Grundlagen der Qualitätssicherung

180.764 Software-Qualitätssicherung

10.10.2024 - WS 2024

Thomas Grechenig
Christina Zoffi

Research Group for Industrial Software (INSO)
https://www.inso.tuwien.ac.at

Agenda

- Motivation
- Software Qualität
- Software Qualitätssicherung
- Qualitätssicherung in der Softwareentwicklung

Motivation

- Qualitätssicherung ist ein wesentlicher Bestandteil moderner Software Entwicklung
- Ihr Stellenwert ist insb. gewachsen durch
  - Steigende Größe und Komplexität von Software Projekten
  - Hoher Grad an Integration und Vernetzung zwischen Systemen
  - Zusammenspiel vieler Technologien
  - Steigender Einsatz von Technologie im Alltag (IOT)
  - Regulatorische Anforderungen (Vorschriften, Standards, Gesetze)

Motivation

- Mangelnde Qualität bzw. fehlerhafte Software hat nicht nur unmittelbare finanzielle Folgen
- Qualitätsmängel können auch Schaden verursachen
  - Personenschaden
  - Verlust an Reputation eines Unternehmens
  - Verlust des Vertrauens der Benutzer
- Ziel ist die Vermeidung bzw. frühzeitige Identifikation von Qualitätsproblemen

Beispiele für gravierende SW-Fehler

Therac-25 (1985-87)

Patienten erhielten bis zu 100-fache Strahlendosis

Todesfälle und Strahlenverletzungen

Overflow eines Counters bei manueller Übersteuerung

Ariane 5 (1996)

Explosion der Rakete 40 Sek. nach dem Start

500$ Mio Schaden

Fehlerhafte Int-Conversion, Reuse eines Software-Modules aus Ariane 4 ohne gründliche Tests

Amazon (2014)

Preise von 75 Mio. Third-Party-Artikeln stürzten auf 1 Cent ab

Hohe Verluste bis zum Bankrott besonders für kleine Unternehmen

Fehler in Software für automatische Preisanpassung

Airbus (2019)

Airbus muss alle 147 Stunden manuell rebootet werden

Fehler bei internem Timer löst Probleme in sicherheits-kritischem Flugsystem aus

NHS Covid (2020)

Personen mit Kontakt zu Infizierten wurden nicht notifiziert

19 Mio Downloads der App

Fehlerhafte mathematische Berechnung der Kontaktdauer zw. Personen

ELGA (2022)

Anzeige falscher Dosis bei verschriebenen Medikamenten

ca.1000 betroffene Patienten, 33 betroffene Apotheken

Technischer Fehler in 3rd-Party-Software

Software Qualität

Software Qualität

Diskussion

Wann hat eine Software gute Qualität?

Definition – Software Qualität

- Keine einheitliche Begriffsdefinition
- Multidimensionaler Begriff, Gesamtqualität setzt sich aus vielen einzelnen Qualitätsaspekten zusammen
- Vorstellungen eines „high quality“-Systems sind kontext- und projektabhängig und oft subjektiv geprägt
- Komplexität auf zwei Ebenen
  - Welche Eigenschaften eines Produkts definieren Qualität?
  - Welche Ausprägung müssen diese Eigenschaften annehmen?
- Messbarkeit und Überprüfbarkeit muss gegeben sein

Definition – Software Qualität (nach IEEE-730)

- *„The degree to which a product or process meets established requirements; […] the degree to which those established requirements accurately represent stakeholder needs, wants, and expectations"* (IEEE 730-2014)

- Software Qualität gem. IEEE-730 bedeutet erfolgreiche Verifikation und Validierung

