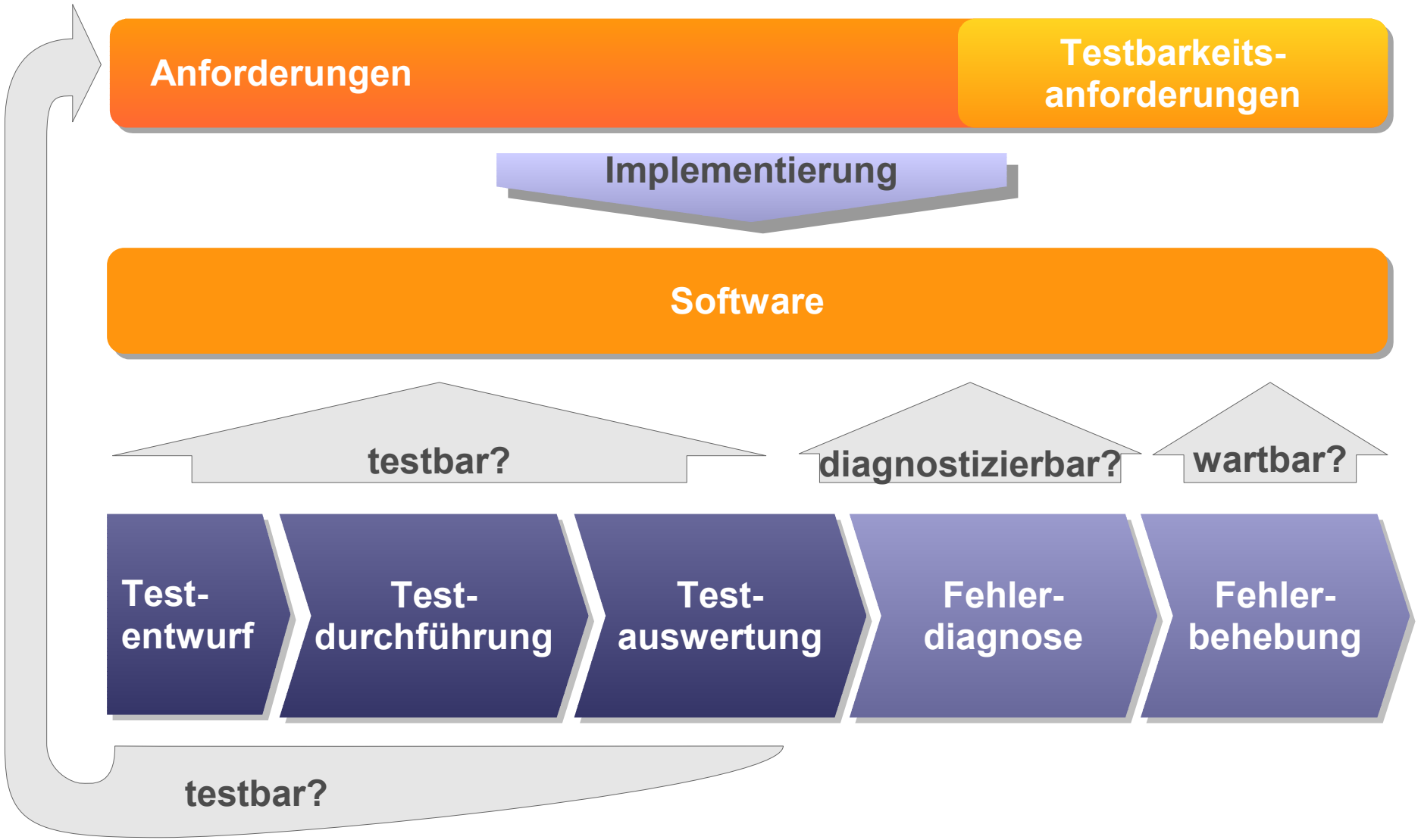
Testbarkeitsanforderungen an die Software

Dr. Stefan Jungmayr
TAV 27, Bad Honnef, 5. Juni 2008

www.testbarkeit.de

There are two ways to write error-free programs; only the third works.
[Alan J. Perlis]

