LE 14
1

Software-Qualitätssicherung

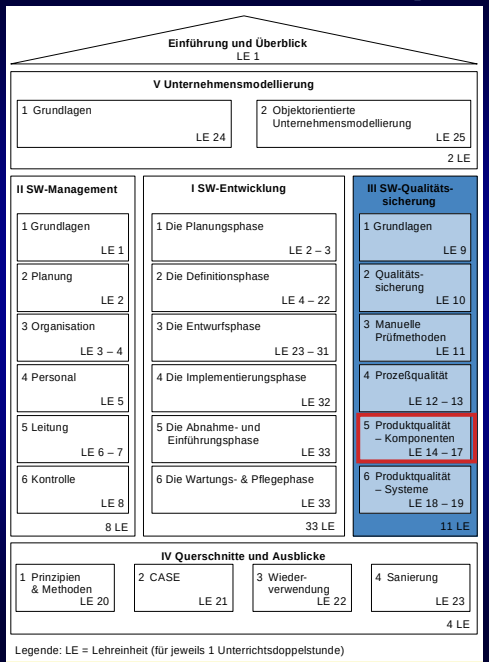
5 Produktqualität – Komponenten: Testende Verfahren 1

[stark gekürzt]

Prof. Dr. Helmut Balzert
Lehrstuhl für Software-Technik
Ruhr-Universität Bochum

© Helmut Balzert 1998

LE 14
2



LE 14
3

Lernziele

- Ein Quellprogramm in einen Kontrollflußgraphen mit der gewünschten Ausprägung umwandeln können
- Aus einem Quellprogramm Testfälle so ableiten können, daß ein vorgegebenes Testkriterium erfüllt wird
- Für ein zu testendes Programm das passende Testverfahren auswählen können
- Ein Quellprogramm instrumentieren können
- Ein Quellprogramm mit Testfällen ausführen können.

LE 14
4

Inhalt

5 Produktqualität – Komponenten
5.1 Einführung und Überblick
Zusätzlich: Einführendes Beispiel in Java
5.2 Klassifikation analytischer Verfahren
5.3 Der Kontrollflußgraph
5.4 Kontrollflußorientierte Strukturtestverfahren
5.4.1 Das Anweisungs- und das Zweigüberdeckungsverfahren
5.4.2 Die Pfadüberdeckungstestverfahren
5.4.3 Die Bedingunsüberdeckungstestverfahren
5.4.4 Zur Auswahl geeigneter Testverfahren.

LE 14
5

5.1 Einführung und Überblick

- Begriffe
- Prüfling, Testling, Testobjekt
  - Die Software-Komponente bzw. das Programm, das getestet werden soll
  - Prüfling ist dabei der allgemeine Begriff
  - Testling impliziert die dynamische Ausführung eines Programms impliziert
- Testfall
  - Besteht aus einem Satz von Testdaten, der die vollständige Ausführung eines zu testenden Programms bewirkt.

LE 14
6

5.1 Einführung und Überblick

- Testdatum
  - Ist ein Eingabewert, der einen Eingabeparameter oder eine Eingabevariable des Testobjekts mit einem Datum im Rahmen eines Testfalls versorgt
- Testtreiber
  - Testrahmen, der ein interaktives Aufrufen von Prozeduren und Funktionen ermöglicht
  - Können durch Testrahmengeneratoren auch automatisch erzeugt werden
  - Ruft der Testling selbst andere Prozeduren oder Funktionen auf, dann müssen diese beim Testen zur Verfügung stehen.

LE 14
7

5.1 Einführung und Überblick

- Instrumentierung
  - Zum Mitprotokollieren, welche Teile des Prüflings bei der Ausführung eines Testfalls durchlaufen wurden
  - In den Quellcode des Testlings werden Zähler eingefügt
  - Nach dem Testlauf werden die Zählerstände ausgewertet
- Überdeckungsgrad
  - Maß für den Grad der Vollständigkeit eines Tests bezogen auf ein bestimmtes Testverfahren.