Verifikation

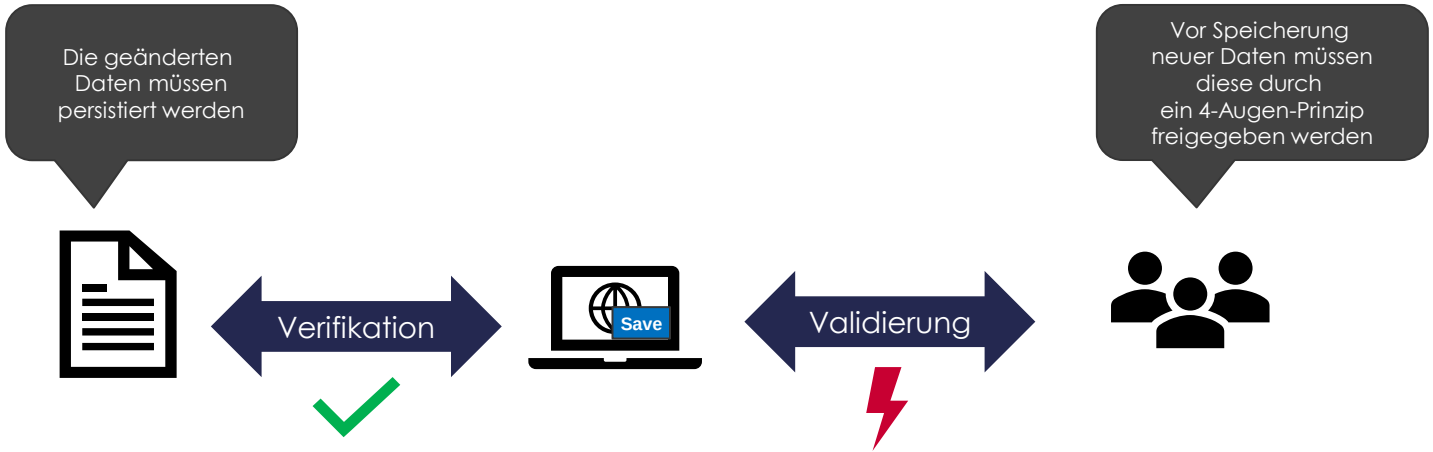
- Konformität gegenüber den spezifizierten Anforderungen
- *Bauen wir das Produkt richtig?*

Validierung

- Konformität gegenüber dem vorgesehenen Zweck und den Bedürfnissen der Stakeholder („Fitness for use")
- *Bauen wir das richtige Produkt?*

Verifikation vs. Validierung Beispiel

Beispiel – Erfolgreiche Verifikation alleine ist nicht ausreichend, um von qualitativer Software zu sprechen



Definition – Software Qualität (nach ISO/IEC 25010)

- ISO/IEC 25010 definiert ein Set an Qualitätsfaktoren und zugehörigen Qualitätskriterien
  - Qualitativ oder quantitativ messbar
  - Helfen bei der Spezifikation von Anforderungen
  - Ermöglichen Beurteilung des Erfüllungsgrades
- Umfassendes Rahmenwerk zur Definition und Bewertung der Software Qualität
- Projektumfeld und Domäne steuern den Stellenwert der einzelnen Qualitätsfaktoren
- Unterteilung in zwei Untergruppen:
  - Funktionale Qualität
  - Strukturelle Qualität

