

Wir kümmern uns.



Pragmatisches Testen mit System

Martin Böhm

06.09.2012

Agenda – Kapitel

1 SALT Solutions GmbH

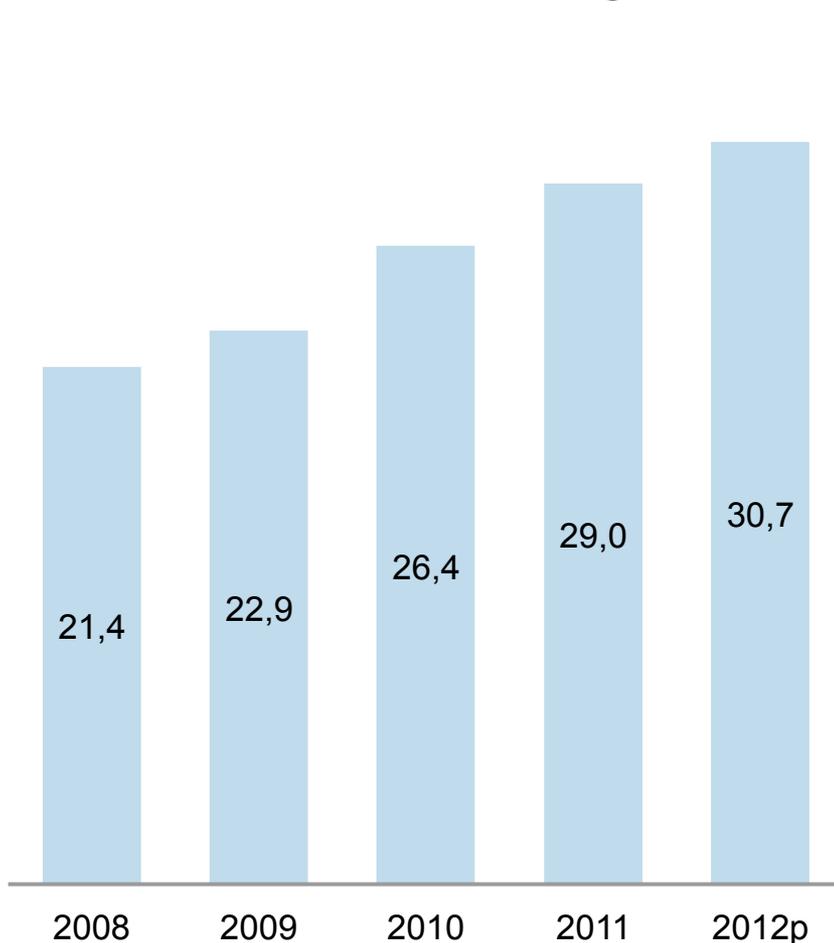
2 Typische Testprobleme

3 Pragmatismus vs. Theorie

4 Techniken, Organisation und Mitarbeiter

5 Fragen und Diskussion

Umsatzentwicklung

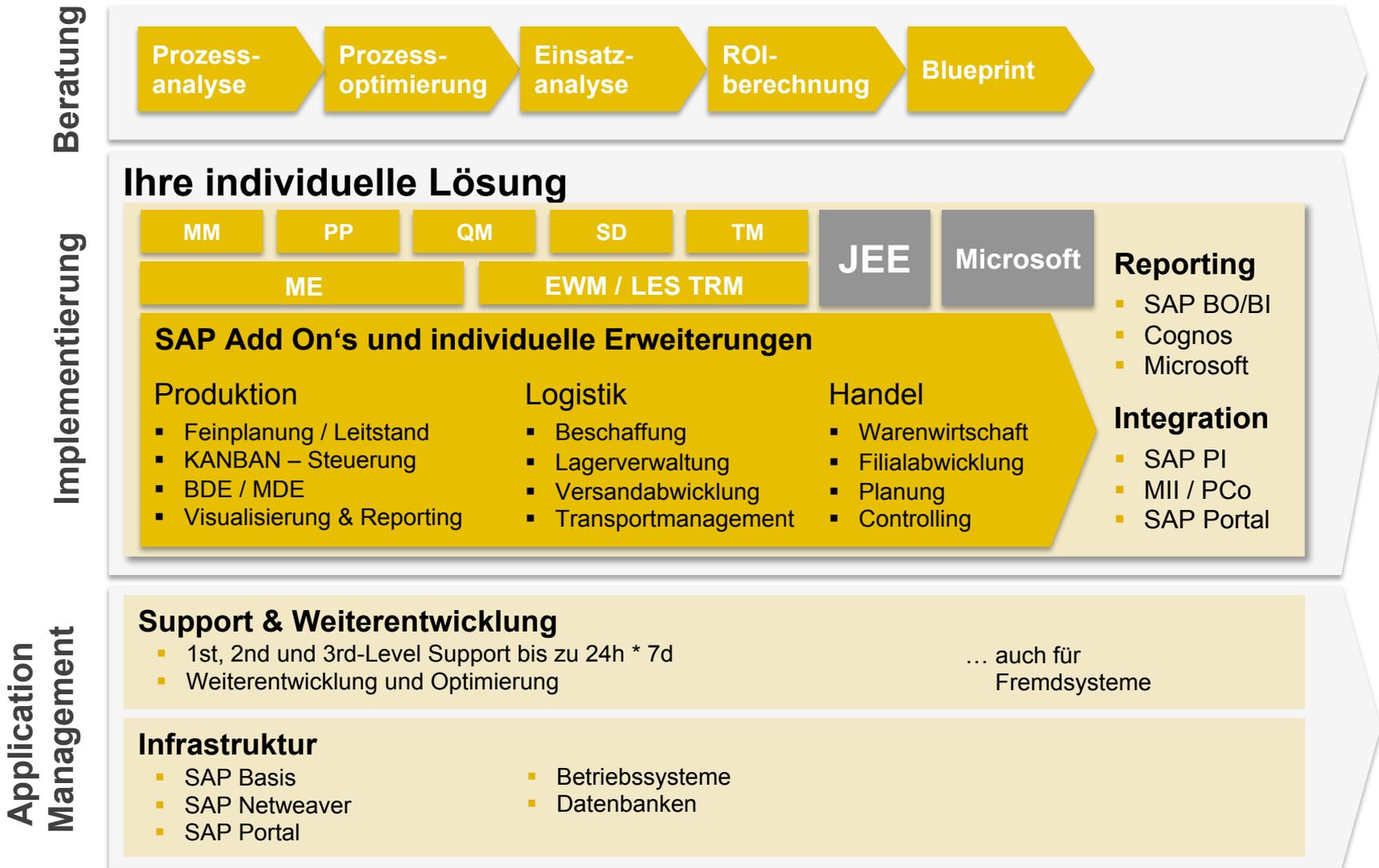


Steckbrief

- Gründung 06/2002
- Standorte Dresden, Düsseldorf, München, Würzburg
- Mitarbeiter (Stand 1.1.2012) 300 (Vollzeit)
- Eigenkapital 4,8 Mio. Euro
EK-Quote 50 Prozent
- Gesellschafter und Geschäftsführer Dr. Bernhard Blüthner
Dieter Heyde
Markus Honold
- Geschäftsfelder Produktion
Logistik
Handel



- **Prozessgestaltung und IT-Lösungen entlang der Wertschöpfungskette**
- **Gesprächspartner für Fachbereich und IT**
- **Internationale Implementierungserfahrungen**
- **Application Management und Support über den kompletten Lebenszyklus**
- **Zertifizierungen: ISO 9001, pharmaindustrievalidiert, SAP AMS**



Diese Unternehmen vertrauen der SALT Solutions



BMW Group



B/S/H/



CLOOS



Dräger



GERRY WEBER
INTERNATIONAL AG



Linde



MAHLE

otto group

Rexroth
Bosch Group



SIEMENS

SportScheck

THALES



voestalpine
EINEN SCHRITT VORAUSS.

