Klassifikationsbaum-Methode

Mathis Kückens, Stefan Kersten

MWSP WS03/04

Gliederung

- Einleitung
- Motivation
- Die Methode
- Tools
- Zusammenfassung
- Übung

Einleitung 1

- **Einsatzgebiete**
  - **Software-Tests**
  - KI – Classification & Regression Trees (CART)
  - Data Mining
  - Medizin
  - Philosophie
  - usw.

Einleitung 2

im Bereich des **Software-Testens**:

1993 erstmals vorgestellt von

Klaus Grimm und **Matthias Grochtmann**

(Daimler-Benz-AG)

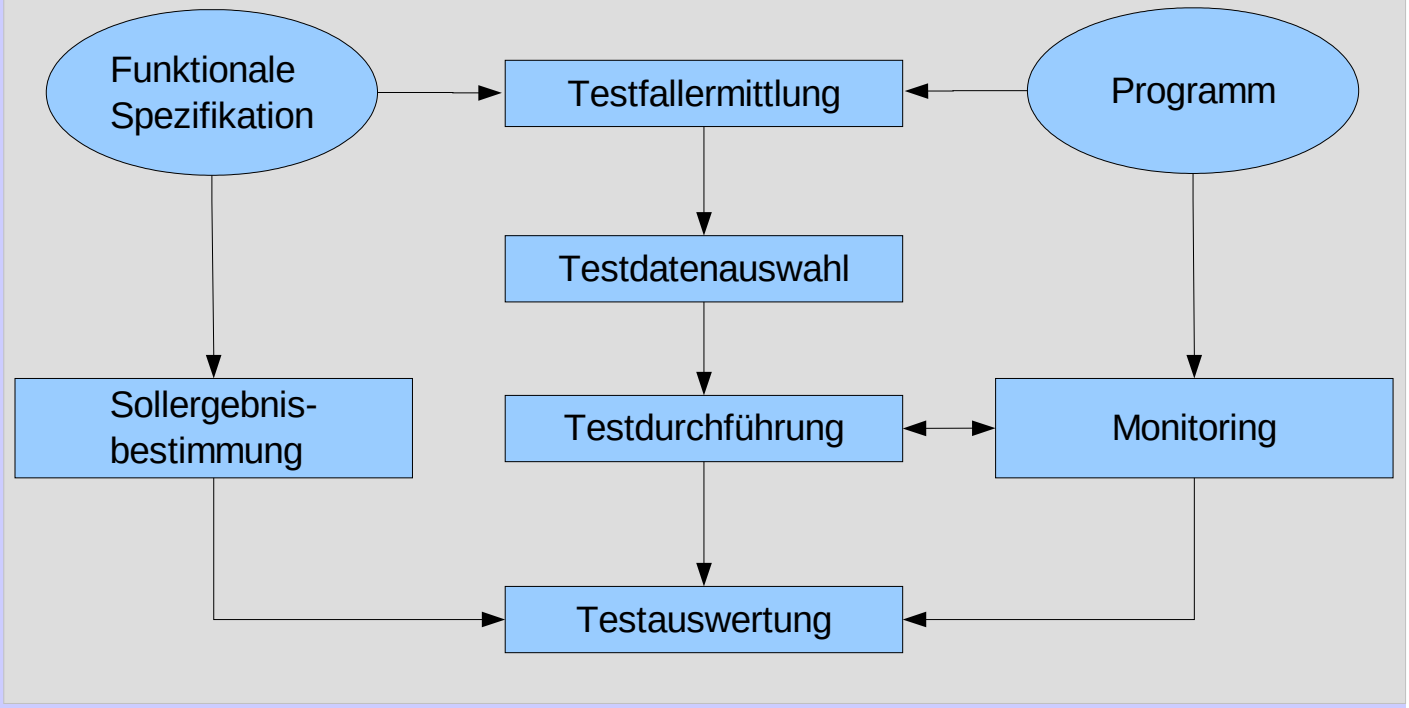
Einleitung 3

- **Anwendungsbeispiele**
  - Eingebettete Systeme
    - Elektronische Flugsteuerung
    - Motorelektronik
    - Mechatronische Systeme (Kfz-Elektronik)
      - ABS, Airbag, Servolenkung
  - mechanische Systeme
    - Vorsortiersystem einer Briefverteilanlage
    - usw.

Einleitung 4

- **Abgrenzung**
  - Dynamisches Verfahren
  - Blackbox-Test (Funktionaler Test)
    - Testfälle anhand der Spezifikation und den Anforderungen
  - eingegliedert in Systematischen Test

Systematischer Test: Ablauf



Motivation 1

- Entwicklung eingebetteter Systeme
  - Codierung 20%
  - Testaufwand 80%

    -> **Systematischer Test**
- keine leistungsfähigen Methoden und Werkzeuge für den funktionalen Test verfügbar

Motivation 2

Vorteile

- **Systematik/Methodik**

  -> redundanzarme Testfälle

  -> fehlerintensive Testfälle

- **grafische Methode**

  -> kompakte Darstellung des Gesamttests

  -> hierarchische Aufgliederung des Problembereichs

  -> leichte Erlernbarkeit

Motivation 3

Weitere Vorteile

- **Werkzeugunterstützung** (CTE/XL, TESSY)

