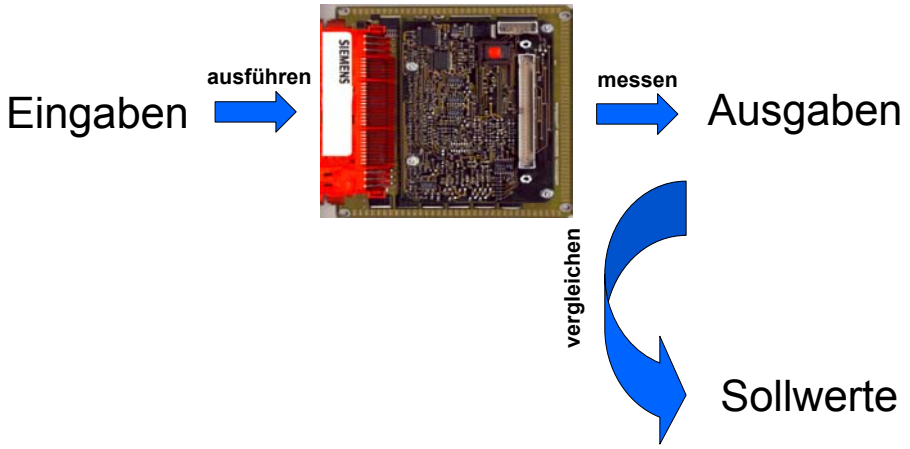
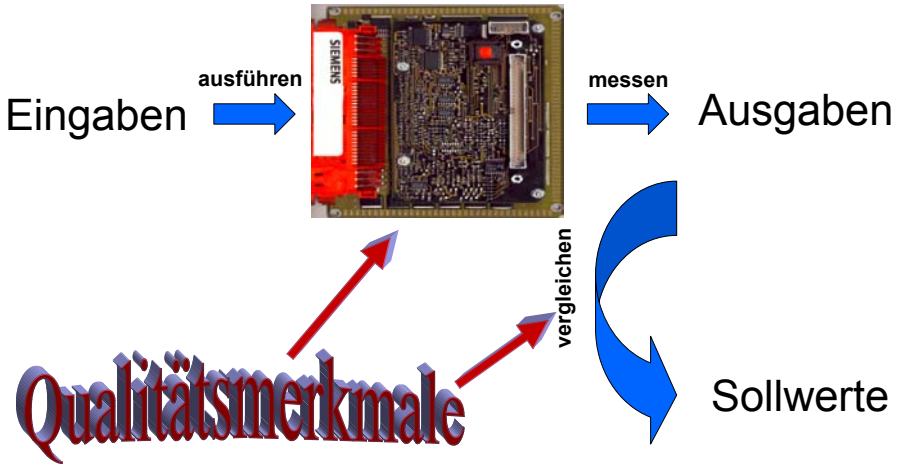
1x1 des Testens

Frank Listing
MicroConsult GmbH

MICROCONSULT

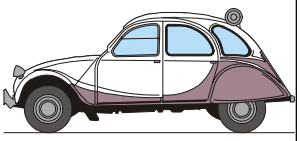
Allgemeines



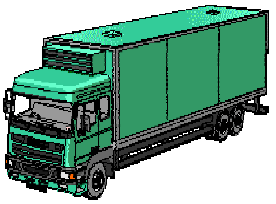


Qualitätsmerkmal	Qualitäts-Teilmerkmal
Funktionalität	Angemessenheit, Richtigkeit, Interoperabilität, Ordnungsmäßigkeit, *Sicherheit*
Zuverlässigkeit	Reife, Fehlertoleranz, Wiederherstellbarkeit
Benutzbarkeit	Verständlichkeit, Erlernbarkeit, Bedienbarkeit
Effizienz	Zeitverhalten, Verbrauchsverhalten
Änderbarkeit	Analysierbarkeit, Modifizierbarkeit, Stabilität, Prüfbarkeit
Übertragbarkeit	Anpassbarkeit, Installierbarkeit, Konformität, Austauschbarkeit

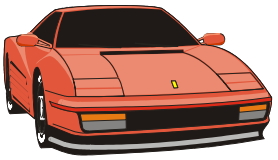
Nach DIN 66272, Oktober 1994 (ISO/IEC 9126 : 1991)



- Spritverbrauch?
- Tragfähigkeit?
- Höchstgeschwindigkeit?



- Spritverbrauch?
- Tragfähigkeit?
- Höchstgeschwindigkeit?



- Spritverbrauch?
- Tragfähigkeit?
- Höchstgeschwindigkeit?