Begriffe und Fokus

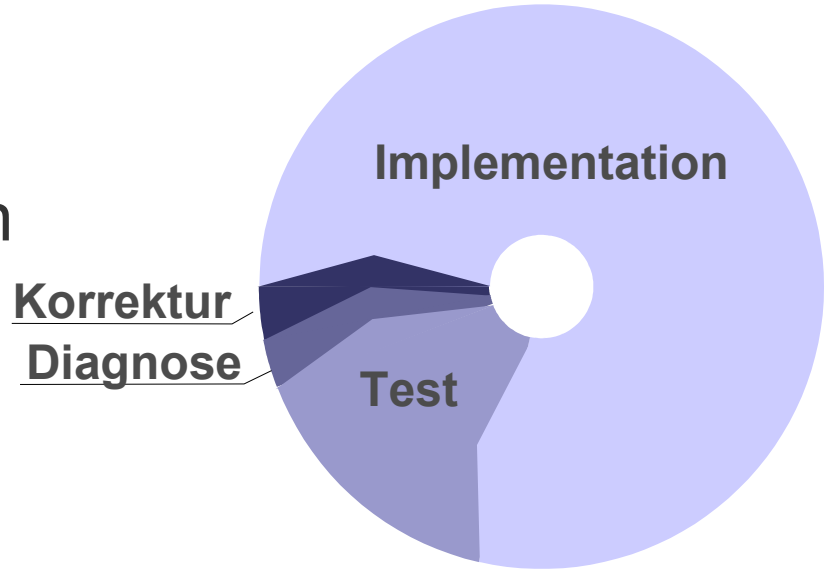


These 1:

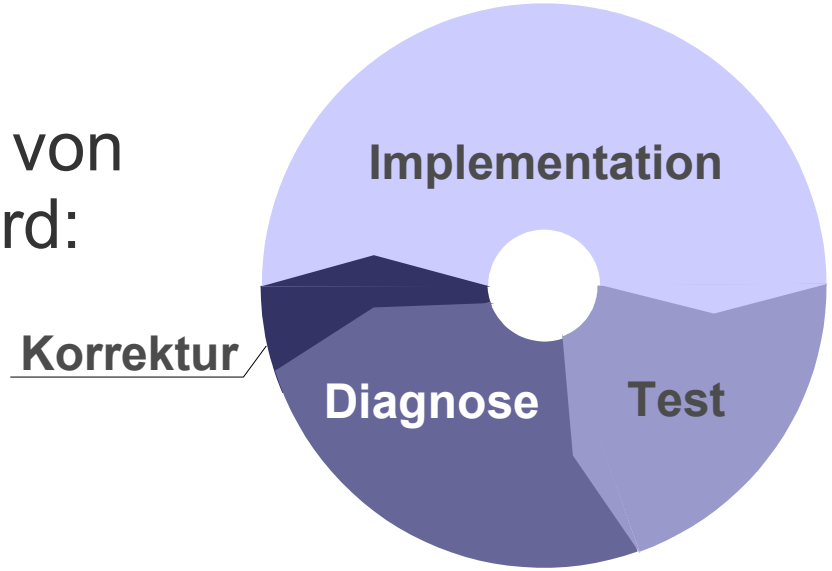
„Testbarkeit birgt Einsparungspotential am Gesamtentwicklungsaufwand von ca. 10%”.

Aufwand durch Probleme bei Test und Diagnose (I)