  -> Automatisierung von Testaktivitäten

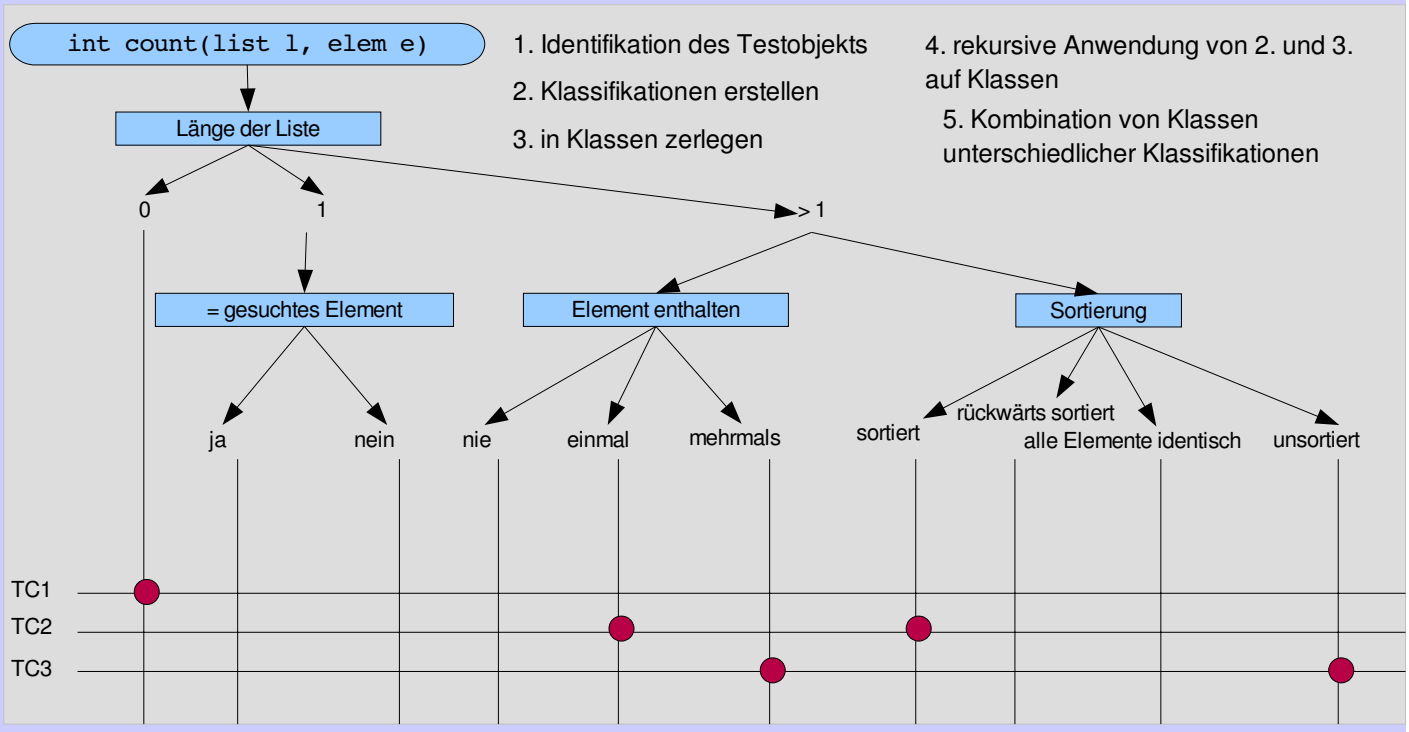
  -> intensive Anwendung der Methode

- **parallel** zum Programm entwickeln

Die Methode 1

- **Grundidee**
  - Eingabedatenbereich unter verschiedenen Gesichtspunkten/Aspekten betrachten, in denen sich das Testobjekt gleich verhält
  - das Testobjekt in die Aspekte zerlegen
  - durch Kombination dieser Zerlegungen zu Testfällen gelangen

Vorgehensweise: K-Baum-Erstellung

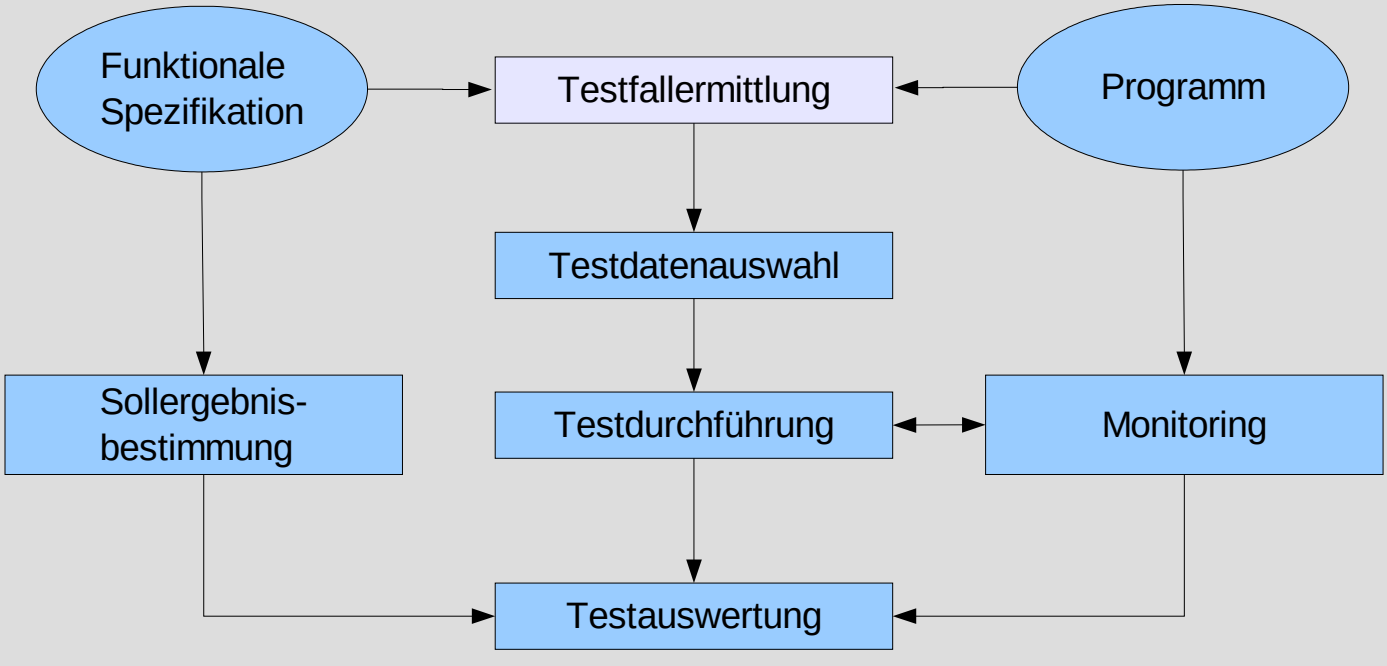


1. Identifikation des Testobjekts
2. Klassifikationen erstellen
3. in Klassen zerlegen
4. rekursive Anwendung von 2. und 3. auf Klassen
5. Kombination von Klassen unterschiedlicher Klassifikationen

Die Methode 3

- **Begriffe**
  - **Klassifikationsbaum**: besteht aus Wurzel und Knoten
  - **Wurzel**: Testobjekt, Name des KB
  - **Knoten**:
    - **Klassifikation**: Variable oder Parameter, erwartetes Ergebnis
    - **Klasse**: Wertebereich einer Klassifikation
      - Wichtig: Klassen dijunkt, Wertebereich der Klassifikation vollständig wiedergeben
  - **Blätter**: Klassen ohne weitere Unterklassifikationen

