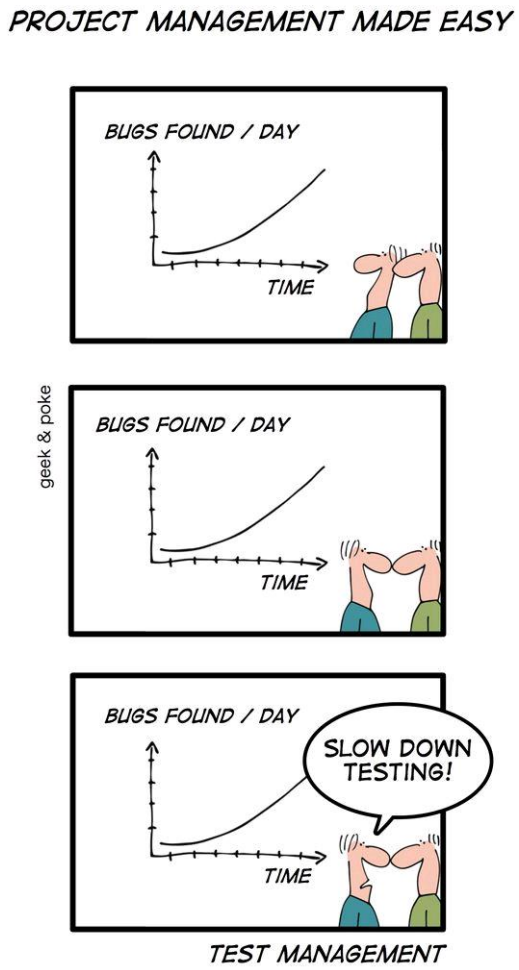
Testmanagement

HAW Hamburg / Fachbereich Informatik

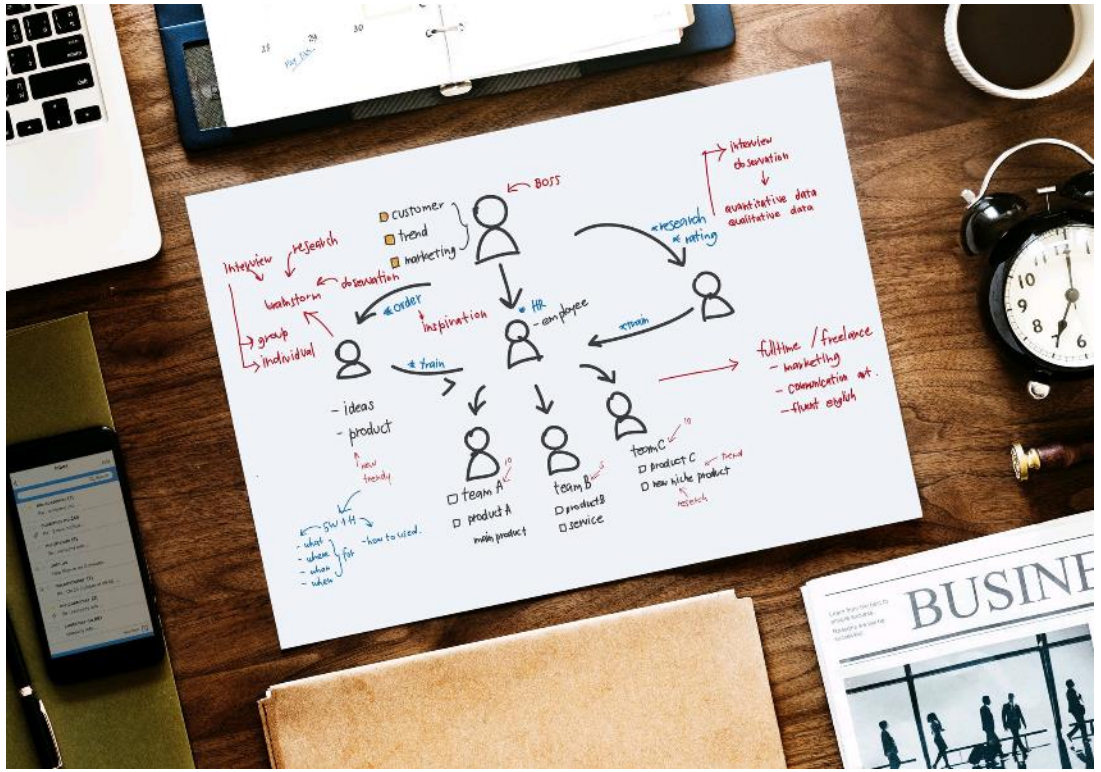
Tim Lüecke

(Tim.Lueecke@haw-hamburg.de)



Team-Organisation

- Prinzipiell gibt es zwei unterschiedliche Organisationsformen für das Testen
  - Test durch Entwickler
  - Test durch unabhängiges Test-Team
- Unabhängiges Test-Team kann auf verschiedene Weise gebildet werden:
  - Teil des Entwicklungsteams, was für ein Release zum Testen abgestellt wird
  - Anderes Entwicklungsteam
  - Eigenes dediziertes Testteam im Projekt oder der Organisation
