

Bachelor of Science (BSc) in Informatik
Modul Advanced Software Engineering 2 (ASE2)

LE 09 – Software Testing

5 Dynamischer Test (Teil 2)

Institut für Angewandte Informationstechnologie (InIT)
Walter Eich (eicw) / Matthias Bachmann (bacn)

<https://www.zhaw.ch/de/engineering/institute-zentren/init/>



Agenda

5 Dynamischer Test

5.1 Blackbox-Testverfahren

5.2 Whitebox-Testverfahren

5.3 Erfahrungsbasierte Testfallermittlung

5.4 Auswahl von Testverfahren

5.5 Wrap-up



Lernziele nach Syllabus ISTQB CTFL (2/2)

4.3 White-Box-Testverfahren

FL-4.3.1 (K2) Anweisungsüberdeckung erklären können

FL-4.3.2 (K2) Entscheidungsüberdeckung erklären können

FL-4.3.3 (K2) Die Bedeutung von Anweisungs- und Entscheidungsüberdeckung erklären können

4.4 Erfahrungsbasierte Testverfahren

FL-4.4.1 (K2) Die intuitive Testfallermittlung erklären können

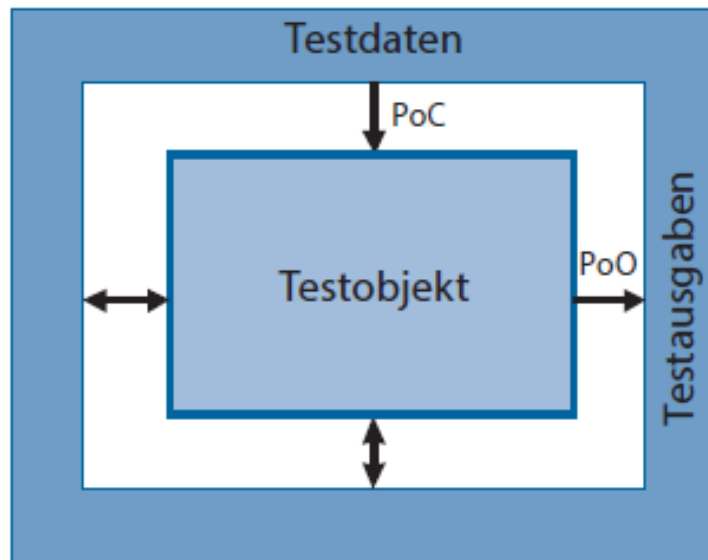
FL-4.4.2 (K2) Exploratives Testen erklären können