LE 14
8

5.1 Einführung und Überblick

- Regressionstest
  - Testwerkzeug speichert alle durchgeführten Testfälle
    - Erlaubt die automatische Wiederholung aller bereits durchgeführten Tests nach Änderungen des Prüflings
    - Führt Soll/Ist-Ergebnisvergleich durch.

LE 14
9

5.2 Klassifikation analytischer Verfahren

- **Merkmale dynamischer Tests**
  - **Übersetztes, ausführbares Programm wird mit konkreten Eingabewerten versehen und ausgeführt**
  - **Test in der realen Umgebung**
  - **Stichprobenverfahren**
    - **Korrektheit des getesteten Programms wird nicht bewiesen**
  - **Herkunft der Testfälle**
    - **Aus dem Kontroll- oder Datenfluß**
      - **Dynamischer Strukturtest bzw. *White Box-Test***
    - **Spezifikation der Systemkomponenten**
      - **Funktionstest, funktionaler Test bzw. *Black-Box-Test***
    - **Vergl. der Ergebnisse von Programmversionen**
      - **Disversifizierende Tests.**

Testende Verfahren

Dynamische Testverfahren

Strukturtest (White Box–Test, Glass Box–Test)

Kontrollflußorientierter Test
- Anweisungsüberdeckungstest
- Zweigüberdeckungstest
- Pfadüberdeckungstest (vollständig, strukturiert, boundary interior)
- Bedingungsüberdeckungstest (einfach, minimal mehrfach, mehrfach)

Datenflußorientierter Test
- Defs-Uses–Verfahren
- Required K–Tupels Test
- Datenkontext–Überdeckung

Funktionaler Test (Black Box–Test)
- funktionale Äquivalenzklassenbildung
- Grenzwertanalyse

LE 14
11

5.2 Klassifikation analytischer Verfahren

Statische Testverfahren
- Manuelle Prüfmethoden (Inspektion, Reviews, Walkthroughs, Schreibtischtest) (siehe Hauptkapitel 3)
- Statische Analysatoren

Verifizierende Verfahren
- Verifikation
- Symbolische Ausführung

Analysierende Verfahren
- Analyse der Bindungsart
- Metriken
- Grafiken und Tabellen
- Anomalieanalyse

LE 14
12

5.3 Der Kontrollflußgraph

- **Beispiel für verschiedene Prüfmethoden**
  - **Programmname: ZaehleZchn**
  - **Aufgabe**
    - **Liest solange Zeichen von der Tastatur, bis ein Zeichen erkannt wird, das kein Großbuchstabe ist**
    - **Oder Gesamtzahl den größten durch den Datentyp int darstellbaren Wert INT_MAX erreicht**
    - **Ist ein gelesenes Zeichen ein Großbuchstabe dann wird Gesamtzahl um 1 erhöht**
    - **Ist der Großbuchstabe ein Vokal, dann wird auch VokalAnzahl um 1 erhöht**
    - **Als Ausgabeparamenter werden Gesamtzahl und VokalAnzahl übergeben.**

LE 14
13

5.3 Der Kontrollflußgraph

```
void ZaehleZchn(int &VokalAnzahl,
                                    int &Gesamtzahl)
{ char Zchn;
  cin>>Zchn;
  while ((Zchn>='A')&&(Zchn<='Z')&&
                (Gesamtzahl<INT_MAX))
  { Gesamtzahl = Gesamtzahl+1;
    if((Zchn=='A')||(Zchn=='E')||(Zchn=='I')||
      (Zchn == 'O')||(Zchn == 'U'))
    {   VokalAnzahl = VokalAnzahl + 1;
    }
    cin>>Zchn;
  } //end while
}.
```