  - Dienstleister
  - Spezialisten für besondere Testaufgaben (z.B. Lasttest)
- Je nach Testphase mag eine unterschiedliche Organisation mehr Sinn machen (Komponenten- vs Systemtest)

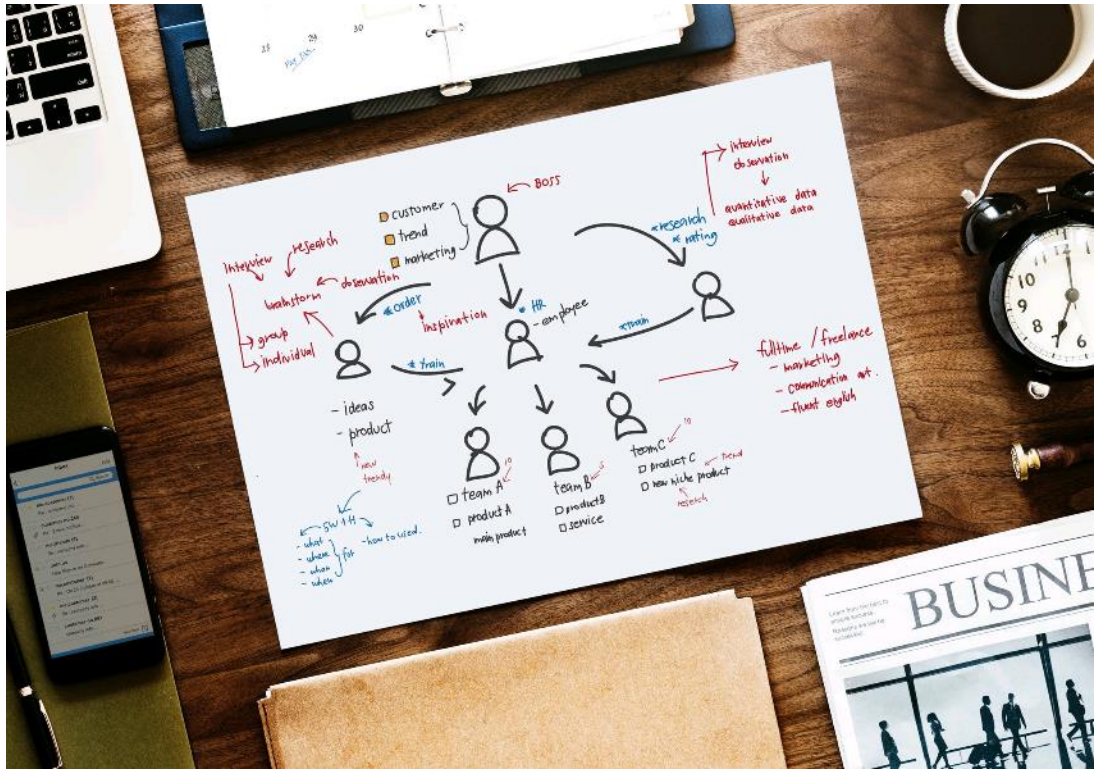


Testen durch unabhängige Teams

- **Vorteile**:
  - Arbeiten objektiver und finden andere Fehler (nicht “betriebsblind”)
  - Hinterfragen von impliziten Annahmen
- **Nachteile**
  - Isolation behindert Kommunikation
  - Flaschenhals bei zu geringen Resourcen
  - Verantwortung der Entwickler lässt nach

