Vorlesung (WS 2014/15)

Softwarekonstruktion

Prof. Dr. Jan Jürjens

TU Dortmund, Fakultät Informatik, Lehrstuhl XIV

Teil 0: Organisatorisches und Einleitung

v. 17.10.2014

Vorlesungswebseite (bitte notieren):
http://www-secse.cs.tu-dortmund.de/secse/pages/teaching/ws14-15/swk/index_de.shtml
(oder: http://jan.jurjens.de : rechte Spalte: „Softwarekonstruktion“; oder: Link im LSF)

Agenda

- **Organisatorisches**
  - **Studienordnung: Einordnung / Kompetenzen / Struktur**
  - **Vorlesung: Bildungsvertrag, Termine, Feedback**
  - **Übung: Konzept / Termine**
  - **Klausur**
- Vorstellung des Fachgebietes
- Vorlesungsinhalte

Ziele dieser Vorlesung

Erlangbare Kompetenzen innerhalb der Vorlesung:

- Vermittlung eines Überblicks über das Spektrum gängiger **Konzepte zur Spezifikation** und **zum Testen.**
- **Einbettung dieser Spezifikationskonzepte** in die Qualitätssicherung.
- **Modellbasiertes Software-Engineering**.
- Erläuterung verschiedener **Testtechniken.**
- **Fortgeschrittene Spezifikations-** und **Verifikationstechniken** (OCL, statische Analyse)

Struktur laut Modulhandbuch (Bachelor)

LEHRSTUHL 14
SOFTWARE ENGINEERING

Bachelor Informatik / Angewandte Informatik: **Wahlpflichtmodul.**

Teilnahmevoraussetzungen:

- **Erfolgreich abgeschlossene Module** „Software-Technik“ (SWT) und „Softwarepraktikum“ (SoPra)
- **Kenntnisse**: Objektorientierung, Programmierpraxis, Mathematische Grundlagen der Informatik

Umfang: 3 SWS (2 SWS Vorlesung, 1 SWS Übung)

4 Credits: 3 Credits Vorlesung, 1 Credit Übung

Aufwand: 120 Stunden über 15 Semesterwochen:

- 45 Stunden Präsenz (15*(2+1))
- 75 Stunden Vor-/Nachbereitung und Hausübungen (15*5)

Veranstaltungssprache: Deutsch.

Bildungsvertrag: Wir bieten ...

- **Fachliche Einführung** in das Thema Softwarekonstruktion.
- **Engagierte Betreuung:**
  - Interessante Vorlesung.
  - Regelmäßige Sprechstunden.
  - Betreute Übungen.
  - Korrigierte Hausübungen.
  - Transparente Anforderungen.
  - Möglichkeiten zum direkten Feedback.
- Möglichkeit zum **Erwerb des Scheins.**

Bildungsvertrag: ...wir erwarten:

Aktives Auseinandersetzen mit den Vorlesungsinhalten:

- Aktive Teilnahme an der Vorlesung.
- Vor- und Nachbereitung der Vorlesung.
- Aktive Teilnahme an den Übungen.
- Bearbeitung der Hausübungen.

Vorlesung
Termin, Webseite

Vorlesungstermin: Mo 12:15 bis 13:45, Otto-Hahn-Str. 14 – E23

Abstimmung: PAUSE ?

Aktuelle Informationen zur Vorlesung (**bitte regelmäßig beachten wegen möglicher Vorlesungsausfälle o.ä.**):

http://www-secse.cs.tu-dortmund.de/secse/pages/teaching/ws14-15/swk/index_de.shtml

(oder: http://jan.jurjens.de : rechte Spalte: „Softwarekonstruktion“).

Vorlesung
Folien, Skript, Aufzeichnung

Vorlesungsfolien werden auf o.g. Webseite zur Verfügung gestellt (planmäßig spätestens 18.00 Uhr am Vortag der Vorlesung).

Enthalten Diskussionsfolien; zugehörige Antwortfolien werden erst nach der Vorlesung online gestellt.

Vollständigkeit der Folien (Klausurvorbereitung) vs. „Textwand“:
Teilweise **Notizfolien** und **Anhänge**:

- Inhalt in Vorlesung nur mündlich wiedergegeben
- Textversion nützlich z.B. für Klausurvorbereitung

Zusätzlich: **Textnotizen** zu ausgewählten Inhalten (“**Skript**”, s. Web)

Vorlesungsaufzeichnung:

- Nur zur Verwendung von Teilnehmern der Vorlesung.
- Zeitaufwändige Nachbearbeitung => V.A. zur Klausurvorbereitung.

Termine (**14-tägig**):

- Dienstags, 10:15 - 11:45, OH14/104
- Mittwochs, 12:15 - 13:45, OH14/304
- Mittwochs, 14:15 - 15:45, OH12/2.063
- Donnerstags, 08:15 - 09:45, OH12/1.056
- Donnerstags, 10:15 - 11:45, OH14/104

Start der Übungen: KW 44 (ab dem 27.10.2014)

Anmeldung:

- Via AsSESS
- http://ess.cs.uni-dortmund.de/ASSESS/index.php?do=lecturelist

- Anmeldung heute ab 15 Uhr möglich.
- **Anmeldung bis 13.10.2014, 14:00 Uhr.**
- Verteilung mittels Algorithmus (Solver).
  (nicht priorisiert nach zeitlichem Eingang der Anmeldung)
- **Bekanntgabe der Verteilung am 15.10.2014.**

Übung
Formalia

- Ablauf der Übungen und Erwerb der Studienleistung
  - siehe Webseite zur Vorlesung oder 1. Übungsblatt

Online-Übungen

LS 14 hat für SWK **Web-basiertes System für automatische Bewertung von Übungsaufgaben** entwickelt (teilweise Finanzierung durch „QVM-Mittel“- **DANKE !**).

Zwei Einsatzszenarien:

- Zusätzliches Angebot für freiwillige Bearbeitung von Übungsaufgaben, z.B. **Klausurvorbereitung**.
- Einsatz im Rahmen von **2 Übungsblättern** am Semester-Ende: Erlaubt zusätzliches Angebot dieser Übungsblätter mit kurzfristigem Feedback durch automatisches System (sonst aus Resourcengründen nicht realisierbar).

Inpud-Forum

Ziel: **Diskussion** der Studierenden untereinander.

Zusätzl. Kommunikation mit Veranstaltern für Fragen zum Inhalt:

- **Keine garantierten Antwortzeiten**
- Für Dringendes oder Vertrauliches: **Mail** oder **Sprechstunde**

Organisatorische + inhaltliche FAQ

- Für Fragen von Studierenden, die auch für andere interessant sein könnten.

Moderation durch Veranstalter.

Prüfung
Studienordnung

Bachelor Informatik / Angewandte Informatik**:**

- **Studienleistung**: Übungsschein
- **Modulprüfung**: Klausur (bei bestandener Studienleistung)

=> **Bearbeitung der Übungsaufgaben** ist **Voraussetzung** zur Teilnahme an der Modulprüfung.

Diplom:

- **Prüfungsleistung**: Klausur

Prüfung: Klausur

- schriftlich
- 60 Minuten

Klausurtermine:

- Mittwoch, 25.02.2015, 15:00 - 16:30 Uhr
  Audimax, E 29
- Vorauss.:
  Mittwoch, 25.03.2015, 16:00 - 17:30 Uhr
  Audimax, E 29

Feedback

Wir **bitten um vorlesungsbegleitendes Feedback**, um Verbesserungen semesterbegleitend durchführen zu können.

Übliche Kontaktmöglichkeiten:

- Nach der Vorlesung
- E-mail jan.jurjens@cs.tu-dortmund.de
- Tel.: 0231 755-7208
- Sprechstunde: Mo 11-12 (bitte vorher per email anmelden)
- Anonymes Kontaktformular:
  http://www-secse.cs.tu-dortmund.de/secse/pages/teaching/feedback_de.shtml
  (s. Link von Vorlesungswebseite). SWK ankreuzen !

Darüberhinaus: interne Umfragen für vorlesungsbegleitendes Feedback.

Kontakt

Vorlesungsseite (bitte regelmäßig beachten):

http://www-secse.cs.tu-dortmund.de/secse/pages/teaching/ws14-15/swk/index_de.shtml

(oder: http://jan.jurjens.de : rechte Spalte: „Softwarekonstruktion“; oder: Link im LSF)

Ansprechpartner (WWW: verlinkt von o.g. Webseite):

- Vorlesung:
  - Jan Jürjens
- Übung:
  - Jens Bürger
  - Nina Harmuth
  - Janine Hemmers
- Inpud-Forum: http://inpud.cs.uni-dortmund.de
- Übungsanmeldung: http://ess.cs.uni-dortmund.de/ASSESS

Agenda

- Organisatorisches
- **Vorstellung des Fachgebietes**
  - **Forschung**
  - **Abschlussarbeiten, Hiwi-Jobs, weiteres Lehrangebot**
- Vorlesungsinhalte

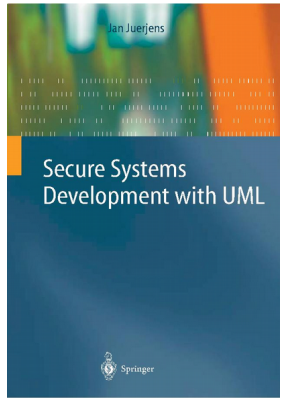
Vertrauenswürdige Software-Systeme

Erwartungen an Vertrauenswürdigkeit von Software in letzten 10 Jahren stark gestiegen.

→ **Oft nicht erfüllt**, viele Software-Systeme angreifbar.

- **Teil des Problems**: System- und Software-Entwicklungsmethoden konnten mit gestiegenen Erwartungen bei steigender Systemkomplexität nicht mithalten.