VOSS

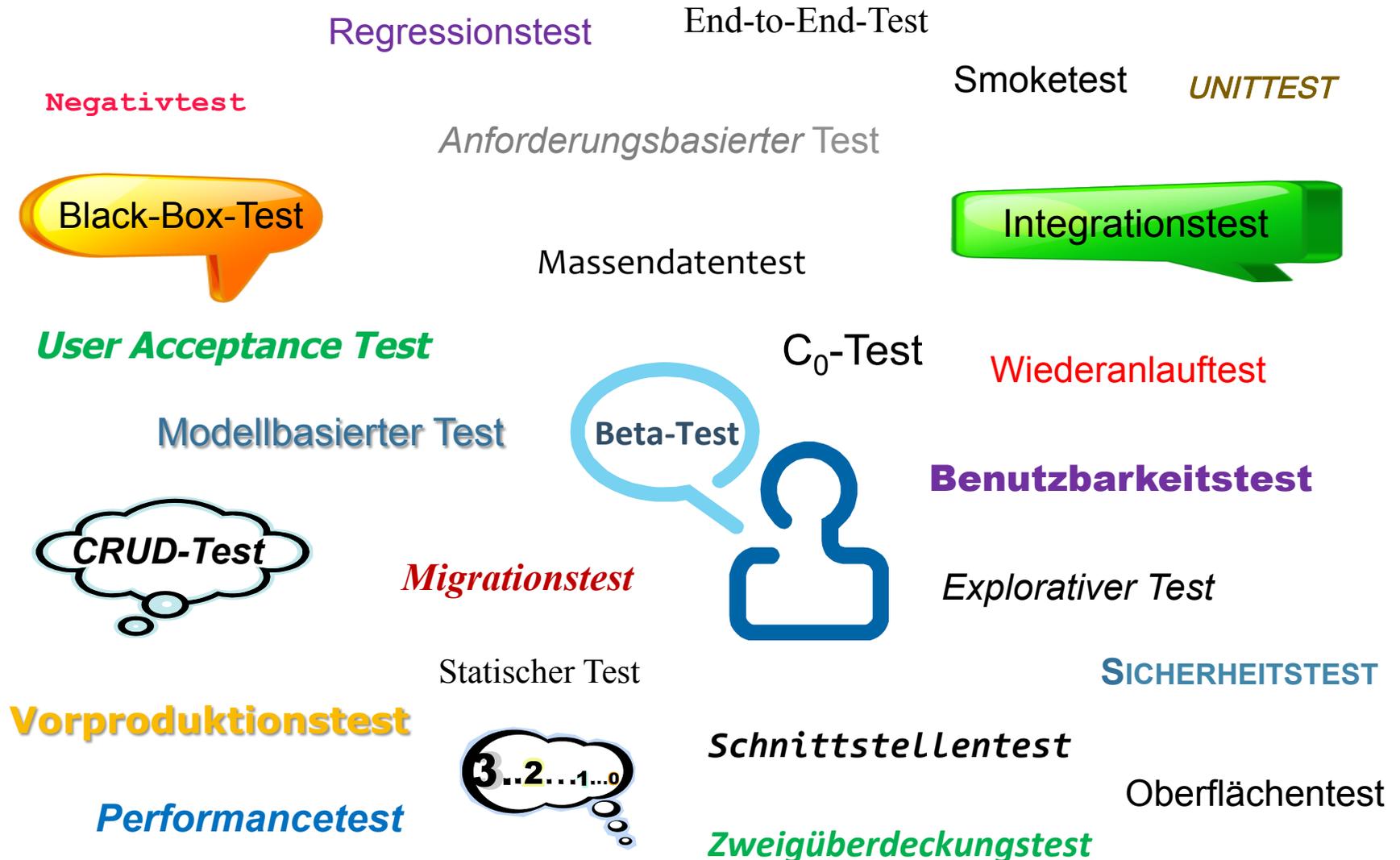


WÜRTH

Typische Probleme beim Testen

- Test wird zwar postuliert, aber nicht verstanden und methodisch vorgegangen
- Komplexität und Vorbereitungsdauer werden unterschätzt
- Notwendiges Know-How wird unterschätzt!; Testen wird auf elementare Aspekte reduziert: „Anwendung an! → Funktioniert! → Fertig!“)
- Später oder gar kein Test
- Es werden die falschen Tools zum falschen Zeitpunkt eingesetzt
- Es wird etwas falsches getestet; Anzahl oder Umfang der Testfälle ist unpassend
- Die Tester sind überlastet, weil die Spezifikation oder Durchführung der Tests aufwändig ist.
- Test wird als Kostentreiber empfunden, weil wirklich schwere und teure Fehler zu spät gefunden werden
- Testen wird auf die notwendigsten Aktivitäten reduziert; das Testmanagement wird abgeschafft
- Die Folgen unzureichenden Tests bzw. Testmanagements werden unterschätzt
- „Die entscheidenden Fehler findet man sowieso nur im Live-Betrieb!“

Test-Buzzwords

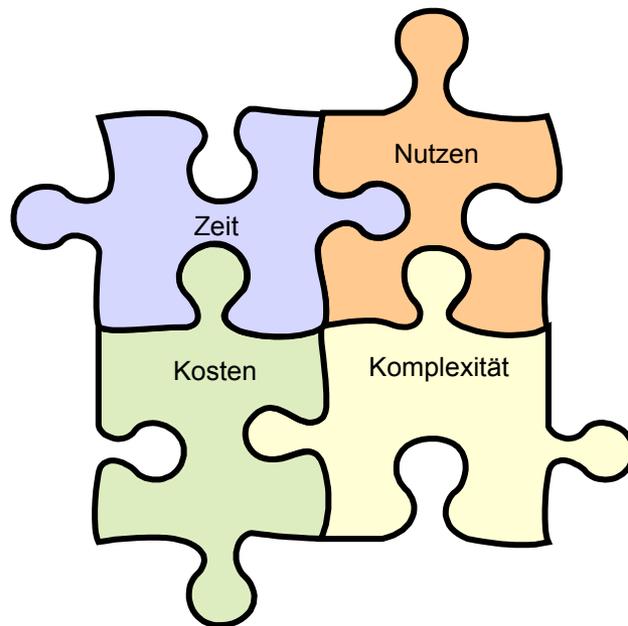


Fehlannahmen über das Testen, Zitate

- „Testen ist Selbstzweck und kein Kernziel des Projekts, deswegen will der Kunde es nicht in Rechnung gestellt haben“
 - Ursache:
Mangelnde Teststrategie → Mangelnde Testanforderungen → schlechte Tests → Verfehlte Testziele → Höherer Testaufwand → Sinkende Testeffizienz → Gefährdeter Projekterfolg → Kunde verzichtet auf „überflüssigen“ Aufwandsposten Test
- „Desto mehr Tests wir durchführen, desto mehr Fehler spüren wir auf.“
 - Ursache:
Test aus der Fehlerperspektive
Kernanforderungen, Projekt- und Produktrisiken werden beim Test vergessen
- „Mit dem richtigen Testwerkzeug und den gewählten Testtechniken ist Testen ein Kinderspiel.“
 - Ursache:
Falsche Annahmen über die Testumstände und Fehlerfindungsquoten
Fehlendes Test-Know-How („A fool with a tool...“)