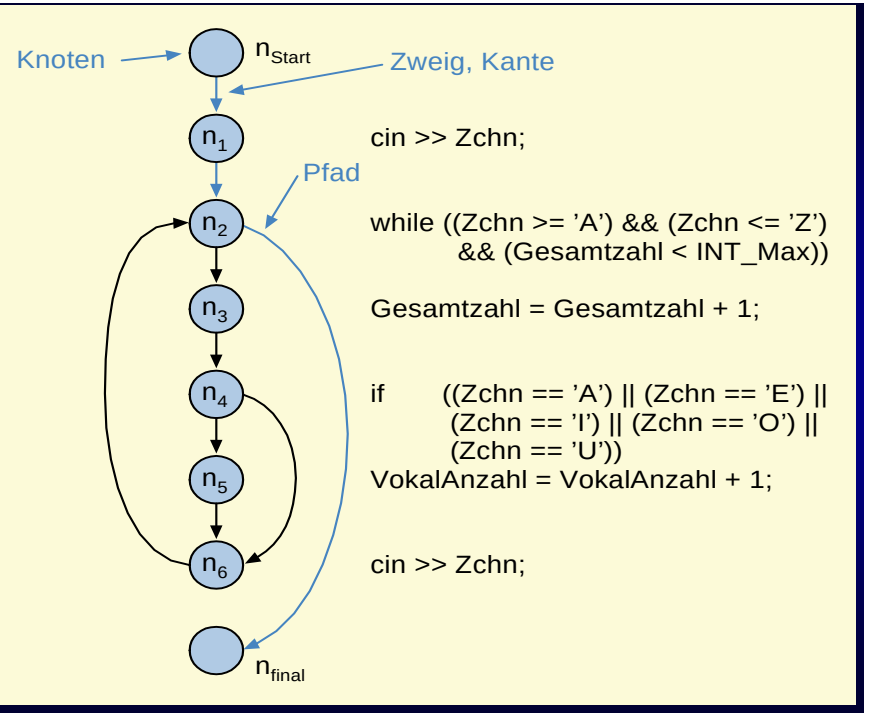

LE 14
14

5.3 Der Kontrollflußgraph

- Eine gerichtete Kante von einem Knoten i zu einem Knoten j beschreibt einen möglichen **Kontrollfluß** von Knoten i zu Knoten j
- Die gerichteten Kanten werden als **Zweige** bezeichnet
- Eine abwechselnde Folge von Knoten und Kanten, die mit dem Startknoten beginnt und mit einem Endeknoten endet, heißt **Pfad**

◆ Kontrollflußorientierte Testverfahren
- Kontrollflußgraphen, bei denen jeder Knoten eine ausführbare Anweisung darstellt .

LE 14
15



LE 14
16

5.4 Kontrollflußorientierte Strukturtestverfahren

- **Einordnung**
  - Basieren auf der Kontrollstruktur des zu testenden Programms
  - Gehören zu den **Strukturtest-** oder ***White Box-*** bzw. ***Glass Box*-Verfahren**
    - Struktur muß sichtbar sein
  - Gehören zu den **dynamischen Testverfahren**
    - Das übersetzte Programm wird beim Testen ausgeführt
  - Es handelt sich um **testende Verfahren**
    - Anhand von einzelnen Testfällen die Fehlerfreiheit überprüft wird.

LE 14
17

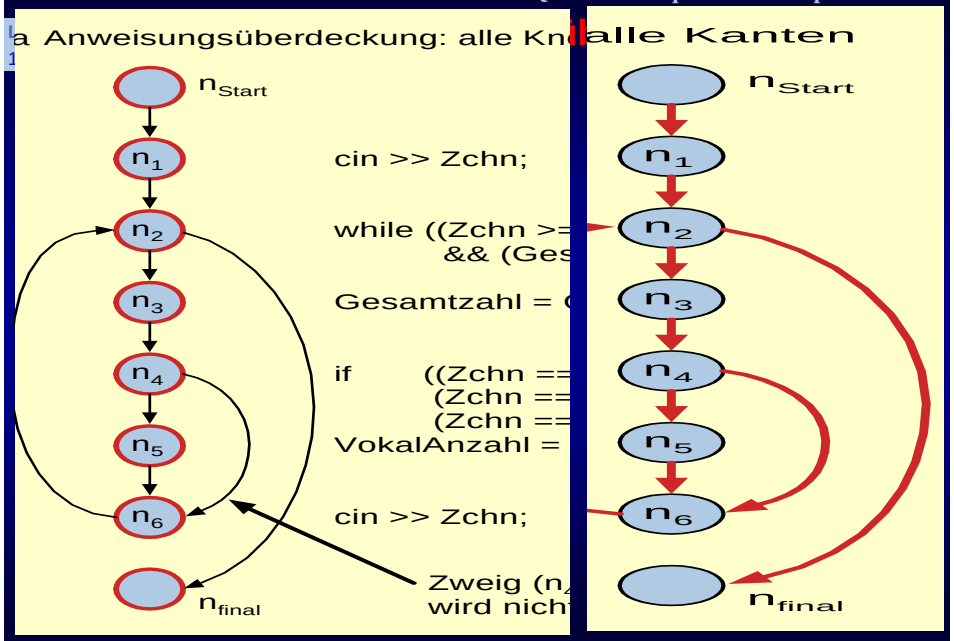
5.4 Kontrollflußorientierte Strukturtestverfahren