Modellbasierte Entwicklung



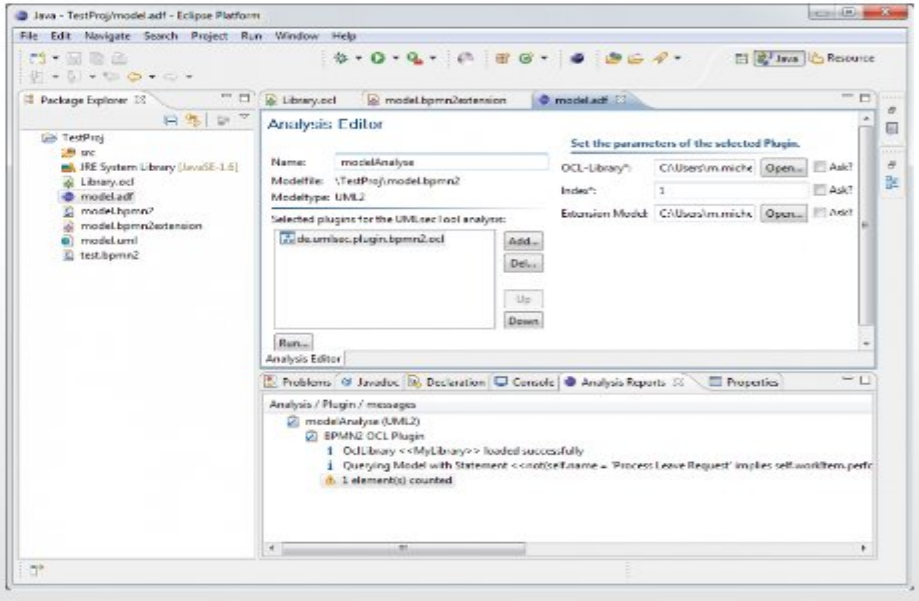
Werkzeugunterstützung

CARiSMA

Home - News - Team - Contact
Features - Checks - Docs - Installation

http://carisma.umlsec.de

2012-02-06: A new update of CARiSMA has been released!



Welcome to CARiSMA!

Modeling offers an unprecedented opportunity for high-quality critical systems development that is feasible in an industrial context. CARiSMA enables you to perform:

- **compliance** analyses,
- **risk** analyses, and
- **security** analyses

of software models.[1)]

Since CARiSMA is a reimplemented variant of the former UMLsec tool it natively supports UML models. Due to its EMF-based implementation CARiSMA can also support **domain-specific modeling languages** such as BPMN.

CARiSMA is fully **integrated into Eclipse** and can thus become part of the modeling tool of your choice including but not limited to TOPCASED, Papyrus MDT, IBM Rational Software Architect, and many others.

A flexible **plugin architecture** makes CARiSMA extensible for new languages and allows users to implement their own compliance, risk, or security checks.

Anwendungen in Unternehmen

- Digitaler Formularschrank [SAFECOMP 03]
- Internes Informationssystem [ICSE 07]
- Biometrisches Authentisierungssystem: mehrere Schwachstellen aufgedeckt [ACSAC 05, Models 09]
- Gesundheitskarte: Architektur mit UMLsec untersucht, Schwachstellen aufgedeckt [Jour. Meth. Inform. Medicine 08]
- Common Electronic Purse Specifications (Globaler Standard für elektr. Geldbörsen): mehrere Schwachstellen aufgedeckt [IFIPSEC 01, ASE 01]
- Gesundheitsinformationssysteme [Caise 09]
- Return-on-Security Investment Abschätzung
- Analyse Elektronische-Signatur-Architektur
- IT-Sicherheits-Risikomodellierung
- Smart-card Software-Update Plattform

Aktuell:

- Cloud-Anwender Compliance/Sicherheit
- Sicherheitsökonomische Analysen
- Cloud-Anbieter Compliance/Sicherheit

Weitere relevante Lehrveranstaltungen

WiSe 2014/15:

- Vorlesung “Software-Konstruktion” (Bachelor-Wahlpflicht)
- **Vorlesung „Sicherheit: Fragen und Lösungsansätze“ (Bachelor-Wahlvorlesung)**
- Proseminar “Werkzeugunterstützung für sichere Software“
- Seminar „Software-Engineering und Sicherheit“

SoSe 2015 (**unter Vorbehalt**):

- **Fachprojekt „Softwaretechniken für sichere Cloud-Computing-Systeme”**
- **Proseminar “Werkzeugunterstützung für sichere Software“**
- **Vorlesung „Software-Engineering für langlebige Systeme“ (Bachelor-Wahlvorles.)**
- **[Vorlesung “Methodische Grundlagen des Software Engineering” (Master-Basismodul Software) => erst wieder SoSe 2016 wg Forschungssemester]**
- **[Seminar „Software-Engineering und Sicherheit“ => erst wieder WiSe 2015/16 wg Forschungssemester]**

Forschungsbereich Master: Software, Sicherheit und Verifikation

Schwerpunktgebiete Diplom: Sicherheit und Verifikation, Software-Konstruktion

Informationen unter: http://www-secse.cs.tu-dortmund.de/secse/pages/teaching/index_de.shtml (oder http://jan.jurjens.de , Link: Lehre).

Einige Beispiel-Themen für Abschlussarbeiten

- Mustererkennung und Identifizierung in Versionlogs
- Aspektorientierte Realisierung von UMLsec-Stereotypen am Beispiel des Secure Links-Stereotyps
- Adaption von UMLsec-Stereotypen zur Modellierung von Anforderungen an die Datensicherheit nach UML 2.x
- Evolutionen und Co-Evolutionen für ReL
- Differenz-basierte Sicherheitsverifikation
- Entwicklung eines Prüfdokument-Generators am Beispiel von Sicherheitszertifizierungen
- Sicherheit und Evolution von Informationssystemen
- Automatisiertes Mapping von Event-Log-Bezeichnern auf Aktivitäten zur Unterstützung von Compliance-Analysen

Aktuelle Informationen unter:

http://www-secse.cs.tu-dortmund.de/secse/pages/teaching/thesis/index_de.shtml

(oder: Vorlesungswebseite, linke Spalte: „Bachelor/Master/Diplomarbeiten “).

Abschlussarbeiten auch in Zusammenhang mit **Forschungsprojekten** am Fraunhofer ISST oder LS 14 / TUD möglich.

Bei Interesse **bitte bei mir melden** !

Hiwi-Tätigkeiten

Hiwi-Jobs fortlaufend zu vergeben:

- Unterstützung von **Forschungsprojekten** (z.B.: Seconomics, SecVolution, ClouDAT):
  z.B. Java-Programmierung für UML-Analyse-Werkzeug, konzeptuelle Arbeiten zu modellbasierter Sicherheitsanalyse
- Unterstützung in der **Lehre** (Tutorien, Folienerstellung etc)

Weitere Informationen:

http://www-secse.cs.tu-dortmund.de/secse/pages/home/jobs_de.shtml

(oder: http://jan.jurjens.de ; rechte Spalte: HiWi-Stellen).

Auch Hiwi-Beschäftigung mit inhaltlichem Bezug zu **Abschlussarbeiten** möglich.

Bei Interesse **bitte bei mir melden** !

Auslandaufenthalte, Stipendien etc.

Vielfältige internationale Kontakte für Auslandsaufenthalte, z.B.:

- EU-Projekt Seconomics: Unis Trento (I), Aberdeen (UK), Madrid (S), Anadolu (TR); Firmen Atos Origin (F), National Grid (UK), Deep Blue (I), Barcelona Transport (S).

→ Bei Interesse **bitte bei mir melden**.

Als ehemaliger Geförderter: Vorschlagsrecht für Aufnahme in Förderung der Studienstiftung des deutschen Volkes.

→ Bei Interesse **bitte bei mir melden**.

Und danach ?

[http://www.fraunhofer.de/de/presse/presseinformationen/2013/Januar/fraunhofer-schafft-wieder-ueber-1000-neue-arbeitsplaetze---press.html]

"**Fraunhofer schafft wieder über 1000 neue Arbeitsplätze.** Fraunhofer wird auch 2013 wieder stark wachsen und über 1000 zusätzliche Stellen schaffen. ... »Unser Wachstum speist sich sowohl aus öffentlichen Förderprogrammen als auch aus Aufträgen mit der Wirtschaft. Das belegt, wie leistungsfähig und erfolgreich Fraunhofer am Forschungsmarkt agiert.«

[http://www.fraunhofer.de/de/presse/presseinformationen/2012/april/ertraege-aus-der-wirtschaft.html]

Fraunhofer ist ein beliebter Arbeitgeber. ... 86 Prozent der Mitarbeiterinnen und Mitarbeiter sind stolz darauf, bei Fraunhofer zu arbeiten. Im Durchschnitt sagen das in Deutschland nur 60 Prozent über ihren Arbeitgeber."

http://www.randstad-award.de/randstad-award-deutschland/presse/news/news/items/349.html

"**Der Randstad Award für den attraktivsten Arbeitgeber geht in diesem Jahr an die Fraunhofer-Gesellschaft.**"

→ Regelmäßig Einstellungsmöglichkeiten als wissenschaftlicher Mitarbeiter am Lehrstuhl oder am Fraunhofer ISST. Insbesondere Promotion möglich.

Agenda

- Organisatorisches
- Vorstellung des Fachgebietes
- **Vorlesungsinhalte**

Software Engineering is...

