

Softwarecontrolling mittels Kennzahlen

Proseminar IT–Kennzahlen und Sofwaremetriken

Timo Besenreuther

21. Juni 2010

Überblick

Motivation

- Projektmanagement im Wandel
- Allgemeines zu Qualität und Software Controlling

Qualitätsmodelle

- McCall, ISO/IEC 9126
- Goal Question Metric
- Capgemini sd&m Kennzahlensystem
- ConQAT

Einsatz

- Ansätze zum Einsatz von Kennzahlen
- Kennzahlen im Entwicklungsprozess

Motivation

Projektmanagement im Wandel

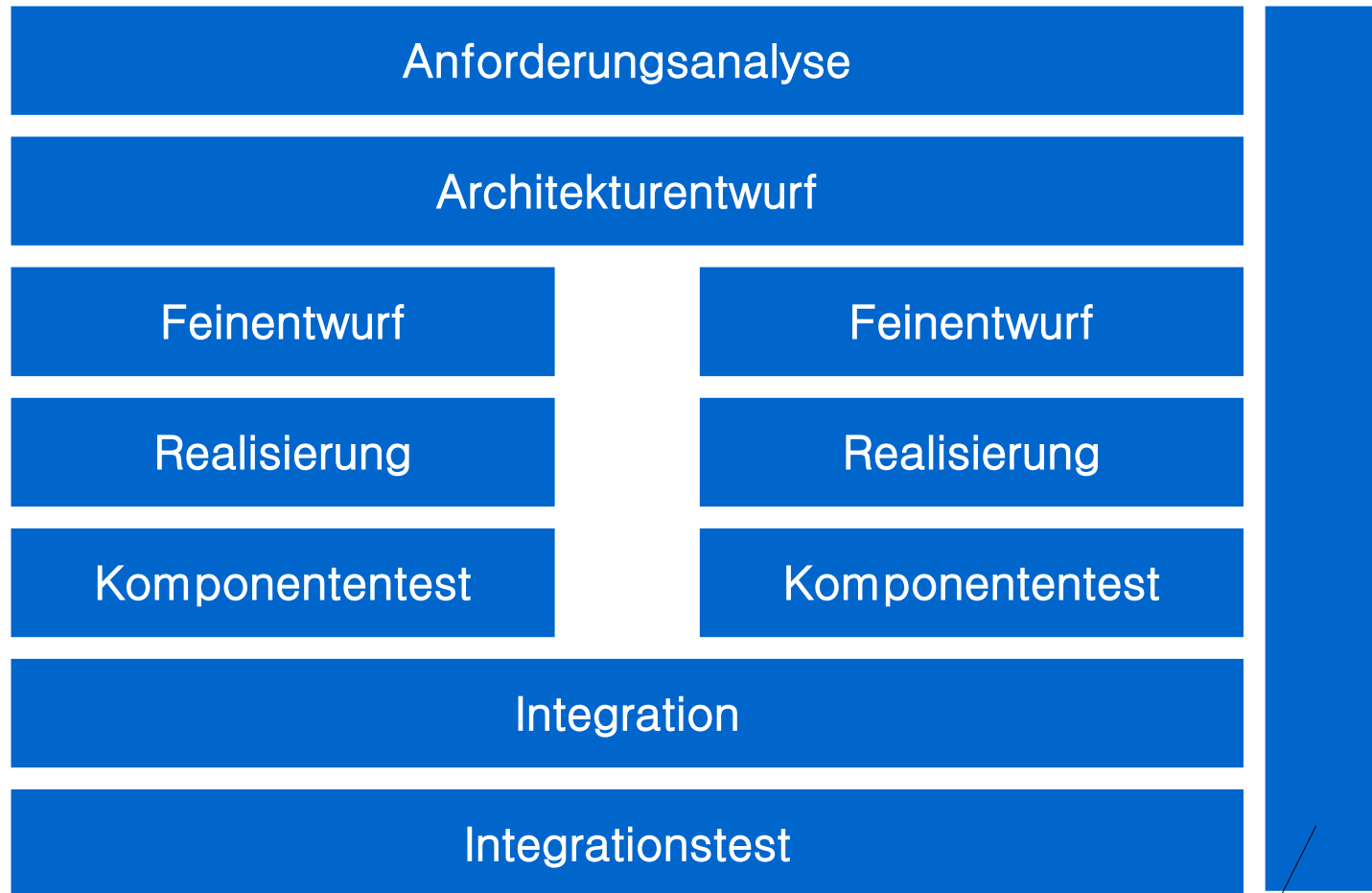
Klassisches Verständnis

- Einhaltung des Zeitplans
- Einhaltung des Budgetplans
- Erfüllung von Anforderungen (Funktionalität, Effizienz)

Weiterentwicklung der Software Branche

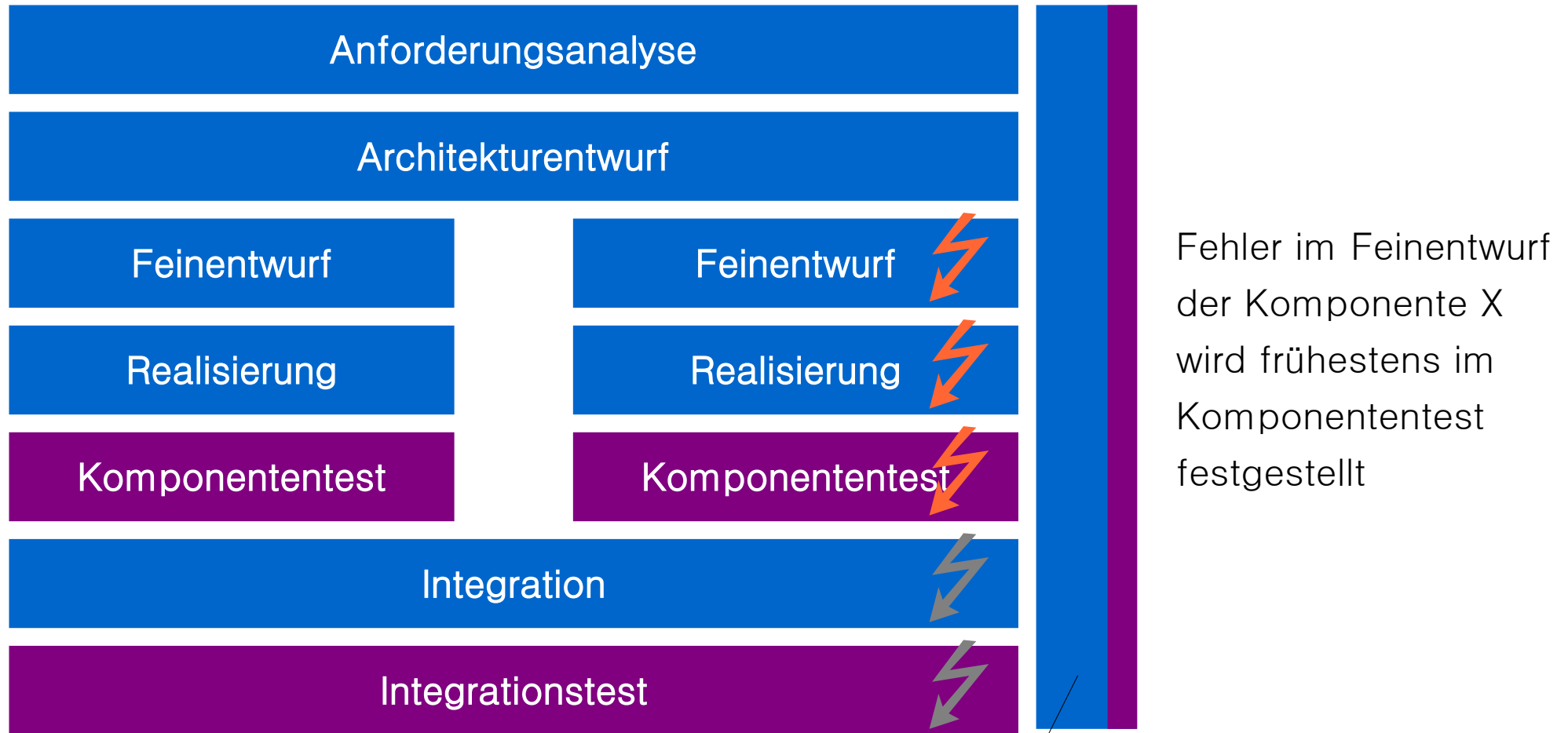
- Verteilte Systeme
 - Verstreute Teams
 - Spezialisierte Rollen
 - Kurze Release-Zyklen
 - Hoher Termin- und Budgetdruck
- => Softwareentwicklung muss gut geplant und verlässlich sein

Messpunkte für Qualität?



