Foto: Devon Divine auf Unsplash

Testing

Teil 4 - Dynamische Tests

INNOVET
KI B³
Künstliche Intelligenz in die berufliche Bildung bringen

GEFÖRDERT VOM
INNOVET
Bundesministerium für Bildung und Forschung
bibb Bundesinstitut für Berufsbildung

Gefördert als InnoVET-Projekt aus Mitteln des Bundesministeriums für Bildung und Forschung.

Übersicht über dynamische Tests

- Testobjekt: Ein ausführbarer Teil des Codes (komplettes System, Teilsystem, einzelne Komponente)
- Ziele der Prüfung:
  - Fehlerorientiert, d.h. Aufdeckung von Fehlern.
    - Wo ist es wahrscheinlich, dass Fehler gemacht worden sind?
  - Konformitätsorientiert, d.h. Prüfung, ob das Prüfobjekt mit seiner Spezifikation übereinstimmt.
    - Test der erforderlichen Funktionalität.
    - Auch genannt: Konformitätstest oder Abnahmetest.
  - In der Praxis wird eine Mischung aus beidem verwendet:
    - Fehlerorientierte Tests überprüfen auch die in der Spezifikation beschriebene Funktionalität,
    - Konformitätsorientierte Tests sollten versuchen, Fehler aufzudecken.
- Prüfstrategie: Verschiedene Techniken für die Erstellung von Testfällen (Whitebox, Blackbox).

Unstrukturiertes vs. systematisches Testen

- Unstrukturierte Tests:
  - Ausprobieren:
    - Entwickler erstellt und startet sein Programm.
    - Schließlich läuft das Programm und der Entwickler kann es ausprobieren.
    - Entwickler stolpert nur über Ergebnisse, die offensichtlich falsch sind.
  - Wegwerftest:
    - Jemand führt das Programm aus und gibt Daten ein, die auf bestimmten Annahmen beruhen,
    - Er/sie analysiert die Ausgabedaten und beobachtet möglicherweise Fehler, aber die Tests sind nicht wiederholbar.

Unstrukturiertes vs. systematisches Testen

Systematische Tests

- Identifizieren Sie die Ziele und Voraussetzungen für jeden Test.
- Entwicklung: Konstruieren Sie Testfälle mit definierten Eingabewerten und erwarteten Ergebnissen unter Verwendung einer gewählten Testentwurfsstrategie.
- Planung: Skizzieren Sie die genauen Schritte für jeden Test, zusammen mit der Reihenfolge, in der sie durchgeführt werden sollen.
- Ausführung: Führen Sie die Anwendung unter Verwendung der in den Testszenarien beschriebenen Eingaben aus.
- Bewertung: Vergleichen Sie die tatsächlichen Ergebnisse mit den erwarteten Ergebnissen.
- Ergebnisdokumentation: Dokumentieren Sie die Konfiguration der Software, die für die Prüfung verwendeten Testdaten und die erzielten Ergebnisse.

Definition der Begriffe

Testfall: Enthält detaillierte Informationen über:

- Vorbedingungen und Nachbedingungen des Testfalls,
- Eingabedaten (pro Testfall genau ein Satz von Eingabedaten) und erwartete Ausgabedaten.

Test-Skript (oft auch Testfall oder ausführbarer Testfall genannt):

- Detaillierte Schritte zur Ausführung eines Testfalls (für die Ausführung durch Menschen oder Computer).

Test-Suite: Satz von Testfällen

- Strukturierung der Testfälle in Gruppen (z. B. auf der Grundlage der abgedeckten Anforderungen oder der Systemstruktur).
- Kann eine (optimale) Reihenfolge der Ausführung festlegen.

Testlauf: Ausführung einer Reihe von Testfällen für eine bestimmte Version/Konfiguration des Testobjekts.

Zur Erinnerung: Dynamische Tests auf den unteren Teststufen

- Dynamisches Testen erfordert die Ausführung des Testobjekts:
  - Ein ausführbares Programm muss vorhanden sein!