… the establishment and use of **sound engineering principles** in order to obtain **economically** software that is **reliable** and works **efficiently** on real machines. [NR68]

... the **systematic** approach to the **development**, **operation**, **maintenance**, and **retirement** of **software**.“ [IEEE83]

Defizite in der Praxis

Entwicklungsprozesse der Software-Industrie bei weiten nicht so planbar, zuverlässig, effektiv, effizient und flexibel wie die „althergebrachter“ Industrien:

- **Keine zuverlässige Herstellung** von Software im industriellen Maßstab.
- Kosten- und Terminüberschreitungen.
- Bei Auslieferung: **ungenügende Softwarereife.**
- **Keine Produktivitätskontrolle** wie in anderen industriellen Fertigungsbereichen.
- **Keine Qualitätskontrolle** wie in anderen industriellen Fertigungsbereichen, gerade das Testen ist immer noch unterbewertet.

Probleme: Verspätung, Abbruch

LEHRSTUHL 14
SOFTWARE ENGINEERING

Anzahl Function	Früher als geplant	Termingerecht	Verspätet	Abgebrochen
Points1 FP	14,86%	83,16%	1,92%	0,25%
10 FP	11,08%	81,25%	5,67%	2,00%
100 FP	6,06%	74,77%	11,83%	7,33%
1.000 FP	1,24%	60,76%	17,67%	20,33%
10.000 FP	0,14%	28,03%	23,83%	48,00%
100.000 FP	0,00%	13,67%	21,33%	65,00%
Durchschnitt	5,53%	56,94%	13,71%	23,82%

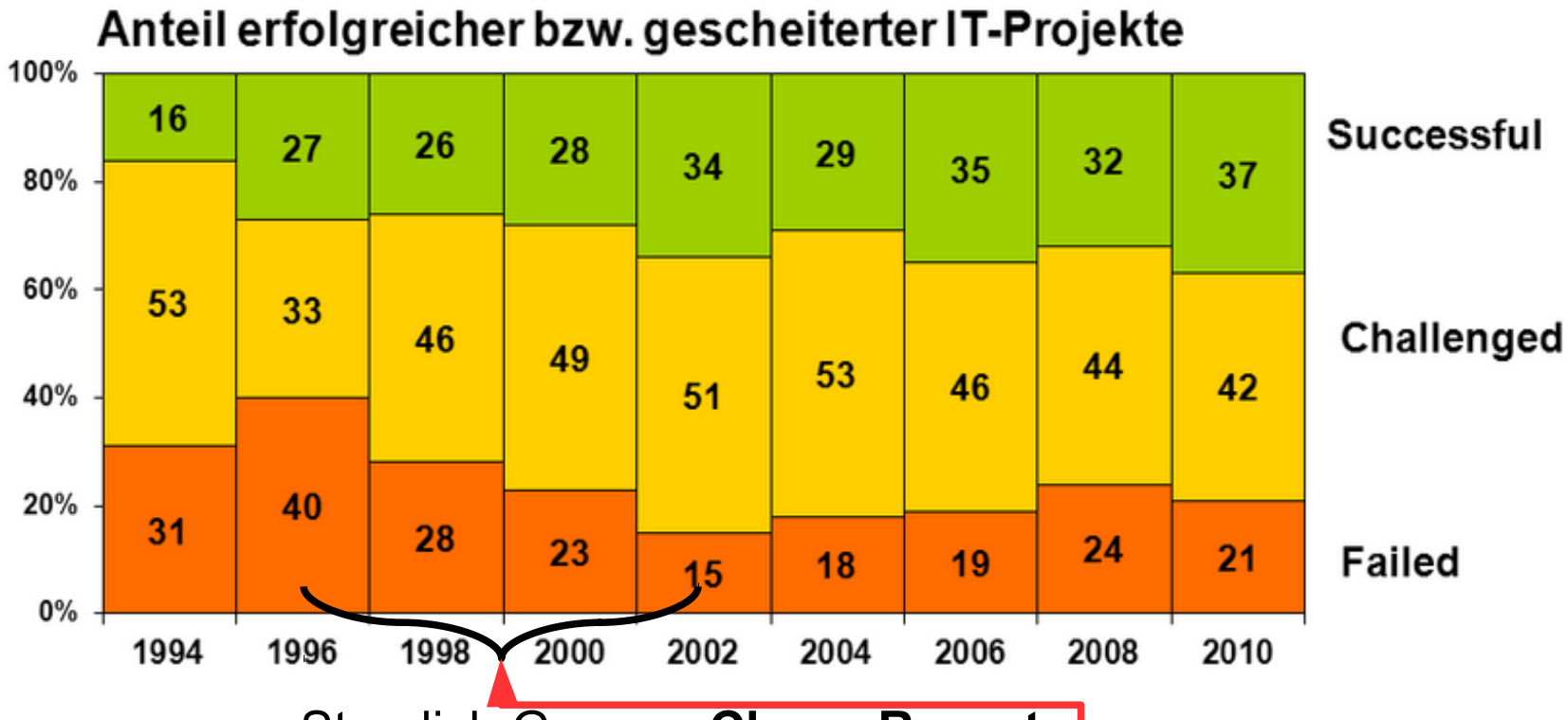
Abbruchrate großer Projekte nach [Jon96]

Andere Quelle: **Erfolgreich** durchgeführte **Software-Projekte**:

- 16% im Jahre 1994; 34% im Jahre 2003

1 Function Point (FP): ca. 55 Lines of Code (LOC) in C++/Java, 20 LOC Perl, 13 LOC SQL, etc.
Desktop-Projekt mit 1 FP dauert ca. eine Woche mit einem Entwickler
Allgemeines Projekt mit 100.000 FP und 100 Entwicklern: ca. 17 Monate

Probleme: … aktuellere Zahlen



Standish Group – **Chaos Report**

Erfolgreich durchgeführte **Software-Projekte**:

- 35% seit 15 Jahren

Gescheiterte Software-Projekte:

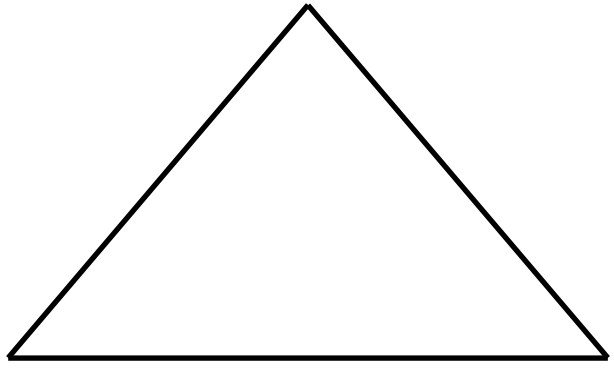
- 20%

Abnahme gescheiterter Projekte: Geänderte Kriterien

Quelle: http://www.ikmt.de/public/homepage.htm#de/service/management-forum.htm#showthread.php@&tid=334&pid=#pid

Interessenkonflikte: Termine, Kosten, Qualität

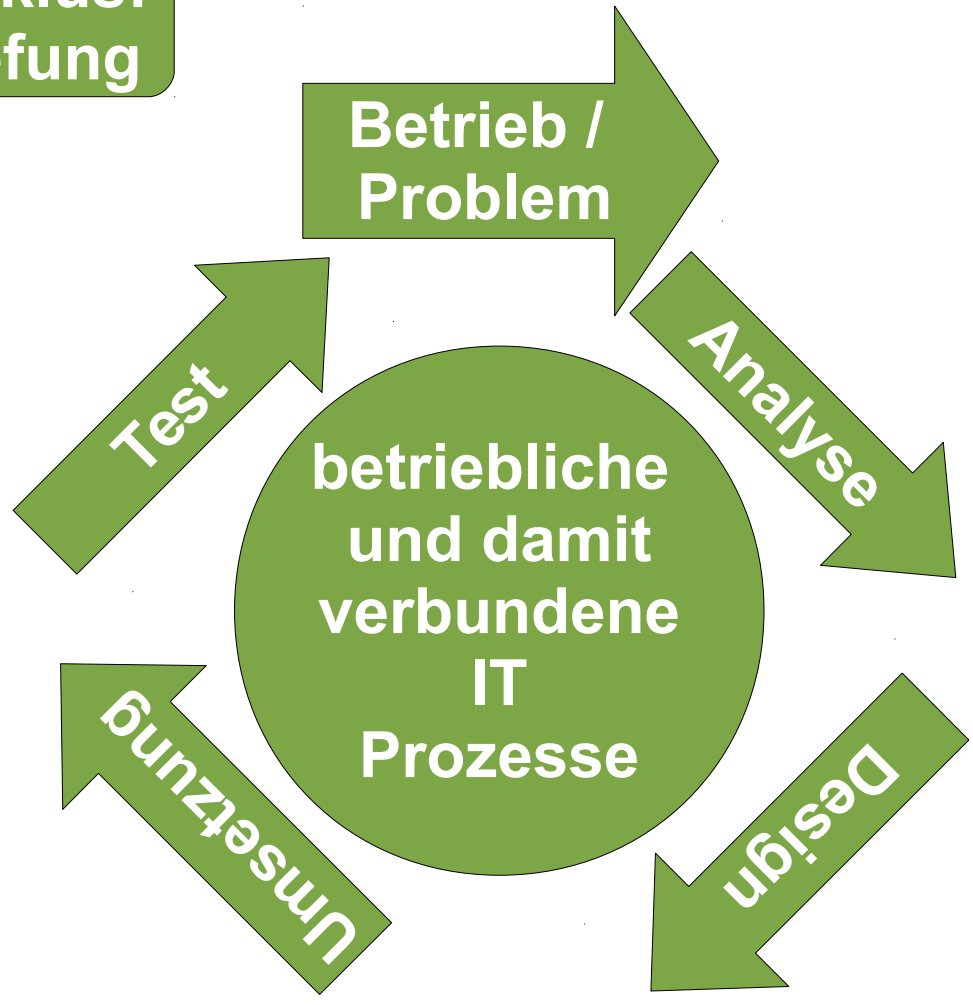
Qualität: korrekt
(funktionsfähig)