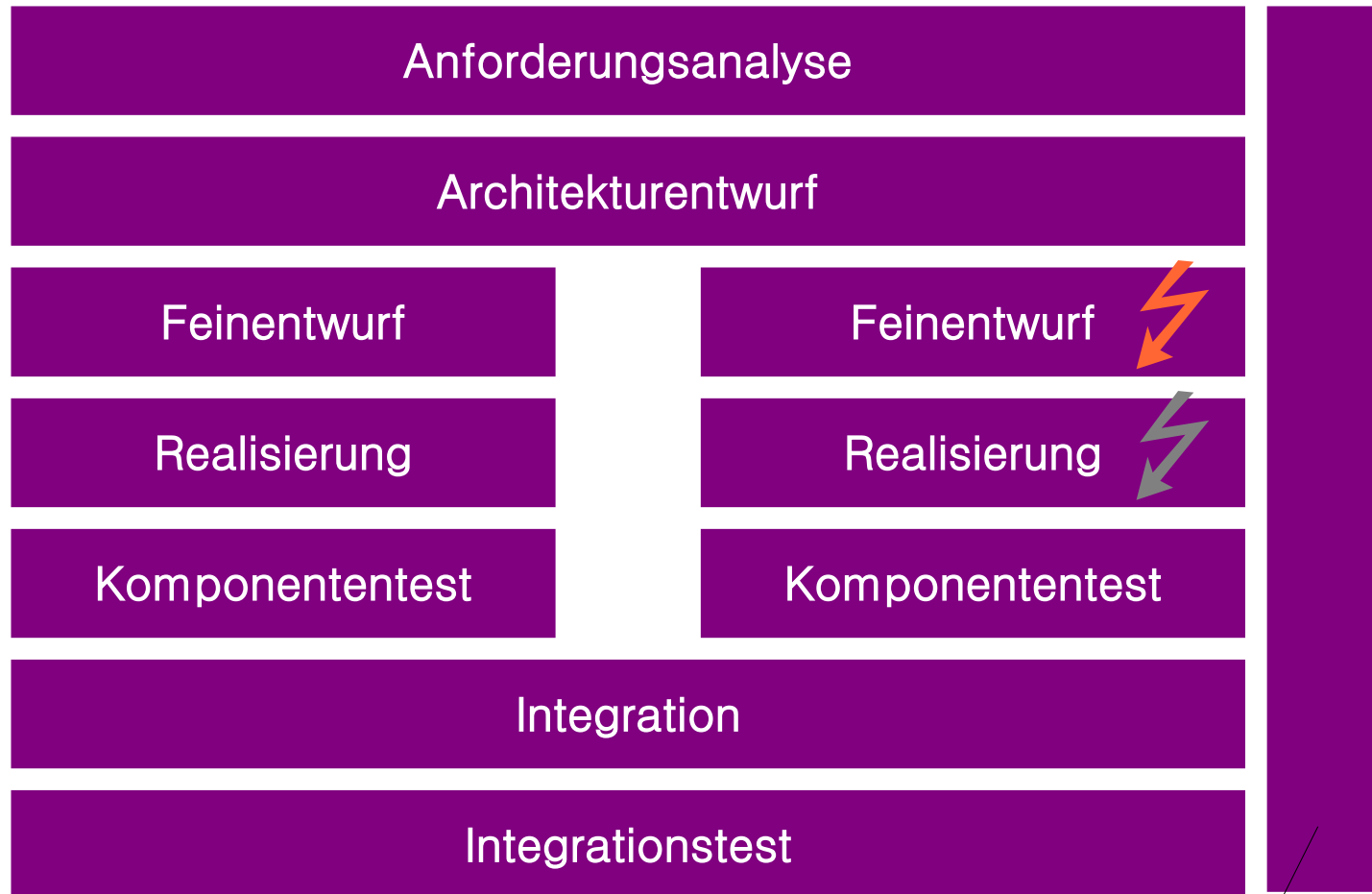
Projekt-Unterstützungssystem
Timetracking, Issuetracking...

Klassische Messpunkte für Qualität



Projekt-Unterstützungssystem
Timetracking, Issuetracking...

Moderne Messpunkte für Qualität



Das Risiko von Fehlern, die durch Fehler früherer Phasen verursacht werden wird minimiert

Projekt-Unterstützungssystem
Timetracking, Issuetracking...

Software Controlling

Erweiterung des Qualitätsverständnisses

Von Projektmanagement zu **Software Controlling**:
Risiken frühzeitig erkennen und Einfluss nehmen

Erweiterte Betrachtung

Status, Qualität & Zeiteinhaltung von Zwischenprodukten

- Lastenheft, Architekturentwurf, Kern-Komponenten
- Interne Qualitätsmerkmale wie Dokumentation & Metriken