- Einzelne Klassen, Module (Komponententests) und integrierte Teilsysteme (Integrationstests) sind oft nicht allein ausführbar:
  - Abhängigkeiten zu anderen Softwareelementen, da externe Methoden/Funktionen aufgerufen werden,
  - Tests benötigen Methoden/Funktionen, die keine Benutzeroberfläche haben, aber zum Testen von einem anderen Softwareelement aufgerufen werden können.
- In den unteren Teststufen (Komponententest, Integrationstest) muss das Testobjekt in eine Testumgebung eingebettet werden, um es ausführbar zu machen und die Tests zu steuern.

Techniken für den Testfallentwurf

- **Black-Box-Techniken:** Basiert auf den Spezifikationen und Anforderungen der Software.
  - **Äquivalenz-Partitionierung:** Unterteilt die Eingabedaten der Anwendung in Partitionen äquivalenter Daten, aus denen Testfälle abgeleitet werden können.
  - **Grenzwertanalyse:** Konzentriert sich auf die Werte an den Grenzen von Äquivalenzpartitionen.
  - **Entscheidungstabelle:** Verwendet eine Tabelle zur Darstellung logischer Beziehungen zwischen Eingaben (Bedingungen) und Aktionen (Regeln), um die Testszenarien zu bestimmen.
  - **Prüfung von Zustandsübergängen:** Testen des Übergangs von einem Zustand in einen anderen, um sicherzustellen, dass die Übergänge die erwarteten Ergebnisse liefern.
  - **Use Case Testing:** Ableitung von Testfällen auf der Grundlage von Anwendungsfällen, die die Interaktionen zwischen einem Benutzer und dem System beschreiben.

Techniken für den Testfallentwurf

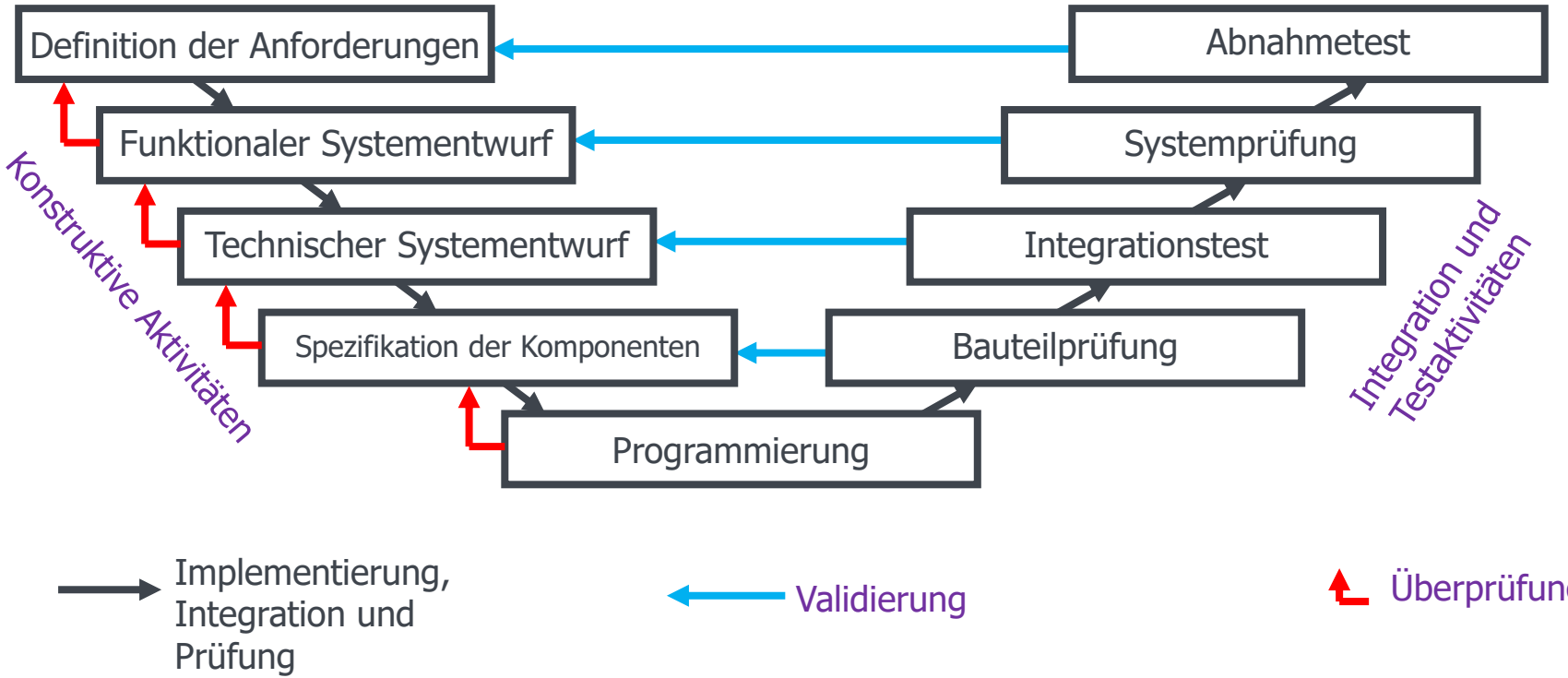
- **White-Box-Techniken:** Basiert auf der internen Logik und Struktur des Codes.
  - **Anweisungsabdeckung:** Stellt sicher, dass jede Codezeile mindestens einmal ausgeführt wird.
  - **Verzweigungsabdeckung:** Stellt sicher, dass bei jeder Verzweigung alle möglichen Zweige getestet werden. Erweitert die Anweisungsabdeckung.
  - **Pfadabdeckung:** Stellt sicher, dass alle möglichen Pfade durch den Code ausgeführt werden, also alle möglichen Kombinationen von Zweigen. Erweitert die Pfadabdeckung.
  - **Bedingungsabdeckung:** Stellt sicher, dass jede Bedingung in einer Verzweigung mindestens einmal *wahr* und einmal *falsch* war. Erweitert die Anweisungsüberdeckung.