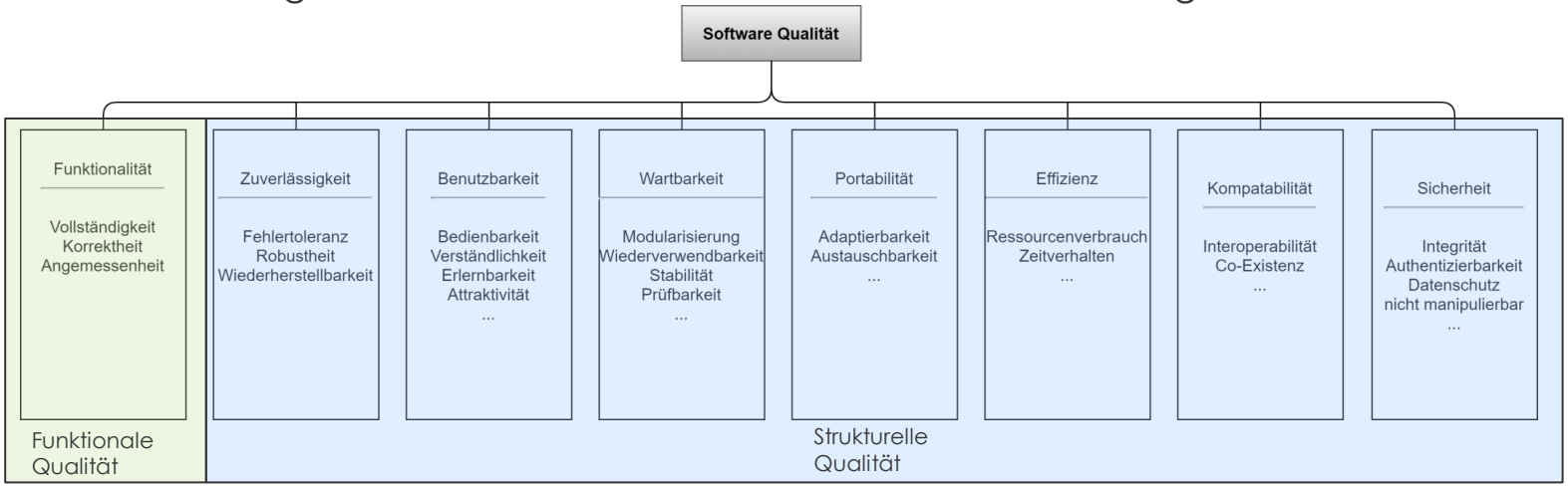
Qualitätsfaktoren

Funktionale Qualität

- Die äußere Sicht (Benutzersicht) auf ein System
- Externe Merkmale des Systems
- Definieren das "was?"
  - → Bezug zu funktionalen Anforderungen

Strukturelle Qualität

- Die innere Sicht auf ein System
- Interne Merkmale und Eigenschaften eines Systems
- Definieren das "wie gut?"
  - → Bezug zu nicht-funktionalen Anforderungen



Qualitätsfaktoren

Beispiel – Befundsystem

Ihr Unternehmen wurde mit der Entwicklung eines Befundsystems beauftragt, das Patienten und Ärzten Laborbefunde bereitstellen soll.

Welche (strukturellen) Qualitätsfaktoren haben hier besonders hohen Stellenwert und warum?

Software Qualitätssicherung

Definition – Software Qualitätssicherung

- *„Aktivitäten, die darauf fokussieren, Vertrauen in die Erfüllung der Qualitätsanforderungen zu erzeugen.“ (ISTQB)*
- *“A planned and systematic pattern of all actions necessary to provide adequate confidence that an item or product conforms to established technical requirements.” (IEEE 610.12-90)*

- Maßnahmen und Methoden, die der Überprüfung und Überwachung der Software Qualität dienen
- Sicherstellung, ob und zu welchem Grad die Qualitätsfaktoren erfüllt sind
- Kontinuierlicher Prozess, der den gesamten SW-Lebenszyklus begleitet

Klassifikation Qualitätssicherung

Organisatorische Methoden

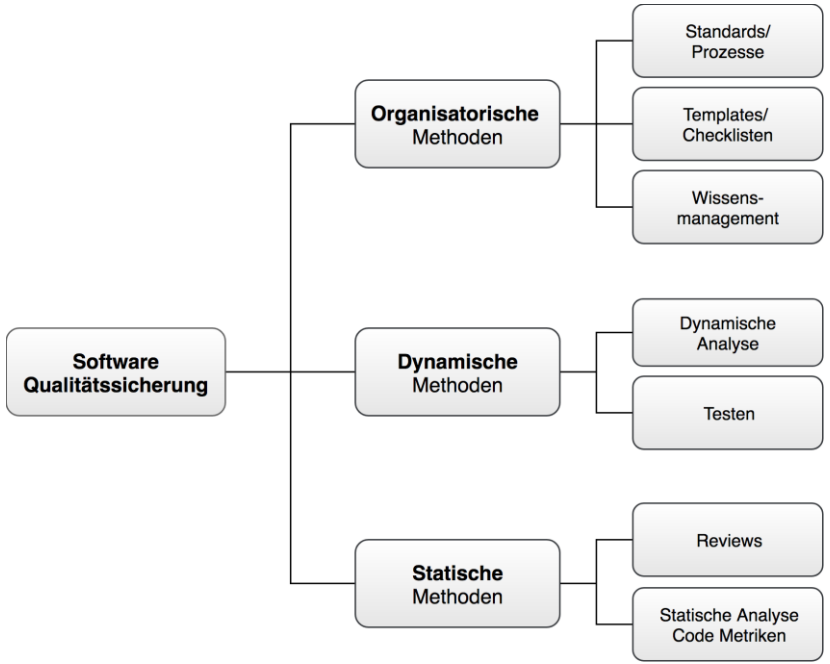
- Schaffen notwendige Infrastruktur und Rahmenbedingungen
- Definieren und steuern den Qualitätsprozess