FL-4.4.3 (K2) Checklistenbasiertes Testen erklären können



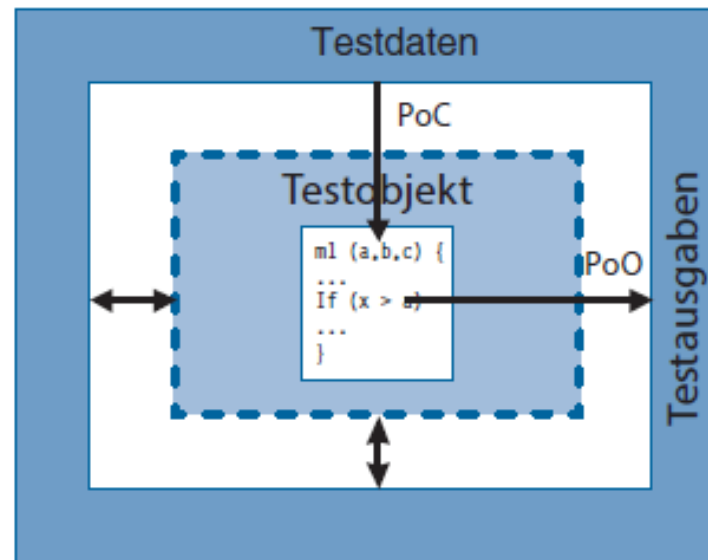
5 Blackbox- vs. Whitebox-Testverfahren

Blackbox-Verfahren



PoC und PoO »außerhalb«
des Testobjekts

Whitebox-Verfahren



PoC und/oder PoO »innerhalb«
des Testobjekts

PoC = Point of Control PoO = Point of Observation



5.2 Whitebox-Verfahren

- **Whitebox-Test**
 - Test unter **Nutzung von Information über Interna** des Testobjekts.
 - Gesucht werden «**fehleraufdeckende**» **Stichproben** der möglichen Programmabläufe und Datenverwendungen.
- **Whitebox-Testverfahren**
 - Alle Testverfahren, die zur Herleitung oder Auswahl der Testfälle sowie zur Bestimmung der Vollständigkeit der Prüfung (Überdeckungsgrad) Information über die innere Struktur des Testobjekts (z.B. Zweige, Pfade, Daten) heranziehen.
 - Daher auch **strukturelle oder strukturbasierte Testverfahren** genannt.
 - Ein weiter Begriff ist **codebasierte Testverfahren**.
 - Whitebox-Testverfahren können in allen Teststufen genutzt werden, aber die im Folgenden beschriebenen codebasierten Verfahren werden **am sinnvollsten in der Komponententeststufe** eingesetzt.



5.2 Whitebox-Verfahren

Es lassen sich folgende **Whitebox-Testverfahren** unterscheiden:

- **Anweisungstest**
- **Entscheidungstest (Zweigtest)**
- **Test der Bedingungen**
 - (einfacher) Bedingungstest
 - Mehrfachbedingungstest
 - Modifizierter bzw. minimaler Bedingungs-/Entscheidungstest
- **Pfadtest**



5.2.1 Anweisungstest und Anweisungsüberdeckung

- Die **einzelnen Anweisungen** (engl. «Statements») des Testobjekts stehen im Mittelpunkt der Untersuchung.
- Es sind Testfälle zu identifizieren, die eine zuvor **festgelegte Mindestquote** oder auch alle Anweisungen des Testobjekts zur Ausführung bringen.
- Das Verfahren (und der Entscheidungstest) kann sehr gut an einem **Kontrollflussgraphen** veranschaulicht und erklärt werden.
- Kriterium zur Beendigung der Tests:
 - Anteil der ausgeführten Anweisungen (statement coverage)
 - Auch als C0-Mass bezeichnet



5.2.1 Anweisungstest und Anweisungsüberdeckung

- Dynamisches, kontrollflussbasiertes Testverfahren, das die **mindestens einmalige Ausführung aller Anweisungen** des Testobjekts fordert.

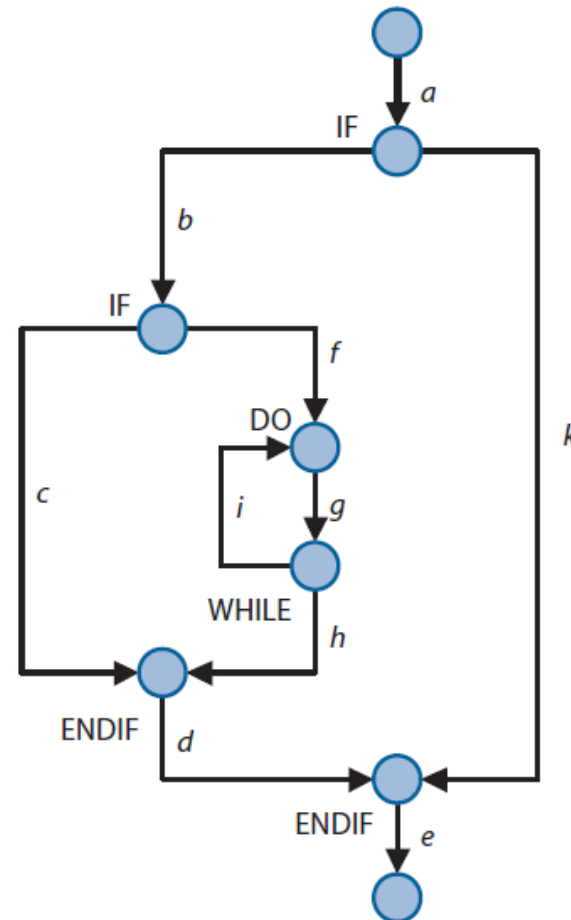
$$\text{Anweisungsüberdeckung} = \frac{\text{Anzahl durchlaufener Anweisungen}}{\text{Gesamtzahl Anweisungen}} * 100 \%$$

- Jeder Testfall wird anhand eines Pfads durch den Kontrollflussgraphen bestimmt.
 - Bei dem Testfall müssen die auf dem Pfad liegenden Kanten des Graphen durchlaufen werden, d.h. die Anweisungen (Knoten) in der entsprechenden Reihenfolge zur Ausführung kommen.
 - Bei der Berechnung wird nur gezählt, ob eine Anweisung bei der Ausführung überhaupt durchlaufen wurde, die Häufigkeit der Ausführung spielt keine Rolle.
- Wenn der zuvor festgelegte Deckungsgrad erreicht ist, wird der Test als ausreichend angesehen und beendet.