Was halten Sie von dem Versuch, einen

- Formel 1 Rennwagen
- zum Containertransport als
- 3-Liter Auto

zu bauen?

- Was halten Sie von dem Versuch, eine
  - sehr speichereffiziente Software mit
  - sehr hoher Zuverlässigkeit und
  - sehr guter Anpassbarkeit
  - mit grafischer, konfigurierbarer Bedienoberfläche

  zu realisieren?

Was halten Sie von dem Versuch, einen

- Formel 1 Rennwagen
- zum Containertransport als
- 3-Liter Auto

zu bauen?

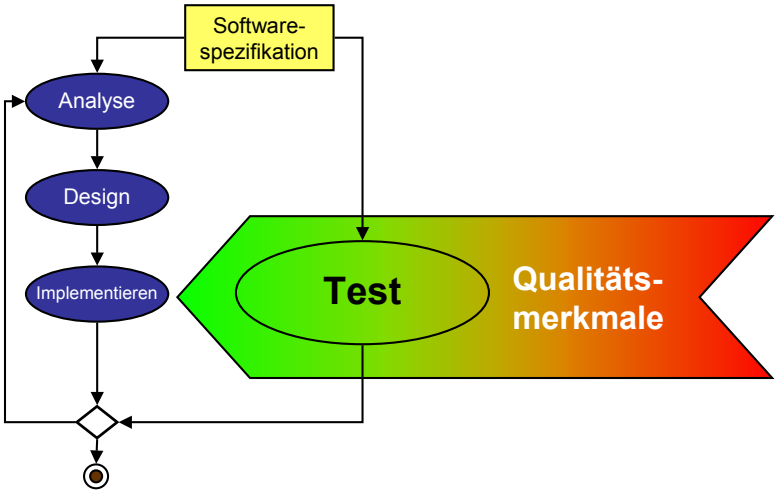
- Was halten Sie von dem Versuch, eine
  - sehr speicher- und zeiteffiziente Software mit
  - sehr hoher Zuverlässigkeit und
  - sehr guter Wartbarkeit
  - mit grafischer, konfigurierbarer Bedienoberfläche

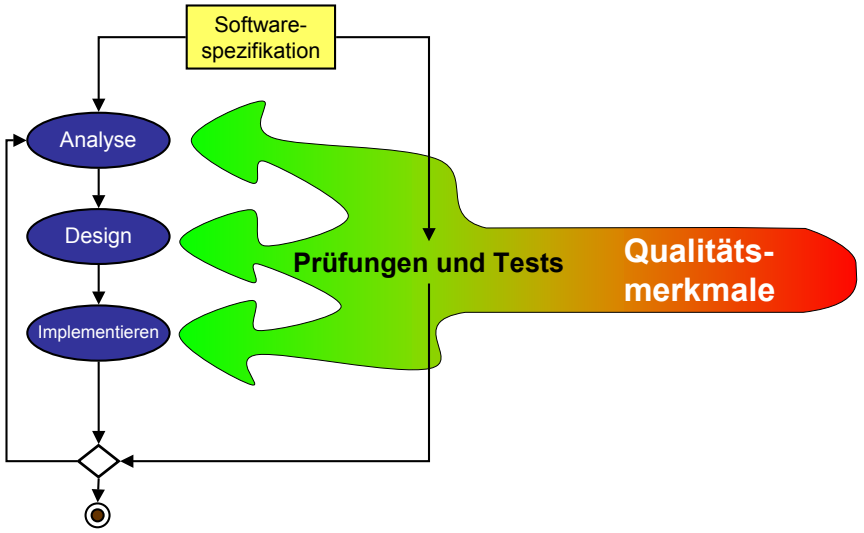
  zu realisieren?

Qualitätsmerkmal	Qualitäts-Te
Funktionalität	Angemessenheit, R… …erabilität, Ordnungsmäßig…
Zuverlässigkeit	Reife, Feh… …erherstellbarkeit
Benutzbarkeit	Verst… …nbarkeit, Bedienbarkeit
Effizienz	…rbrauchsverhalten
Änderbarkeit	…arkeit, Modifizierbarkeit, Stabilität, …keit
Übertra…	…passbarkeit, Installierbarkeit, Konformität, Austauschbarkeit