Termin:
rechtzeitig

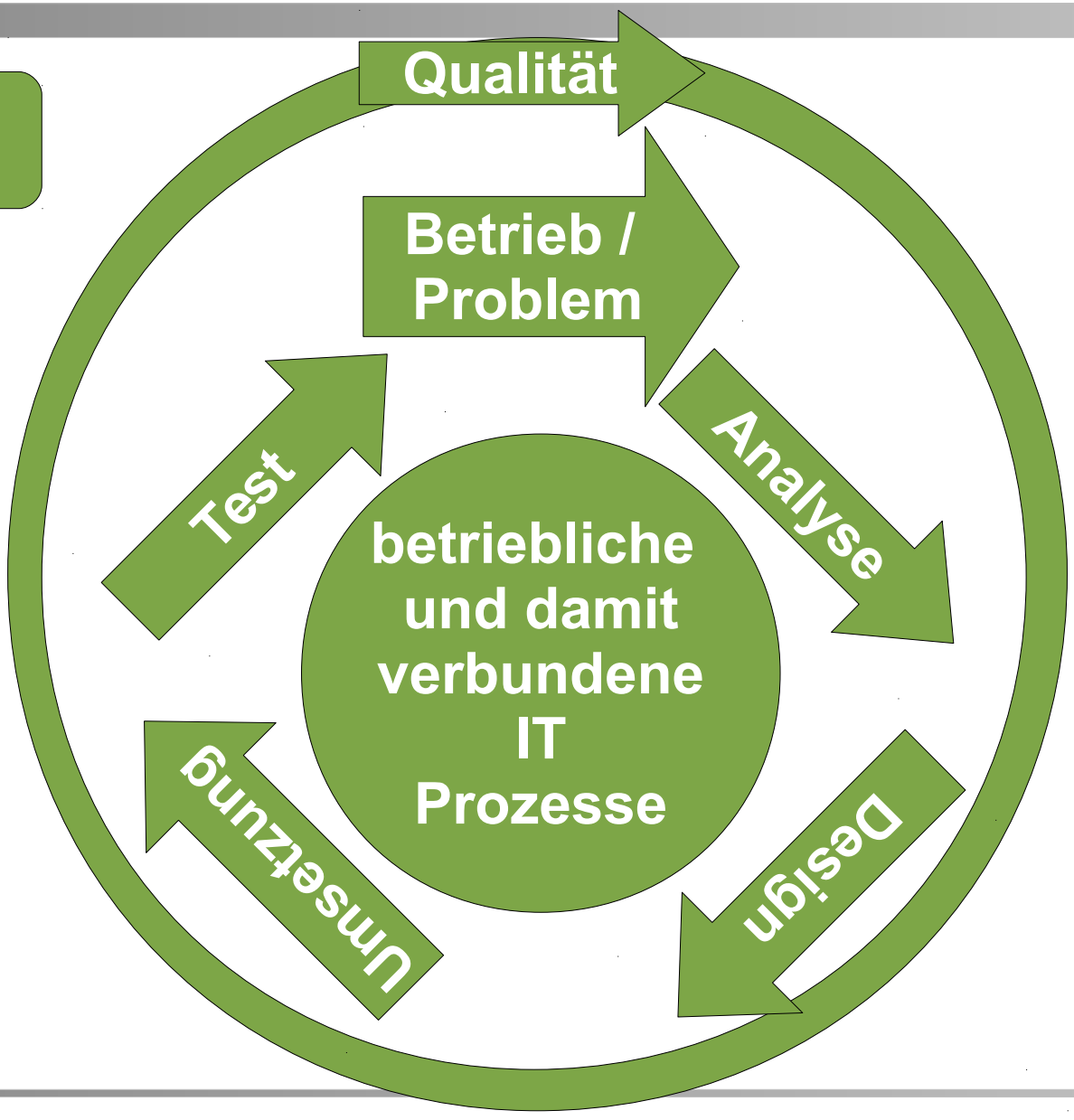
Kosten:
im Budget

Der SE Lebenszyklus: Punktuelle Vertiefung



Vorlesungsüberblick
Inhaltlicher Zusammenhang

Umfassend: Qualität



Themen der Vorlesung

Kapitel 1: Modellbasierte Softwareentwicklung

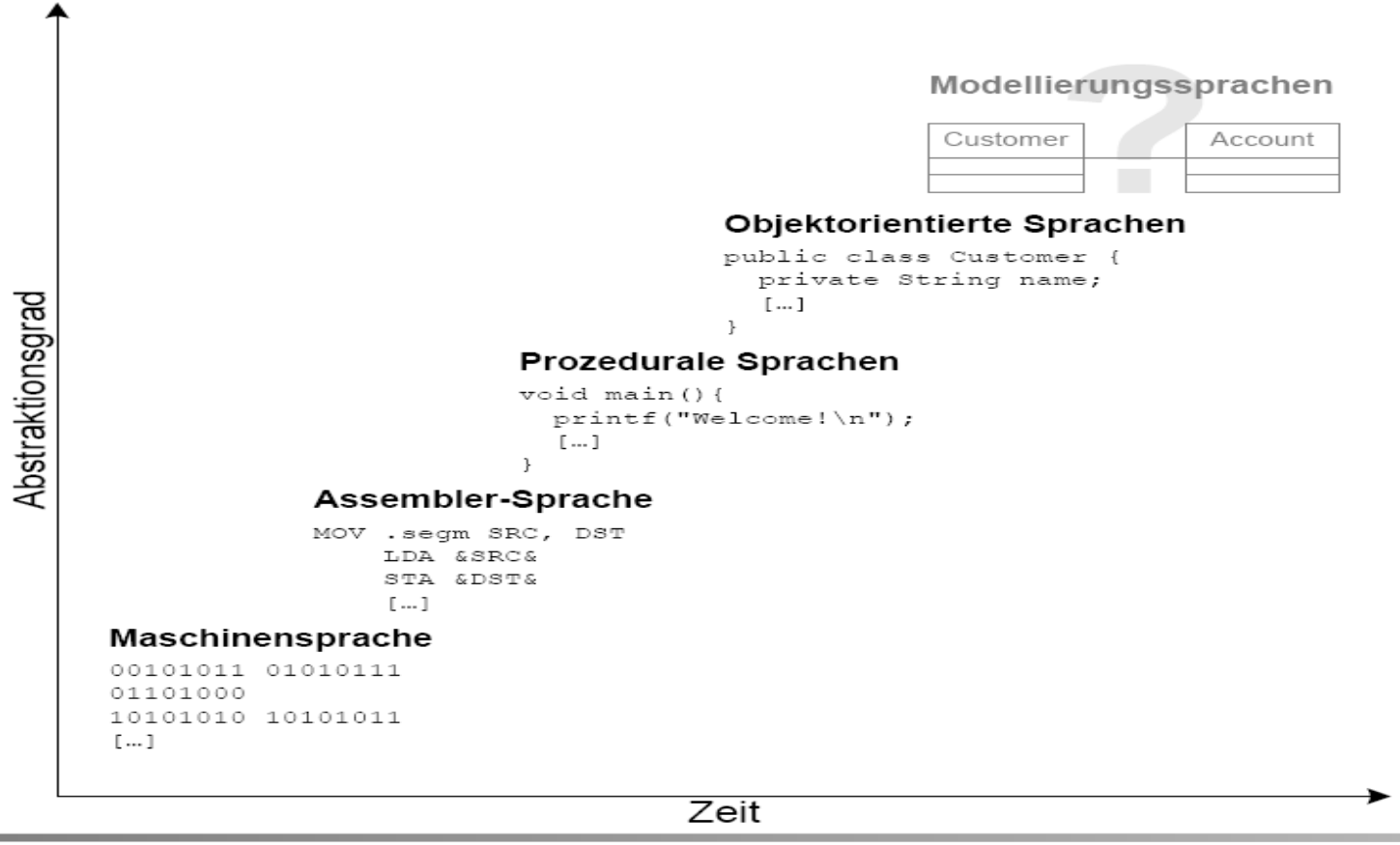
* Teil 1.0: Modellbasierte Softwareentwicklung: Einführung
* Teil 1.1: Modellbasierte Softwareentwicklung
* Teil 1.2: Grundlagen Object Constraint Language (OCL)
* Teil 1.3: Ereignisgesteuerte Prozessketten (EPK)
* Teil 1.4: Petrinetze
* Teil 1.5: Eclipse Modeling Foundation (EMF)

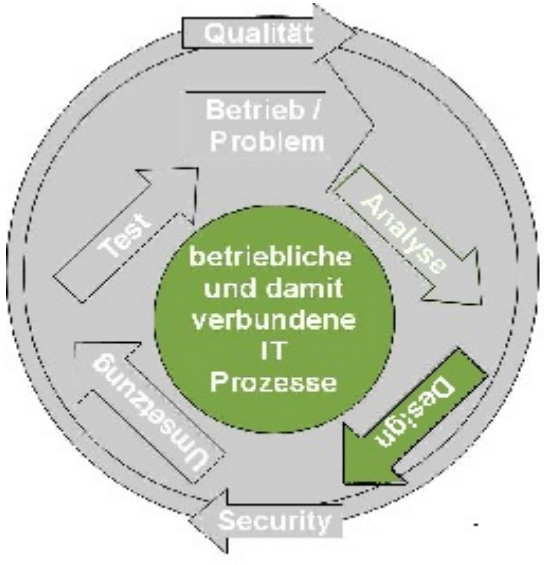
Kapitel 2: Qualitätsmanagement

* Teil 2.0: Einführung Qualitätsmanagement
* Teil 2.1: Grundlagen der Softwareverifikation
* Teil 2.2: Softwaremetriken
* Teil 2.3: Black Box Test
* Teil 2.4: White Box Test
* Teil 2.5: Testen im Softwarelebenszyklus