5.2.1 Anweisungsüberdeckung - Beispiel

- 1 Testfall: a, b, f, g, h, d, e



Legende:

-  Knoten, Anweisung
-  Kante, Kontrollfluss
- n* Benennung der Kanten



5.2.1 Diskussion Anweisungsüberdeckung

- Die **Anweisungsüberdeckung** ist ein in der Aussage **schwaches Kriterium**.
 - Für die Anweisungsüberdeckung ist eine «leere» Kante, d.h. eine Kante, die lediglich ein oder mehrere Knoten überbrückt, ohne Bedeutung.
 - Beispiele: (ELSE-)Kante (zwischen IF und ENDIF) mit leerem ELSE Teil; Rücksprung zum Anfang einer Repeat-Schleife; BREAK
 - Möglicherweise fehlende Anweisungen in dem enthaltenden Programmteil werden nicht erkannt!
- Eine 100%ige Überdeckung der Anweisungen ist in der Praxis nicht immer erreichbar.
 - Z.B. wenn Ausnahmebedingungen im Programm vorkommen, die während der Testphase nur mit erheblichem Aufwand oder gar nicht herzustellen sind.
 - Kann aber auch auf nicht erreichbare Anweisungen («dead code») hindeuten (ggf. statische Analyse durchführen).



5.2.2 Entscheidungstest und Entscheidungsüberdeckung

- Ein weiter gehendes Kriterium im Whitebox-Test ist die **Überdeckung der Entscheidungen**.
- **Entscheidungen** im Programmtext stehen im **Mittelpunkt** der Untersuchung.
- Nicht die Ausführung der einzelnen Anweisungen wird betrachtet, sondern **die Auswertung einer Entscheidung**.
- Aufgrund des Ergebnisses wird entschieden, welche Anweisung als nächste ausgeführt wird.
- Kriterium zur Beendigung der Tests:
 - Anteil der ausgeführten Programmzweige (branch coverage)
 - Grundlage bildet in der Regel der Kontrollflussgraph (Kantenüberdeckung)
 - Auch als C1-Mass bezeichnet



5.2.2 Entscheidungstest und Entscheidungsüberdeckung

- Kontrollflussbasiertes Testverfahren, fordert die **Überdeckung aller Entscheidungen bzw. Zweige** eines Testobjekts zu allen Fällen.
- Zweigüberdeckung (branch coverage) zielt darauf ab, alle Zweige im Kontrollflussgraphen abzudecken, jeder Testfall wird anhand eines Pfads durch den Kontrollflussgraphen bestimmt. Durch die Testfälle müssen alle Zweige (Kanten) des Graphen durchlaufen werden (auch die «leeren» Zweige, die Knoten bzw. Anweisungen «überbrücken») – jede Entscheidung wird also zu allen Möglichkeiten ausgewertet.

$$\text{Zweigüberdeckung} = \frac{\text{Anzahl durchlaufener Zweige}}{\text{Gesamtzahl Zweige}} * 100 \%$$



