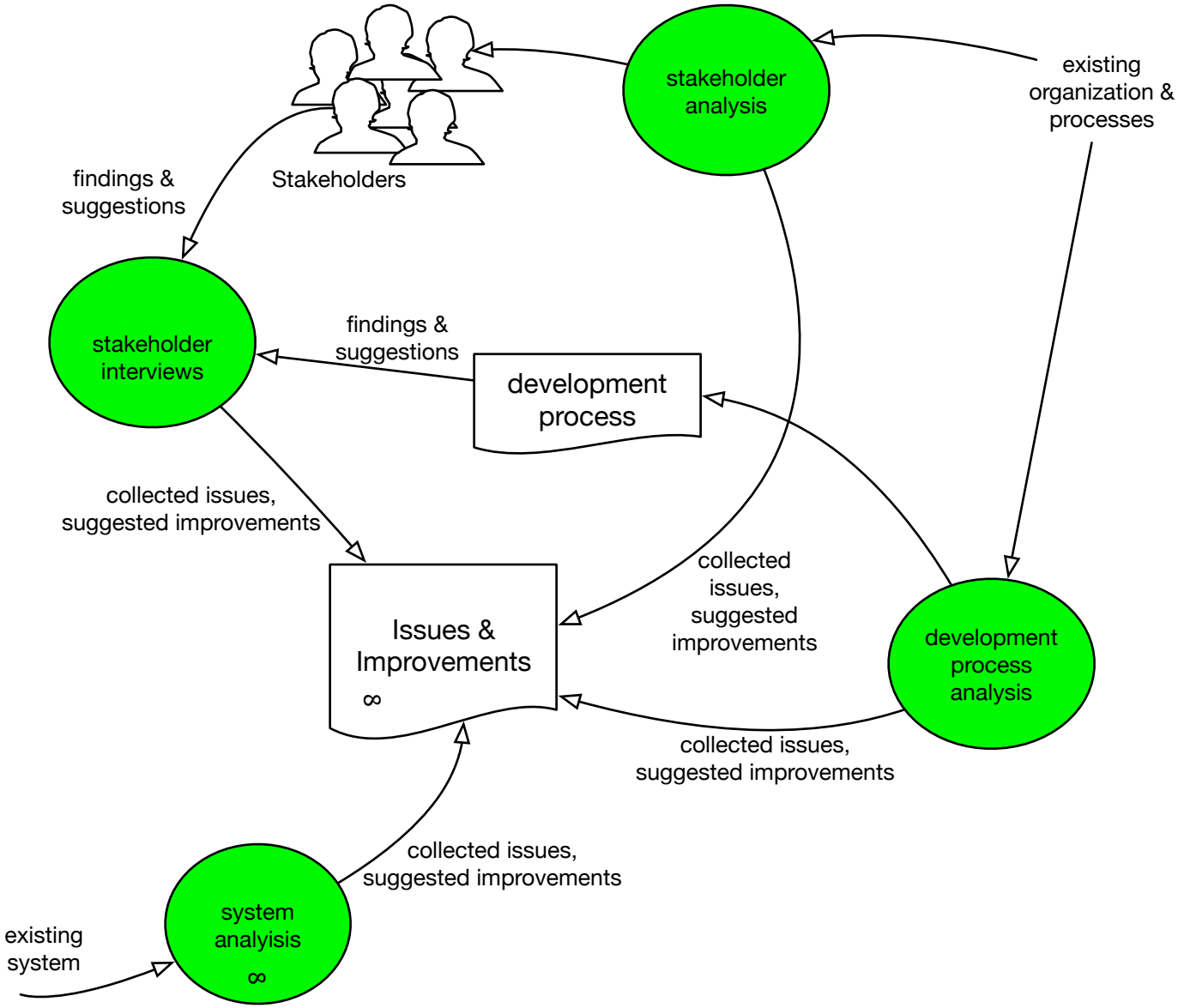
Historie von Code untersuchen

- 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

aim42

Fazit (3)

Vorsicht: Mikroskop-Falle

Fazit (4)

Vorsicht: neue Feinde...