Software Controlling

Definitionen

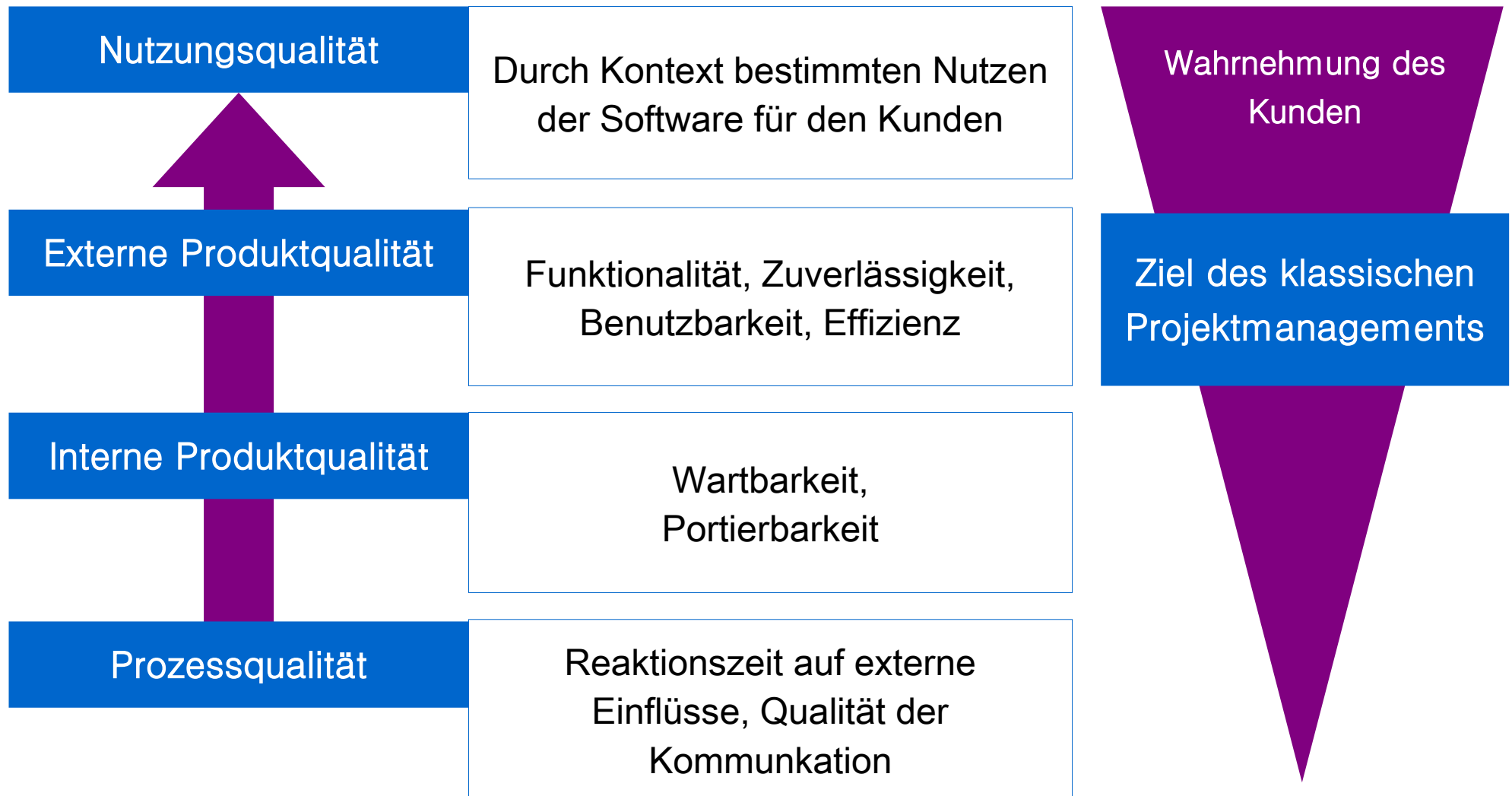
- Funktions- und bereichsübergreifendes **Koordinationsystem**
- “Als Ziele werden **Wirtschaftlichkeit und Effektivität der Planung, Steuerung und Kontrolle** aller IT-Prozesse, deren Ressourcen und der Infrastruktur in der Organisation definiert.” [Kütz05]
- **Allgemeiner Controlling Ansatz** [Kargl99]
Unterstützung, Informationsversorgung & Koordination von
 - Zielfindung
 - Planung
 - Überwachung

Software Controlling

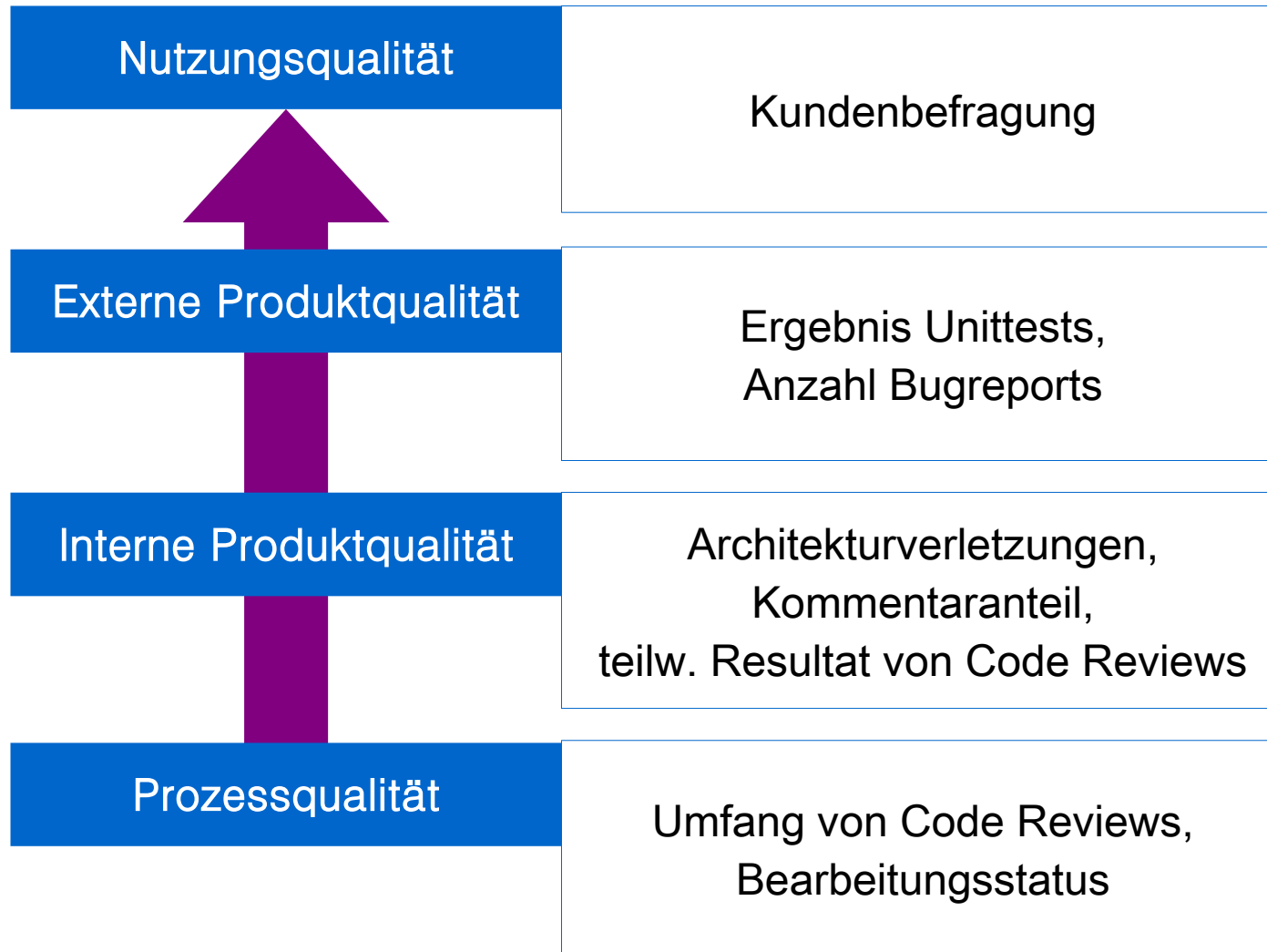
Ziele des Software Controllings

- Transparenz über den Zustand der Software und des Projekts
- Stärkung des Qualitätsbewusstseins der Beteiligten
- Überwachung der Kosten
- Stärkung des Vertrauens beim Kunden
- Schaffen von Wettbewerbsvorteilen gegenüber Unternehmen ohne effektives Software Controlling

Qualität und ihre Sichtweisen



Beispiele für Metriken



Erinnerung

Eigenschaft
des Objekts
(vorherige Folie)

Metrik

Zahlenwert

Qualitäts-Controlling durch Kennzahlensystem

Kennzahl / Metrik

- Indikator für einen Qualitätsaspekt
- Hat alleine wenig Aussagekraft

Kennzahlensystem

- Zusammengesetzt aus vielen Kennzahlen
- Gewichtet und interpretiert einzelne Werte
- Soll den Prozesszustand widerspiegeln
 - Fortschritt gegenüber Plan
 - Einhaltung von definierten Prozessabläufen
- Soll Auskunft über Risiken / Effizienzhemmnisse geben
 - Unreife Zwischenprodukte
- Zukunftsperspektive (Wartbarkeit, Erweiterbarkeit)