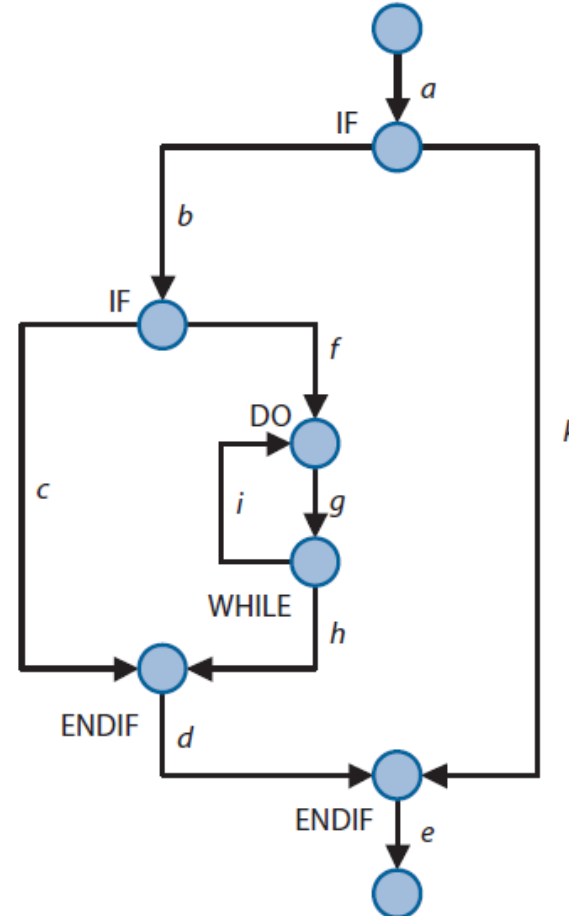
5.2.2 Entscheidungsüberdeckung - Beispiel

+3 Testfälle:

a, b, c, d, e

a, b, f, g, i, g, h, d, e

a, k, e



Legende:

● Knoten, Anweisung

→ Kante, Kontrollfluss

n Benennung der Kanten



5.2.2 Diskussion

Entscheidungsüberdeckung

- Beim Entscheidungs- oder Zweigtest ist die Ausführung von **mehr Testfällen als beim Anweisungstest** erforderlich.
- Wie viel mehr, das hängt von der **Struktur des Testobjekts** ab (Hinweis durch die zyklomatische Zahl).
- Im Gegensatz zum Anweisungstest können mit dem Entscheidungstest **fehlende Anweisungen in leeren IF-Statements erkannt** werden.
- **100% Entscheidungsüberdeckung garantiert 100% Anweisungsüberdeckung**, aber nicht umgekehrt.
- Entscheidungsüberdeckung ist das **umfassendere Kriterium**.
- Die einzelnen Entscheidungen werden unabhängig voneinander betrachtet und es werden **keine bestimmten Kombinationen** der einzelnen Programmteile gefordert.



5.2.3 Test der Bedingungen (Exkurs)

- Wahrheitswerte: false, true (oft auch 0, 1)
- Atomare Teilbedingungen
 - Variablen vom Typ boolean
 - Operationen mit Rückgabewert vom Typ boolean
 - Vergleichsoperationen
 - Z.B. flag; isEmpty(); size > 0
- Zusammengesetzte Bedingungen
 - Verknüpfen atomare Teilbedingungen mit booleschen Operatoren
- Entscheidungen sind zusammengesetzte Bedingungen, die den Programmablauf steuern
- In Java:
 - &, |, ^ Bitweise und, oder und exklusiv-oder-Verknüpfung
 - &&, || Wie oben, aber lazy evaluation, z.B. a && b a ? b : false
 - if ((size > 0) && (inObject != null)) {...} else {...}



5.2.3 Test der Bedingungen

- Bei der **Zweig-/Entscheidungsüberdeckung** wird **ausschliesslich der ermittelte Ergebnis-Wahrheitswert** einer Bedingung berücksichtigt.
 - Anhand dieses Wertes wird entschieden, welche Verzweigung im Kontrollflussgraphen verfolgt wird bzw. welche Anweisung als nächste im Programm zur Ausführung kommt.
 - Problem: Setzt sich eine Bedingung aus mehreren Teilbedingungen zusammen, die über logische Operatoren miteinander verknüpft sind, so muss im Test die strukturelle Komplexität der Bedingung berücksichtigt werden.
- Hierbei werden unterschiedliche Anforderungen und damit auch Abstufungen der Testintensität mit Berücksichtigung der zusammengesetzten Bedingungen unterschieden.