Rollen

- Testmanager
  - Steuert und plant die Tests sowie die benötigten Resourcen
  - Legt Test-Strategie fest
  - pendant zum Projektmanager für die Tests
- Testdesigner (Testanalyst)
  - Analysiert Anforderungen
  - Erstellt Testspezifikationen und Testdaten
- Testautomatisierer:
  - Automatisiert spezifizierte Tests
- Testadministrator
  - Installiert Software und stellt Testumgebungen bereit
- Tester
  - Review und Durchführung von Tests



Testplanung

- Testen sollte nicht abseits der anderen Qualitätsmaßnahmen gesehen werden, sondern diese umfassen!
- Artefakte:
  - Qualitätssicherungsplan (s. Vorlesung Qualität)
  - Testkonzept, d.h. Festlegung von
    - Teststrategie
    - Testautomatisierung
    - Testumgebungen
    - Testplan (inkl. Aufwand und Kosten)
  - Priosierung der Tests
  - Testende-Kriterieren
    - Testumfang
    - Produktqualität
    - Rest-Risiko
    - Wirtschaftliche Rahmenbedinungen

Priorisierung von Tests

- Tests werden **priorisiert**, um diese nach dieser Reihenfolge auszuführen
- Auf diese Weise lässt sich der **meiste Nutzen** aus der Testphase ziehen!
- Mögliche **Faktoren** für die Priorisierung:
  - Nutzungshäufigkeit bzw. Wahrscheinlichkeit
  - Fehlerrisiko (s. Definition in Risiko-Management!)
  - Fehler-Wahrnehmung
  - Priorität der Anforderungen
  - Qualitätsmerkmale
  - Komplexität der Komponenten

Management der Testarbeiten

- Systematische Tests werden in **Zyklen** durchgeführt:
  - Fehleridentifikation
  - Fehlerbehebung
  - Nachtest
- Jeder Testzyklus muss **für sich geplant** und kontrolliert werden
- **Metriken** für die Verfolgung:
  - *Fehlerbasiert* (z.B. Fehler/BT)
  - *Testfallbasiert* (Anzahl der ausgeführten Testfälle vs. blockiert und fehlgeschlagen)
  - *Testobjektbasiert* (z.B. Codeabdeckung)
  - *Kostenbasiert* (z.B. geschätzte Kosten vs. erwarteter Nutzen)

Steuerung der Testszyklen

- Ergebnisse jedes Testszyklus müssen in einem **Report** protokolliert werden
- Bei **Verzögerungen** oder Abweichungen zum Plan, muss der Test-Manager reagieren:
  - Streichen von niedrig-priorisierten Tests
  - Zusätzliche Test-Resourcen
  - Vereinfachung von hoch-priorisierten Tests (Weglassen von Varianten)
  - Verlängerung der Testphase
  - Abbruch des Releases