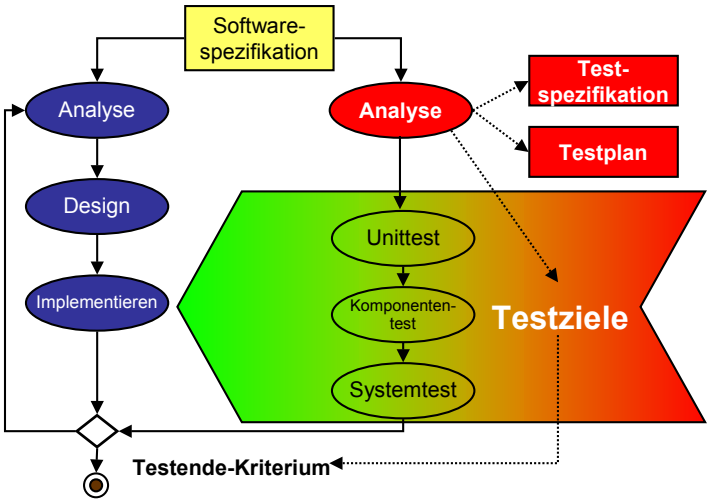
Maximal 4 aus 6

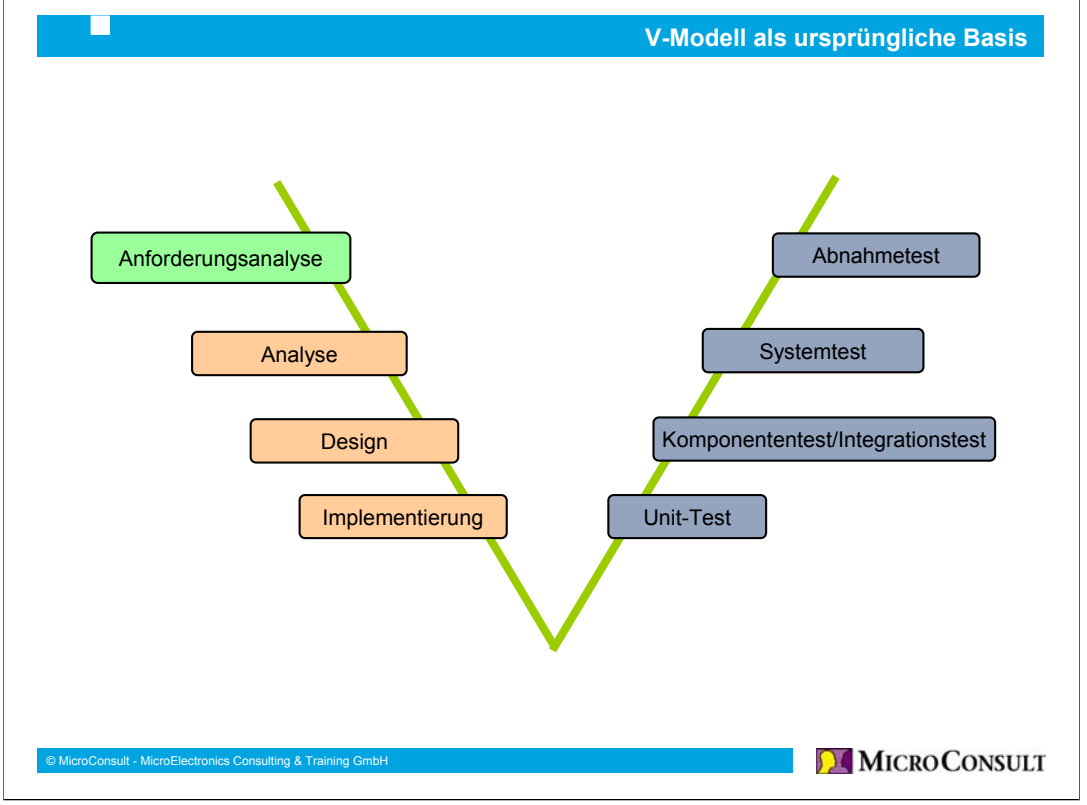
Nach DIN 66272, Ok… …O/IEC 9126 : 1991)