Dr. Gernot Starke

gs@gernotstarke.de

http://gernotstarke.de
http://innoq.com

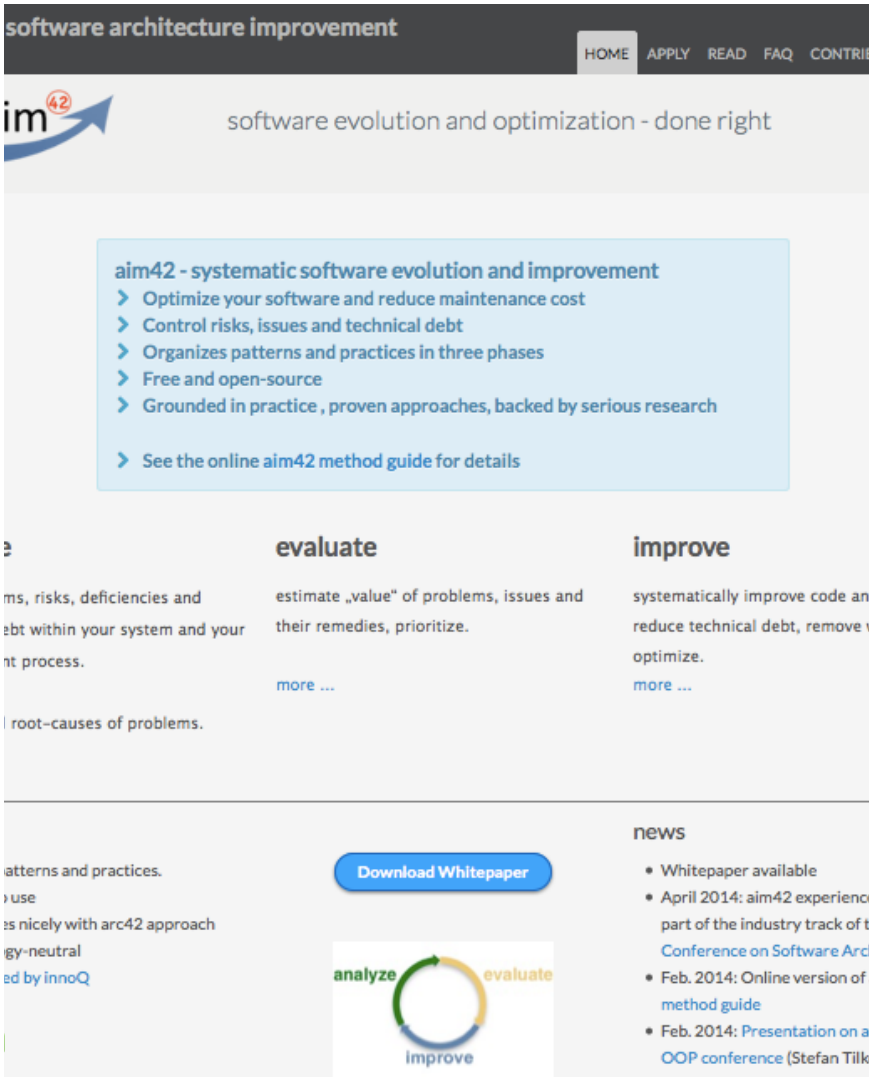
Projekt

Praktisch eingesetzt...

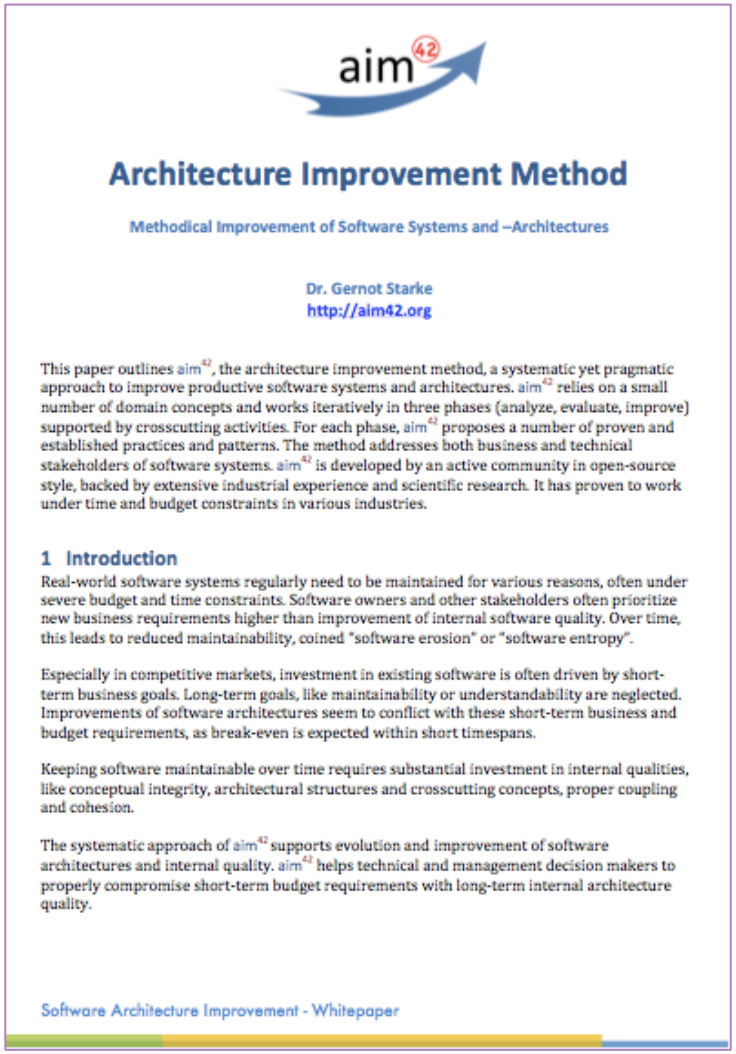
- Automotive: „Multimedia-Framework“
- Rail-Service „Infrastruktur“
- Mobilfunk „Billing“
- Airport-Operations „Luggage Handling“
- Systemsoftware für Maschinenbau / Lebensmittelindustrie

- 2014:
  - Logistik-Konzern (Audit Online-Sales System, >1MLOC)
  - ERP-Hersteller (Audit und Rebuild Kernsystem, 2 MLOC)
  - Finanzdienstleister, Modernisierung Core, 500kLOC

Website



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.