Systematischer Test: Ablauf



Testfallermittlung 1

- **Begriffe**
  - **Testfall**
    - allgemein:

      Auswahl der Eingabevariablen und der erwarteten Ausgaben
    - in Bezug auf KB:

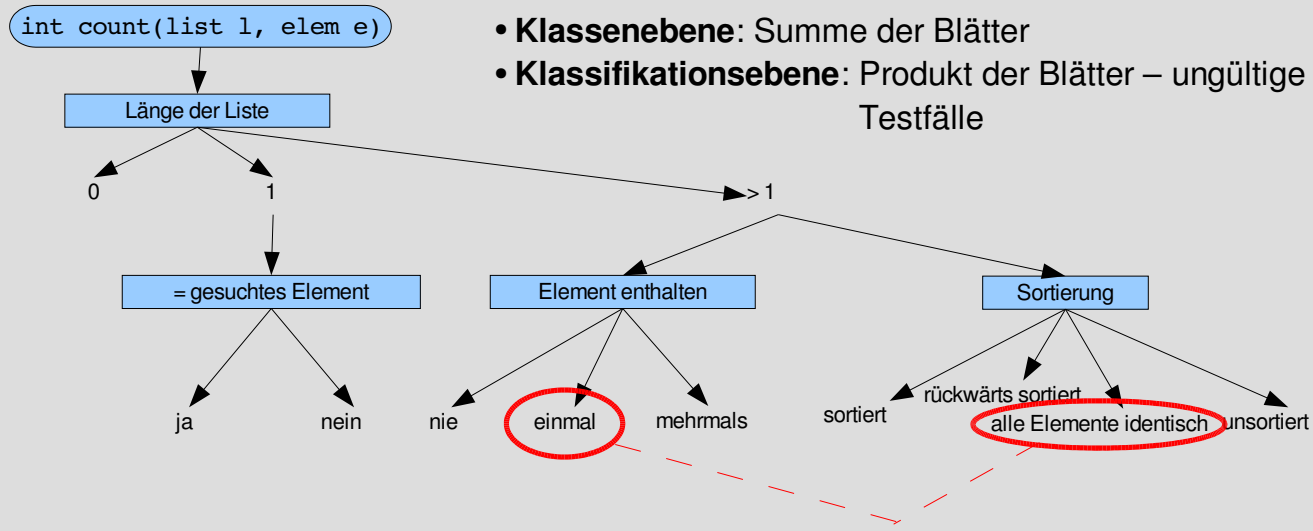
      Auswahl einer bestimmten Kombination der Blätter des Baumes

Testfallermittlung 2

- **Potentieller Testfall**
  - Klassen- bzw. Blätterauswahl, in der aus jeder Klassifikation genau eine Klasse ausgewählt wird
- **Gültiger Testfall**
  - Potentieller Testfall, dessen Klassenauswahl konsistent mit der Spezifikation und der semantischen Bedeutung des Testobjektes ist
- **Ungültiger Testfall**
  - Potentieller Testfall, der nicht gültig ist

Testfallermittlung 3

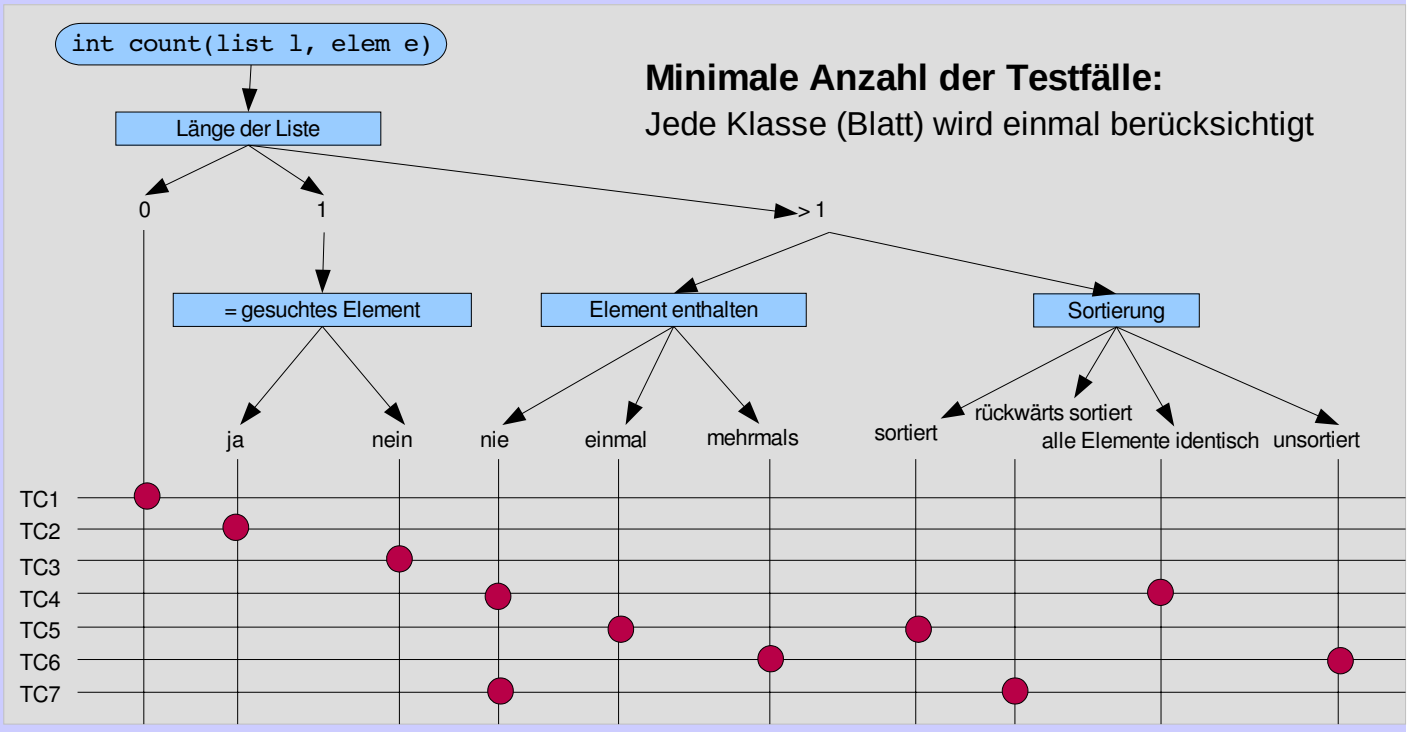
Maximale Anzahl der Testfälle:



- **Klassenebene**: Summe der Blätter
- **Klassifikationsebene**: Produkt der Blätter – ungültige Testfälle

max(TF) = 1 + 2 + (3 * 4) - 1 = 14

Testfallermittlung 4



Minimale Anzahl der Testfälle:
Jede Klasse (Blatt) wird einmal berücksichtigt

Testfallermittlung 6

KBM liefert für jeden Testfall eine **Testfallspezifikation**:

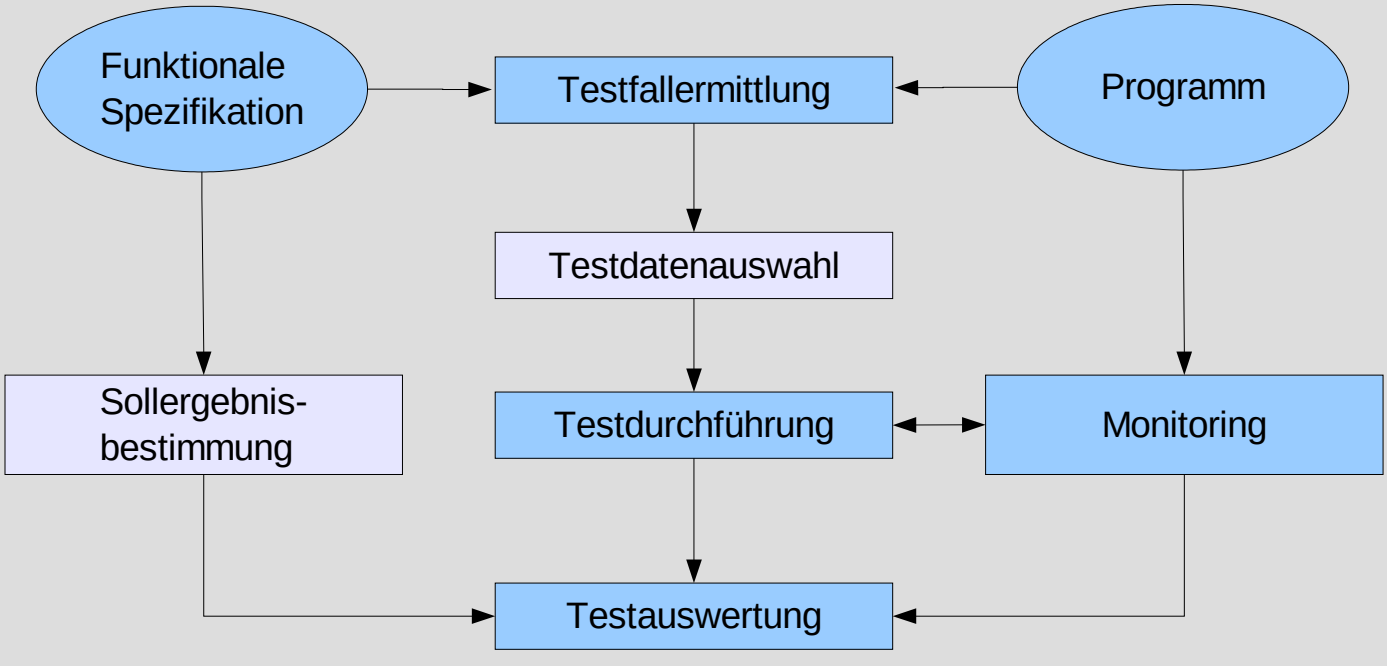
Bsp:

```
Länge der Liste: > 1
Element enthalten: einmal
Sortierung: sortiert
```

semiformale textuelle Beschreibung der den Testfall konstituierenden Klassen und Klassifikationen

-> automatisierbar (CTE)