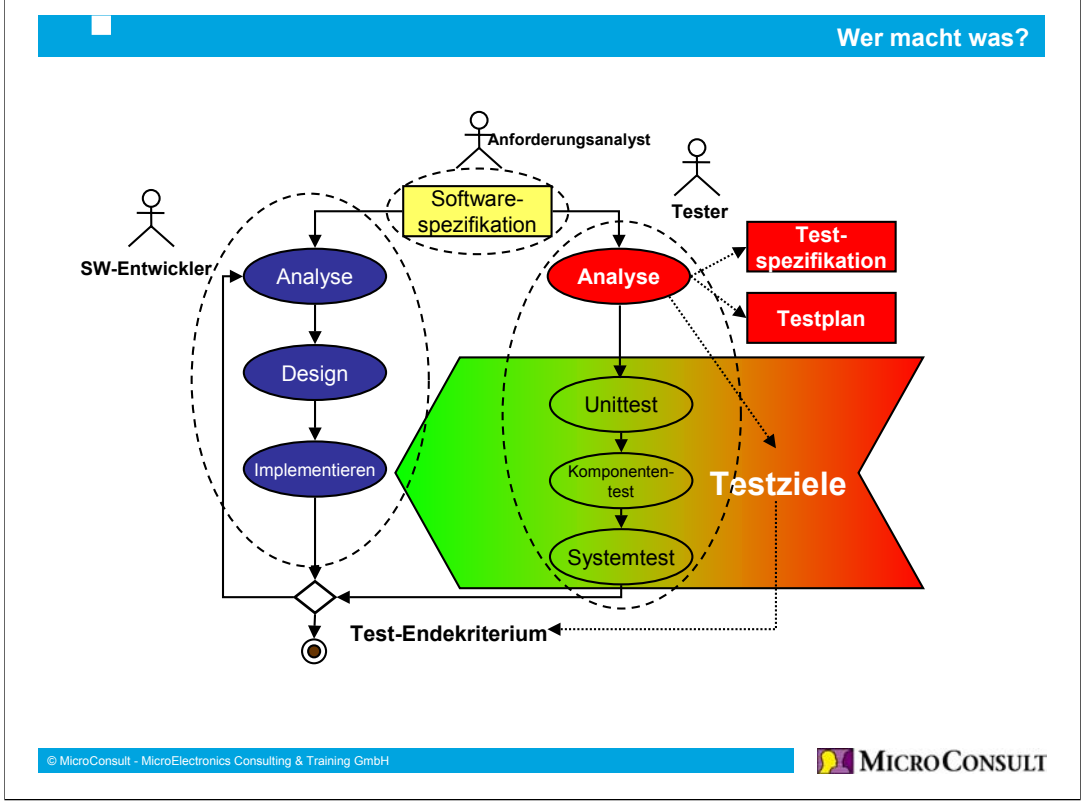
Testziele





Das V-Modell dient als Referenzmodell für das Testen. Heute wird vor allem für Realtime-Systeme kein reines V-Modell verwendet. Realtime-Systeme benötigen meist Optimierungsphasen. Somit wird das „V“ n-mal durchlaufen. Das heißt, SW-Entwicklung wird heute erfolgreich iterativ in Inkrementen betrieben.

Für die Entwicklung einer Teststrategie ist das V-Modell auch heute noch voll gültig. Jeder Phase der Entwicklung wird eine Phase des Test gegenübergestellt. Auf der linken Seite des V bewegt sich der SW-Entwickler von oben nach unten. Auf der rechten Seite bewegt sich der Tester von unten nach oben.



Personen übernehmen Rollen in einem Entwicklungsprozess.

An eine Rolle sind Aufgaben gebunden.

Eine Person kann mehrere Rollen in einem Projekt übernehmen.

Rollen können einander ausschließen.

(Optimal wäre SW-Entwickler und Tester sind verschiedene Personen)

Anforderungsanalyst

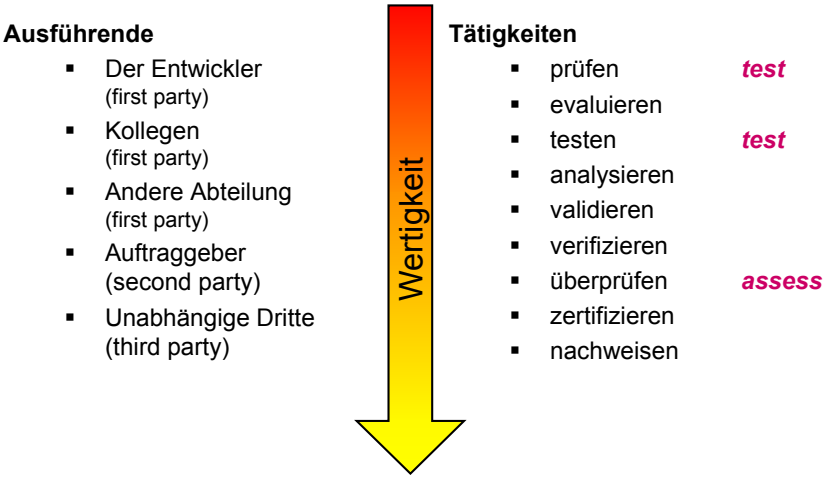
Analysiert die Aufgabe (Lastenheft) und erstellt daraus eine Anforderungsspezifikation.