Teil 1.0: Modellgetriebene SW-Entwicklung: Einführung

- **Modelle und Modellierung: Begriffe**
- **UML: Wiederholung**
- **Szenarien der Modellgetriebenen SW-Entwicklung**

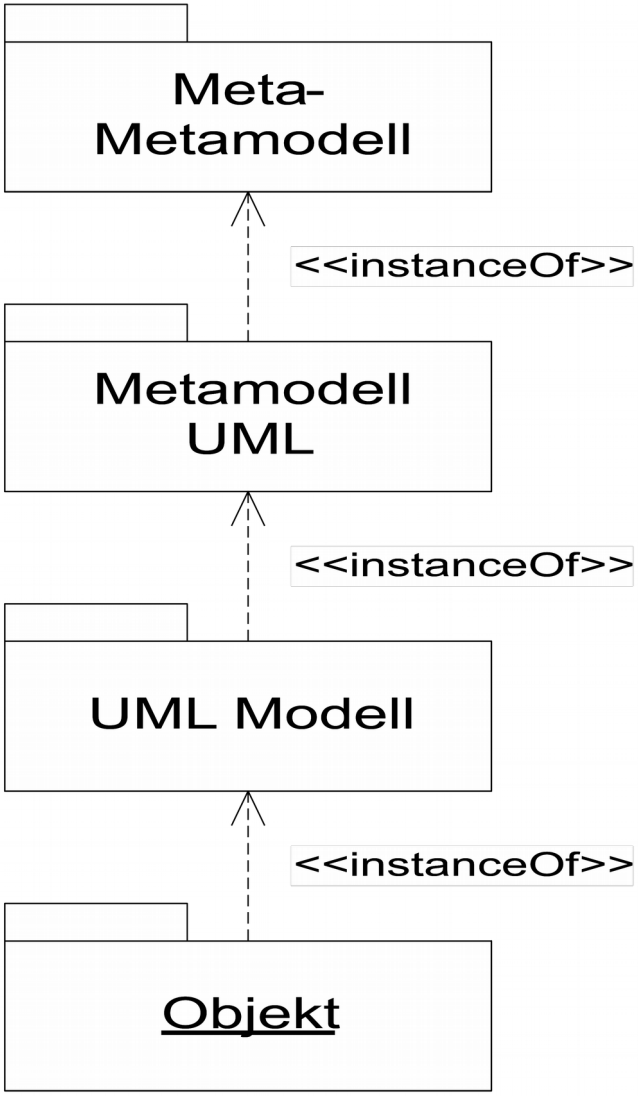


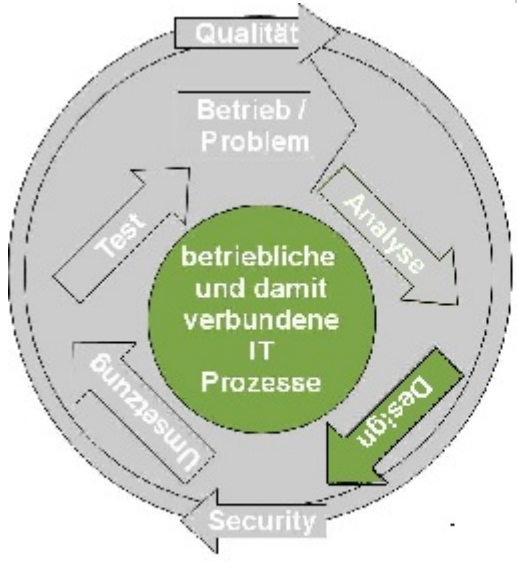


Teil 1.1: Modellbasierte SW-Entwicklung

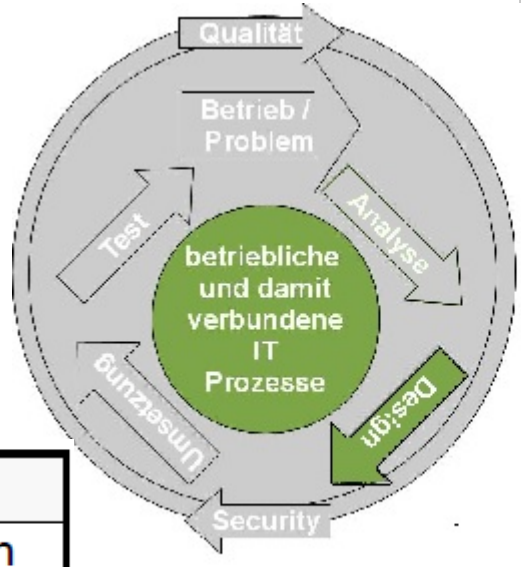
UML schon bekannt aus SWT und SoPra. Hier neu:

- **Metamodellierung**
- **Spezifikation der UML**
- **Model-Driven Architecture**



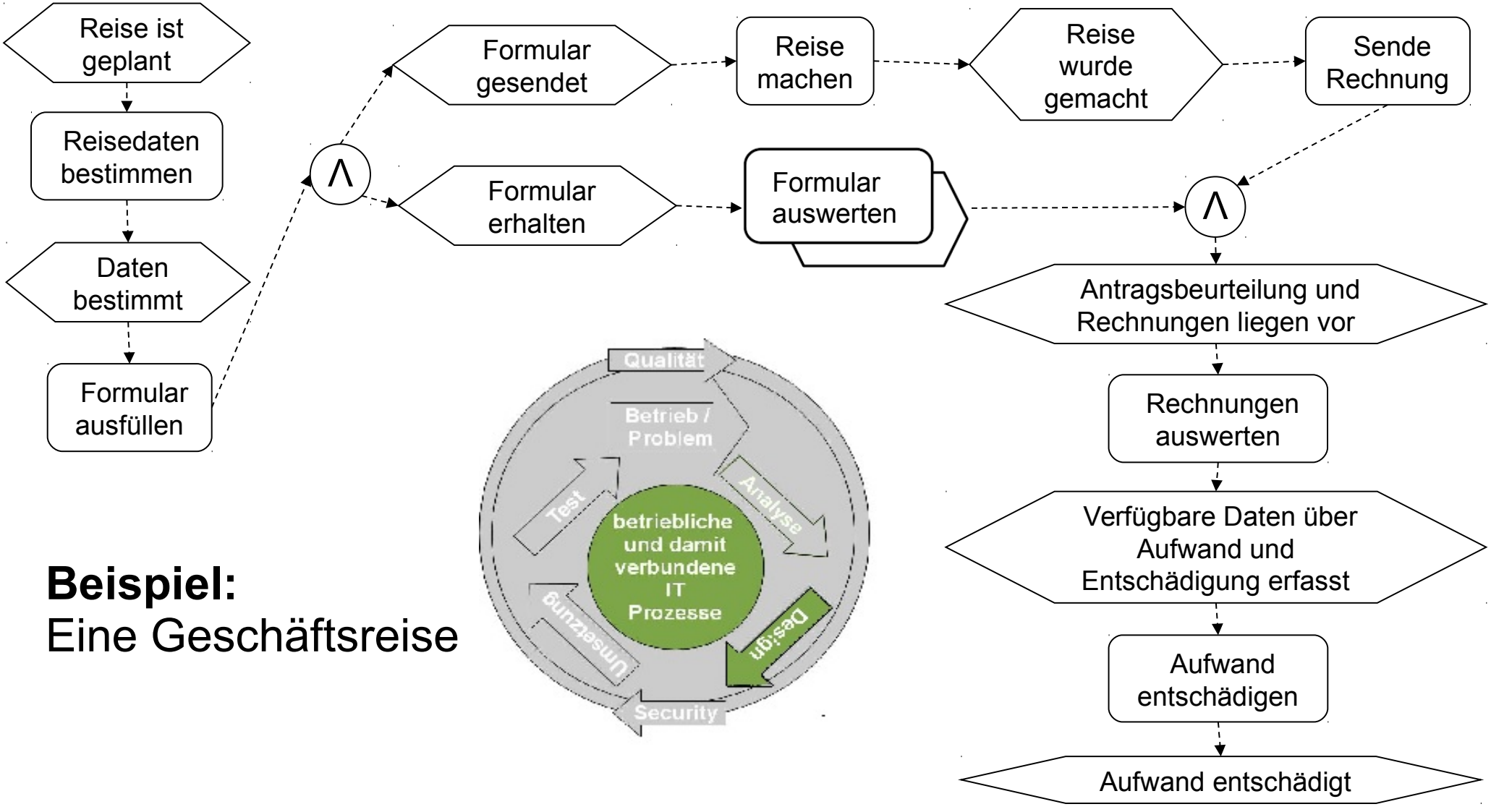


Teil 1.2: Object Constraint Language (OCL)

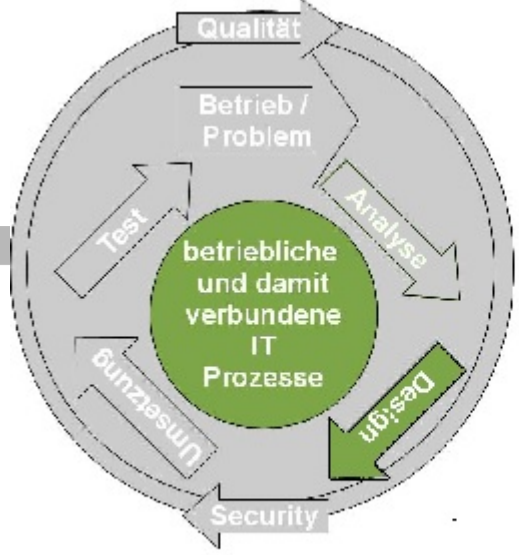


Operation	Description
hasReturned() : Boolean	True if type of template parameter is an operation call, and the called operation has returned a value.
result()	Returns the result of the called operation, if type of template parameter is an operation call, and the called operation has returned a value.
isSignalSent() : Boolean	Returns true if the OclMessage represents the sending of a UML Signal.
isOperationCall() : Boolean	Returns true if the OclMessage represents the sending of a UML Operation call.
parameterName	The value of the message parameter.