Testablauf: Überblick



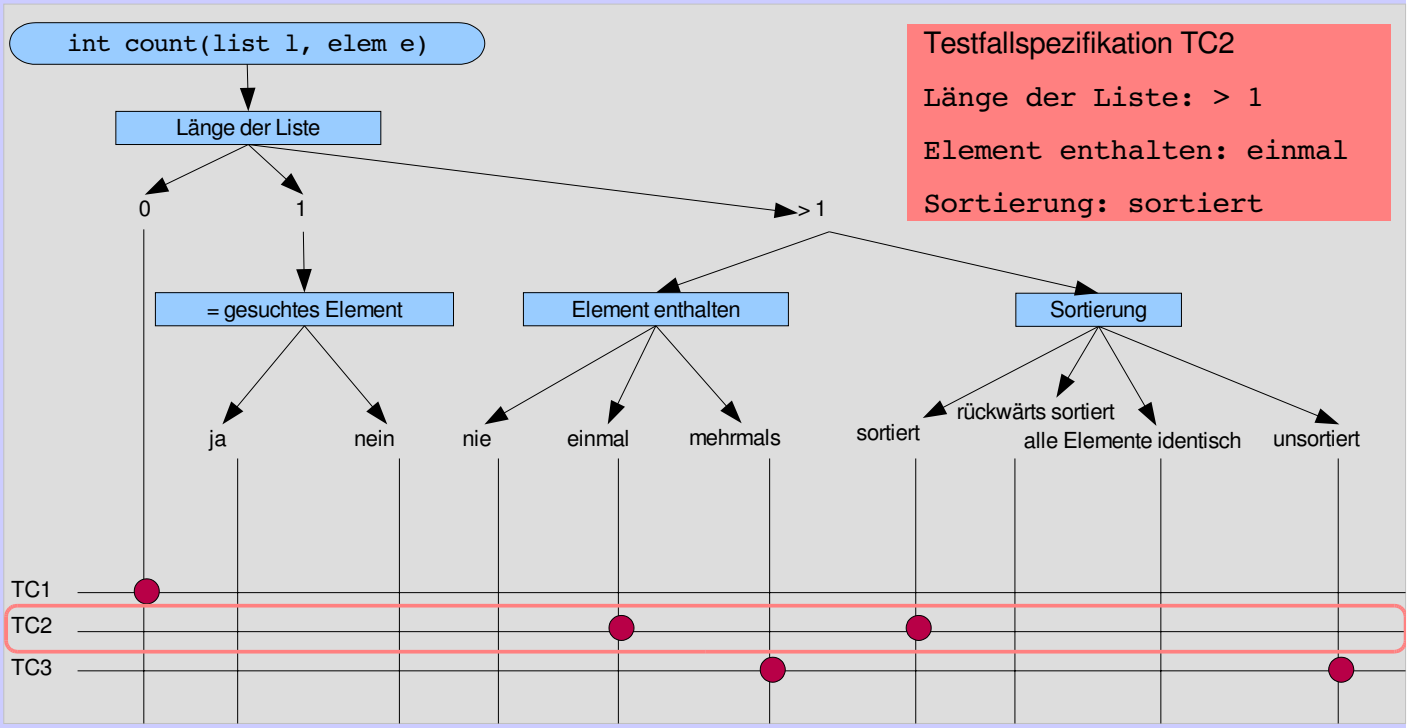
Begriffe

- **Testfall**: Kombination unterschiedlicher Klassen des Klassifikationsbaums
- **Testobjekt**: zu testende Einheit (Methode, Motor, eingebettetes System)
- **Testdaten**: Eingabevektor für Testobjekt und Systemzustand
- **Testpaket**: Gruppe von Testfällen, die das gleiche Testobjekt unter gleichen Aspekten testen

Testdatenauswahl 1

- Assoziation aller einen Testfall konstituierender Klassen mit konkreten Daten
- zugewiesenes Datum muss in allen Testfällen gleichen Wert besitzen
- Erzeugung weiterer Unterklassifikationen zum Testen spezieller Werte

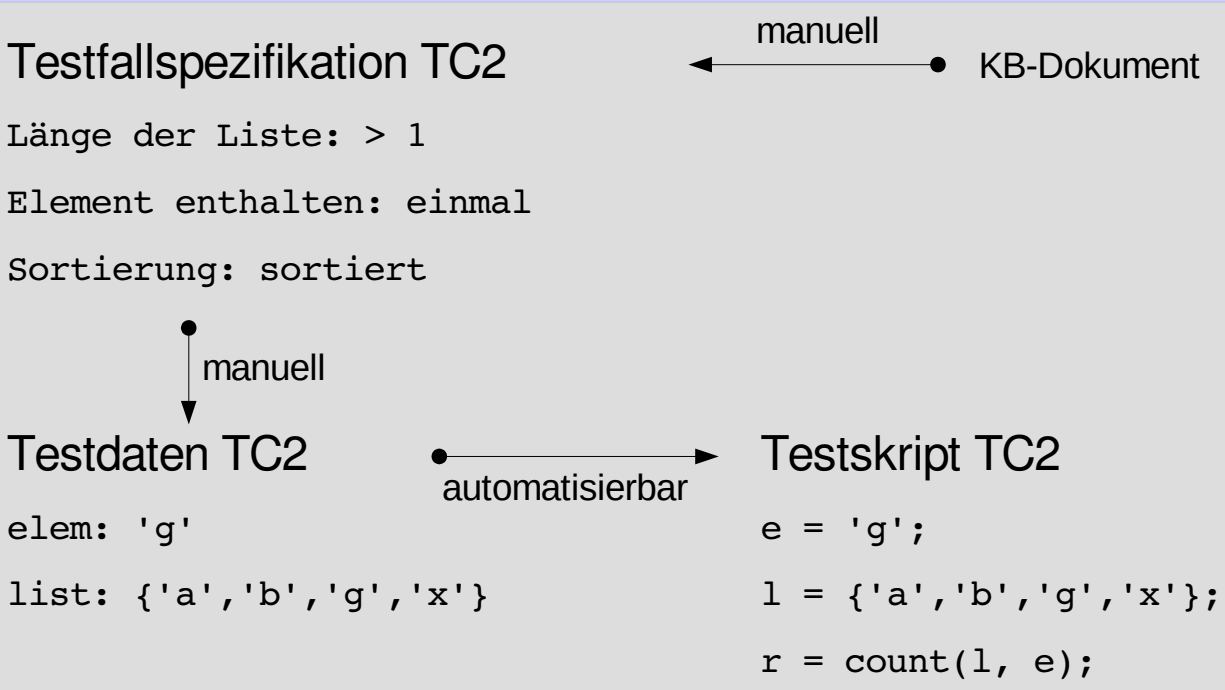
Beispiel: `count`



Testdatenauswahl 2

- Manuelle Bestimmung aus der Testfallspezifikation
- Problem:
  - **Semantische Lücke** zwischen Spezifikation und konkreten Testdaten
  - Keine **Verwendung des Klassifikationsbaumes** für Testautomatisierung
  - steigende **Komplexität** bei umfangreichen Klassifikationsbäumen