SW-Entwickler

Entwickelt aus der Anforderungsspezifikation eine Architektur, ein Designmodell und schreibt den Code.

Tester

Entwickelt aus der Anforderungsspezifikation Testfälle, bestimmt die vorzunehmenden Prüfungen, führt, dokumentiert und wertet die Tests aus.



- **Prüfen**
  Technischer Vorgang, der aus dem Bestimmen eines oder mehrerer Merkmale eines bestimmten Erzeugnisses, Verfahrens oder einer Dienstleistung besteht und gemäß einer vorgeschriebenen Verfahrensweise durchzuführen ist. (EN 45 020 : 1993, 12.1).
- **Evaluieren**
  Analysieren und bewerten eines Sachverhalts.
- **Testen**
  Ein Programm wird ausgeführt, um Fehler zu finden.
- **Analysieren**
  Verstehen und Erfassen der Sachverhalte einer Aufgabenstellung, sowie die Entscheidung über die Durchführbarkeit und die Erfolgsaussichten eines künftigen Projektes.
- **Validieren**
  Tut das System das Richtige? Ein Fachmann/frau prüft die Spezifikation oder das gesamte System
- **Verifizieren**
  Tut das System das richtig? Prüfen gegen die Spezifikation.
- **Überprüfen**
  Prüfen der Prüfung. Wurde der Test richtig durchgeführt?
- **Zertifizieren**
  Bestätigung der Einhaltung bestimmter Verfahren durch einen unabhängigen Dritten.
- **Nachweisen**
  Beweisen eines Sachverhaltes (z.B. Programm erfüllt die Anforderungen).

Validieren – Verifizieren (am Beispiel eines Eigenheimes)

Die **Validation** untersucht die Frage:
Wird das richtige Haus gebaut?

Die **Verifikation** untersucht die Frage:
Wird das Haus richtig gebaut?

MICROCONSULT

Gehört von M. Müllerburg, GMD

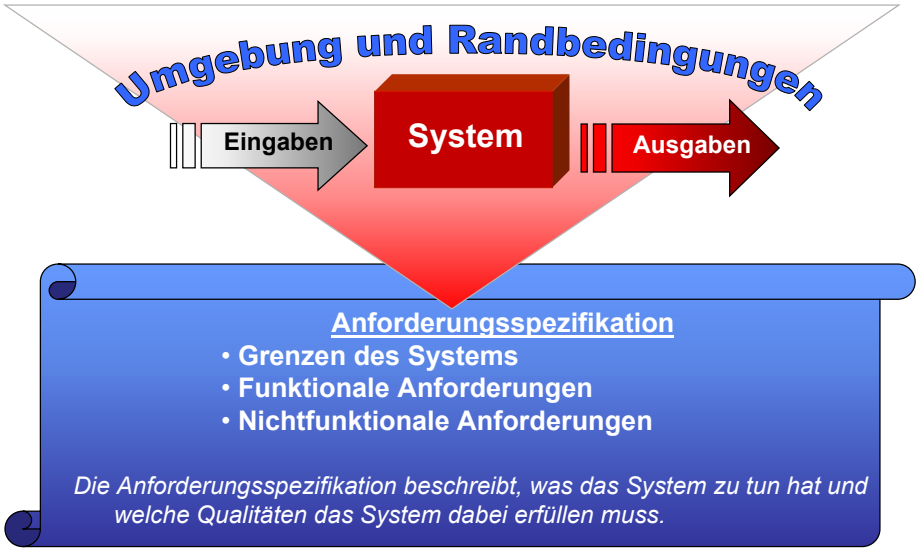
Requirementanalyse und Test

Requirements (Anforderungen) müssen existieren!

- mündlich (typisch)
- schriftlich (empfohlen)
- modellbasiert (die Zukunft)