5.2.3 Test der Bedingungen

- Als **Überdeckungskriterien** werden **Verhältnisse zwischen den bereits erreichten und allen geforderten Wahrheitswerten** der (Teil-) Bedingungen gebildet.
- Bei den Verfahren, welche die Komplexität der Bedingungen im Programmtext des Testobjekts in den Mittelpunkt der Prüfung stellen, ist es sinnvoll, eine vollständige Prüfung (100%ige Überdeckung) anzustreben.
- Wir betrachten:
 - **Einfache Bedingungsüberdeckung** (engl. Condition Testing)
 - **Mehrfachbedingungsüberdeckung** (engl. Multiple Condition Testing)
 - **Modifizierter bzw. minimaler Bedingungs-/Entscheidungstest** (engl. Modified Condition Decision Coverage Testing, MCDC)



5.2.3 Einfache Bedingungsüberdeckung

- Kontrollflussbasiertes, dynamisches Testverfahren, das die **Überdeckung der atomaren Teilbedingungen einer Entscheidung** mit «wahr» und «falsch» fordert.
 - Teste jeden atomaren Ausdruck einmal zu wahr und einmal zu falsch.
- Bei n atomaren Ausdrücken mindestens 2, höchstens 2^n Testfälle

$$\text{Bedingungsüberdeckung} = \frac{\text{Anzahl zu wahr und falsch getesteten atom. A.}}{\text{Gesamtzahl atomarer Ausdrücke}} * 100 \%$$



5.2.3 Einfache Bedingungsüberdeckung

Teste jeden atomaren Ausdruck einmal zu wahr und einmal zu falsch!

A	B	$A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

2 Ausdrücke, 2 Testfälle

A	B	$A \vee B$
0	0	0
0	1	1
1	0	1
1	1	1

2 Ausdrücke, 2 Testfälle

A	B	C	$A \wedge B \wedge C$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

3 Ausdrücke, 2 Testfälle



5.2.3 Mehrfachbedingungsüberdeckung

- Teste **jede Kombination der Wahrheitswerte aller atomaren Ausdrücke**. Bei n atomaren Ausdrücken (atomare A.) ist die Testfall-Anzahl = 2^n
 - Wächst exponentiell mit der Anzahl unterschiedlicher atomarer Ausdrücke!

$$\text{Mehrfachbedingungsüber.} = \frac{\text{Anzahl getesteter Kombinationen atom. A.}}{2^{\text{Gesamtzahl atomarer Ausdrücke}}} * 100 \%$$

- Bei der Auswertung der Gesamtbedingung ergeben sich auch beide Wahrheitswerte.
 - Die Mehrfachbedingungsüberdeckung erfüllt somit auch die Kriterien der Anweisungs- und Entscheidungsüberdeckung.
 - Sie ist ein umfassenderes Kriterium, da sie auch die Komplexität bei zusammengesetzten Bedingungen berücksichtigt.

5.2.3 Mehrfachbedingungsüberdeckung

Teste jede Kombination der Wahrheitswerte aller atomarer Ausdrücke!

A	B	$A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

2 Ausdrücke, 4 Testfälle

A	B	$A \vee B$
0	0	0
0	1	1
1	0	1
1	1	1

2 Ausdrücke, 4 Testfälle

A	B	C	$A \wedge B \wedge C$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

3 Ausdrücke, 8 Testfälle



5.2.3 Modifizierter Bedingungs- /Entscheidungstest

- Teste den (Gesamt-)Ausdruck einmal zu wahr und einmal zu falsch sowie jede Kombination von Wahrheitswerten, bei denen die Änderung des Wahrheitswertes eines atomaren Ausdrucks den Wahrheitswert des zusammengesetzten Ausdrucks ändern kann (MM-Kombinationen)!

$$\text{Minimale Mehrfachbeding.} = \frac{\text{Anzahl getesteter MM-Kombinationen}}{\text{Gesamtzahl MM-Kombinationen}} * 100 \%$$



5.2.3 Modifizierter Bedingungs- /Entscheidungstest

Teste den (Gesamt-)Ausdruck einmal zu wahr und einmal zu falsch sowie jede Kombination von Wahrheitswerten, bei denen die Änderung des Wahrheitswertes eines atomaren Ausdrucks den Wahrheitswert des zusammengesetzten Ausdrucks ändern kann!