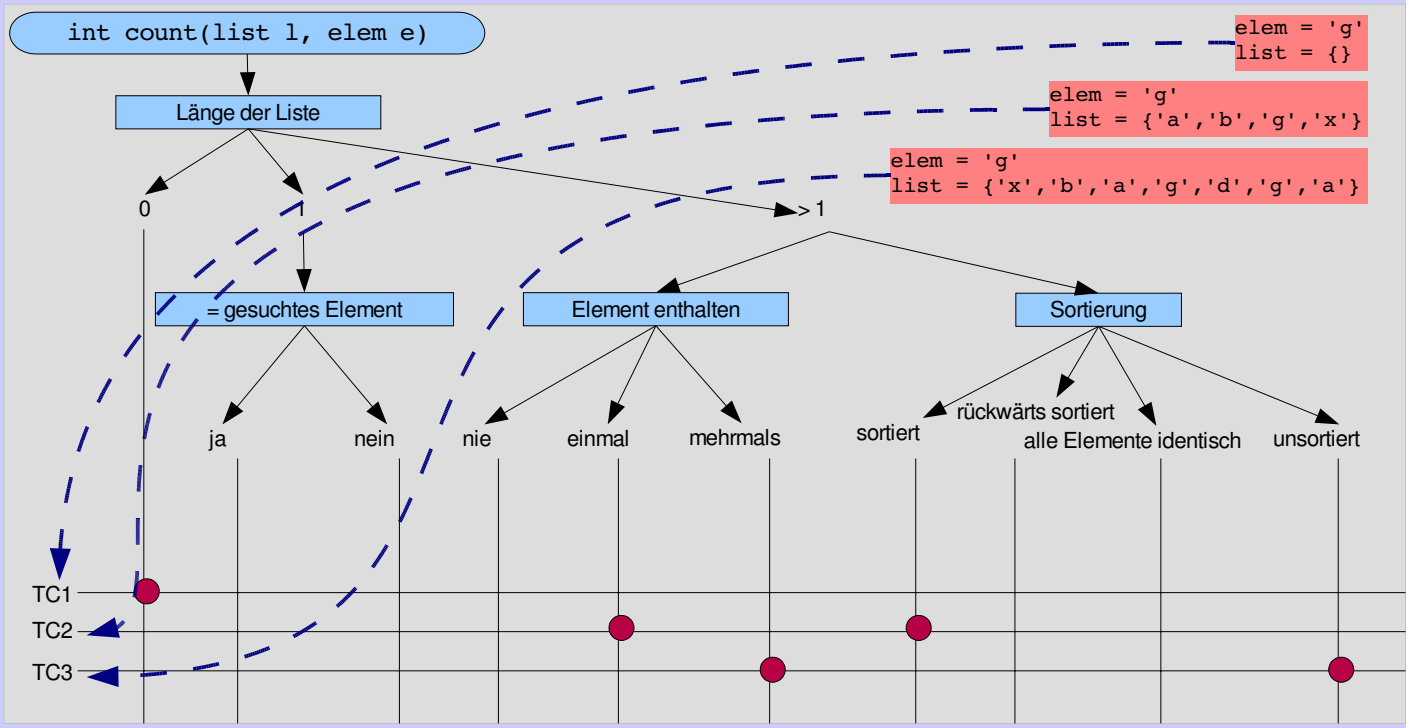
Testdatenauswahl 3



Testdatenauswahl 4

- Attribute an den Testfällen
  - Verknüpfung der erforderlichen Daten mit jedem Testfall
  - Gemeinsames Dokument für Testfallerstellung und Testdatenbestimmung
  - **keine** Verwendung der Semantik des Klassifikationsbaumes

Beispiel: `count`



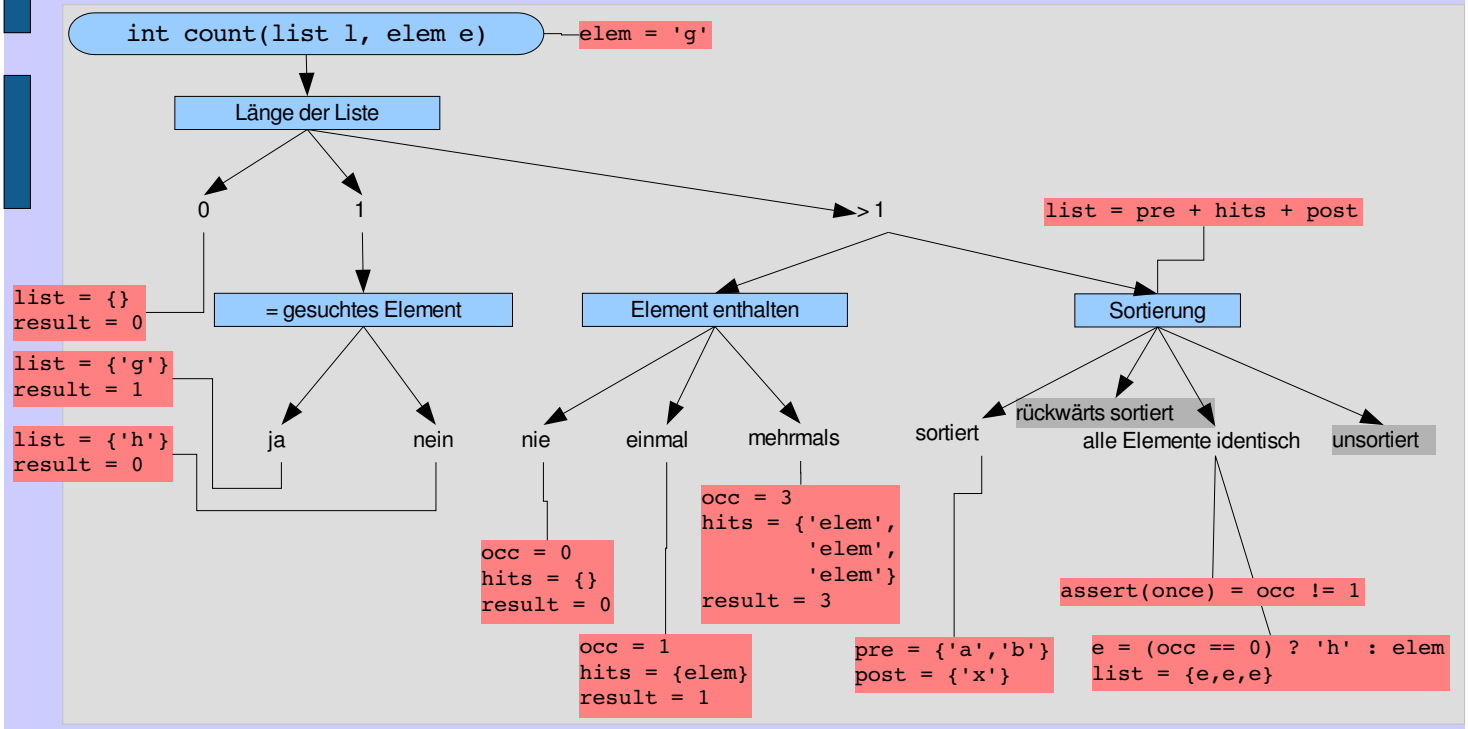
Testdatenauswahl 5

- Attribute am Klassifikationsbaum
  - Klassen im Klassifikationsbaum werden mit notwendigen und relevanten Testdaten attributiert
- Erweiterungen
  - Vererbung und Überschreibung von Attributen
  - Ausdrücke in Attributen
  - Zusicherungen
  - Angabe der Solldaten