Techniken für den Testfallentwurf

- **Gray-Box-Techniken:** Kombiniert sowohl Black-Box- als auch White-Box-Techniken, um Testfälle zu entwerfen.

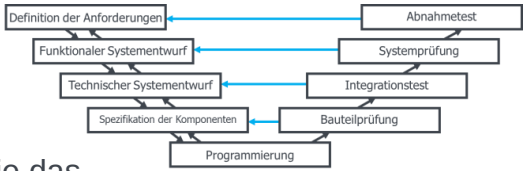
  - **Paarweises Testen (All-Pairs Testing):** Testet alle möglichen diskreten Kombinationen von zwei Eingabeparametern.
  - **N-Tuple Testen:** Erweitert die paarweisen Tests auf *n-Tupel*, um komplexere Kombinationen und Szenarien abzudecken. Es werden alle möglichen Kombinationen von *n* Parametern getestet

Dynamische Prüfung und V-Modell



Geschichte des V-Modells

- V-Model (1. Version 1987) ist das Ergebnis eines Softwareentwicklungsprojekts (1986) des Bundesministeriums der Verteidigung.

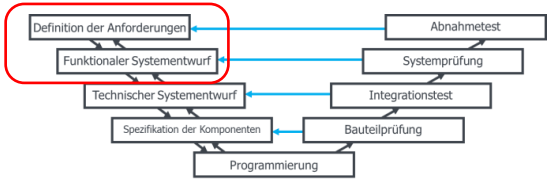


- Die (erste) zivile Variante des V-Models wurde 1991 entwickelt.
  - Andere Regierungsstellen waren mit denselben Softwareproblemen konfrontiert wie das Verteidigungsministerium.
- 1997 Aktualisierung des V-Modells zur Unterstützung neuer Softwareentwicklungsstrategien
- Feb. 2005 wurde das V-Modell XT (XT = eXtreme Tailoring) veröffentlicht. Das V-Modell XT
  - bezieht die Rolle des Kunden in das Modell ein,
  - unterstützt die Modularisierung von Software, und
  - unterstützt agile und inkrementelle Softwareentwicklungsstrategien.

Konstruktive Aktivitäten

- Die Aktivitäten auf dem linken Zweig sind konstruktive Aktivitäten

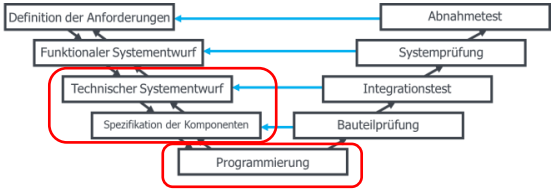
  (wie aus dem Wasserfallmodell bekannt).