Dynamische Methoden

- Software bzw. Teile davon müssen ausführbar sein
- Fokus auf das Verhalten zur Laufzeit ("äußere Qualität")

Statische Methoden

- Software muss nicht ausführbar bzw. integriert sein
- Fokus auf die interne Struktur („innere Qualität")

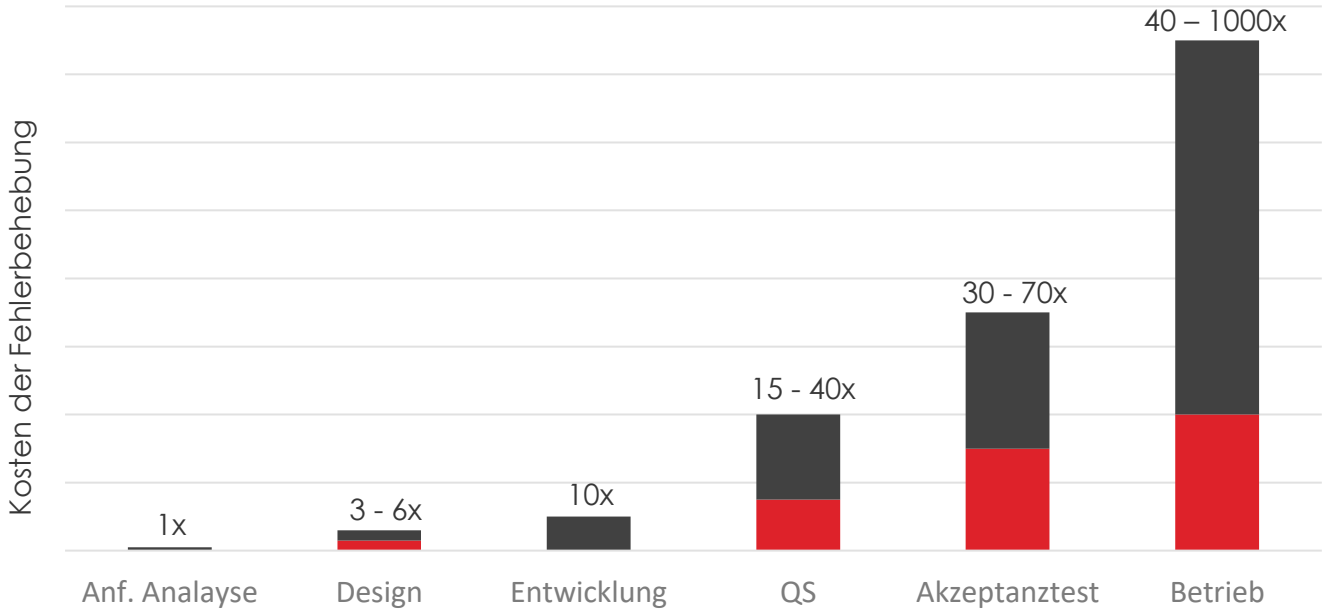


Klassifikation Qualitätssicherung

- Organisatorische Methoden werden auch als **konstruktive** QS-Maßnahmen bezeichnet
- Statische und dynamische Methoden werden auch als **analytische** QS-Maßnahmen bezeichnet
- Ziel:
  - Prävention von Qualitätsproblemen
  - Identifikation von Qualitätsmängeln
  - Kontinuierliche Überwachung einzelner Qualitätsaspekte
- Ergänzen einander durch unterschiedliche Sichtweisen
  - Äußere Qualität: Benutzersicht, Systemverhalten
  - Innere Qualität: interne Sicht, u.a. Architektur, Wartbarkeit, Verständlichkeit, …

Fehlerkosten

- QS-Maßnahmen sollten möglichst frühzeitig und begleitend zu den Phasen eines SW-Projekts stattfinden
- Fehlerkosten steigen, je später ein Fehler identifiziert/behoben wird