A	B	$A \wedge B$
0	0	0
0	1	0
1	0	0
1	1	1

A	B	$A \vee B$
0	0	0
0	1	1
1	0	1
1	1	1

A	B	C	$A \wedge B \wedge C$
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

2 Ausdrücke, 3 Testfälle

2 Ausdrücke, 3 Testfälle

3 Ausdrücke, 4 Testfälle



5.2.3 Modifizierter Bedingungs-/Entscheidungstest - Beispiel

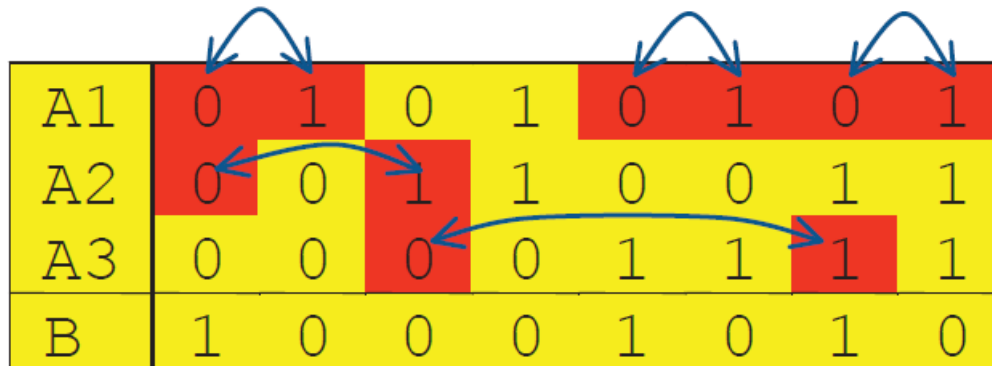
- Seien A_1 , A_2 , A_3 atomare Ausdrücke und $B=f(A_1, A_2, A_3)$ der durch die unten abgebildete Wahrheitswerttabelle definierte Gesamtausdruck.
- Frage:** Welche Werte-Kombinationen von A_1 , A_2 und A_3 sind nach der minimalen Mehrfach-Bedingungsüberdeckung mit Testfällen zu testen?

A1	0	1	0	1	0	1	0	1
A2	0	0	1	1	0	0	1	1
A3	0	0	0	0	1	1	1	1
B	1	0	0	0	1	0	1	0



5.2.3 Modifizierter Bedingungs-/Entscheidungstest - Beispiel

- **Antwort:** Mit Testfällen alle Kombinationen (Spalten) testen, in denen mindestens eine rote Markierung ist.
- Da hier MM-Kombinationen existieren, ist sowohl B=0 als auch B=1 abgedeckt und daher kein gesonderter Testfall notwendig, der den (Gesamt-)Ausdruck einmal zu wahr und einmal zu falsch auswertet.



The diagram shows a truth table with 4 rows (A1, A2, A3, B) and 8 columns. Red cells indicate specific combinations of variables. Blue arrows show dependencies between columns: A1 depends on columns 1, 2, 5, 6, 7, and 8; A2 depends on columns 1, 3, and 4; A3 depends on columns 3, 6, and 7.

A1	0	1	0	1	0	1	0	1
A2	0	0	1	1	0	0	1	1
A3	0	0	0	0	1	1	1	1
B	1	0	0	0	1	0	1	0



5.2.3 Pfadtest (Exkurs)

- Der **Pfadtest** (engl. path coverage) fordert die **Ausführung aller unterschiedlichen Pfade** durch ein Testobjekt.
- Ein Pfad beschreibt die **mögliche Abfolge von einzelnen Programmteilen** in einem Programmstück.
- Im Gegensatz dazu werden Zweige unabhängig voneinander betrachtet, jeder für sich.
- Die **Pfade berücksichtigen Abhängigkeiten** zwischen den Zweigen, wie zum Beispiel bei Schleifen, bei denen ein Zweig an den Anfang eines anderen Zweiges zurückführt.
- Kommen im Programmstück noch Schleifen hinzu, so zählt **jede mögliche Anzahl von Schleifenwiederholungen als ein möglicher Pfad** durch das Programmstück.
- Es ist klar, dass eine **100%ige Überdeckung aller Pfade** in einem Programm **während der Testphase nie erreichbar** ist, sobald das Programm nicht trivial ist.



5.2.4 Allgemeine Bewertung der Whitebox-Verfahren

- Grundlage aller hier beschriebenen Whitebox-Verfahren ist der **vorliegende Programmtext**.
- In **Abhängigkeit der Komplexität** der Programmstruktur können die adäquaten Testverfahren ausgewählt und angewendet werden.
- Anhand des Programmtextes und der ausgewählten Verfahren wird die **Intensität der Tests** festgelegt.
- Die dargestellten Whitebox-Verfahren eignen sich für die **unteren Teststufen**.
- Bei den **codebasierten Whitebox-Verfahren** wird gefordert, dass bestimmte Programmteile zur Ausführung kommen bzw. Entscheidungen unterschiedliche Wahrheitswerte annehmen.
- Zur **Ermittlung der Überdeckungsmasse** können entsprechende Werkzeuge genutzt werden (Stichwort Instrumentierung).