Qualität und Test

- Qualität = Eignung eines Produkts oder Dienstleistung bestimmte Anforderungen zu erfüllen
- Test = Vorgehen zur Bestimmung der Qualität eines Produkts oder einer Dienstleistung



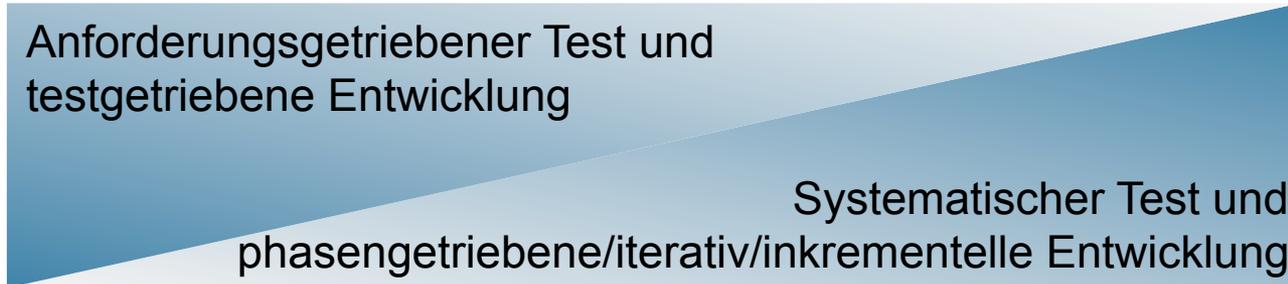
**Qualität =
Anforderungserfüllungsgrad**



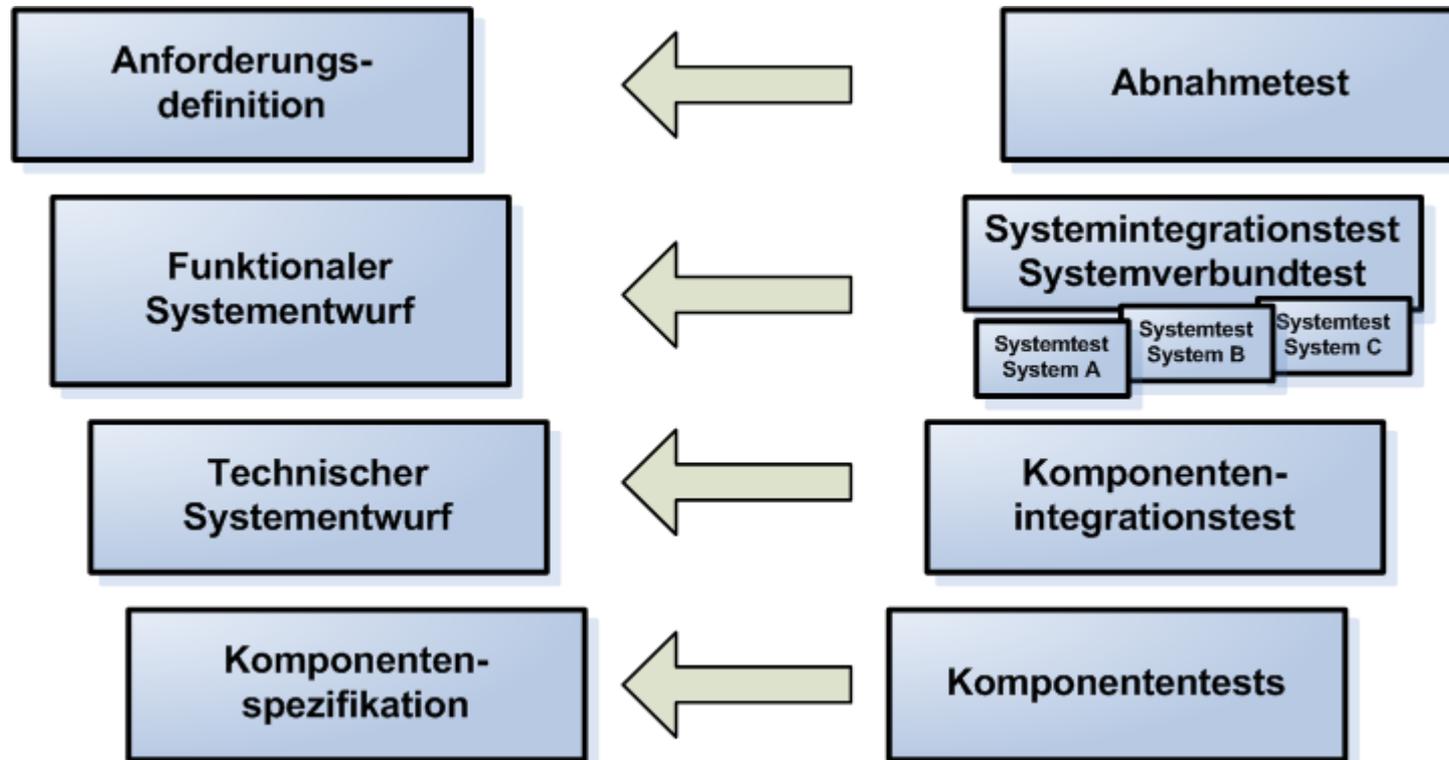
**Test = Messung des
Anforderungserfüllungsgrades**