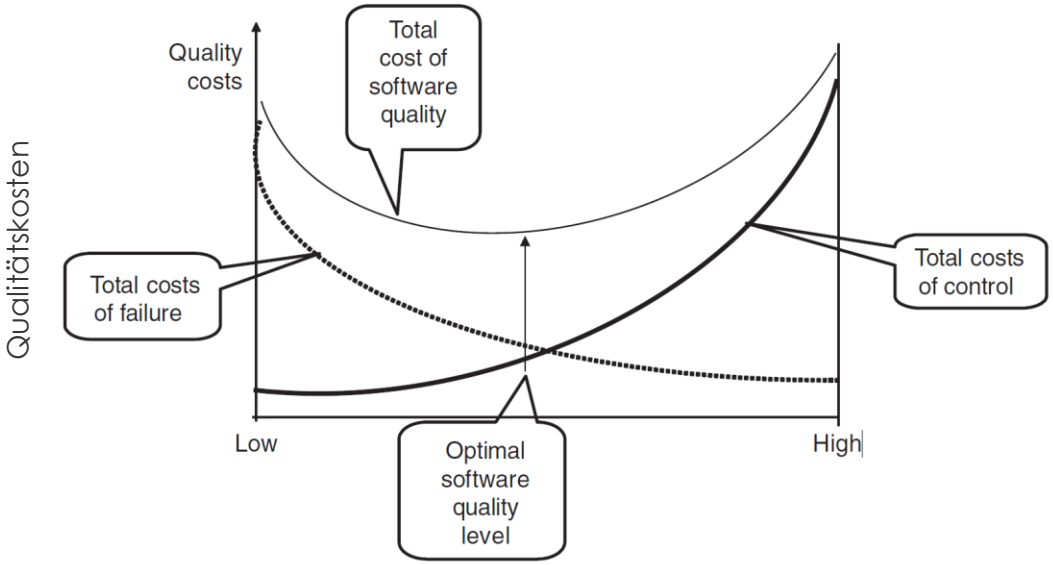
Qualitätskosten

- *„The real question is not how much a quality management system costs, but what is the cost of not having one."* (Harold Geneen)

- Gesamtheit aller geplanten und ungeplanten Kosten der Qualitätssicherung
  - Prüfkosten (geplante Kosten)
  - Fehlerkosten (ungeplante Kosten)
- → auch unterlassene Qualitätssicherung verursacht Kosten

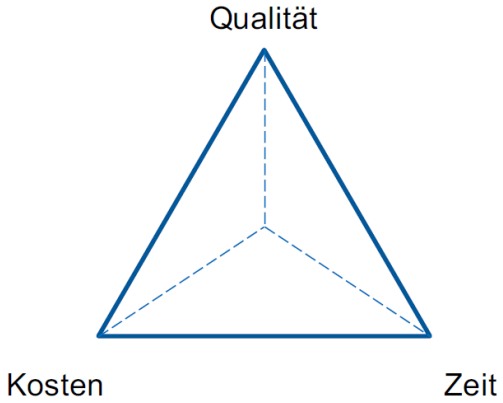
Qualitätskosten – Balance

- Proaktive Investition in Qualitätssicherung vs. hohe Fehlerkosten
- Ziel ist ein ausgewogenes Verhältnis zwischen Prüfkosten und Fehlerkosten