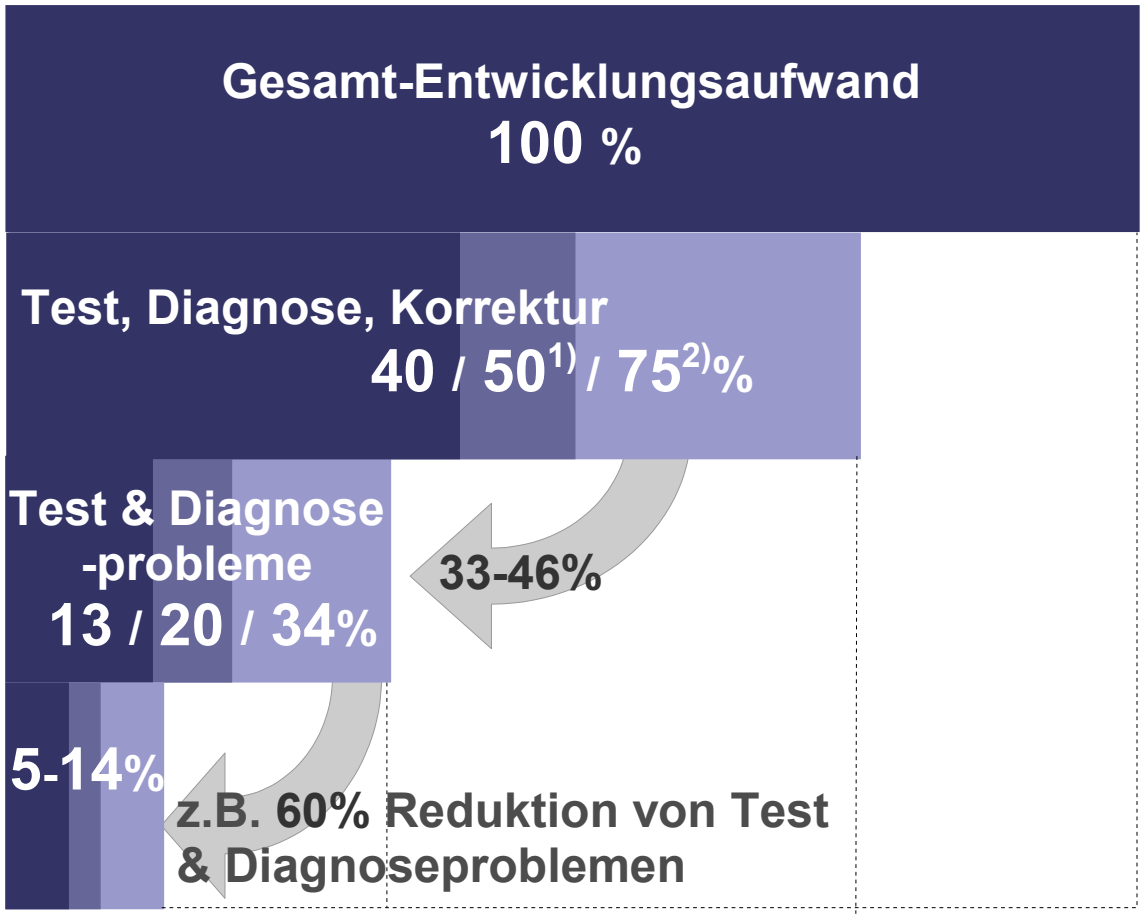
Implementierungszyklus, wie ihn das Management vermutet:



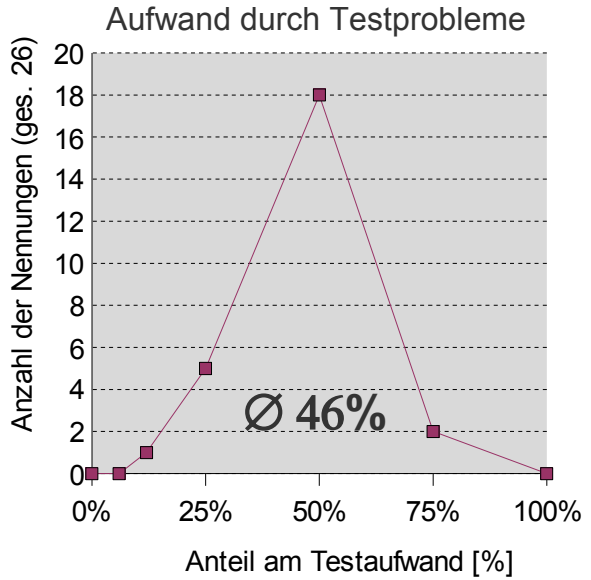
Implementierungszyklus, wie er von den Entwicklern erlebt/gelebt wird:

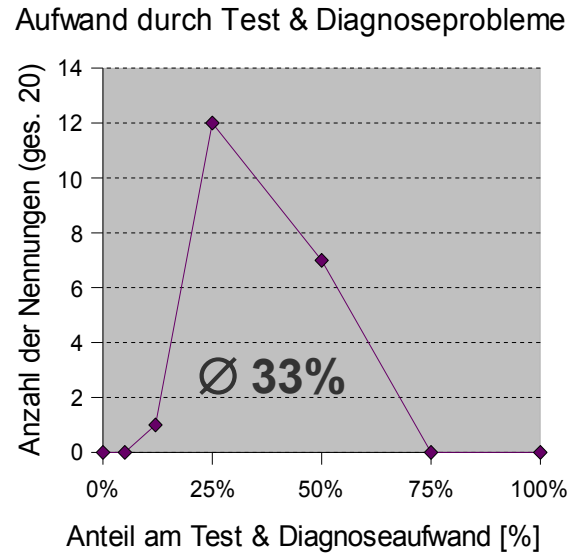


Aufwand durch Probleme bei Test und Diagnose (II)



[1] [Beiz90], [2] [Hail02]





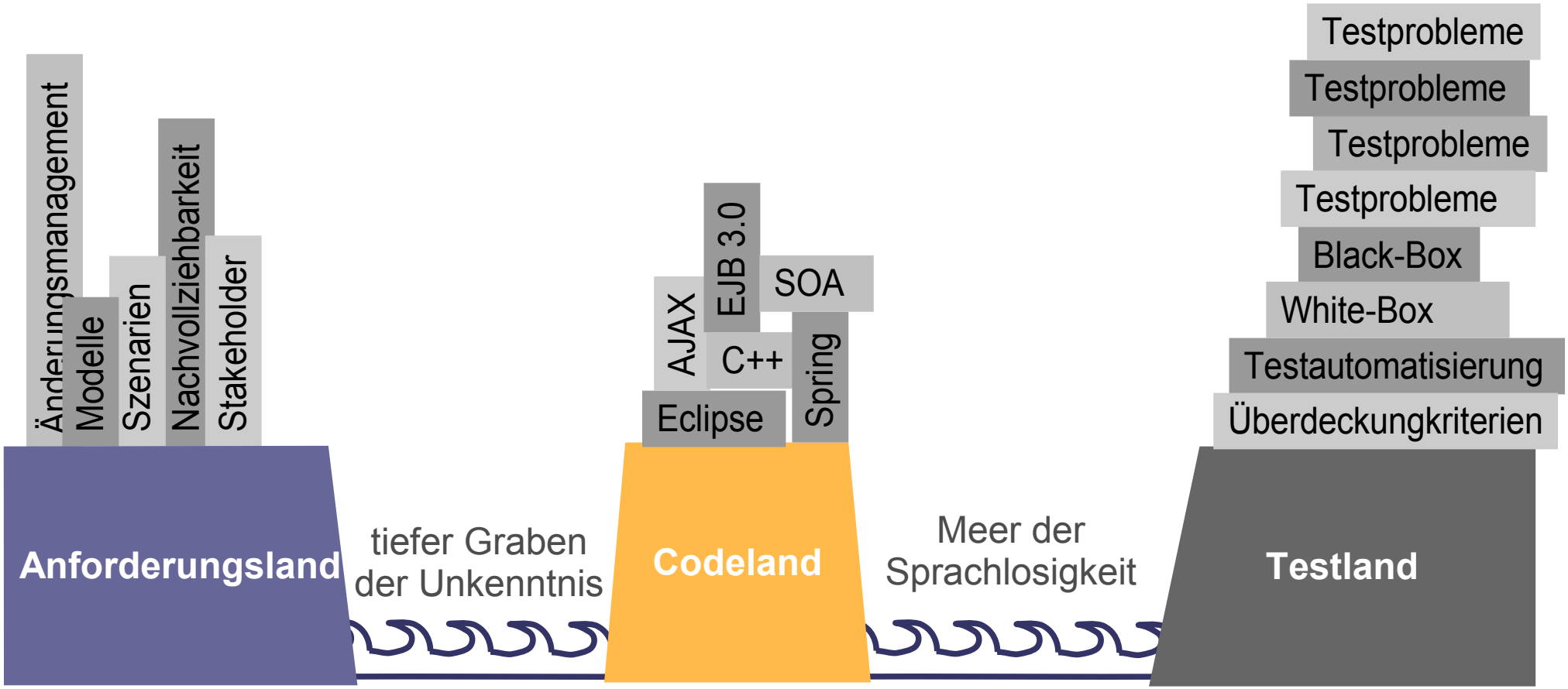
Aufwand durch Probleme bei Test und Diagnose (III)