- Das Softwaresystem wird auf jeder Ebene der konstruktiven Aktivitäten detaillierter.

- Definition der Anforderungen
  - Die Bedürfnisse und Anforderungen des Kunden/Benutzers werden erfasst, schriftlich festgehalten und genehmigt.
  - Der Zweck des Systems und die gewünschten Merkmale und Eigenschaften werden definiert.
- Funktionaler Systementwurf
  - Anforderungen werden auf Funktionen und Benutzerinteraktionen des Softwaresystems abgebildet.

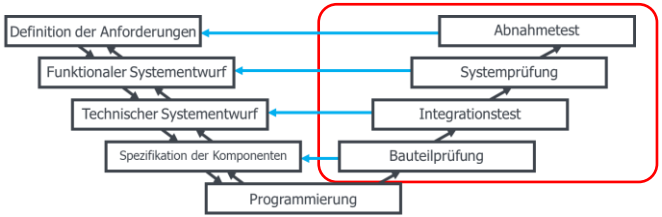
Konstruktive Aktivitäten



- Technischer Systementwurf:
  Die Implementierung des Systems ist entworfen: Benutzeroberfläche.
  - Zerlegung in Teilsysteme/Entwurf einer Systemarchitektur.
- Spezifikation der Komponenten:
  Für jedes Teilsystem/Komponente: Definition von
  - Aufgabe,
  - Verhalten,
  - Innere Struktur,
  - Schnittstellen zu anderen Teilsystemen.
- Programmierung:
  - Implementierung der Komponente als Modul in einer Programmiersprache.

Test-Aktivitäten

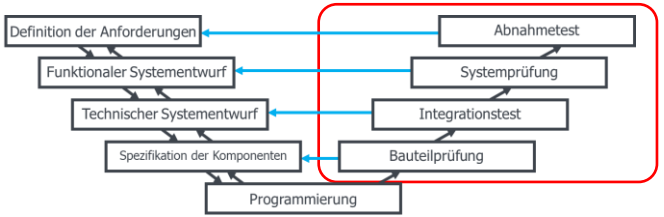
- Der einfachste Weg, Fehler zu finden, die bei konstruktiven Tätigkeiten gemacht wurden, ist, sie auf derselben Ebene zu identifizieren, auf der sie gemacht wurden.



- Der rechte Zweig des V-Model enthält eine Testebene für jede Konstruktionsebene..
- Bauteilprüfung:
  - Prüft, ob jede Komponente gemäß ihrer Komponentenspezifikation funktioniert.

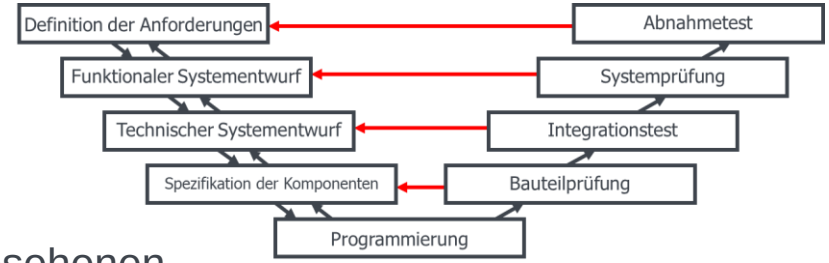
Test-Aktivitäten

- Integrationstest:
  - Prüft, ob Gruppen von Komponenten gemäß dem technischen Systementwurf zusammenarbeiten.
- Systemprüfung:
  - prüft, ob das System als Ganzes gemäß dem funktionalen Systementwurf funktioniert.
- Abnahmetest:
  - Prüft, ob das System als Ganzes gemäß der Anforderungsdefinition (die oft Teil eines Vertrags ist) funktioniert, d. h. aus Sicht der Kunden/Nutzer.