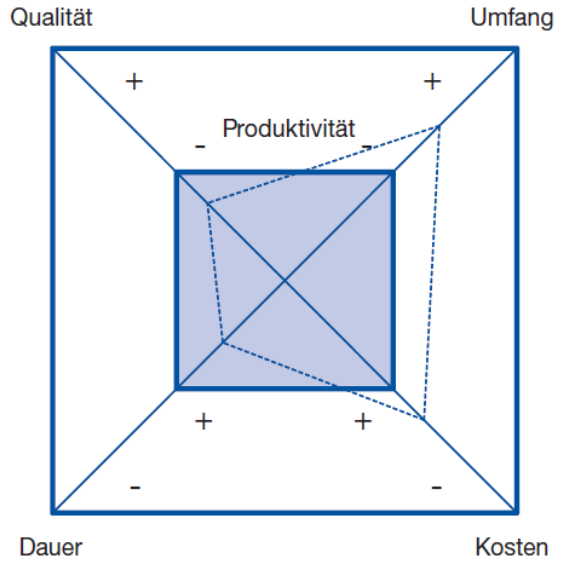
Einflussfaktoren – Magisches Dreieck

- In einem Projekt konkurrieren die Faktoren Qualität, Zeit und Kosten um die verfügbare Gesamtproduktivität
- Qualität (Umfang und Inhalt) steht im Spannungsverhältnis mit den anderen Faktoren
- Verzerrungen in eine Dimension gehen zu Lasten einer oder beider anderen



Einflussfaktoren – Teufelsquadrat nach Sneed

- Erweiterung des magischen Dreiecks
- Trennung von Qualität und Umfang in separate Dimensionen
- Ermöglicht mehr Handlungsoptionen bei Verzerrung von Dauer oder Kosten:
  - Qualität kann gehalten werden, unter Reduktion des Umfangs
  - (Umfang kann gehalten werden, unter Reduktion der Qualität)
- Die Grundfläche des Quadrats ist fix und symbolisiert die verfügbare Produktivität



Qualitätssicherung in der Softwareentwicklung

Vorgehensmodelle in der Softwareentwicklung

- Vorgehensmodelle geben SW-Projekten einen einheitlichen Rahmen und Ablauf
- Im Laufe der Zeit sind unterschiedliche Vorgehensmodelle entstanden
  - Sequenzielle (historische) Modelle (z.B. Wasserfall, V-Modell)
  - Iterative/Inkrementelle Modelle (z.B. Spiralmodell, V-Modell XT)
  - Agile Modelle (z.B. Scrum, Kanban)
- Es gibt kein Universalmodell, das für jedes Projekt erfolgsbringend angewandt werden kann
  - Wahl und Anpassung des Vorgehensmodells je nach Projekt
- Fundamentale Phasen sind in unterschiedlicher Form in jedem Modell aufgegriffen

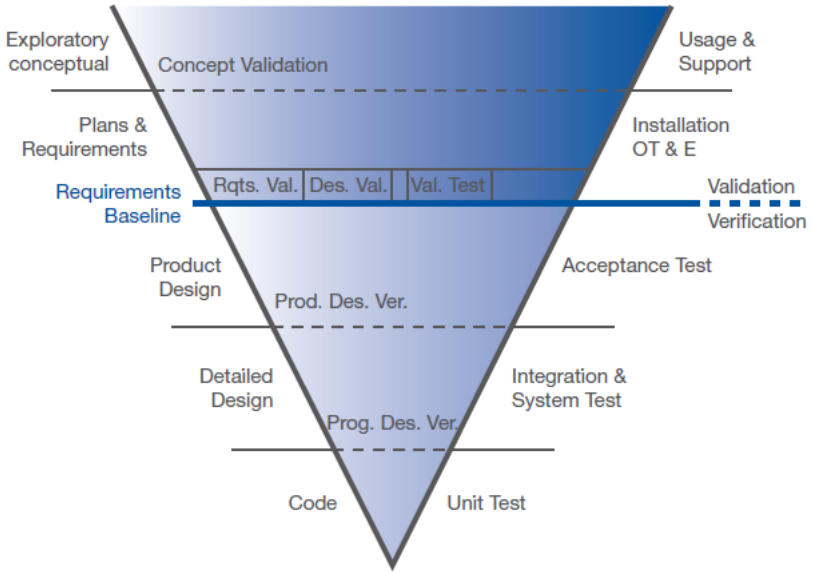
Einbettung Qualitätssicherung

- QS in der Softwareentwicklung erfordert systematisches Vorgehen nach bewährten Methoden und Maßnahmen
- Qualität kann nicht (oder nur durch enorme Kosten) erst am Ende eines Projekts hinzugefügt werden
- QS ist ein kontinuierlicher Prozess, der alle Phasen der Softwareentwicklung begleitet
- QS-Aktivitäten finden in jedem Vorgehensmodell Anwendung, sind aber unterschiedlich gelagert und integriert