Mündliche Requirements

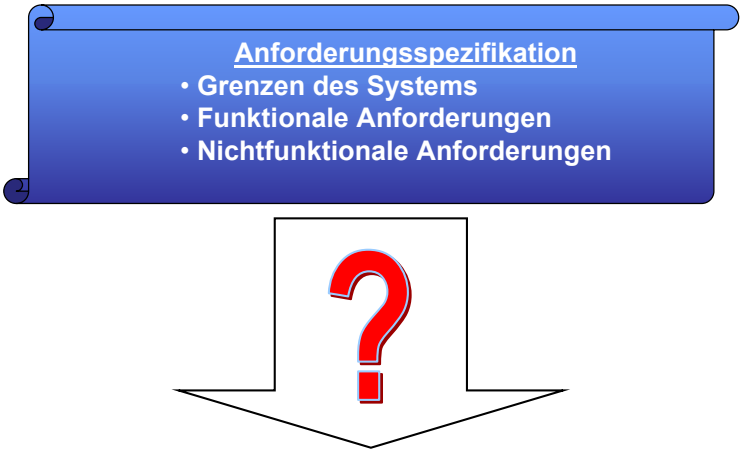
- müssen für den Test analysiert werden
- erfordern eine **ausführlichere** Testspezifikation!



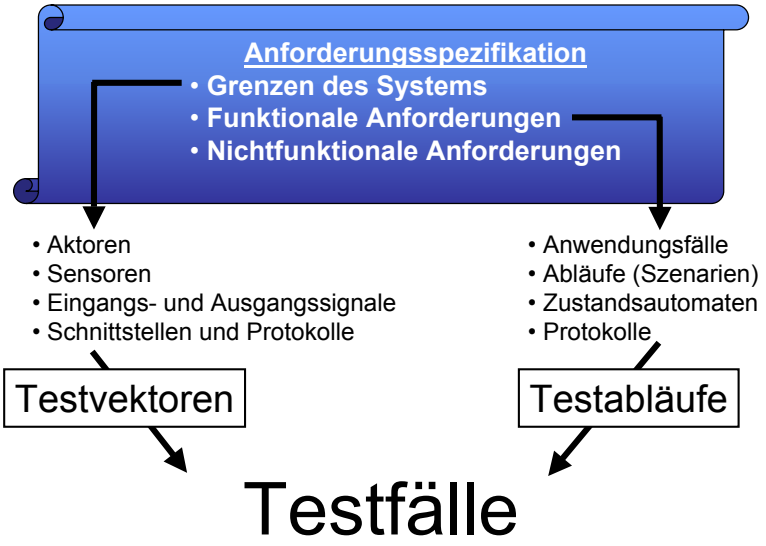
Anforderungsspezifikation

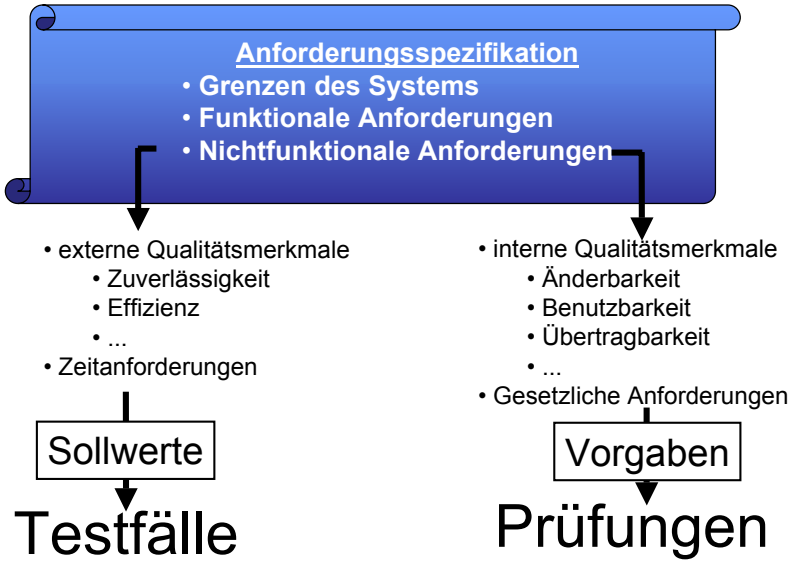
- Grenzen des Systems
- Funktionale Anforderungen
- Nichtfunktionale Anforderungen

Die Anforderungsspezifikation beschreibt, was das System zu tun hat und welche Qualitäten das System dabei erfüllen muss.