Agenda

5 Dynamischer Test

5.1 Blackbox-Testverfahren

5.2 Whitebox-Testverfahren

5.3 Erfahrungsbasierte Testfallermittlung

5.4 Auswahl von Testverfahren

5.5 Wrap-up



5.3 Erfahrungsbasierte Testfallermittlung

- Der **Erfolg systematischer Testverfahren hängt stark von der Qualität der Dokumente ab**, auf deren Grundlage Testfälle definiert werden.
 - Qualität bedeutet Lesbarkeit, Testbarkeit, Vollständigkeit, Eindeutigkeit, Widerspruchsfreiheit, ...
 - Oft sind auch Formalisierungsgrad und Granularität nicht adäquat, um formale bzw. systematische Testverfahren anzuwenden.
- Grundidee erfahrungsbasierter Verfahren
 - Erfahrung/Wissen der Tester wird bei der Definition von Testfällen einbezogen.
 - Kompensation der mangelnden Qualität der Eingangsdokumentation durch die Erfahrung und durch das Wissen der Tester.
- Beispiele für **erfahrungsbasierte Verfahren** sind: **Intuitives Testen** (Error Guessing), **checklistenbasiertes Testen** und **exploratives Testen**



5.3 Intuitive Testfallermittlung (Error Guessing)

- Testentwurfsverfahren, bei dem die **Erfahrung und das Wissen der Tester genutzt** werden, um vorherzusagen, welche Fehlerzustände in einer Komponente oder einem System aufgrund der Fehlhandlungen vorkommen, und um Testfälle so abzuleiten, dass diese Fehler aufgedeckt werden.
- Methode und Herleitung oder Auswahl der Testfälle, gestützt auf Erfahrung und Wissen des Testers.
- Systematische Vorgehensweise
 - Erstelle einen Fehlerkatalog mit möglichen Fehlerzuständen und Fehlerwirkungen
 - Entwerfe Testfälle, die auf diese Fehler abzielen



5.3 Checklistenbasiertes Testen

- Hier dienen **Checklisten als Grundlage** zur Testfallerstellung.
- Eine Checkliste enthält **Aspekte (Testbedingungen)**, die getestet werden müssen. Zu diesen Aspekten gehören beispielsweise **Fehlerbehandlungen und die Ausfallsicherheit**.
- In der Regel werden die Listen im **Laufe der Zeit erstellt** und aktualisiert.
- Sie basieren überwiegend auf den **Erfahrungen der Tester**.
- Auch hier kann eine Art **Überdeckung** in Abhängigkeit der geprüften Checklisteneinträge angegeben werden.
- Wenn **alle Punkte** aus der Checkliste geprüft sind, ist eine **100%ige Überdeckung** erreicht.



5.3 Exploratives Testen (1/2)

- Bei «schlechter» oder fehlender Testbasis (Spezifikation etc.), kann sogenanntes «**exploratives Testen**» helfen.
- Beim explorativen Testen werden die **Testaktivitäten nahezu «parallel»** durchgeführt.
- Die möglichen Elemente des Testobjekts, die einzelnen Aufgaben und **Funktionen werden «erforscht»**.
- Dann wird **entschieden, welche Teile getestet** werden sollen.
- Dabei werden **wenige Testfälle zur Ausführung** gebracht und das **Ergebnis wird analysiert**.
- Mit der Ausführung wird sozusagen das «**unbekannte**» Verhalten des **Testobjekts weiter geklärt**.
- **Auffälligkeiten und weitere Informationen** dienen zur Erstellung der nächsten Testfälle.