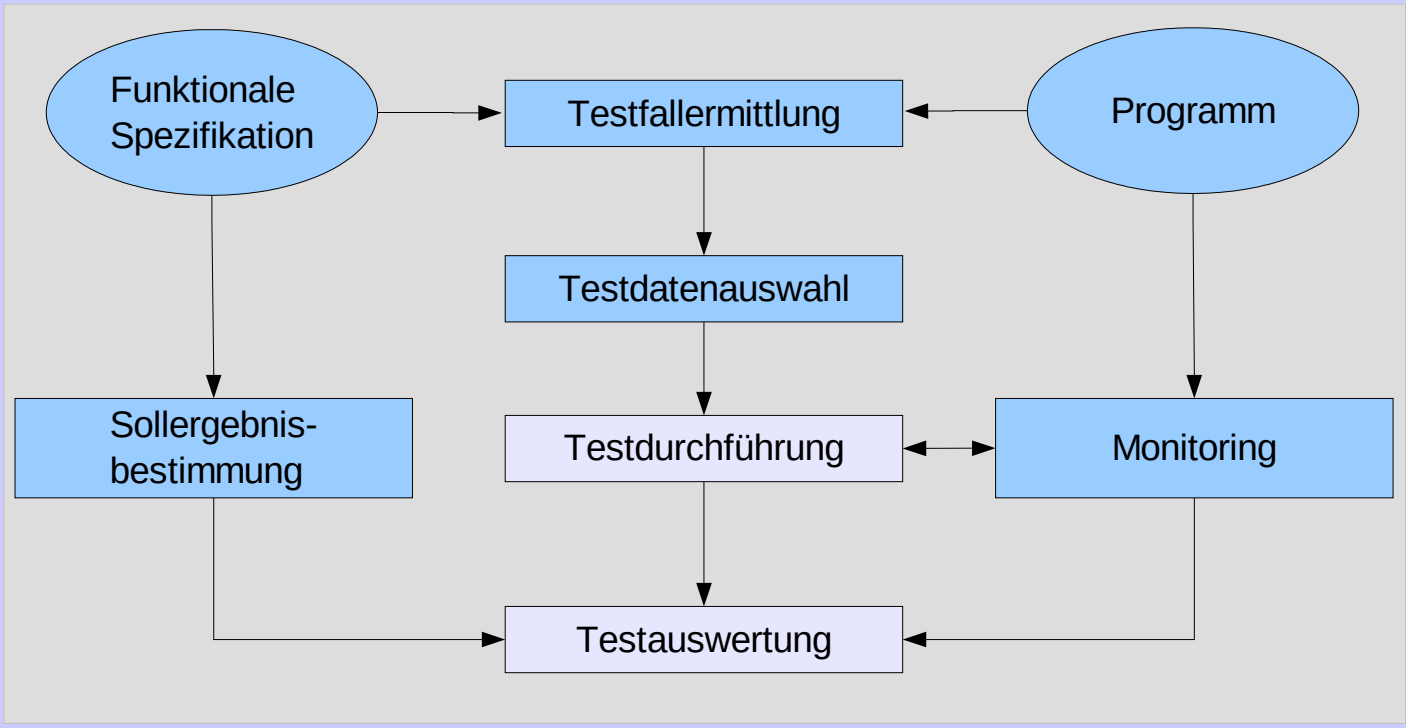
Solldatenbestimmung

- Grundlage: funktionale Spezifikation
- Attributierung des Klassifikationsbaumes
- Erweiterungen
  - Wertebereichstoleranzen
  - Iterationstoleranzen