Qualitätsmodelle

Frühes Qualitätsmodell: McCall (1977)

Unterteilung nach Tätigkeit

Produktnutzung

Korrektheit, Zuverlässigkeit, Effizienz, Integrität, Nutzbarkeit

Produkttransition

Portierbarkeit, Wiederverwandbarkeit, Plattformunabhängigkeit

Produktüberarbeitung

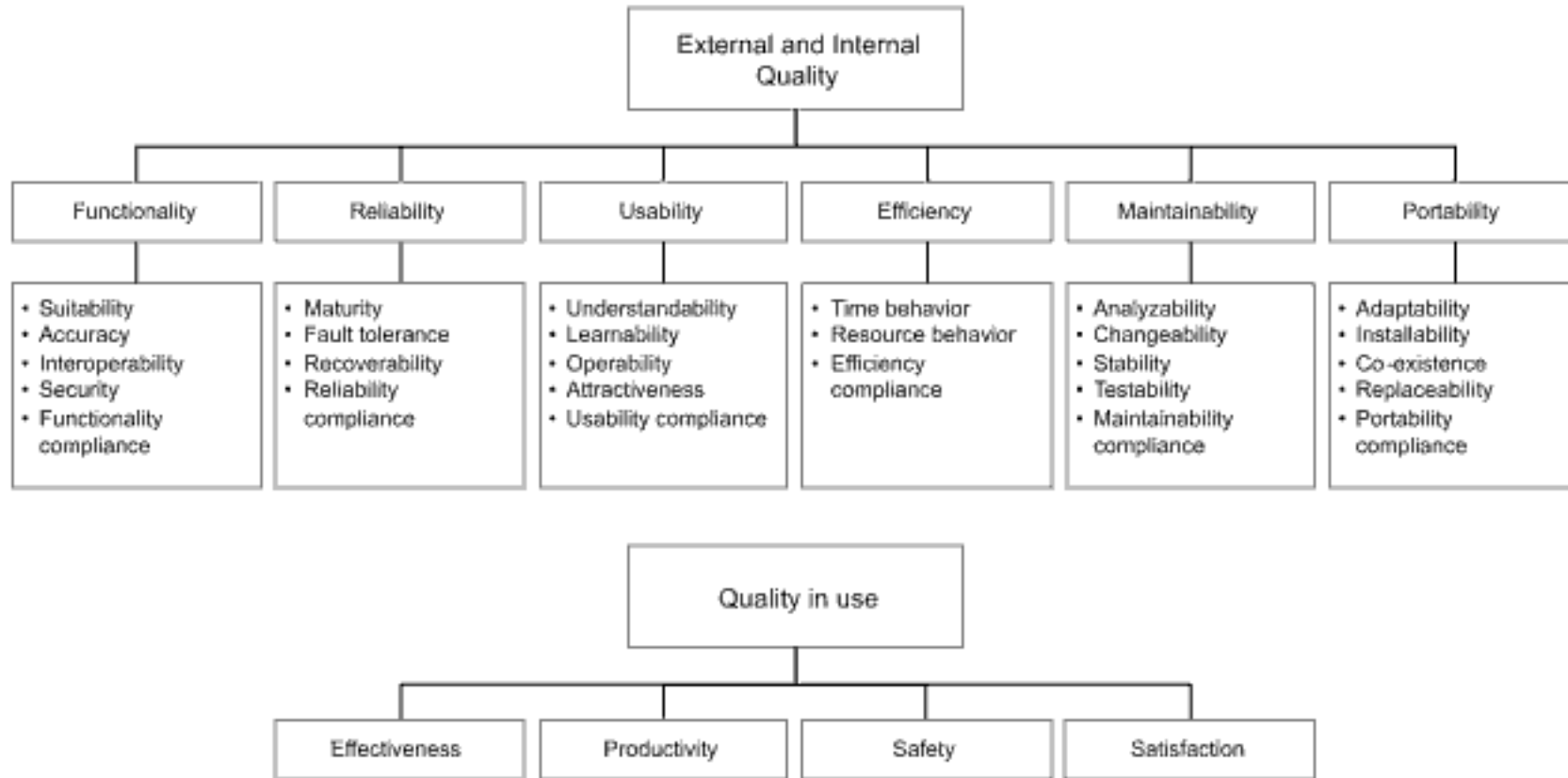
Wartbarkeit, Flexibilität, Testbarkeit

Viele der Faktoren können nicht genau gemessen werden, sondern werden auf einer Checkliste subjektiv mit Werten von 0 bis 10 belegt. Der Endwert ist die gewichtete Summe der Faktoren.

Vollständige Liste der Faktoren:

http://www.csse.monash.edu.au/courseware/cse3308/cse3308_2005/assets/McCall_Checklist.pdf

Heutiges Modell: ISO/IEC 9126



Definiert Begriffe, schafft Klarheit und Standards

Hohes Abstraktionsniveau, ebenfalls nicht direkt messbar

Kritik an Qualitätsmodellen

Fehlende Metriken

Viele Eigenschaften sind zu weich, um gemessen zu werden

Unpassende Metriken

Mehrdeutigkeit, Redundanz

In der Praxis nicht erhebbar

Falsches Testobjekt

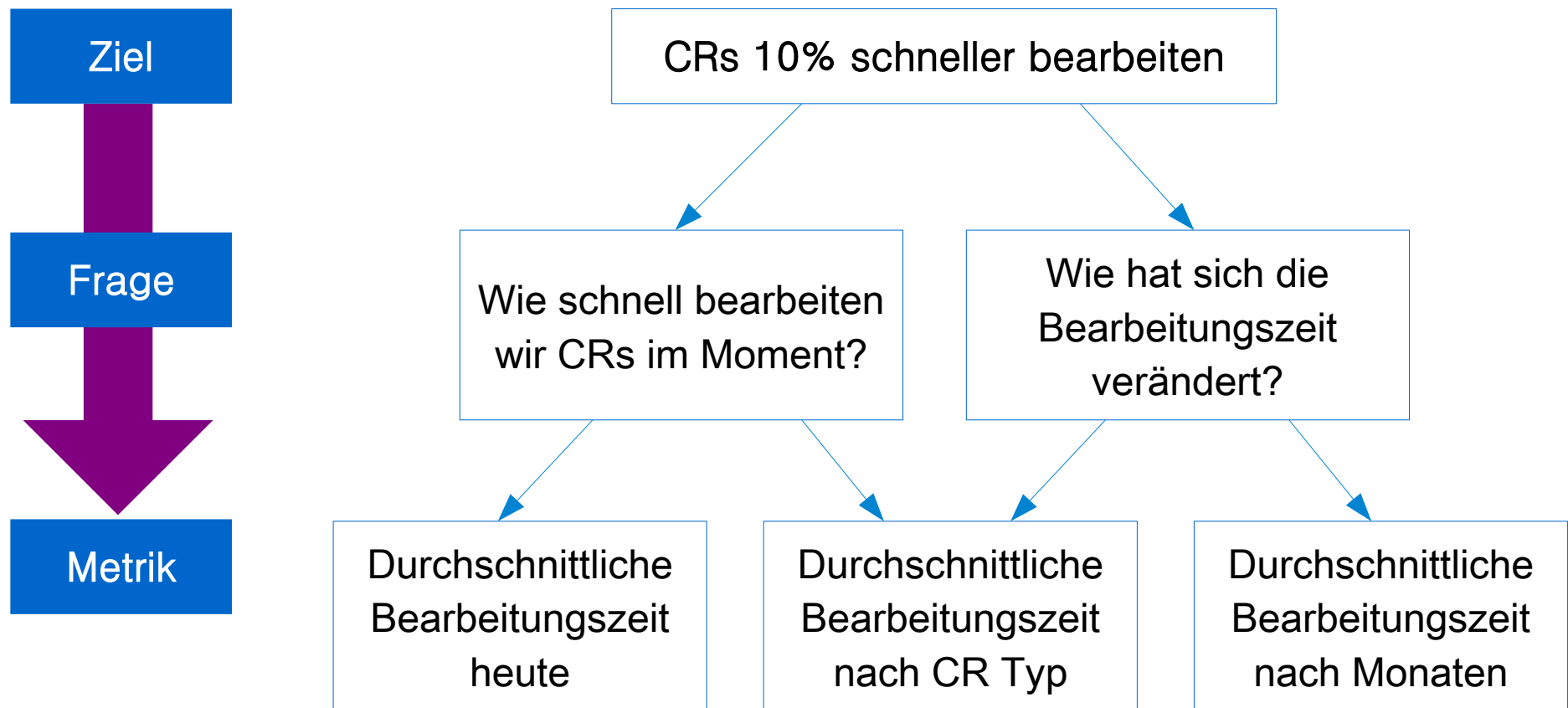
Fertige Software statt Zwischenprodukte oder Prozesse