- **Bekannteste Testverfahren**
  - **Anweisungs**überdeckungstest
  - **Zweig**überdeckungstest
  - **Pfad**überdeckungstest
- **Ziel**
  - Mit einer Anzahl von Testfällen alle vorhandenen **Anweisungen, Zweige** bzw. **Pfade** auszuführen
- **Pfadüberdeckungstest**
  - Umfassendste kontrollflußorientierte Testverfahren
  - Unpraktikabel wegen der hohen Pfadanzahl realer Programme
  - Verschiedene Testverfahren nähern sich dem Pfadüberdeckungstest an.

LE 14
18

5.4.1 Anweisungs- & Zweigüberdeckungstest

- **Anweisungsüberdeckungstest**
  - Auch C_0-Test (C = *Coverage)*
  - Ausführung aller Anweisungen, d.h. aller **Knoten** des Kontrollflußgraphen
- **Zweigüberdeckungstest**
  - Auch C_1-Test
  - Ausführung aller Zweige, d.h. aller **Kanten** des Kontrollflußgraphen.



LE 14
20

5.4.1 Anweisungs- & Zweigüberdeckungstest

- **Testfall für alle Knoten**
  - Aufruf von `ZaehleZchn` mit: `Gesamtzahl = 0`
  - Eingelesene Zeichen: `'A', '1'`
  - Durchlaufener Pfad:
    $(n_{start}, n_1, n_2, n_3, n_4, n_5, n_6, n_2, n_{final})$
- **Testfall für alle Kanten**
  - Aufruf von `ZaehleZchn` mit: `Gesamtzahl = 0`
  - Eingelesene Zeichen: `'A', 'B', '1'`
  - Durchlaufener Pfad:
    $(n_{start}, n_1, n_2, n_3, n_4, n_5, n_6, n_2, n_3, n_4, n_6, n_2, n_{final})$.

LE 14
21

5.4.1 Anweisungs- & Zweigüberdeckungstest

- Nachteile des Zweigüberdeckungstests
  - Er ist **unzureichend** für den Test von **Schleifen**
  - Er berücksichtigt nicht die **Abhängigkeiten zwischen Zweigen**, sondern geht davon aus, daß alle Zweige unabhängig voneinander sind
  - Er ist unzureichend für den Test komplexer, d.h. **zusammengesetzter Bedingungen**.

LE 14
22

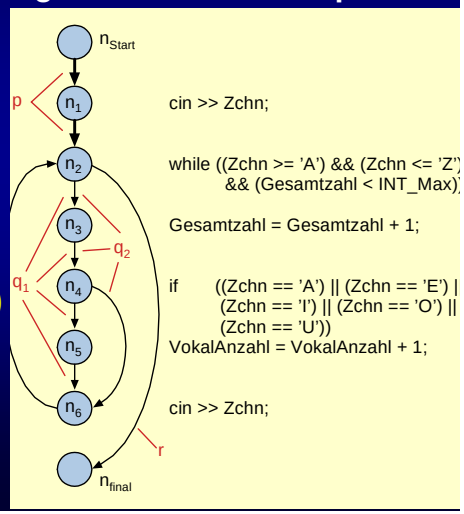
5.4.2 Die Pfadüberdeckungstestverfahren

- Pfadüberdeckungstestverfahren
  - Entwickelt, um Programme mit Wiederholungen bzw. Schleifen ausreichend testen zu können
  - **Pfadüberdeckungstest**
    - Fordert die Ausführung aller unterschiedlichen Pfade des zu testenden Programms.