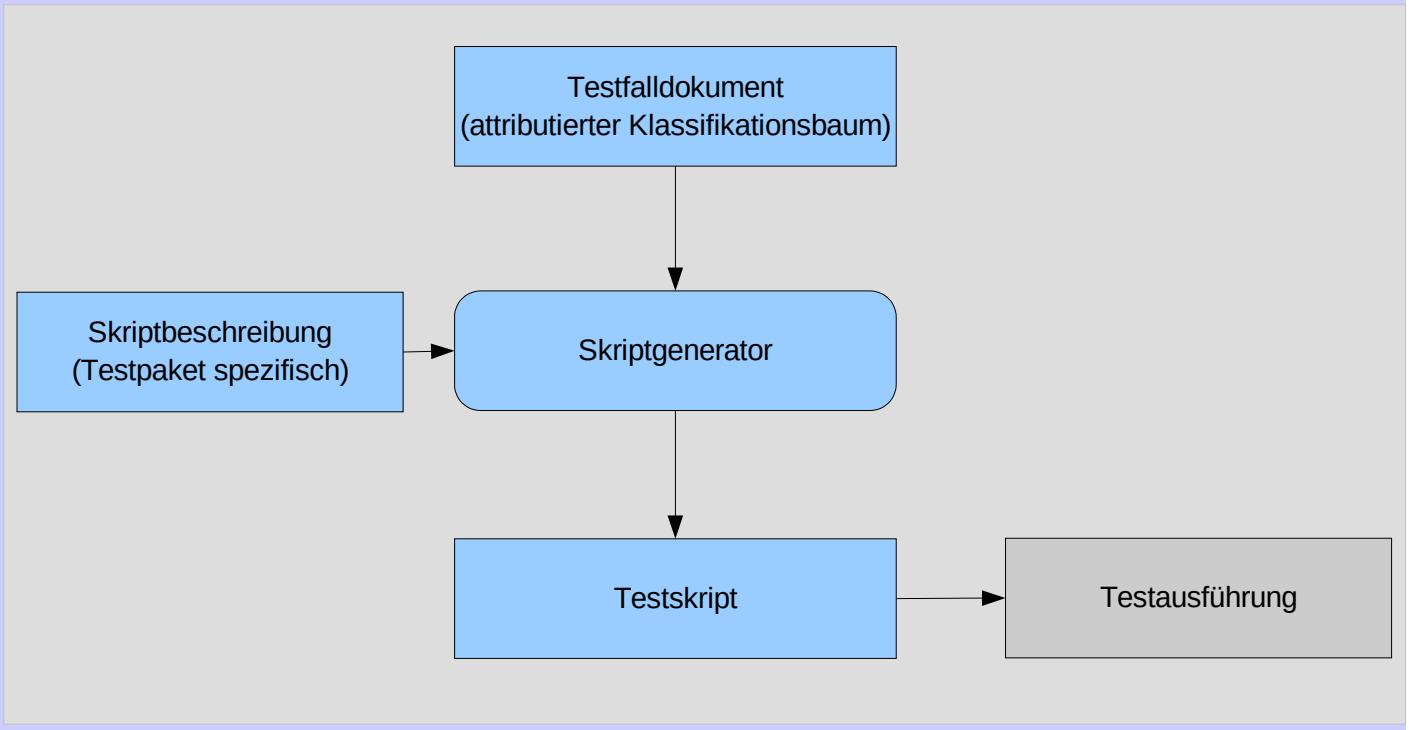
Beispiel: `count`



Testablauf: Überblick



Testskriptgenerierung

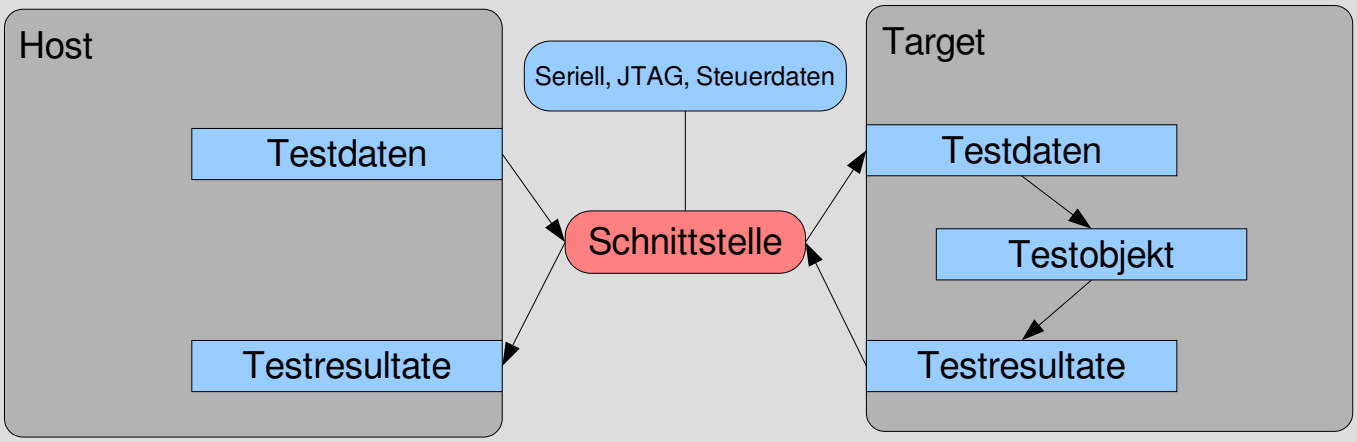


Testausführung 1

- Ausführung des Testskriptes auf der Zielplattform
- Besonderheiten beim Test von **eingebetteten Systemen**
  - beschränkte Ressourcen
  - Schnittstelle zum Testobjekt (Target-Test)
  - Datenkonvertierung (z.B. Fließkomma -> Festkomma)

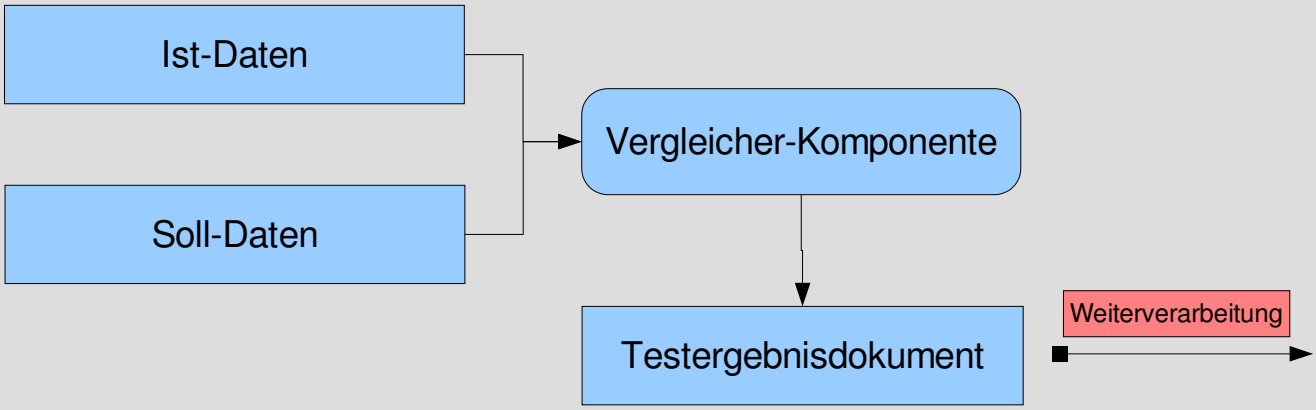
Testausführung 2

Client-Server-Architektur



Testauswertung

- Vergleich von Ist- und Solldaten
- Berücksichtigung der Solltoleranzen



Tool: CTE/XL

- **C**lassification **T**ree **E**ditor (e**X**tended **L**ogics)
- graphisches Tool zum Erstellen von Klassifikationsbäumen
- Erweiterungen
  - Formulierung logischer Abhängigkeiten
  - Kombinationsregeln
  - Automatisierte Testfallerzeugung
  - hierarchische Gliederung umfangreicher K-Bäume

Testsystem: TESSY

- Integratives Unit-Test-System
- Einbettung in zyklischen Entwicklungs-/Testprozess
- Features
  - Klassifikationsbaum-Methode
    - Testfallermittlung
    - Test- und Solldateneingabe
    - Testdurchführung mit Monitoring
    - Testauswertung und -dokumentation

Testsytem: TESSY

- Erweiterungen
  - Projekt- und Dokumentverwaltung
  - Schnittstellenerkennung, Targettest
  - White-Box-Test zur Qualitätsüberprüfung generierter Testfälle
  - evolutionäres Testen für Performancemessungen

Zusammenfassung

- Klassifikationsbaum-Methode unterstützt systematisches, strukturiertes Testen auch umfangreicher Systeme
- umfangreiche Toolunterstützung
- Anwendung und Erfahrungswerte in der Praxis
- intuitive Erzeugung konsistenter Testfälle
- erleichterte Pflege und Erweiterung von Testsuiten
- Einbindung in Softwarezyklus (Entwicklung, Wartung)

Übung

- Testobjekt

```
int anzahl_vokale(String s);
```

- Aufgabe
  - Erstelle den Klassifikationsbaum.
  - Trage die minimale Anzahl von Testfällen in die Kombinationstabelle ein.
  - Ermittle Test- und Solldaten für zwei Testfälle.

Klassifikationsbaum `anzahl_vokale`