- Beispiele für QS-Aktivitäten:
  - Analyse: Prüfung von Anforderungen und Spezifikationsdokumenten
  - Entwurf: Prüfung von technischer Spezifikation, Ableitung von Testfällen
  - Implementierung: Analyse von Source Code, Implementierung von Tests

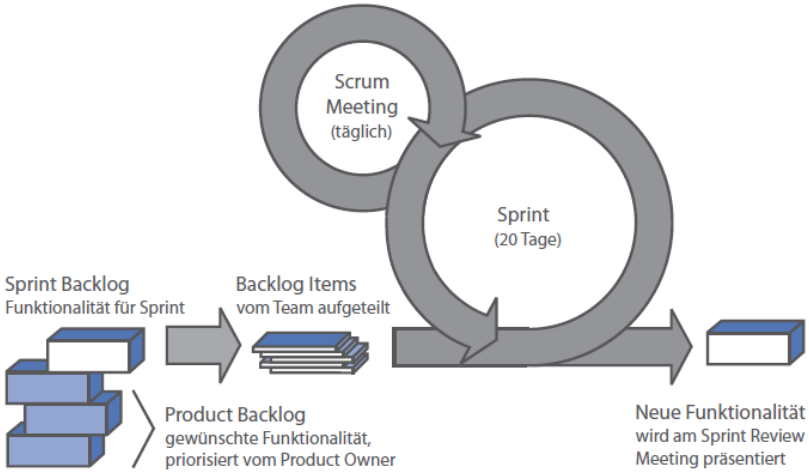
Überblick – QS im V-Modell

- Grundlage für modernere Modelle
- Jede *konstruktive* Phase der SW-Entwicklung (linke Seite) hat eine korrespondierende *prüfende* Phase der Qualitätssicherung (rechte Seite)
- Erstes Modell, das QS-Aktivitäten als Feedback-Schleifen integriert
- Miteinbezug von Verifikation und Validierung
- Risiko der Fehlentwicklung durch späte Validierung bleibt



Überblick – QS bei Scrum

- Repräsentant für agile SW-Entwicklung
  - Leichtgewichtiges Rahmengerüst
  - Iterative Entwicklung in Zyklen (Sprints)
- QS ist kein separater Prozess, sondern eng mit den Prinzipen und Praktiken der agilen SW-Entwicklung gekoppelt
- QS-Aktivitäten sind Teil des Sprints
- Teams sind oft cross-functional und selbstorganisiert



Zusammenfassung

- Es gibt verschiedene Herangehensweisen, den Begriff Qualität zu definieren. Eine Möglichkeit ist die Beschreibung über Qualitätsfaktoren.
- QS ist ein kontinuierlicher Prozess und beinhaltet konstruktive und analytische Maßnahmen und Methoden zur Sicherstellung und Überprüfung der Qualität.
- Qualitätskosten setzen sich aus Prüfkosten und Fehlerkosten zusammen. Auch unterlassene Qualitätssicherung verursacht Kosten.
- Qualität, Umfang, Kosten und Zeit stehen in einem Spannungsfeld, Verzerrungen einer Dimension haben Auswirkungen auf die anderen.
- QS ist ein wesentlicher Bestandteil aller Vorgehensmodelle, ist jedoch je nach Modell unterschiedlich verankert.

Referenzen

- Thomas Grechenig, Mario Bernhart, Roland Breiteneder, Karin Kappel: „Softwaretechnik: Mit Fallbeispielen aus realen Entwicklungsprojekten“, Pearson Studium, 2009
- ISO/IEC 25010., ISO/IEC 25010:2011, Systems and software engineering, Systems and software Quality Requirements and Evaluation (SQuaRE), System and software quality models, 2011
- IEEE Standard for Software Quality Assurance Processes, in IEEE Std 730-2014 (Revision of IEEE Std 730-2002), 2014
- Neil Walkinshaw, „Software Inspections, Code Reviews, and Safety Arguments“, 2017
- Daniel Galin, “Software Quality: Concepts and Practice”, IEEE Computer Society, Inc., 2018
- Gernot Starke, “Kolumne: Quality-driven Software Architecture: Herausforderungen mit Qualität”, Javamagazin, 1/2023, S.50-52
- Claude Laporte, Alain April, “Software Quality Assurance”, Wiley, 2018