Prüfungen und Tests





Testplanung und Testspezifikation

Testplan (Beispiel nach IEEE 829)

- Testplan ID
- Einführung
- Zu testende Komponenten
- Zu testende Funktionen
- Nicht zu testende Funktionen
- Vorgehensweise
- Pass / Fail Kriterien
- Produkte
  - Testtätigkeiten
  - Testumgebung
  - Zuständigkeiten
  - Personal
  - Zeitplan
  - Risiken und Risikomanagement
  - Genehmigungen

Testspezifikation (Beispiel nach IEEE 829)

- Testspezifikation ID
- Zu testende Funktionen
  - Identifikation des zu testenden Objektes
  - Aufführung der Funktionen und der Kombinationen der Funktionen
  - Verlinkung mit der Anforderungs- und der Design-Spezifikation
- Testverfahren
  - Ausführung der Test
  - Auswahl der Testfälle
  - Beschreibung der Umgebung für die Testfälle
- Testskripte und Testfälle
  - Tabellarische Beschreibung der Testfälle
  - Benennung der Testskripte
- Pass / Fail Kriterien

Testplan (Beispiel nach IEEE 829)

- Testplan ID
- Einführung
- Zu testende Komponenten
- Zu testende Funktionen
- Nicht zu testende Funktionen
- Vorgehensweise
- Pass / Fail Kriterien
- Produkte
- Testtätigkeiten
- Testumgebung
- Zuständigkeiten
- Personal
- Zeitplan
- Risiken und Risikomanagement
- Genehmigungen

Testspezifikation (Beispiel nach IEEE 829)

- Testspezifikation ID
- Zu testende Funktionen
  - Identifikation des zu testenden Objektes
  - Aufführung der Funktionen und der Kombination
  - Verlinkung mit der Anforderungs- und d
- Testverfahren
  - Ausführung der Test
  - Auswahl de
  - R
- Testski
  - ng der Testfälle
  - Testskripte
-

Eine gedruckte Kopie des Testplans sollte von den verantwortlichen Personen unterschrieben werden!

Die Teststrategie legt

- die Konzepte zur Findung aussagekräftiger Testfälle fest und
- schränkt die Menge der Testfälle ein.

Entwicklung der Strategie anhand

- der Qualitätsmerkmale
- der Komponenten
- der Risiken

Früh mit den Tests im Entwicklungsprozess beginnen, um Fehler so früh wie möglich zu beseitigen.

	Funktionalität	Zuverlässigkeit	Benutzbarkeit	Effizienz	Übertragbarkeit
Wichtigkeit	40%	30%	10%	10%	10%
Source-Code	+			++	
Unit-Test	+			+	
Komponententest	++	+			+
Systemtest	++	++	+		
Abnahmetest	++	++	++		

	Funktionalität	Zuverlässigkeit	Benutzbarkeit	Effizienz	Übertragbarkeit
Wichtigkeit	40%	30%	10%	10%	10%
Komponente A	+	+		++	
Komponente B	++		++		+
Komponente C	+	++			+

Qualitätsmerkmale lassen sich den Projektphasen zuordnen. Diese müssen aus der Anforderungsspezifikation entnommen oder eingefordert werden. Ein Gewichtung ist unumgänglich. Hier ist der Tester gefordert. Den Projektphasen kann nun das Qualitätsmerkmal zugeordnet werden.

	Funktionalität	Zuverlässigkeit	Benutzbarkeit	Effizienz	Übertragbarkeit
Wichtigkeit	40%	30%	10%		10%
Source-Code	+			++	
Unit-Test	+			+	
Komponententest	++				+
Systemtest	++	++	+		
Abnahmetest	++	++	++		

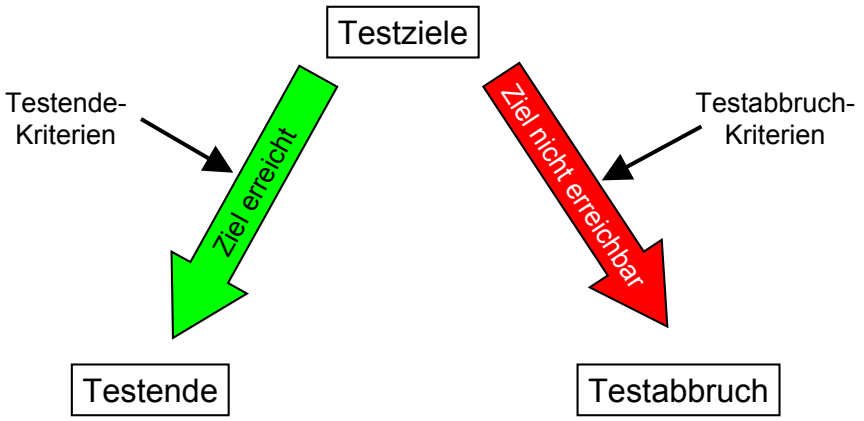
Qualitätsmerkmale und Projektphasen

	Funktionalität	Zuverlässigkeit	Benutzbarkeit	Effizienz	Übertragbarkeit
Wichtigkeit	40%	30%		10%	10%
Komponente A	+	+		++	
Komponente B	++		++		+
Komponente C		++			+