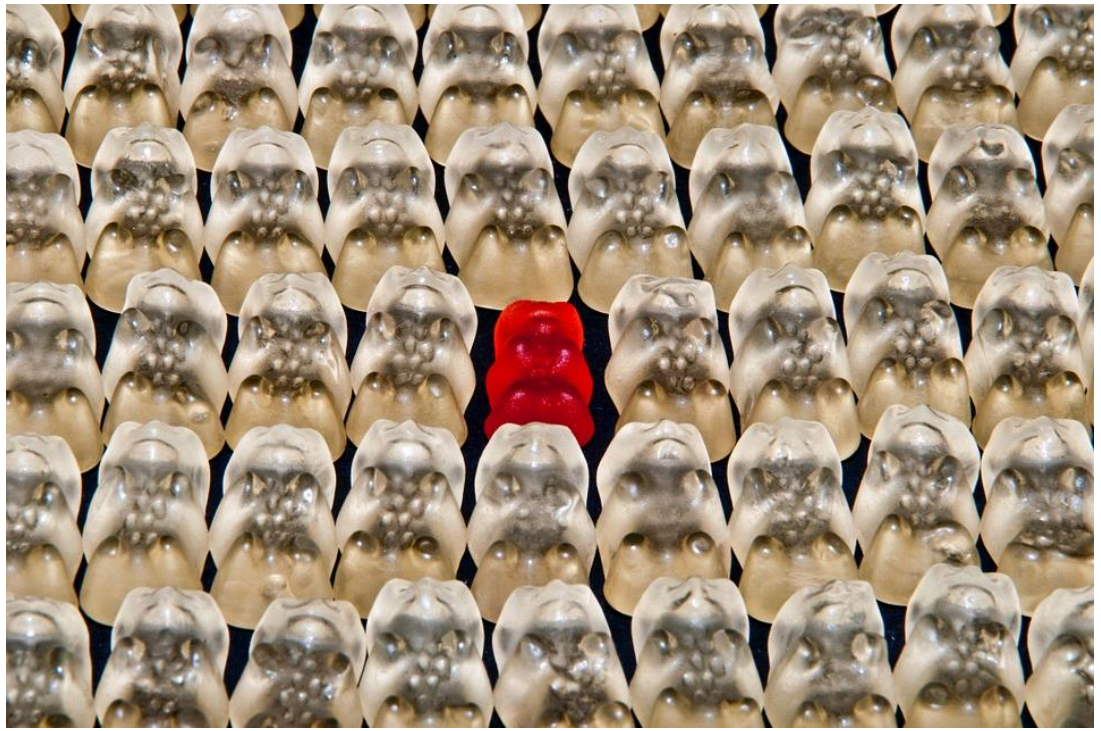
Fehlermanagement

- Das Fehlermanagement umfasst **alle Aktivitäten** zur Erfassung, Behandlung und Verfolgung von Fehlern im System
- **Erfassung** muss systematisch mit einem einheitlichen Schema erfolgen (s. Tabelle)
- **Klassifikation** erfolgt aus Sicht des Nutzers und separat aus Sicht der Projektmanagements (Klasse vs. Priorität)
- **Verfolgung** wird systematisch über ein Zustandsmodell gesteuert (nur der Tester darf einen Fehler schließen!)

	Attribut
Identifikation	Nummer
	Testobjekt
	Version
	Plattform
	Reporter
	Datum
Klassifikation	Status
	Klasse
	Priorität
	Anforderung
	Fehlerquelle
Problembeschreibung	Testfall
	Problem
	Komentar
	Verwei

Klassifikation von Fehlern

- **Klassifikation aus Sicht des Nutzers**
  - Klasse 1: Mangel der Dokumentation
  - Klasse 2: leichte, umgehbare Fehler
  - Klasse 3: schwere umgehbare Fehler
  - Klasse 4: Fehler, ohne deren Behebung eine wirtschaftliche Nutzung des Systems unmöglich ist
- **Priorisierung als separates Attribut:**
  - Priorität 1: Patch, sofortiges Beheben durch z.B. Hotfix
  - Priorität 2: Nächste Version
  - Priorität 3: Gelegentlich, sobald ohnehin an der Komponente gearbeitet wird
  - Priorität 4: Offen – Korrektur muss noch geplant werden



Zusammenfassung

- Organisatorische Trennung von Testern hilfreich – muss aber sinnvoll gestaltet sein
- Testmanager als Pendant des Projektmanagers für die Qualitätssicherung
- Priorisierung der Tests, um Lösungsalternativen bei Verzögerungen schnell finden zu können
- Systematisches Fehlermanagement als Basis für das Testmanagement