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

aim42

Architecture Improvement Method

Code
Issues 25
Pull Requests 0
Wiki
Pulse
Graphs
Network
Settings

HTTPS clone URL

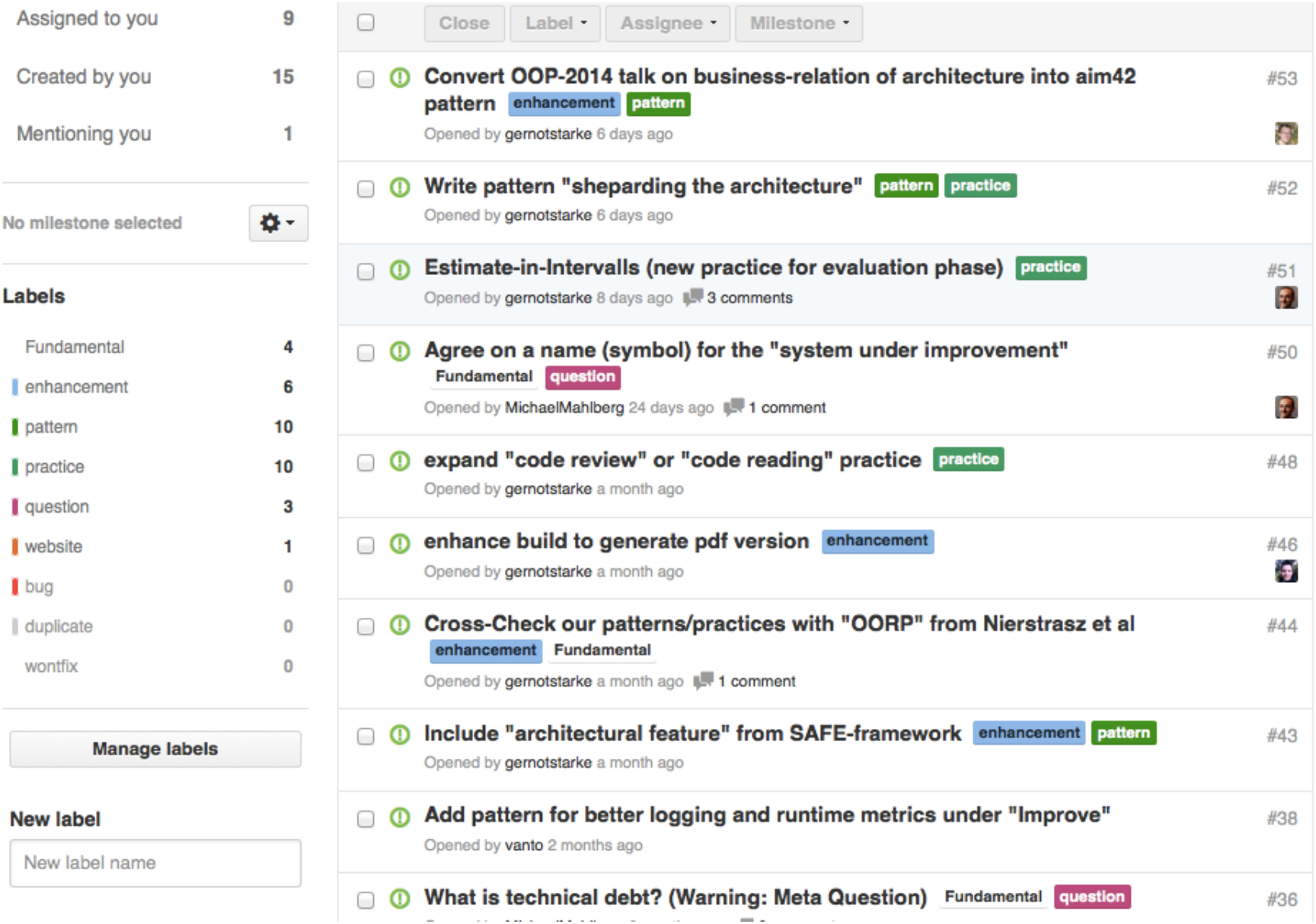
https://github.com/

You can clone with HTTPS, SSH, or Subversion.

Clone in Desktop

Download ZIP

Issues: viele...



Contributors...

Graph | Members

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

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