Kleine Projekte

Große Projekte

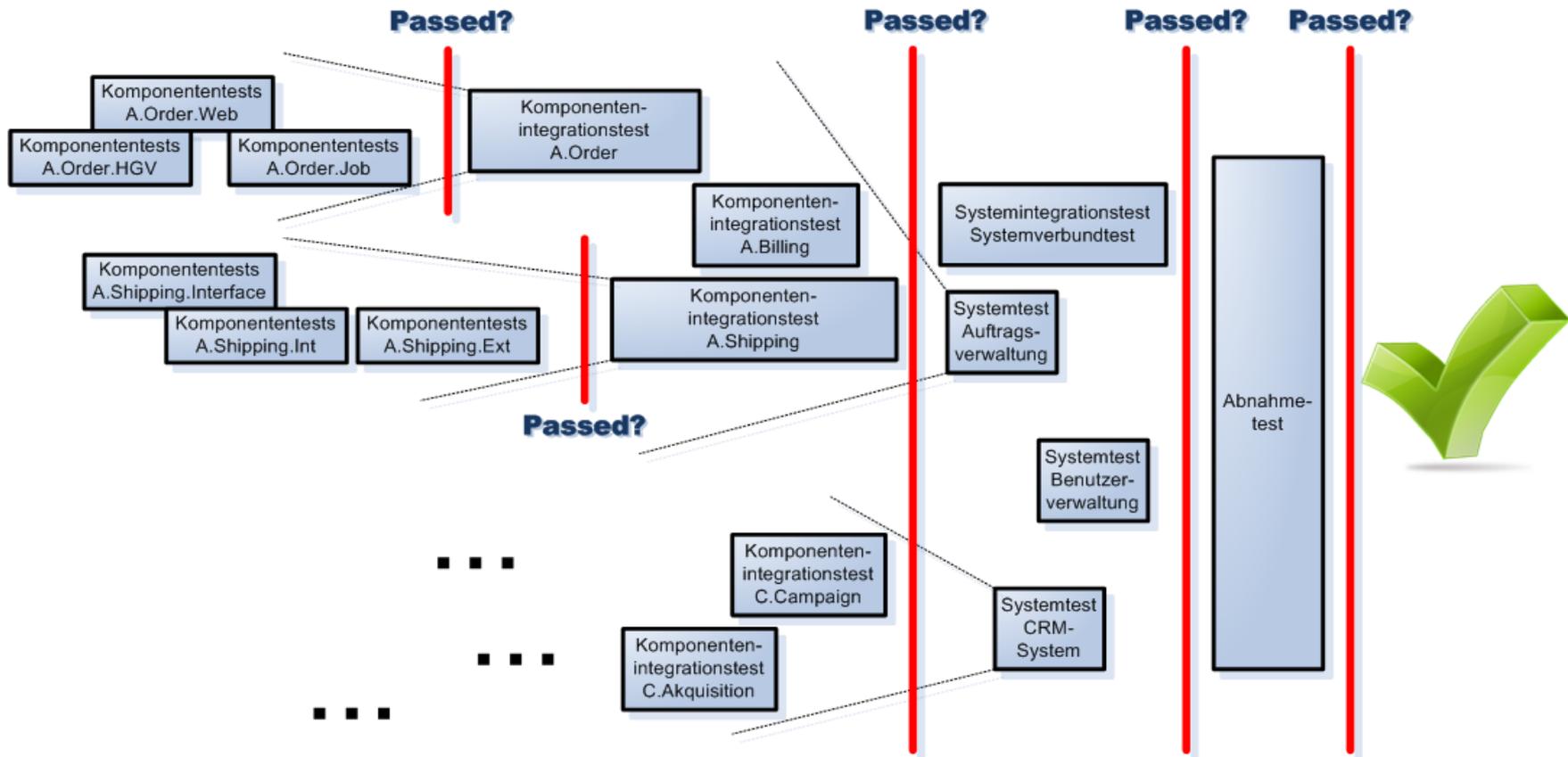


- Tendenziell:
Weil kleine Projekte wachsen und sich Anforderungen häufig noch ändern, muss die Testvorgehensweise angepasst werden.
Die Komplexität von großen Projekten erfordert i. d. R. eher eine systematische Testvorgehensweise.
- Für beide gilt:
 - Test soll unter Kosten-Nutzen-Aspekten geführt werden
 - Die Qualität des Gesamtsystems hängt von der Qualität der Komponenten und Teilsysteme und Qualität des Softwareentwicklungsprozesses ab



*ISTQB = International Software Testing Qualifications Board

Qualitative Abhängigkeit



Was steckt hinter den Teststufen?