„After visiting hundreds of companies and chatting with thousands of developers I'm left with a qualitative sense that code debug burns about half the total engineering time for most products.
That suggests minimizing debugging is the first place to focus our schedule optimization efforts." [Gans01]

These 2

“Testbarkeitsanforderungen werden unzureichend formuliert“

Arbeitsteilung und ihre Wirkung



Testbarkeitsanforderungen von Testern

- Verständlichkeit
- angemessene Komplexität
- Kontrollierbarkeit (Steuerbarkeit)
- Beobachtbarkeit
- Isolierbarkeit
- Automatisierbarkeit
- Robustheit (der Software)
- Reproduzierbarkeit
- Lokalisierbarkeit
- Anonymisierbarkeit

Testbarkeitsanforderungen von Entwicklern

- Ausgabe von Fehlermeldungen
- nachvollziehbare Fehlermeldungen
- nachvollziehbarer Systemzustand
- konfigurierbares Logging
- selbstdefinierte Abfragen
- Fehlerstatistiken
- Trennung von Fehlermanagement und Geschäftslogik
- einfache Fehlermanagement-Logik
- offene System-Architektur

Testbarkeitsanforderungen von Benutzern

- zugänglicher Systemzustand
- verständlicher Systemzustand & Fehlermeldung
- Unterscheidung Benutzerfehler/Systemfehler
- Selbst-Test
- frühzeitige Fehlererkennung
- Fehlerprognosen
- Unterstützung bei Fehlerbeschreibung
- automatische Weiterleitung der Fehlermeldung
- Diagnose-Qualität
- Vorwarnzeit vor Systemausfall

These 3

„Testbarkeitsanforderungen sind mehrheitlich funktional“

Funktional versus Nicht-Funktional

Tester

Verständlichkeit	Q
Angemessene Komplexität	Q
Kontrollierbarkeit (Steuerbarkeit)	F
Beobachtbarkeit	F
Isolierbarkeit	Q
Automatisierbarkeit	Q/F
Robustheit (der Software)	Q
Reproduzierbarkeit	Q/F
Lokalisierbarkeit	F
Anonymisierbarkeit	F

Entwickler

Ausgabe von Fehlermeldungen	F
Nachvollziehbare Fehlermeldungen	Q/F
Nachvollziehbarer Systemzustand	F
Konfigurierbares Logging	F
Selbstdefinierte Abfragen	F
Fehlerstatistiken	F
Trennung von Fehlermanagement und Geschäftslogik	Q
Einfache Fehlermanagement-Logik	Q
Offene System-Architektur	Q