Für Software Controlling ist aber beides nötig

Aggregation

Viele Standards machen keine Angaben zum Zusammenführen der Daten zu einer höherwertigen Aussage

Ansatz: Goal Question Metric



Ansatz: Goal Question Metric

Ziele haben fünf Aspekte

- Messobjekt
- Zweck
- Qualitätsfokus
- Blickwinkel
- Kontext

Analysiere die CR Bearbeitung
zum **Zwecke** der Verbesserung
in **Bezug auf** Bearbeitungszeiten
vom **Blickwinkel** des Entwicklers
im **Kontext** des Projekts X

Das Capgemini sd&m Kennzahlensystem

Hintergrund

- Entwicklung aus der Industrie (Capgemini sd&m), kommerziell
- Maßnahme gegen erkannte Unzulänglichkeit bisheriger Systeme
- Verbindet Systeme zu einem großen Kennzahlensystem

Eigenschaften von Software

- Systemumfang und -komplexität
- Technische Code-Qualität
- Effizienz
- Korrektheit und funktionale Vollständigkeit
- Fortschritt
- Allgemeiner Projektstatus

Systemumfang und -komplexität

Kenngroßen

- SLOC
- Anzahl Statements
- Anzahl Klassen
- Anzahl Kommentare
- Anzahl Abhängigkeiten

Motivation

Allgemeine Information, Normalisierung anderer Messgrößen, gibt Auskunft über Qualität des Entwurfs

Quellen

javaNCSS, SonarJ

Technische Code-Qualität

Kenngroßen

- Anzahl Regelverstöße
- Warnungsdichte (nicht kritische Regelverstöße)

Motivation

Einhaltung von Standards und Vorgaben

- Architekturentwurf
- Coding Guidelines
- State-of-the-Art

Quellen

SonarJ, Findbugs

Effizienz

Kenngroßen

- Antwortzeit
- Aufrufhäufigkeit

Motivation

Bewertung von Leistungszielen

Quellen

Performancelog

Stresstest

Korrektheit und funktionale Vollständigkeit

Kenngrößen

- Fehleranzahl
- Fehlerdichte
- Fehlerhafte Testfälle

Motivation

Beurteilung der funktionalen Qualität

Quellen

Bugzilla

xUnit

Fortschritt

Kenngroßen

- Bearbeitungsstatus
- Anzahl offener Issues
- Anzahl Änderungen
- Testüberdeckung
- Codereview-Aufwand
- Codereview-Tiefe

Motivation

Gegenüberstellung

- planerischer Bearbeitungsstatus
- Entwicklungsdynamik der Komponenten
- Umfang an durchgeführten Entwicklungs- und QS-Maßnahmen

Fortschritt

Quellen

- Artefaktliste
- Aktivitätenliste
- Konfigurationsmanagement
- Projektmanagement-Tool
- Evtl. Bugzilla

Allgemeiner Projektstatus

Kenngroßen

- Subjektive Aussagen von Mitarbeitern

Motivation

Vervollständigung des ermittelten Projektstatus

Validierung der Kennzahlensystems

Quellen

Voting im Web durch Mitarbeiter

Fragebögen

Vergleich der Kennzahlen

Produktstruktur (Komponentenebene)

Umfang und Komplexität

Tech. Code-Qualität

Effizienz

Korrektheit

Wert mit Kritikalität der Komponente gewichten (Wichtigkeit der Funktion aus Anwendersicht, Komplexität)

Zeitlicher Verlauf

Fortschritt

Projektstatus

Bezogen auf ganzes Projekt

Vergleich der Kennzahlen

Automatisierte Erhebung

Umfang und Komplexität

Tech. Code-Qualität

Effizienz

Korrektheit

Fortschritt

Manuelle Erhebung

Projektstatus

Bezieht menschliche
Urteilkraft mit ein

Mögliche Fragen

- Wie ist die Qualität?
- Wird der nächste Meilenstein planmäßig erreicht?
- Wie ist die Stimmung?

Vergleich der Kennzahlen

Daten aus Build-Prozess

Umfang und Komplexität

Tech. Code-Qualität

Korrektheit

Manuelle Eingaben

Projektstatus

Tech. Code-Qualität

Fortschritt

Aus Planungswerkzeugen

Projektstatus

Fortschritt

Aus Fehlerverwaltung

Korrektheit

Aus laufendem System

Effizienz

ConQAT

Hintergrund

- Universitäre Entwicklung (TU München, Lehrstuhl Broy)
- Open Source
- Beinhaltet Kennzahlensystem mit Reports
- Gibt auch Überblick über andere Bereiche
 - SVN Statistiken
 - Bugzilla Statistiken
 - Clone Detection

Dashboard



ConQAT: CCSM Release Code Base

CCSM Release Code Base @ Sun Jun 06 02:23:14 CEST 2010

- Overview
- SVN Repository
- Bugzilla
- CCSM Release Code Base
- Architecture Analysis
- Info
 - Log Messages
 - Config Graph
 - Execution Time
 - Config