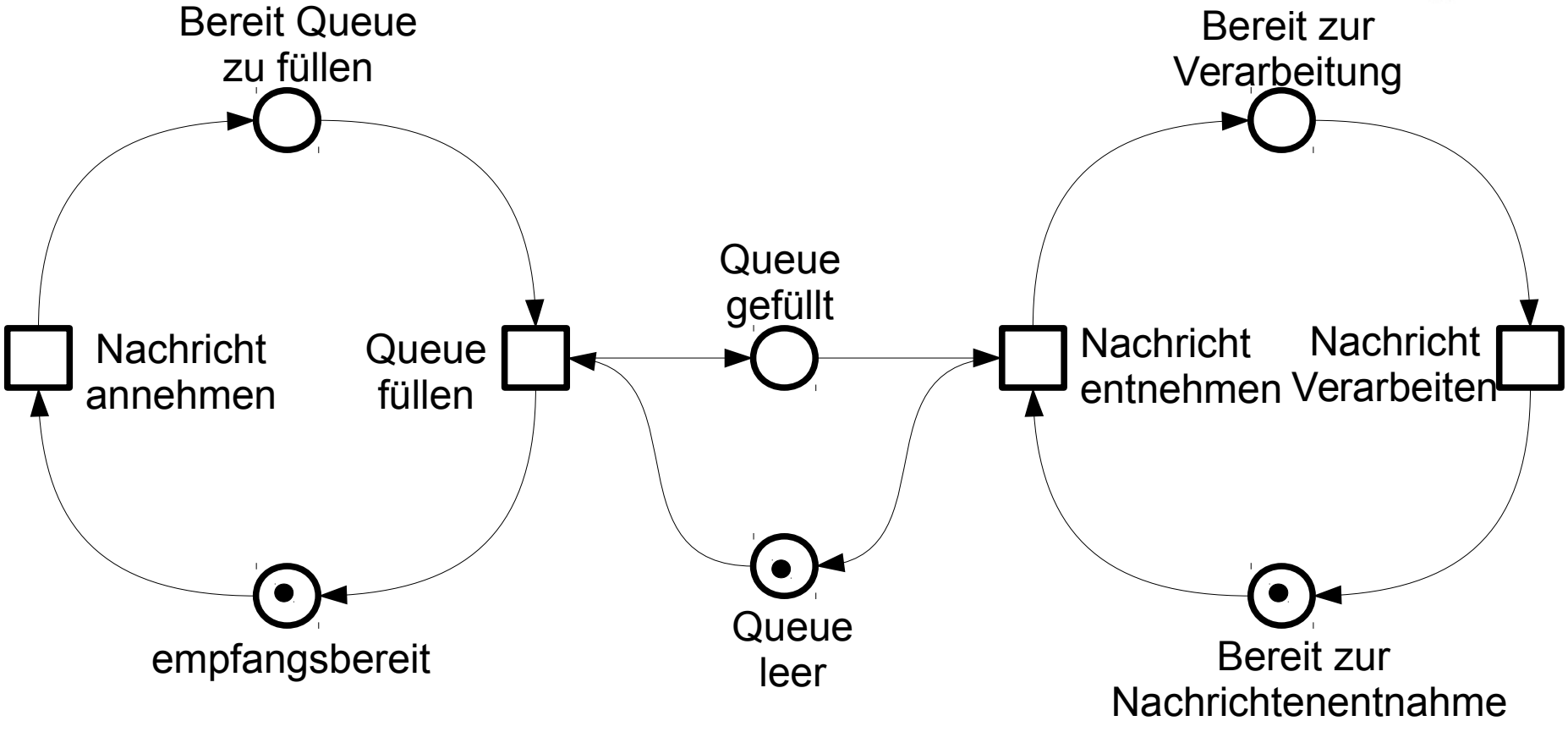
Teil 1.3: Ereignisgesteuerte Prozessketten (EPK)



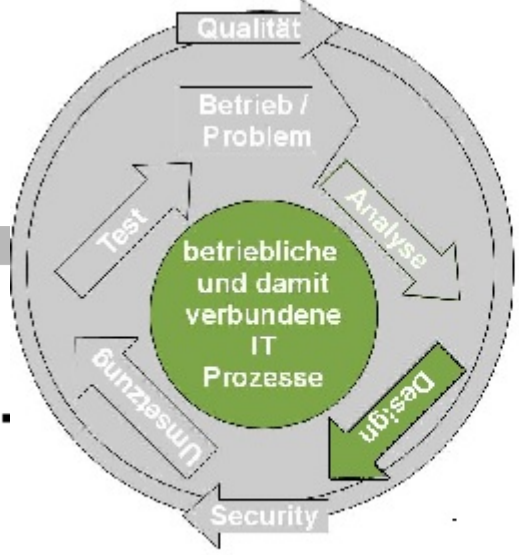
Beispiel:
Eine Geschäftsreise



Beispiel: Nachrichten-Queue



Teil 1.5: Eclipse Modeling Foundation (EMF)



	OMG	EMF
M3 (Metametamodel)	MOF	Ecore (Java classes)
M2 (Metamodel)	UML, CWM, ...	EMF model
M1 (Model)		

EMF

instanceOf

EMF.Edit

EMF codegen

EMF.Codegen

EMF

Java code for manipulation and default serialization of EMF models

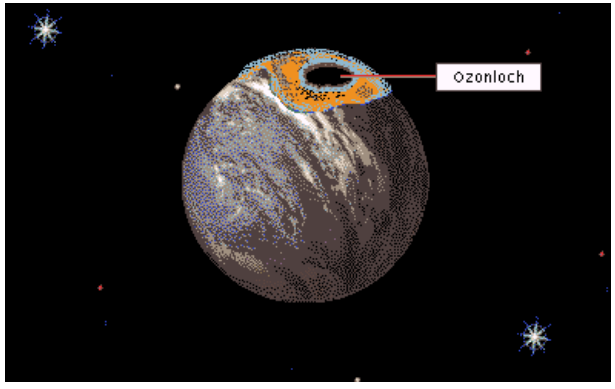
instanceOf

EMF model

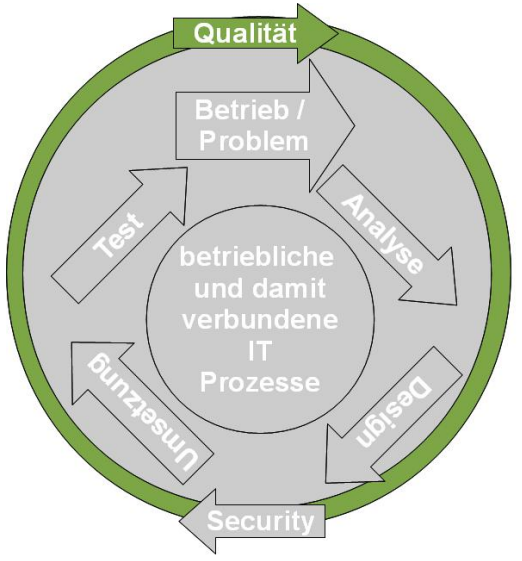
EMF.Edit

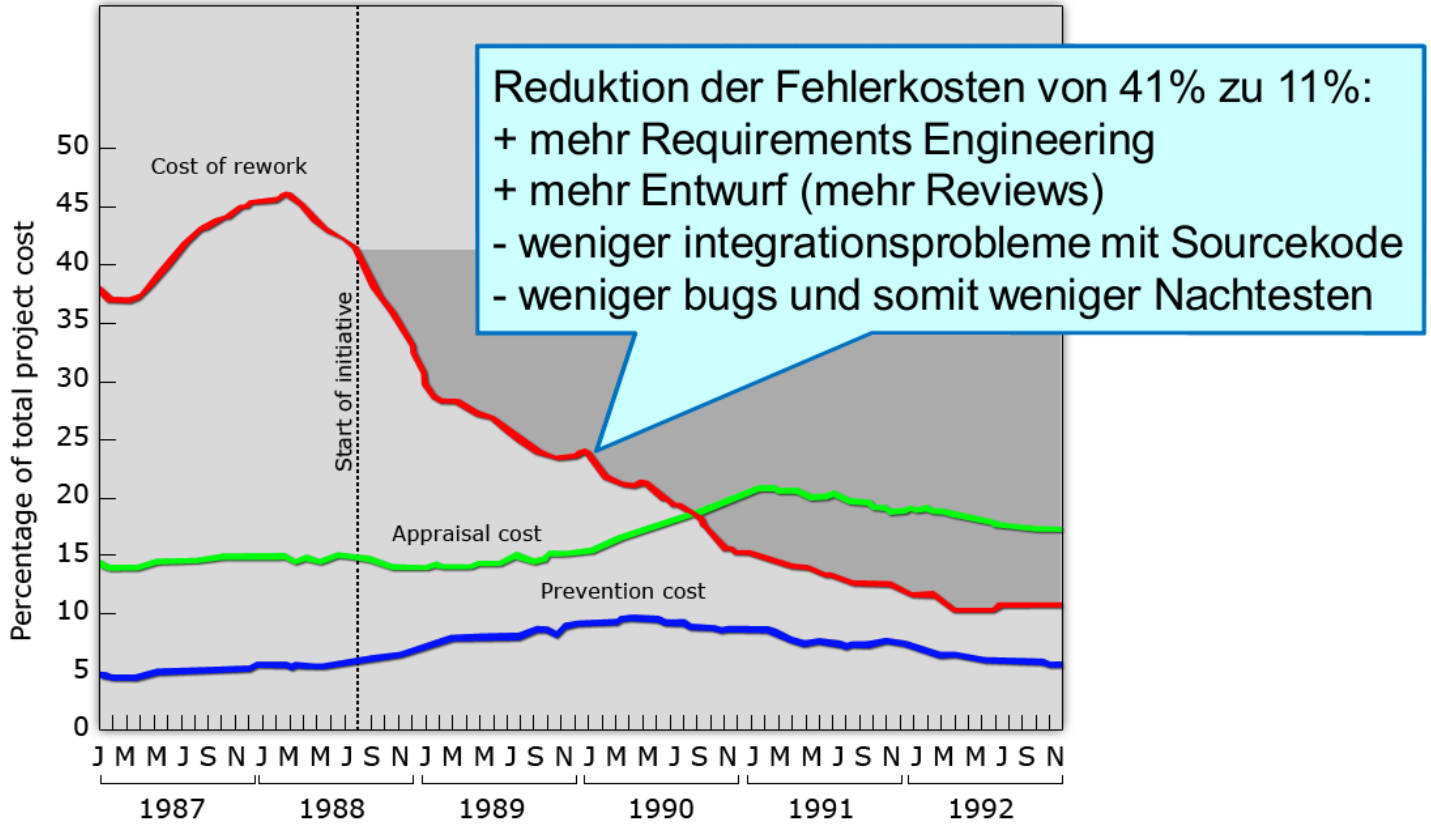
Beispiele für Auswirkungen von Software-Fehlern:

- **NASA - Erdbeobachtungssatelliten** 1979-1985: **Ozonloch** 7 Jahre (!) lang **nicht erkannt.**
- ESA, Kourou, Franz. Guyana, 4. Juni 1996: **Selbstzerstörung der Ariane** 5 beim Jungfernflug 39 Sekunden nach Start.
- Bemannte **NASA-Raumkapsel** Gemini V: **Verfehlte ihren Landeplatz** um ca. 160 Kilometer.



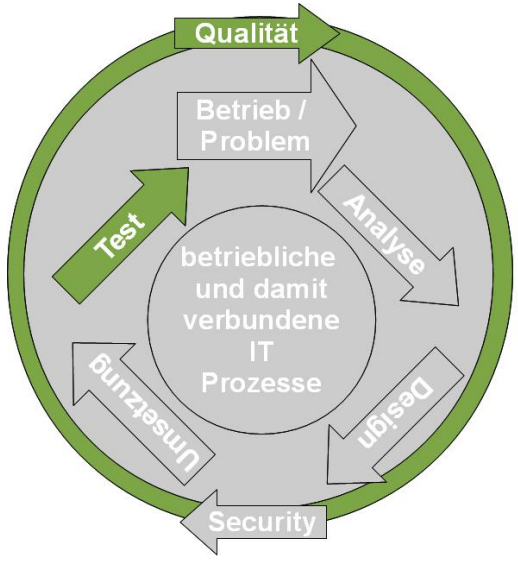
Teil 2.1: Grundlagen der Softwareverifikation

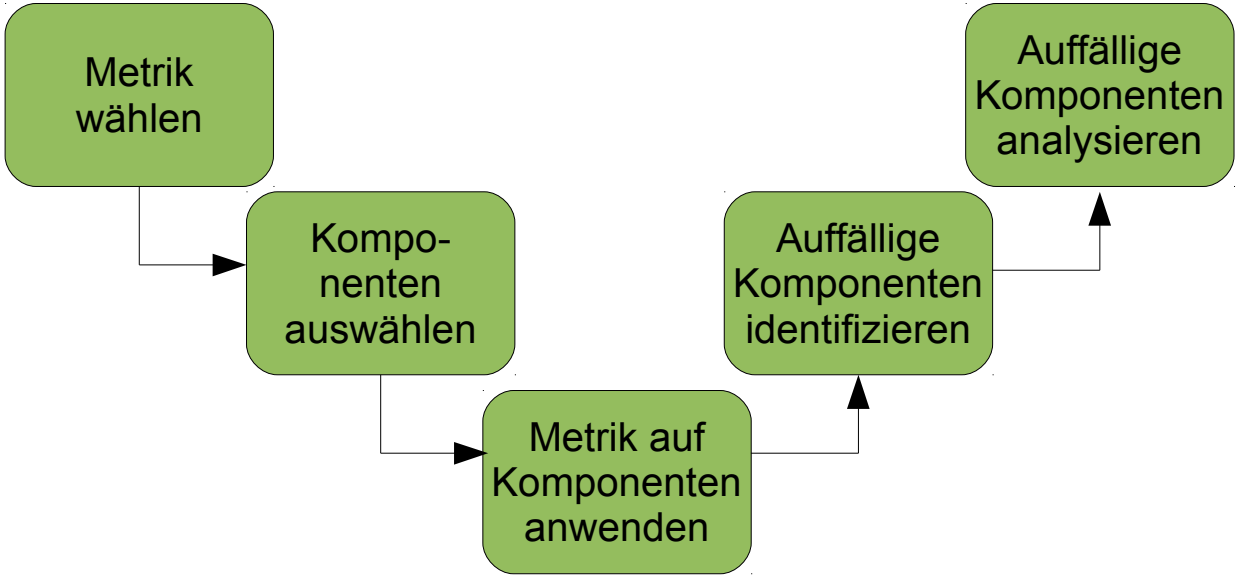




Teil 2.2: Softwaremetriken

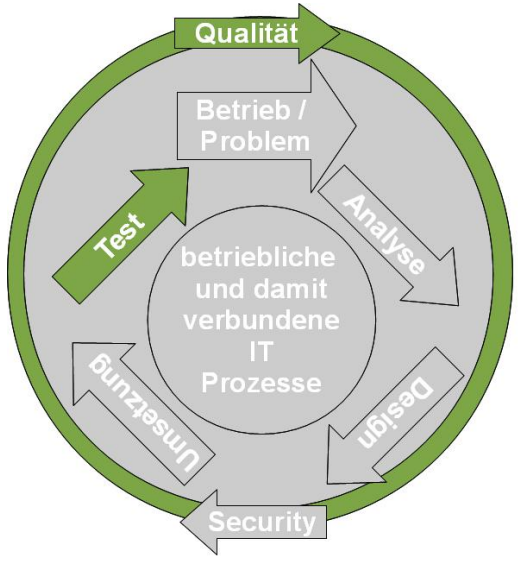
- Metriken
- Direktes und indirektes Messen
- Vorgehensweisen
- Effekte

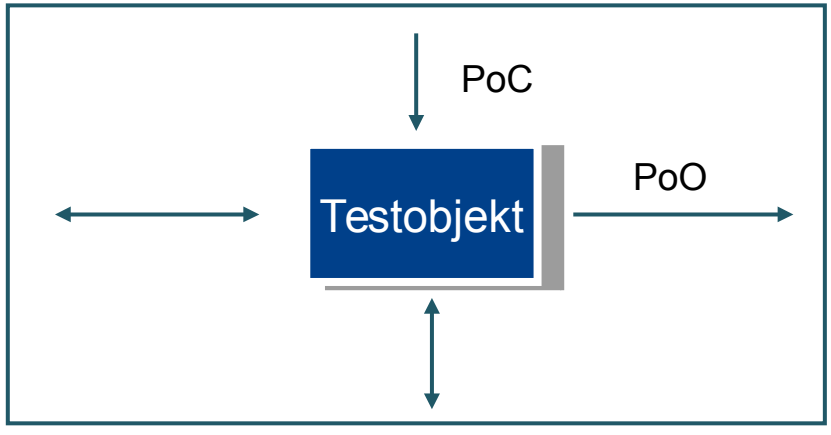




Äquivalenzklassen schon bekannt aus SWT. Hier neu:

- **Grenzwertanalyse**
- **zustandbasiertes Testen**





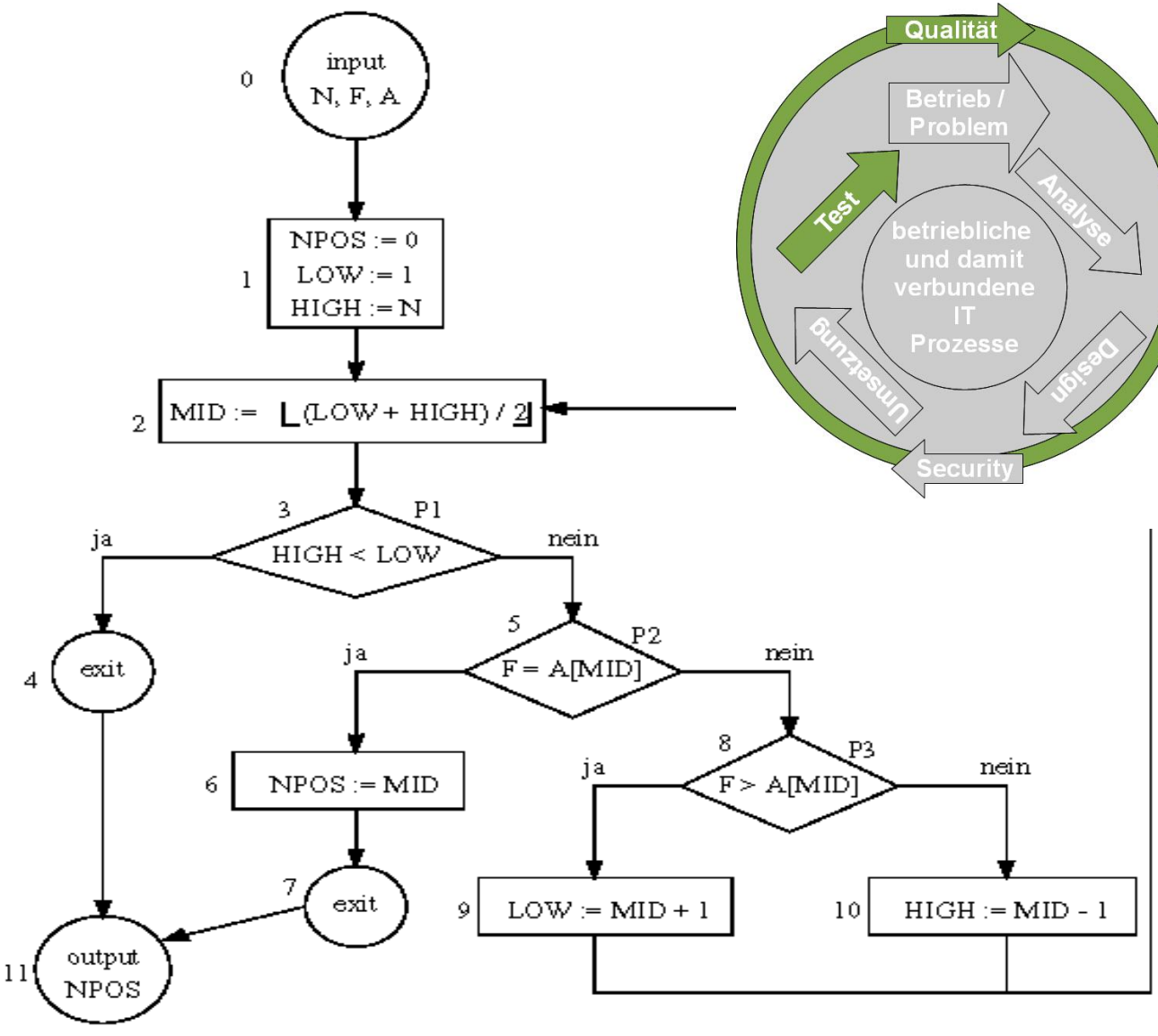
Teil 2.4: White-Box-Test

Daten- & Kontrollfluss-basiertes Testen.

Anweisungs-, Zweig-, Pfad-, und **Mehrfach-bedingungs-überdeckung** schon aus SWT bekannt.

Hier neu:

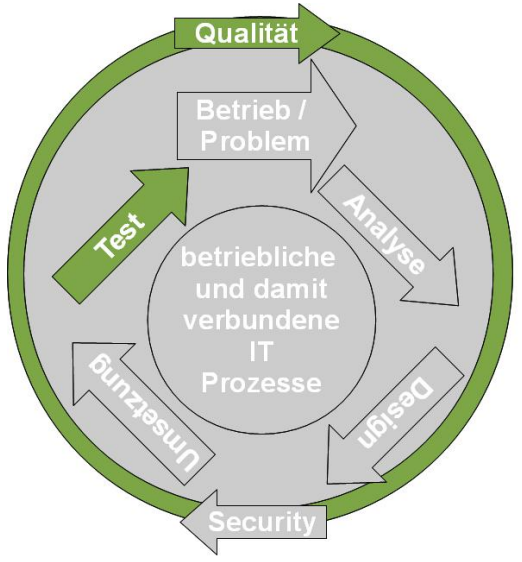
- **datenflussbasierte Testen**
- **statische Analyse**

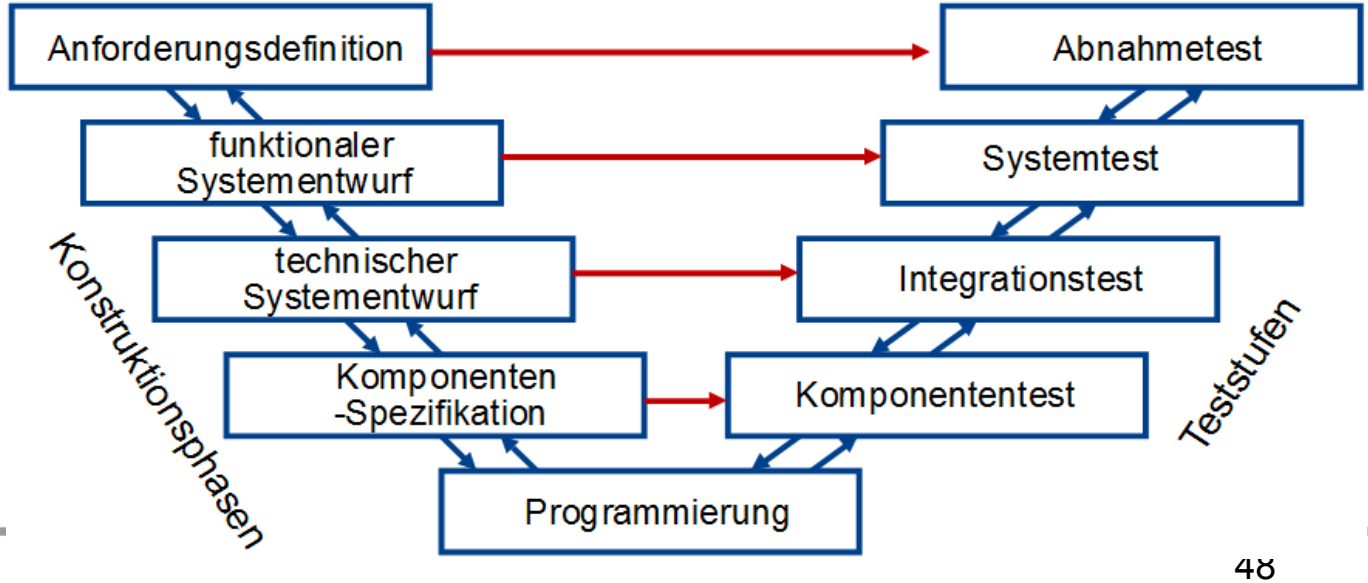


Teil 2.5: Testen im Softwarelebenszyklus

Programmtest und **Klassentest** schon aus SWT bekannt. Hier neu:

- weitere Testarten (**Komponenten-, Integrations-, System- und Abnahmetest**)
- dazugehörige **(Test-)Strategien**.





Voraussetzungen

Was Sie aus „Softwaretechnik“ und dem „Software-Praktikum“ wissen sollten:

- Qualitätsmerkmale von Software
- V-Modell
  - Testverfahren auf den verschiedenen Ebenen
  - Programmtest, Klassentest, Komponententest, …
- UML-Diagramme
  - Klassendiagramme
  - Aktivitätsdiagramme
  - ...

Literatur: Kapitel 1

Teil 1.0-1.2:

- J. Ludewig, H. Lichter: **Software Engineering - Grundlagen, Menschen, Prozesse, Techniken**, dpunkt.verlag, 3. Auflage, 2013. https://www2.swc.rwth-aachen.de/se_buch . **Ab 33 EUR (e-book)**: http://www.dpunkt.de/buecher/4501/software-engineering.html **Unibibliothek**: http://www.ub.tu-dortmund.de/katalog/titel/1416968.
- V. Gruhn, D. Pieper, C. Röttgers: **MDA – Effektives Software-Engieneering mit UML2 und Eclipse**, Xpert.press/Springer-Verlag, 2006. Unibibliothek: http://www.ub.tu-dortmund.de/katalog/titel/1223129
- J. Seemann, J.W. Gudenberg: **Software-Entwurf mit UML2**, Xpert.press/ Springer-Verlag, 2006. Unibibliothek http://www.ub.tu-dortmund.de/katalog/titel/1223020

(Links s. a. Vorlesungswebseite)

Literatur: Kapitel 1

Teil 1.2:

- J. Warmer, A. Kleppe: **The Object Constraint Language: Getting Your Models Ready for MDA**, Addison-Wesley Longman Publ. & Co., Inc., 2003. Unibibliothek: http://www.ub.tu-dortmund.de/katalog/titel/901443
- Object Management Group: **OCL 2.4** http://www.omg.org/spec/OCL/2.4/PDF/

Teil 1.3:

- J. Becker, C. Mathas, A. Winkelmann: **Geschäftsprozessmanagement**. Springer-Verl., 2009. Unibib.: http://www.ub.tu-dortmund.de/katalog/titel/1256897

Teil 1.4:

- H. Reisg: **Petrinetze**. Vieweg, 2010. Teil I. Unibibliothek: http://www.ub.tu-dortmund.de/katalog/titel/1305786

Teil 1.5:

- D. Steinberg: **Eclipse Modeling Framework**, Addison-Wesley, 2008.

Literatur: Kapitel 2

- E. Riedemann: **Testmethoden für sequentielle und nebenläufige Software-Systeme**. Teubner, Stuttgart, 512 S., 1997. PDF herunterladbar von Vorlesungs-Webseite. Unibibliothek: http://www.ub.tu-dortmund.de/katalog/titel/687299
- A. Spillner, T. Linz: **Basiswissen Softwaretest**. 4., überarbeitete Auflage, dpunkt.verlag, 2010, 308 Seiten, 39 Euro (D), ISBN 987-3-89864-642-0 Unibibliothek: http://www.ub.tu-dortmund.de/katalog/titel/1287855.
- Bei Engpässen in der Ausleihe alternativ ähnliche Inhalte als e-Book: A. Spillner, T. Linz, H. Schaefer: **Software Testing Foundations**, 3. Auflage, Rocky Nook, 2011, 296 Seiten, Print ISBN-13: 978-1-933952-78-9 Unibibliothek (**e-Book**): http://www.ub.tu-dortmund.de/katalog/titel/1409780

Agenda

- **Organisatorisches**
  - Studienordnung: Einordnung / Kompetenzen / Struktur / Prüfung
  - Vorlesung: Bildungsvertrag, Termine, Feedback
  - Übung: Konzept / Termine
  - Klausur
- **Vorstellung des Fachgebietes**
  - Forschung
  - Abschlussarbeiten, Hiwi-Jobs, weiteres Lehrangebot
- **Vorlesungsinhalte**

→ **NOCH FRAGEN ?**