5.3 Exploratives Testen (2/2)

- Zur Strukturierung des explorativen Testens können sowohl **definierte Testziele** als auch eine **zeitliche Beschränkung** vorgenommen werden.
- Eine **Beschränkung auf bestimmte Elemente** des Programms, auf bestimmte Aufgaben oder Funktionen ist beim explorativen Testen ebenfalls sinnvoll (beschrieben in einer «Test Charta»).
- Die **Grundideen des explorativen Testens** sind zusammengefasst:
 - Ergebnisse eines Testfalls beeinflussen die Erstellung und die Ausführung der weiteren Testfälle.
 - Während des Testens entsteht ein «mentales» Modell des zu testenden Programms. Das Modell umfasst dabei, was das Programm leistet bzw. wie es «funktioniert» und wie sein Verhalten ist bzw. wie es sein sollte.
 - Gegen dieses Modell wird getestet, wobei die Aufmerksamkeit darauf gerichtet ist, weitere Aspekte und Verhalten des Testobjekts aufzuzeigen, die bisher nicht oder in anderer Form im «mental» Modell vorhanden sind.



Agenda

5 Dynamischer Test

5.1 Blackbox-Testverfahren

5.2 Whitebox-Testverfahren

5.3 Erfahrungsbasierte Testfallermittlung

5.4 Auswahl von Testverfahren

5.5 Wrap-up



5.4 Auswahl von Testverfahren (1/2)

- Allgemeines Ziel ist, mit **wenig Aufwand ausreichend unterschiedliche Testfälle** zu erstellen, die mit einer gewissen Wahrscheinlichkeit vorhandene Fehlerwirkungen nachweisen.
- Die **Verfahren** zur Erstellung der Testfälle sind dabei «**passend**» auszuwählen.
- Einige **Testverfahren** eignen sich besonders gut für den **Einsatz auf einer bestimmten Teststufe**, andere sind **auf allen Teststufen** anwendbar.
- Zur Erstellung der Testfälle nutzen Tester in der Regel eine **Kombination aus verschiedenen Testverfahren**, um mit dem zur Verfügung stehenden Testaufwand (abhängig von Zeit und Budget) die besten Ergebnisse zu erzielen.
- Die Vorgehensweisen bei Testanalyse, Testentwurf und Testrealisierung bei den Testverfahren kann von **sehr informell bis hin zu sehr formal** reichen.



5.4 Auswahl von Testverfahren (2/2)

- Der **passende Grad an Formalität** hängt vom Kontext ab. Beeinflussenden Faktoren sind zum Beispiel:
 - Art des Testobjekts (Komponente oder System)
 - Komplexität der Komponente oder des Systems
 - Einhaltung von Standards
 - Kundenanforderungen oder vertragliche Anforderungen
 - Testziele
 - Risikobewertung (Risikostufe und Risikoarten)
 - Erwartete Nutzung der Software
 - Verfügbare Dokumentation
 - Verfügbare Werkzeuge
 - Kenntnisse und Fähigkeiten des Testers
 - Arten von Fehlerzuständen
 - Softwareentwicklungsmodell
 - Zeit und Budget



Agenda

5 Dynamischer Test

5.1 Blackbox-Testverfahren

5.2 Whitebox-Testverfahren

5.3 Erfahrungsbasierte Testfallermittlung

5.4 Auswahl von Testverfahren

5.5 Wrap-up



Wrap-up

- Allgemeines Ziel ist, **mit wenig Aufwand ausreichend unterschiedliche Testfälle** zu erstellen, die mit einer gewissen Wahrscheinlichkeit **Fehlerwirkungen nachweisen**.
- Vorab zu **klärende Faktoren für die Wahl der Testverfahren**: Art des Testobjektes, formale Dokumentation und Werkzeuge, Einhaltung von Standards, Erfahrung der Tester, Kundenwünsche, Risikobewertung und weitere Faktoren.
- Testen umfasst immer die **Kombination von unterschiedlichen Verfahren**:
 - **Blackbox-Verfahren**: Äquivalenzklassenbildung in Kombination mit Grenzwertanalyse, Zustandsbezogener Test, Test mit Entscheidungstabellen, Anwendungsfallbasierter Test
 - **Whitebox-Verfahren**: Anweisungstest, Zweig- oder Entscheidungstest, Test der Bedingungen, Pfadtest
 - **Erfahrungsbasierte Verfahren**: Intuitive Testfallermittlung, Checklistenbasierte Testfallermittlung und explorative Testen



Ausblick

- Das Thema der nächsten Vorlesung ist:
 - Kap. 6 Testmanagement
 - Kap. 7 Testwerkzeuge