SVN Repository

Commit Messages

Authors

CCSM Release Code Base 4572/222/34

1946/0/25 Findings

747/0/0 JUnit Results

1879/222/96 LEvD Rating CCSM Release Code Base

LEvD Rating Evolution

Findings Trend

Metric Overview

Overview

Architecture-Centric Dashboard

Bugzilla 1488/0/67

CRs

CR Status

CM Users

CR Type

1488/0/67 Assessment

CR Evolution

Architecture Analysis

Architecture description

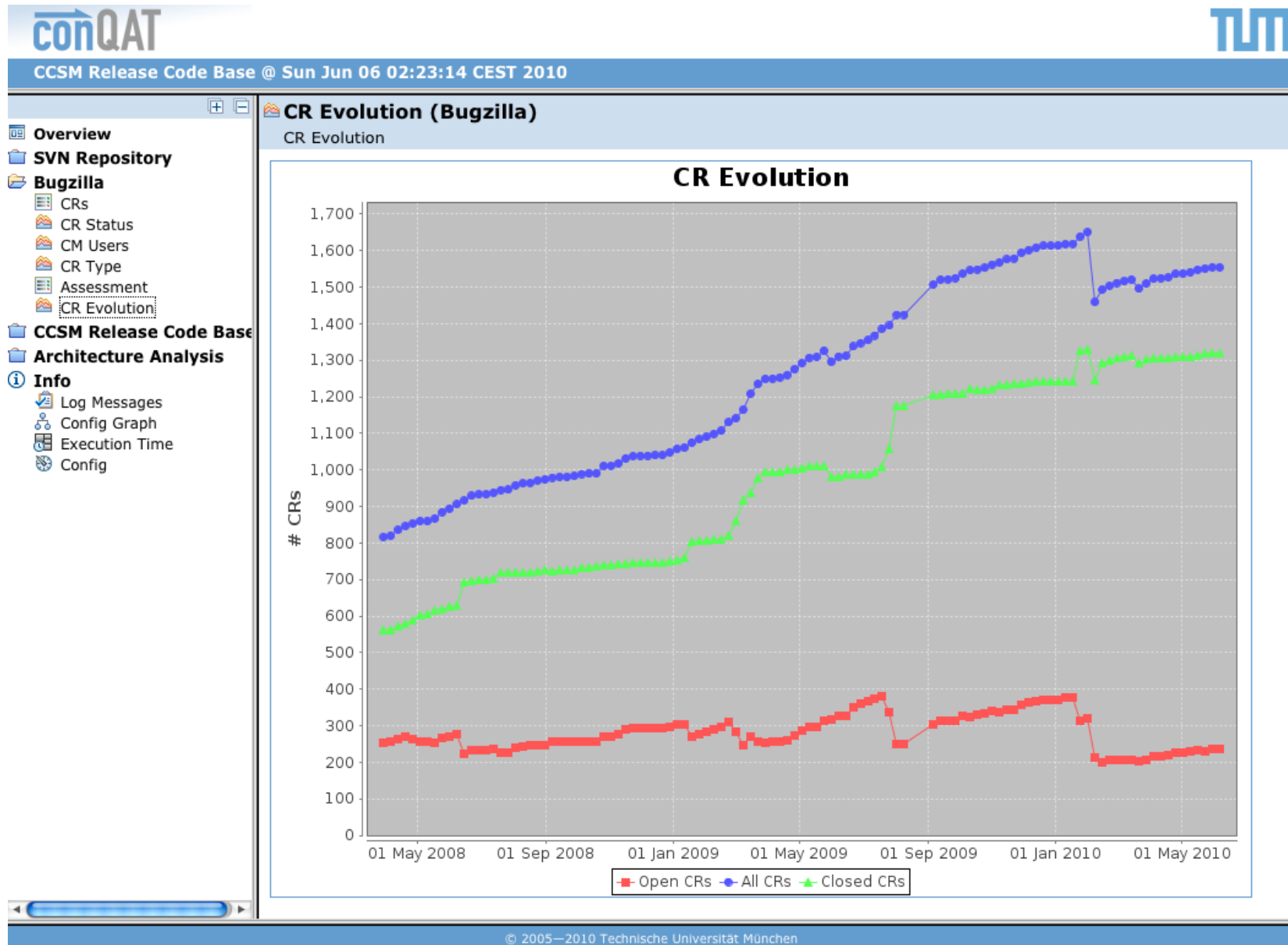
Architecture assessment

Architecture violations

Architecture violations and tolerations

Architecture Elements

Change Requests



Metriken

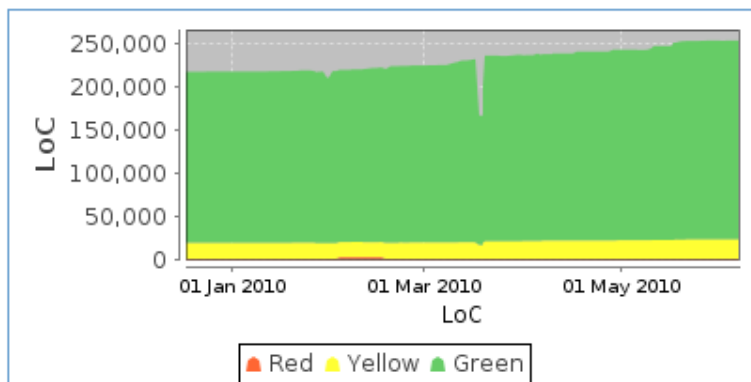
Metric Overview (CCSM Release Code Base)

Metric Overview

Size

Element	files	%files	LoC	%LoC
]16;400]	2,153	97.997	230,186	90.832
]400;1,000]	43	1.957	22,199	8.76
]1,000;1,034]	1	0.046	1,034	0.408
Sum	2,197	100	253,419	100

- Full Source Tree
- Tree Map
- Sorted List

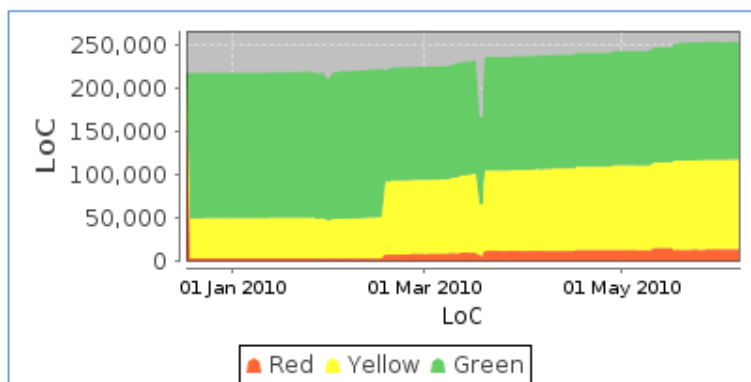


Element	LoC
edu.tum.cs.commons.html.EHTMLElement	1,034
edu.tum.cs.commons.string.StringUtils	886
edu.tum.cs.commons.filesystem.FileSystemUtils	806
edu.tum.cs.cd.core.Model	730
edu.tum.cs.commons.reflect.ReflectionUtils	717
edu.tum.cs.conqat.filesystem.library.FileLibrary	696
edu.tum.cs.commons.options.Options	668
edu.tum.cs.commons.html.EHTMLAttribute	654
edu.tum.cs.scanner.ETokenType	641
edu.tum.cs.commons.filesystem.FileSystemUtilsTest	576

Comment Ratio

Element	files	%files	LoC	%LoC
]0.09;0.2]	56	2.549	13,151	5.189
]0.2;0.4]	595	27.082	103,728	40.931
]0.4;0.94]	1,546	70.369	136,540	53.879
Sum	2,197	100	253,419	100

- Full Source Tree
- Tree Map
- Sorted List



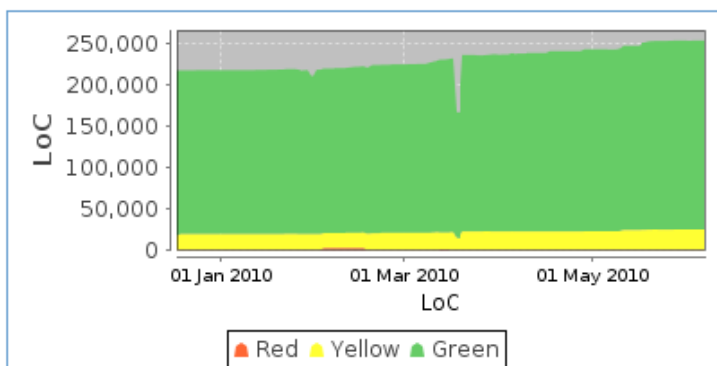
Element	CR
edu.tum.cs.scanner.ETokenType	0.09
edu.tum.cs.cqinspect.findings.views.FindingGroupsView	0.11
edu.tum.cs.cqedit.ui.bundle.DependencySection	0.111
edu.tum.cs.commons.html.EHTMLElement	0.113
edu.tum.cs.conqat.sourcecode.analysis.LongestBlockAnalyzer	0.116
edu.tum.cs.scanner.PythonScannerTest	0.117
edu.tum.cs.conqat.abap.analyzer.CommentToFindingsConverter	0.121
edu.tum.cs.cqedit.ui.editor.ConQATBlockDocumentationComposite	0.124
edu.tum.cs.cqedit.ui.refactoring.participants.RenameBlockParameterParticipant	0.131
edu.tum.cs.conqat.sourcecode.pattern.TokenTypePatternFinder	0.133