LE 14
23

5.4.2 Die Pfadüberdeckungstestverfahren

- Pfadüberdeckungstest: Beispiel
  - Alle möglichen Pfade beginnen mit dem Subpfad **$p = (n_{start}, n_1)$**
  - Darauf folgen **k** Subpfade **q** ($k \geq 0$), gefolgt vom Subpfad **$r = (n_2, n_{final})$**
  - Jeder der **k** Subpfade **q** kann entweder die Form **$(n_2, n_3, n_4, n_5, n_6)$** oder die Form **$(n_2, n_3, n_4, n_6)$** besitzen



LE 14
24

5.4.2 Die Pfadüberdeckungstestverfahren

- Auswertung des Beispiels
  - Anzahl der Schleifendurchläufe
    - Begrnezt durch die maximale Differenz zwischen `Gesamtzahl` beim Prozeduraufruf und `INT_Max`
  - Anzahl der Testpfade
    - Annahme: größte Wert einer Variablen vom Typ **int** 2.147.483.647
    - $2^{2.147.483.647} - 1$ Testpfade
- Praktikabel nur durch Einschränkung der Anzahl der Wiederholungen:
  - *boundary interior* **Pfadtest**
  - **Strukturierter Pfadtest.**

LE 14
25

Der *boundary interior* Pfadtest

- **Einordnung**
  - Eingeschränkte, schwächere Version des Pfadüberdeckungstests
  - Für schleifenfreie Programme ist er mit dem Pfadüberdeckungstest identisch
- **Ansatz**
  - Es wird auf die Überprüfung von Pfaden verzichtet, die durch mehr als einmalige Schleifenwiederholung erzeugt werden

LE 14
26

Der strukturierte Pfadtest

- **Einordnung**
  - Verallgemeinerung des *boundary interior*–Pfadtests
- **Ansatz**
  - Alle Pfade, die eine innerste Schleife – d.h. eine Schleife, die keine weiteren Schleifen enthält – nicht häufiger als k-mal durchlaufen, werden ausgeführt

LE 14
27

5.4.3 Bedingungsüberdeckungstestverfahren

- **Zusammengesetzte und/oder hierarchisch gegliederte Bedingungen**
  - Zweigüberdeckungstest nicht ausreichend
- **Beispiel**
  - `ZaehleZchn` enthält 2 Bedingungen:
    - `((Zchn>='A')&&(Zchn>='Z')AND (Gesamtzahl<INT_Max)))`
      - 3 atomare Bedingungen
    - `((Zchn=='A')||(Zchn='E')||(Zchn=='I') ||(Zchn=='O')||(Zchn=='U'))`
      - 5 atomare Bedingungen
    - Zweigüberdeckungstest kann diese Struktur der Bedingungen nicht geeignet testen.

LE 14
28

5.4.3 Bedingungsüberdeckungstestverfahren

- **Einfache Bedingungsüberdeckung**
  - Alle atomaren Bedingungen: einmal *true* und *false*

Testfälle → | Variablenwerte →

Variablen	1						2	3
Gesamtzahl	0	1	2	3	4	5	0	INT_Max
Zchn	'A'	'E'	'I'	'O'	'U'	'1'	'a'	'D'
Zchn >= 'A'	T	T	T	T	T	F	T	T
Zchn <= 'Z'	T	T	T	T	T	T	F	T
Gesamtzahl < INT_MAX	T	T	T	T	T	T	T	F
Zchn == 'A'	T	F	F	F	F	–	–	–
Zchn == 'E'	F	T	F	F	F	–	–	–
Zchn == 'I'	F	F	T	F	F	–	–	–
Zchn == 'O'	F	F	F	T	F	–	–	–
Zchn == 'U'	F	F	F	F	T	–	–	–

atomare Bedingungen

Wahrheitswerte der atomaren Bedingungen
T (true) – wahr, F (false) – falsch, – (nicht evaluiert)