Validierung

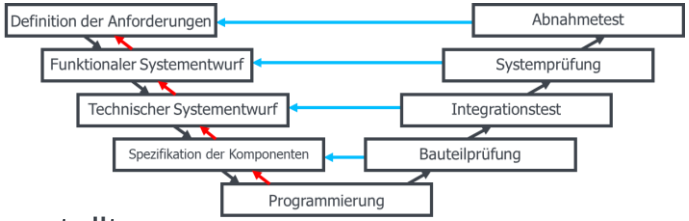
- Jede Testebene prüft, ob das Ergebnis der Entwicklung die auf dieser Abstraktionsebene spezifizierten/relevanten Anforderungen erfüllt.
- Die zu beantwortende Frage:
  - Haben wir das richtige System gebaut?



- Der Prüfer beurteilt, ob ein Produkt für den vorgesehenen Verwendungszweck geeignet ist.
- Da es sich hierbei um eine Überprüfung der ursprünglichen Anforderungen handelt, kann dies als Validierung betrachtet werden..

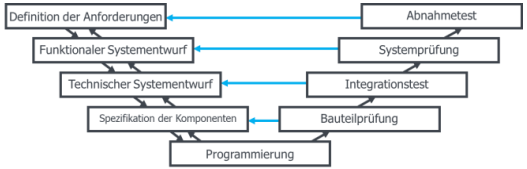
Überprüfung

- Das V-Model enthält auch eine Überprüfung:
  - Die Verifizierung bezieht sich nur auf eine einzige Entwicklungsstufe.
  - Die Verifizierung stellt sicher, dass das Ergebnis jeder Stufe entsprechend den Eingabedokumenten der jeweiligen Stufe erstellt wurde.
  - Es wird geprüft, ob die Eingabespezifikation korrekt in das Ausgabedokument umgewandelt wurde.
    - Es wird jedoch (im Gegensatz zur Validierung) nicht geprüft, ob die Ausgabe für den beabsichtigten Zweck geeignet ist. Wenn z.B. bereits die Eingabe falsch ist, würde eine Transformation, die als korrekt verifiziert wurde, auch eine falsche Ausgabe ergeben.
- Die zu beantwortende Frage:
  - Haben wir das System richtig aufgebaut?



Zusammenfassung

- Umsetzung und Prüfung sind gleichermaßen wichtig.
- "V" veranschaulicht die Testaspekte der Verifizierung und Validierung.



- Unterscheidet verschiedene Teststufen:
  - Nicht nur Tests zu verschiedenen Zeitpunkten eines Projekts, sondern Tests auf verschiedenen Abstraktionsebenen mit unterschiedlichen Zielen, unterschiedlichen Methoden, unterschiedlichen Werkzeugen und unterschiedlichem Personal.
  - Jede Testebene testet gegen die entsprechenden Dokumente der Entwicklungsebene.

- Das V-Model kann den Eindruck erwecken, dass das Testen spät beginnt (nachdem die Programmierung abgeschlossen ist). Dies ist jedoch nicht wahr:
  - Der rechte Zweig des "V" steht für die Testdurchführung (& Integration) selbst.
  - Die Testvorbereitung (Planung, Spezifikation) soll bereits im Rahmen der konstruktiven Tätigkeiten erfolgen (linker Zweig des "V").

INNOVET
KI B³
Künstliche Intelligenz in die berufliche Bildung bringen

Tobias Eisenreich

Universität Stuttgart
Institut für Software Engineering
Empirisches Software Engineering

Umm-e-Habiba

Universität Stuttgart
Institut für Software Engineering
Empirisches Software Engineering

Universität Stuttgart
Institut für Maschinelle Sprachverarbeitung
Institut für Software Engineering

IHK Industrie- und Handelskammer Reutlingen
Reutlingen | Tübingen | Zollernalb

IHK Region Stuttgart

IHK Industrie- und Handelskammer Karlsruhe

LMU Ludwig-Maximilians-Universität München

GEFÖRDERT VOM
INNOVET
Bundesministerium für Bildung und Forschung
bibb Bundesinstitut für Berufsbildung

Gefördert als InnoVET-Projekt aus Mitteln des Bundesministeriums für Bildung und Forschung.

Lizenzbestimmungen

Quellenverzeichnis

Titelfoto: https://unsplash.com/de/fotos/Hzp-1ua8DVE