Grundbegriffe Softwaretesting

Testing

Testen

Testen ist nicht die Suche nach Fehlern, sondern die Suche nach "Symptomen" auch Fehlerwirkung genannt. Zusätzlich wird die Qualität der Software geprüft.

`Anforderungsprüfung, Qualitätsprüfung, Suche nach Fehlerwirkungen`

Debuggen

Hilfsmittel zur Fehlersuche (Defekt im Quellcode, logischer Fehlerzustand)

Statisches Testen

Voraussetzung

Dokumente

Manuelle Testverfahren

bspw. Review == strukturiertes Vorgehen, Meetings zur Dokumentenprüfung

automatisierte Testverfahren

statische Analyse vom Compiler, Prüfung von Programmierrichtlinien

Dynamisches Testen

Voraussetzung

ausführbares Programm

Testfallentwurfsverfahren

Black-Box

Basis = Pflichtenheft, Testorakel
Äquivalenzklassen, Grentzwertverfahren, Anwendungsfälle, Zustände

White-Box Verfahren

Basis = Quellcode
Überdeckung

Teststufen

(wann teste ich?) 1. Modul bzw Unittest
2. Integrationstest
3. Systemtest
4. Abnahmetest
=> eher Anwendung von dynamischen Testverfahren
5. Regressionstest

Qualitätsanforderungen testen

- Performancetest
- Lasttest
- Stresstest
- Tests auf Benutzerfreundlichkeit
- Robustheitstest

Fehlerwirkung

Ein inkorrektes Verhalten, oder welches den Qualitätskriterien widerspricht

Defekt

Tatsächlicher Fehler im Programmcode bzw. im System.

Testorakel

Informationen darüber, was zu testen ist, beispielsweise das Pflichtenheft, Expertenwissen

Tesfall

Ein Testfall umfasst laut GTB-Glossar folgende Angaben:

- die für die Ausführung notwendigen Vorbedingungen
- die Menge der Eingabewerte (ein Eingabewert je Parameter des Testobjekts)
- die Menge der vorausgesagten Ergebnisse
- sowie die erwarteten Nachbedingungen

Testfälle werden entwickelt im Hinblick auf ein bestimmtes Ziel bzw. auf eine Testbedingung, wie z.B. einen bestimmten Programmpfad auszuführen oder die Übereinstimmung mit spezifischen Anforderungen zu prüfen (wie Eingaben an das Testobjekt zuübergeben und Sollwerte abzulesen sind).

Testfallentwurfsverfahren

Bei den hier vorgestellten Testverfahren lassen sich in Abhängigkeit von der Testvoraussetzung in folgende Kategorien einteilen
Dynamische Testverfahren
Statische Testverfahren

Qualitätskriterien

Robusheit

Vermeidung von Systemabstürzen bei Fehlbedienung, falschen Daten oder Hardwarefehlern

Portabilität

Softwareprodukte sollten auf verschiedenen Hardwareplattformen einsetzbar sein (mit Ausnahmen)

Verifizierbarkeit

Integrität

Schutz gegen unberechtigten Zugriff

Benutzerfreundlichkeit

optimaler Bedienungskomfort = Adäquatheit
Klar- und Einfachheit der Benutzerschnittstelle = Erlernbarkeit
Robustheit

Kompatibilität

Einfache Schnittstellen zu anderen Softwareprodukten

Wiederverwendbarkeit

Verwendung von Teilen des Programmcodes in anderen Anwendungen

Erweiterbarkeit

Einfache Anpassung bei Spezifikationsänderungen

Korrektheit

Zugrundeliegende Spezifikation ist erfüllt

Äquivalenzklassenverfahren

Für einen Wertebereich werden Repräsentanten gewählt die für jeden Wert in diesem Bereich bei einem Test den selben Rückgabewert erzielen müssten.

Konstruktion von Testfällen

Ermitteln aller Grenzwerte die zu Fehlerzuständen führen könnten.
Ersetzen durch Repräsentanten

Bestandteile eines Tesfalles

1. Vorbedinungen (müssen vor Testausführung hergestellt werden)
2. Benennung testobjekt und Spezifikation
3. Eingabedaten festlegen
4. Handlungen die zur Durchführung nötig sind
5. erwartete Ergebnisse/Reaktionen
6. erwartete Nachbedingungen (Ergebnis der Durchführung)
7. Prüfanweisungen (wie sind Eingaben an Testobjekt zu übergeben)

Testendkriterium

legen fest wann der Test beendet werden kann. Bspw. eine Testabdeckungsquote von 90%

Testfallüberdeckung

Anzahl getesteter Äquivalenzklassen / Anzahl Äquivalenzklassen

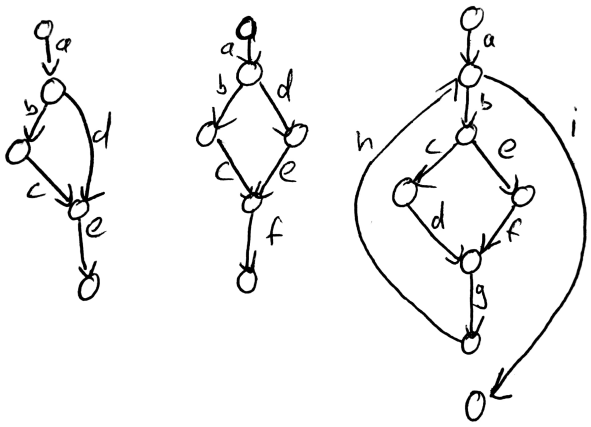
Überdeckung Grenzwerte

Anzahl getesteter Grenzwerte / Anzahl Grenzwerte

Kontrollflussgraph:

Kontrollstrukturen mithilfe von Knoten und Kanten darstellen
Sequenzen in einem Knoten zusammenfassen.

Whiteboxverfahren



Beispiel

C0 Anweisungsüberdeckung

- BSP1: TF1 `A B C E`
- BSP2: TF1 `A B C F`
- BSP2: TF2 `A D E F`
- BSP3: TF1 `A B C D G H I`
- BSP3: TF2 `A B E F G H I`
- BSP3: TF3 `A B C F G H B E F G H I`

C1 Zweigüberdeckung

(=Kanten bzw Pfeile im Kontrollflussgraph)
= Entscheidungsüberdeckung

- BSP1: TF1, TF2 = `A D E`
- BSP2: TF1 und TF 2
- BSP3: Wie C0

Aber C0 != C1, C1 -> C0

C(unendlich) = Pfadüberdeckung

= alle möglichen Wege/Pfade durch den Kontrollflussgraphen
=> vollständiger Test in der Regel nicht möglich

Kompromisslösung:

Cn Test, n>1
n=3
=> 0 - 3 mal wiederholen -> hierbei alle möglichen Pfade.