Metriken

Length of Longest Method

Element	files	%files	LoC	%LoC
]0;40]	2,115	96.268	230,745	91.053
]40;100]	81	3.687	22,181	8.753
]100;105]	1	0.046	493	0.195
Sum	2,197	100	253,419	100

- Full Source Tree
- Tree Map
- Sorted List

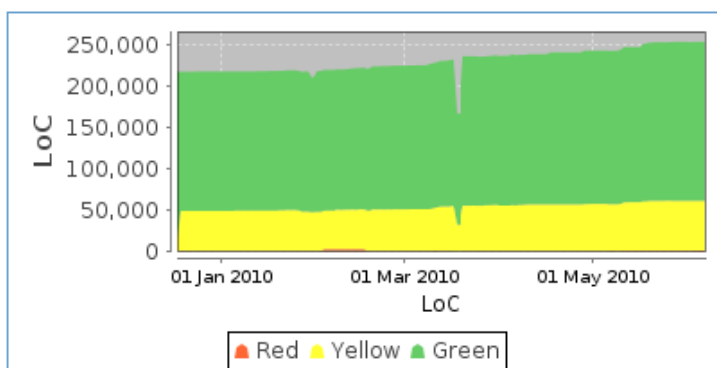


Element	LML
edu.tum.cs.conqat.clonedetective.detection.suffixtree.ApproximateCloneDetectingSuffixTree	105
edu.tum.cs.conqat.model_clones.detection.util.AugmentedModelGraph	97
edu.tum.cs.simulink.util.SimulinkModelWalker	97
edu.tum.cs.scanner.PythonScannerTest	89
edu.tum.cs.conqat.clonedetective.Index.ReadOnlyCloneIndex	88
edu.tum.cs.cqinspect.findings.mapper.FileMapperConfigDialog	87
edu.tum.cs.scanner.PLSQLScannerTest	86
edu.tum.cs.conqat.commons.findings.xml.FindingReportReaderHandler	83
edu.tum.cs.conqat.driver.Instance.ProcessorInstance	81
edu.tum.cs.conqat.model_clones.detection.clustering.CloneClusterer	77

Nesting Depth

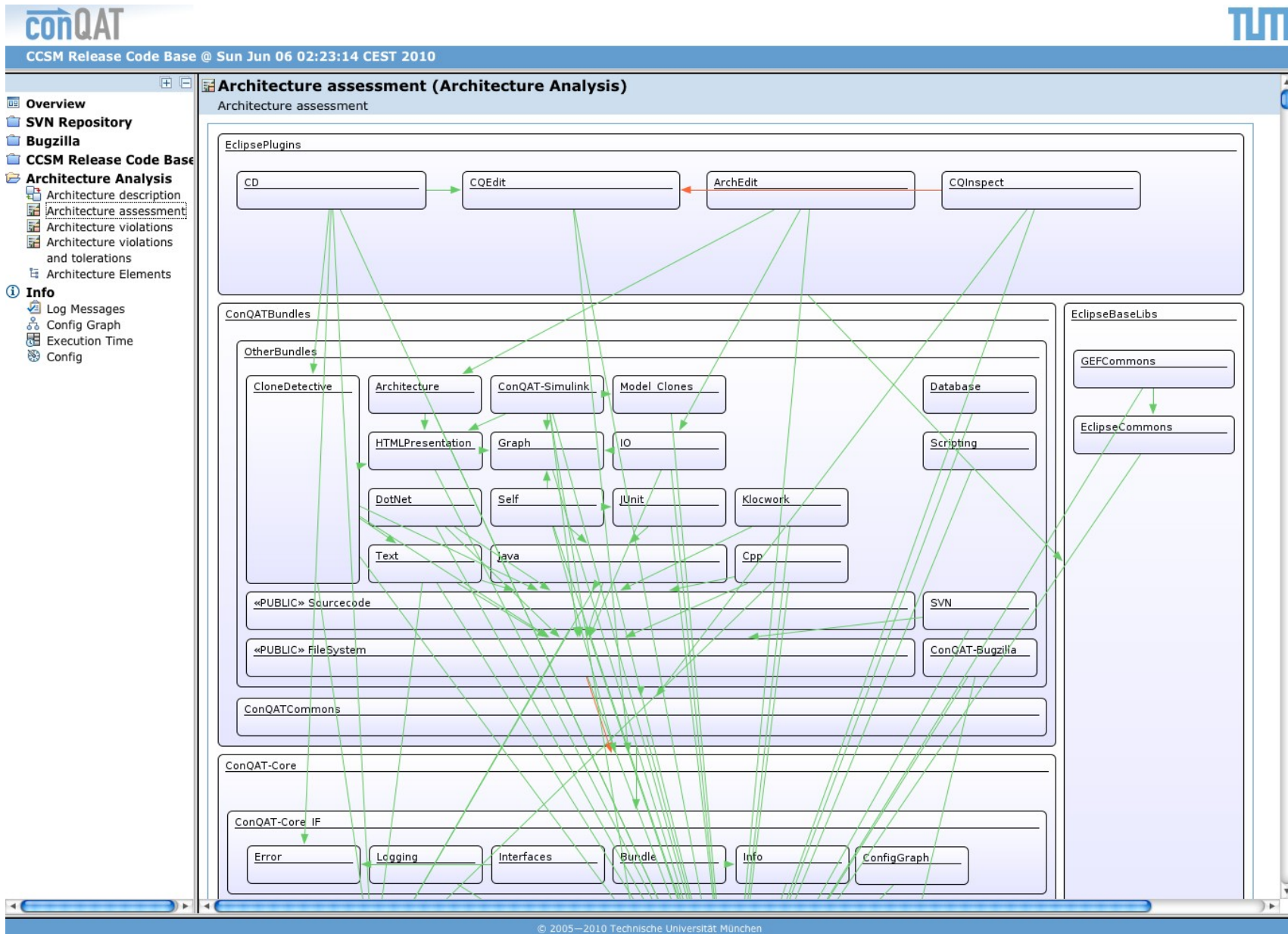
Element	files	%files	LoC	%LoC
]1;5]	2,118	96.404	234,760	92.637
]5;7]	75	3.414	17,653	6.966
]7;9]	4	0.182	1,006	0.397
Sum	2,197	100	253,419	100

- Full Source Tree
- Tree Map
- Sorted List



Element	NestingDepth
edu.tum.cs.emf.commons.tooltips.TableViewerAttributeTooltip	9
edu.tum.cs.emf.commons.tooltips.TreeViewerAttributeTooltip	9
edu.tum.cs.cqedit.ul.bundle.DependencySection	8
edu.tum.cs.cqinspect.findings.mapper.FileMapperConfigDialog	8
edu.tum.cs.cqedit.ul.qlaunch.scopes.JavaScopeAdapter	7
edu.tum.cs.cqedit.ul.editor.layout.LayoutAction	7
edu.tum.cs.cqedit.ul.refactoring.extractblock.ExtractionData	7
edu.tum.cs.conqat.distributed.RemoteJobDistributor	7
edu.tum.cs.conqat.sourcecode.analysis.LongestBlockAnalyzer	7
edu.tum.cs.conqat.graph.bullder.ScopeGraphCreator	7