LE 14
29

5.4.3 Bedingungsüberdeckungstestverfahren

- **Einfache Bedingungsüberdeckung**
  - Enthält nicht die Zweigüberdeckung
  - Enthält nicht die Anweisungsüberdeckung
  - Als alleiniges Testverfahren daher ungeeignet
- **Mehrfach-Bedingungsüberdeckungstest**
  - Alle Variationen der atomaren Bedingungen.

LE 14
30

5.4.3 Bedingungsüberdeckungstestverfahren

- **Mehrfach-Bedingungsüberdeckungstest**

Zchn	'A'	'E'	'I'	'O'	'U'	alle anderen Zeichen	?	?	?
Zchn == 'A'	T	F	F	F	F	F	T	...	T
Zchn == 'E'	F	T	F	F	F	F	T	...	T
Zchn == 'I'	F	F	T	F	F	F	F	...	T
Zchn == 'O'	F	F	F	T	F	F	F	...	T
Zchn == 'U'	F	F	F	F	T	F	F	...	T
Bedingung b	T	T	T	T	T	F	T	...	T

6 Variationen (prinzipiell herstellbar)

26 Variationen (nicht herstellbar)

32 Variationen

Wahrheitswerte der Bedingungen
T (true) – wahr F (false) – falsch

- Nur 6 der 32 denkbaren Variationsmöglichkeiten sind herstellbar
- 26 sind nicht herstellbar.

LE 14
31

5.4.3 Bedingungsüberdeckungstestverfahren

- **Minimale Mehrfach-Bedingungsüberdeckung**
  - Jede Bedingung – ob atomar oder nicht – muß mindestens einmal *true* und *false* sein
  - Auch die Überdeckung der nicht atomaren Bedingungen, einschließlich der Gesamtbedingung ist gefordert
  - Berücksichtigt die Struktur von Bedingungen erheblich besser als die einfache Bedingungsüberdeckung
    - Alle Schachtelungsebenen einer komplexen Bedingung werden gleichermaßen beachtet.

LE 14
32

5.4.3 Bedingungsüberdeckungstestverfahren

- **Minimale Mehrfach-Bedingungsüberdeckung**

Testfälle

Variablen	1							2	3
Gesamtzahl	0	1	2	3	4	5	6	0	INT_MAX
Zchn	'A'	'E'	'I'	'O'	'U'	'B'	'1'	'a'	'D'
Zchn >= 'A'	T	T	T	T	T	T	F	T	T
Zchn <= 'Z'	T	T	T	T	T	T	T	F	T
Gesamtzahl < INT_MAX	T	T	T	T	T	T	T	T	F
Bedingung a	T	T	T	T	T	T	F	F	F
Zchn == 'A'	T	F	F	F	F	F	–	–	–
Zchn == 'E'	F	T	F	F	F	F	–	–	–
Zchn == 'I'	F	F	T	F	F	F	–	–	–
Zchn == 'O'	F	F	F	T	F	F	–	–	–
Zchn == 'U'	F	F	F	F	T	F	–	–	–
Bedingung b	T	T	T	T	T	F	–	–	–

Bedingungen

Wahrheitswerte der Bedingungen
T (true) – wahr
F (false) – falsch
(nicht evaluiert)

Zusätzliches Testdatum zur vollständigen Überdeckung von Bedingung b bei vollständiger Bedingungsevaluation

- Danke!
- Aufgaben

- Diese Präsentation bzw. Teile dieser Präsentation enthalten Inhalte und Grafiken des Lehrbuchs der Software-Technik (Band 2) von Helmut Balzert, Spektrum Akademischer Verlag, Heidelberg 1998