Spezifikation ("linker Ast")	Ergebnisse (i. w. S. Testobjekte)	Teststufe ("rechter Ast")	Ziel	Bedeutung (nach ISTQB Glossar)	Zielgruppe, Entscheider	was neben der Software noch so abfällt...	Beispiele
Anforderungen, User Stories	HW+SW+(Daten, z. B. aus Migration)	Abnahmetest	"Endgültige Vertrauenshöhe des Kunden in das Produkt bzw. das System messen."	Formales Testen hinsichtlich der Benutzeranforderungen und -bedürfnisse bzw. der Geschäftsprozesse. Wird durchgeführt, um einem Auftraggeber oder einer bevollmächtigten Instanz die Entscheidung auf der Basis der Abnahmekriterien zu ermöglichen, ob ein System anzunehmen ist oder nicht.	Auftraggeber, Anwender, Produktmanager	Anwendungssystem; Benutzerdokumentation; Verfahrensanweisungen	Durchführung der Geschäftsprozesse am System; Check der Testdurchführungsprotokolle und der HW-Requirements; Revision der Projektdurchführung
Schnittstellen- und Betriebs-spezifikation	HW+SW	System-integrations-test	"Qualität der Prozessabwicklung mit externen Systemen prüfen."	Testen der Integration von Systemen und Paketen ; Testen der Schnittstellen zu einer externen Organisation; Testen eines Systemverbunds; u. a. auch zur Zertifizierung	Auftraggeber, Anwender, Produktmanager, IT-Betrieb, externe Organisationen	Schnittstellendokumentation; Betriebs-spezifikation	Test von Schnittstellen zu externen Systemen; EDI-Schnittstellen; SAP-Schnittstellen; Webservices; Hardware
funktionaler Systementwurf	SW	Systemtest	"Äußere Qualität des Systems messen."	Testen eines integrierten Systems, um sicherzustellen, dass es spezifizierte Systemanforderungen erfüllt.	Produktmanager, IT-Betrieb, techn. Projektleiter	SW-System mit Installations- und Betriebshandbuch; Installationskripte; Testtreiber;	Klassischer GUI-Klicktest; Klassischer Performancetest
technischer Systementwurf	SW-Module	Komponenten-integrations-test	"Qualität der Komponenten aus Sicht benachbarter Komponenten messen."	Testen wird durchgeführt mit dem Ziel, Fehlerzustände in den Schnittstellen und dem Zusammenwirken der integrierten Komponenten aufzudecken.	Entwickler, SW-Architekt	JAR/WAR/EAR; JavaDoc-ZIP; Testtreiber	Pseudounittest (z. B. Unittest mit Persistenz, Unittests über HTTP); EAR-Deploy; Spring- oder OSGi-Container starten
Komponentenspezifikation	SW-Komponenten	Komponenten-test	"Innere Qualität der Komponente aus technischer Sicht messen."	Testen einer (einzelnen) Komponente.	Entwickler	Sourcecode inkl. JavaDoc; Entwicklungsrichtlinien, XMLs und Konfigurationsfiles	Isolierter Unittest; Checkstyle; Findbugs

- Aktuelles Testvorgehen und Testumgebung auflisten
 - Was ist vorhanden?
 - Wie schaut der Testprozess aktuell aus?
 - Welche Tests (inkl. Reviews, Checklisten, Kundenworkshops nach einem Sprint) werden zur Zeit durchgeführt?
 - Welche Testtechniken oder Testmethoden werden dabei angewendet?
z. B. Regressionstests und Codereviews mittels CI-System; Defect Driven Test; Adhoc-Tests; Kundenvorführung
- Verantwortlichkeiten, Aufgaben und Rollen feststellen und auflisten

Beispiel

- System und Komponenten werden täglich 16:00 Uhr gebaut
- Die Entwickler entwickeln Unittests und führen diese lokal aus (... , schlagen aber häufig fehl, weil Abhängigkeiten zu einer schlecht gepflegten Datenbank bestehen)
- Regressionstests werden selten durchgeführt (... , weil sie vom Buildmanager häufig deaktiviert werden).
- A.Shipping hat die meisten Unittests (... es werden dort aber selten Abweichungen detektiert)
- Codereview wird durchgeführt (...die Behebung der Abweichungen findet aber nur selten statt)
- Der Kunde testet, wenn wir die Software aufs Produktivsystem ausgeliefert haben (... findet aber kritische Fehler häufig erst nach langer Produktivzeit)

- Herausforderungen:

- Schnelle Lösung der Testprobleme
- Behebung am lebenden Projekt
- Es soll wenig kosten
- Maximaler Testeffekt



- Möglichkeiten:

- TPI-Accessment (Test Process Improvement)
- Ursache-Wirkungsanalyse
- Die „5-Warum-Fragen“ stellen



- Warum tut man sich beim Entwickeln von System- und Abnahmetests so schwer, wohingegen die Entwicklung und Durchführung von Unittests eher leicht von der Hand geht?
- Gründe:
 - Die Entwicklung von Unittests erfolgt ad-hoc, während der Entwicklung. Der Zyklus Anforderung-Test-Entwicklung-Feedback ist sehr kurz.
 - Im Gegensatz dazu ist die Spezifikation von Systemtestfällen sehr abstrakt. Der Zeitraum von der Anforderungsformulierung bis zum tatsächlichen Feedback ist im Vergleich zu den Entwicklertests eher lang.

- Deswegen funktionieren eher agile Entwicklungsmodelle wie Scrum, Prototyping etc. so gut



Am Beispiel eines fiktiven Kampagnenmanagers

Anforderung	Typ
Der Nutzer muss am System sehen können, wann der nächste Kampagnentermin geplant ist, um eine Kampagne vorzubereiten.	Funktional
Der Nutzer muss am System einen neuen Kampagnentermin einstellen können, um damit eine Kampagne planen zu können.	Funktional
Das System muss die Kampagnentermine in benutzerdefinierten Abständen sichern, damit bei Systemausfall die Kampagnentermine wiederhergestellt werden können.	Funktional
Der Nutzer muss die Sicherungseinstellungen der Kampagnentermine ändern können, damit diese vom System automatisch gesichert werden können.	Funktional
Das System muss 80000 Termine im Mittel von 12 Sekunden sichern, damit das System von den angesprochenen Käufern akzeptiert wird.	Nicht-Funktional
Das Anwender muss beim Betrieb mit nicht konfigurierter Sicherung eine regelmäßige Warnung erhalten, damit der Anwender vor Datenverlust gewarnt wird.	Funktional

Testobjekte identifizieren (1)

■ Testobjekte des Abnahmetest

- Testobjekt Liefergegenstand: Anforderungen, Handbücher, Software
- Testobjekt Geschäftsprozesse:
 - Kampagne durchführen mit den Teilaktivitäten: Kampagnentermin anlegen, Kampagnenverantwortliche einrichten, Kontakte in Kampagne übernehmen
 - System in Betrieb nehmen: System installieren, Datenbankverbindung einrichten, Sicherung konfigurieren
 - System updaten: Sicherung manuell durchführen, Datenbankverbindung entfernen, System herunterfahren, Update installieren



Testobjekte identifizieren (2)

■ Testobjekte des Systemtest des CRM

- Testobjekte aus statischer Sicht: Kampagnenterminansicht des Kampagnenmanagers
- Testobjekte aus Sicht der Aktivitäten und Funktionen:
 - Kampagnentermin anlegen
 - Systemsicherung konfigurieren
 - System automatisch sichern

■ Testobjekte des Komponentenintegrationstest CRM-Kampagnenmodul

- Testobjekte aus Sicht der Systemkomponenten:
 - Kampagnen-Export (technischer Export)
 - Kampagnen-Terminierung (technische Ermittlung der Termine)
 - Kampagnen-GUI
 - Kampagnen-WebGUI



Testmethoden und Testtechniken

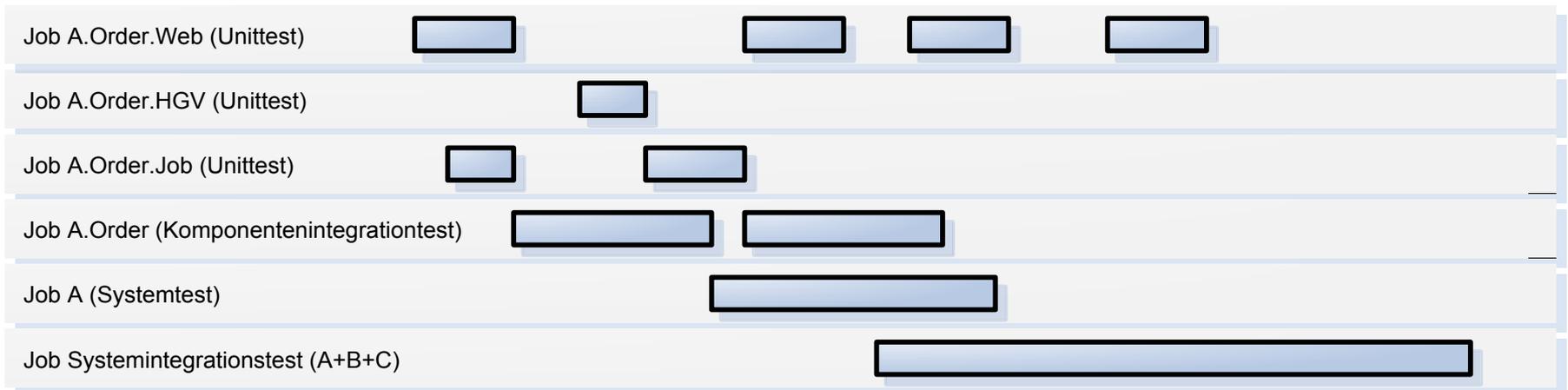
Eine kleine Auswahl von Möglichkeiten:

- Grenzwertanalyse und Äquivalenzklassenbildung z. B. für die Prüfung von Eingabefelder
 - z. B. Eingabefelder des Dialogs „Kampagnentermin anlegen“.
- Anweisungs- und Pfadüberdeckungstests z. B. für White-Box-Tests und Code-Analyse
 - z. B. Sicherheitskritische Software.
- Zustandbasierte Testmethode z. B. zur vollständigen Ermittlung von Testfällen
 - z. B. Zustandsautomat über die Zustände eines Kampagnentermins (angelegt, publiziert, bestätigt, abgelaufen, abgesagt)

Auswahl der Testmethoden und Testtechniken ist von den Anforderungen und Risiken abhängig!

Realisierung und Bestückung der Teststufen mit konkreten Tests

- Teststufen sind quasi Quality Gates
 - Die Teststufen bilden Testeingangs- und Testendekriterien für die jeweils vor- und nachgelagerten Teststufen
- Continuous-Integration-Plattform zur Automatisierung der Testdurchführung
 - z. B. mit Jenkins/Hudson, Maven, Ant
- Abhängigkeiten mittels Job-Trigger





- Was nicht testbar ist, dessen Qualität kann nicht gemessen werden!

- Lösungsmöglichkeiten:

- Anforderungen bzw. Verhalten i. S. von User-Stories formulieren (siehe oben)
- Nicht-funktionale Anforderungen nicht nur klären sondern auch festhalten (Performance-, Security-, Usability-Anforderungen)
- Teile-und-Herrsche-Prinzip anwenden (isoliert messen)
- Systemverständnis verbessern und komplizierte Sachverhalte auflösen



- Qualität und Test ganzheitlich über Kunde, Anbieter, Projektleitung und Team betrachten
- Kleine Steps vorantreiben (5-Warum-Fragen)

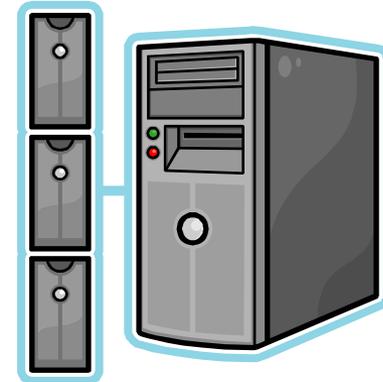


- Werkzeuge für
 - Testdurchführung
 - Testmanagement
 - Reporting
- Auswahl sollte sich an den Anforderungen des Projekts ausrichten
- Grundsätzliche Überlegungen
 - Trennung zwischen Testfallerstellung und Testdurchführung
 - Erstellung und Durchführung von Testszenarios unter Berücksichtigung von parallelen Releases
 - Bevorzugtes Vorgehen für Pflege der Testfälle



Flaschenhals Testumgebung

- Heterogene Systemlandschaft
 - Duplizierung für Test?
 - Virtuelle Umgebung?
- Mehrere Datenbanken / Unterschiedliche Testdatenbestände
 - Testdatengenerierung?
 - Testdatenrücksetzung?
- Mehrere Entwicklungszweige / Releases
 - Integrationstests
 - Abnahmetests
 - Vorproduktionstests
- Begrenzte Testressourcen
 - Integrationstests auf dem Buildsystem?
 - Test-Labor?



- Bewusstsein für Qualität und Quality Gates schaffen
- Argumentationsvorlagen vorbereiten:
 - Kontra fehlendes Bewusstsein für Qualität und deren Messung:
„Test dient nicht der Fehlerfindung, sondern dem Qualitätsnachweis.“
 - Kontra Verlagerung der Testverantwortung auf einen Tester oder Testoutsourcing:
„Jeder Projektteilnehmer braucht ausreichendes Testwissen, um seinen Qualitätsnachweis für das Produkt als auch dessen Systemteile zu erbringen.“
 - Kontra Testargumentation ist graue Theorie, die keine praktische Ansätze zur Kostenreduktion liefert:
„Man testet immer, und auch das Testen lässt sich verbessern und optimieren.“
„Jeder Testansatz ist legitim, sofern er eine Bewertung der Qualität ermöglicht.“
- Wohl überlegter Einsatz neuer Testmethoden (Stichwort: Buzzwords und Testwerkzeuge)
- Kosten-Nutzen-Denken bei der Auswahl der Testtechniken und Testwerkzeuge entsprechend der Anforderungen des Kunden bzw. des Softwareentwicklungsprozesses
- Der Rest ist praktisches Doing im Team!