Architektur



Architektur

conQAT
CCSM Release Code Base @ Sun Jun 06 02:23:14 CEST 2010

Overview

- SVN Repository
- Bugzilla
- CCSM Release Code Base
 - Architecture Analysis
 - Architecture description
 - Architecture assessment
 - Architecture violations
 - Architecture violations and tolerations
 - Architecture Elements
 - Info
 - Log Messages
 - Config Graph
 - Execution Time
 - Config

HTMLPresentation | Graph | IO | Scripting

DotNet | Self | JUnit | Klocwork

Text | Java | Cpp

«PUBLIC» Sourcecode | SVN

«PUBLIC» FileSystem | ConQAT-Bugzilla

ConQATCommons

ConQAT-Core

ConQAT-Core IF

Error | Logging | Interfaces | Bundle | Info | ConfigGraph

BaseLibs

Scanner | Simulink | Bugzilla

«PUBLIC» CCSMCommons

as SVG

Element	assessment	dependencies
CQInspect -> CQEdit	●	edu.tum.cs.cqinspect.findings.qlaunch.FindingReportOutputHandler -> edu.tum.cs.cqedit.ui.model.CQxmlParameter edu.tum.cs.cqinspect.findings.qlaunch.FindingReportOutputHandler -> edu.tum.cs.cqedit.ui.model.CQxmlBlockSpecModel edu.tum.cs.cqinspect.findings.qlaunch.FindingReportOutputHandler -> edu.tum.cs.cqedit.ui.model.CQxmlAttribute edu.tum.cs.cqinspect.findings.qlaunch.FindingReportOutputHandler -> edu.tum.cs.cqedit.ui.qlaunch.output.QLaunchOutputHandlerBase edu.tum.cs.cqinspect.findings.qlaunch.FindingReportOutputHandler -> edu.tum.cs.cqedit.ui.model.CQxmlProcessor
FileSystem -> ConQAT-Core	●	edu.tum.cs.conqat.filesystem.generators.DuplicateFileDeletionScriptGenerator -> edu.tum.cs.conqat.build.ANTWriter

© 2005–2010 Technische Universität München

Einsatz von Kennzahlen

Zieldefinierte Verwendung

Zieldefiniert heißt

Ausgehend von gezielten Fragen zur Qualität

Es werden zum Beantworten

- Kennzahlen gemessen
- Reports erstellen

Bsp: Qualität des täglichen Builds

- Verstöße gegen Guidelines
- Fehlgeschlagene Testfälle

Bsp: Kontrolle der Qualitätssicherungsmaßnahmen

- Review-Aufwände
- Testabdeckung

Explorative Verwendung

Explorativ heißt

- Überblick über Entwicklung der Kennzahlen (Trends)
- Vergleich der Komponenten
- Auffälligkeiten finden, Untersuchungen anstoßen
- Vor allem für Qualitätsverantwortliche

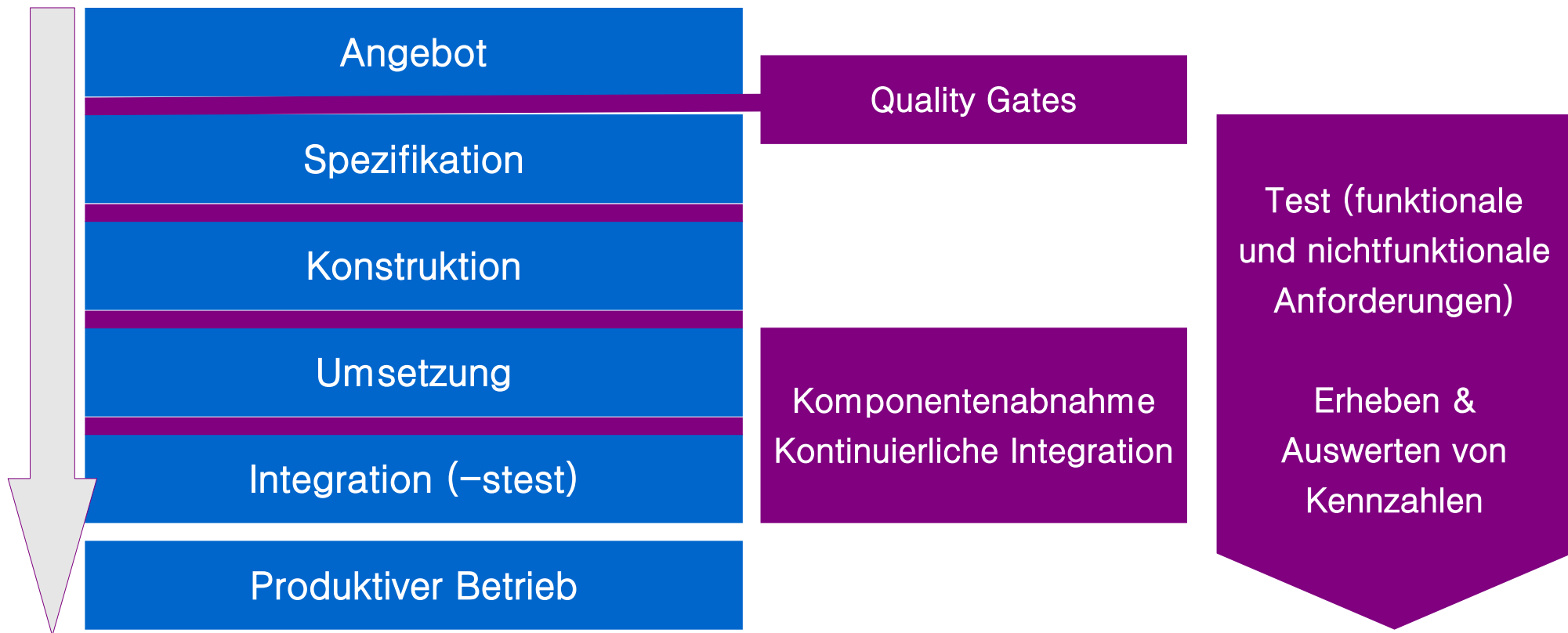
Beispiel

- Komponente hat *vergleichsweise* hohe Fehlerdichte
- Testüberdeckung prüfen (besonders viele Tests?)
- Maßnahmen ergreifen

Das Kennzahlensystem im Entwicklungsprozess

Durchführen von Analysen

- Kontinuierlich (eher explorativ): **Projektleitstand**
- Zu Meilensteinen (eher zieldefiniert)
- An Quality Gates (eher zieldefiniert)



Wichtige Bemerkungen

Aussagekraft der Kennzahlen

- Die Zahlen dienen als Orientierung, nicht als Ziel gegen das optimiert wird
- Die Werte müssen immer interpretiert und in Kontext gesetzt werden
- Die Kennzahlen sind zwar ein Frühwarnsystem, garantieren aber an sich keine hohe Qualität

Aggregation und Automatisierung

- Aggregation zu Gesamtwerten nicht automatisch möglich
- Selbst Indikatoren (wie Ampeln) sind trügerisch

Vielen Dank für die Aufmerksamkeit

Quellen

- Bennis et al. – Software Controlling
Informatik Spektrum 31.06.2008
- Kütz05: **Martin Kütz – IT Controlling für die Praxis** (1. Auflage 2005)
- Kargl99: Herbert Kargl – DV Controlling (4. Auflage 1999)
- Peter Liggesmeyer – Softwarequalität:
Testen, analysieren und verifizieren von Software
- de.wikipedia.org/wiki/Goal_Question_Metric