Benutzer

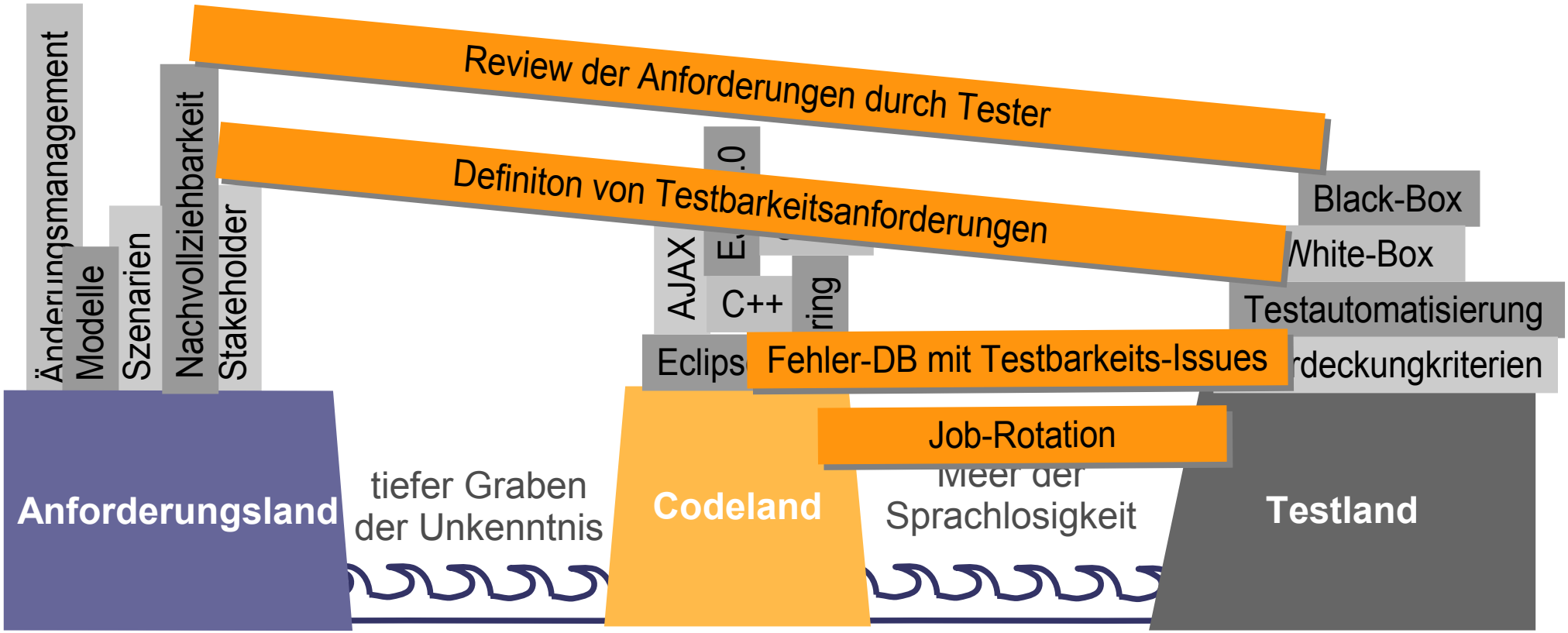
zugänglicher Systemzustand	F
verständlicher Systemzustand & Fehlermeldung	Q
Unterscheidung Benutzerfehler/Systemfehler	F
Selbst-Test	F
frühzeitige Fehlererkennung	F
Fehlerprognosen	F
Unterstützung bei Fehlerbeschreibung	F
automatische Weiterleitung der Fehlermeldung	F
Diagnose-Qualität	F
Vorwarnzeit vor Systemausfall	F

Fazit

- Thesen:
  1. „Testbarkeit birgt Einsparungspotential am Gesamtentwicklungsaufwand von ca. 10%”
  2. “Testbarkeitsanforderungen werden unzureichend formuliert“
  3. „Testbarkeitsanforderungen sind mehrheitlich funktional“
- Hilfestellung existiert
  - initiale Liste mit Testbarkeitsanforderungen
- Future Work (was fehlt?)
  - Aufwandsdaten für Fehlersuche während Implementierung
  - Aufwandsdaten für Test- und Diagnoseprobleme

Fazit

Mehr Brücken sind notwendig!



Zeit für Diskussion ...

„Good designers build creations that work as planned; great designers construct systems that both work and are debuggable.“ [Gans98]

Referenzen

- [Beiz90] B. Beizer. Software Testing Techniques, 2nd Edition, International Thomson Computer Press, 1990.
- [Gans98] Jack Ganssle. Debuggable Designs. Embedded Systems Programming, Dezember, 1998.
- [Gans01] Jack Ganssle. Proactive Debugging. Embedded Systems Programming, Januar 2001.
- [Hail02] B. Hailpern, P. Santhanam. Software debugging, testing, and verification. IBM Systems Journal, vol 41, no 1, 2002.

Kontakt

Stefan Jungmayr
http://www.testbarkeit.de