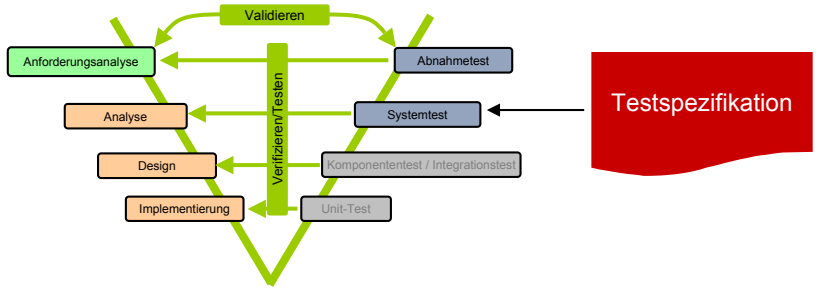
Qualitätsmerkmale und Komponenten

Quelle: nach Testing Embedded Software
Broekmann/Notenboom S. 82

Wann ist der Test zu Ende?

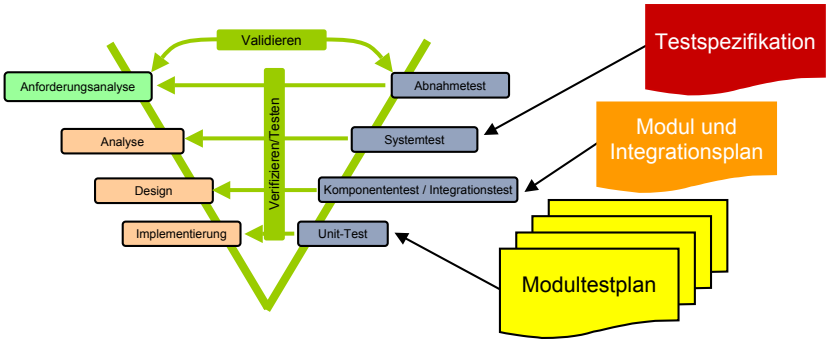


Fokussierung auf den Systemtest



Aufwand für den Test hält sich in Grenzen

- Finden maskierter Fehler schwierig
- Qualität der Komponenten orientiert sich an den Features den Projektes
- Lokalisierung der Fehler ist für den Entwickler schwierig



Einfache Lokalisierung der Fehler durch den Entwickler

Module werden einzeln getestet

Höherer Dokumentationsaufwand

Statische Prüfung

Statische Prüfungen

- Statische Prüfungen sind Verfahren, die eine Bewertung des Codes vornehmen, ohne ihn auszuführen.
- Unterschieden werden manuelle Prüfungen, wie Fagan Inspektion und Walkthroughs, sowie automatische Prüfungen, wie z.B. Compiler, LINT.

Software-Teilprodukte

--- SW Anforderungen ---
Software Anforderungsspezifikation
Software Validierungsplan
--- SW Architektur ---
Software Architekturbeschreibung
Testspezifikation Software-Architektur-Integration
Bedienanleitung Entwicklungswerkzeuge
Codier Richtlinie
--- SW System ---
Software Systembeschreibung
Testspezifikation Software-System-Integration
--- SW Module ---
Software Modul Entwurfs Spezifikation
Software Modul Test Spezifikation
--- Coding ---
Form des Source Codes (händisch, automatisch)
Software Modul Test Report
Software Code Review Report

--- SW Integration ---
Software Modul Integration Test Report
Software System Integration Test Report
Software Architektur Integration Test Report
--- SW Operation and Maintenance ---
Benutzerhandbuch
Bedienanleitung
Wartungsanleitung
--- SW Validation ---
Validierungsreport
--- SW Modification ---
SW Modifikations-Anleitung
SW Modifikations-Anforderung
SW Modifikations-Auswirkungsanalyse
SW Modifikations-Bericht
--- Phasenübergreifende ---
SW Qualitätsplan
SW Verifikationsplan
Verifikationsbericht
Plan zum Nachweis der Qualität
Bericht über den Qualitätsnachweis

MICRO CONSULT

Aus unseren akt. Herstellerchecklisten basierend auf 61508-1

--- SW Anforderungen ---
Software Anforderungsspezifikation
Software Validierungsplan
--- SW Architektur ---
Software Architekturbeschreibung
Testspezifikation Software-Architektur-Integration
Bedienanleitung Entwicklungswerkzeuge
Codier Richtlinie
--- SW System ---
Software Systembeschreibung
Testspezifikation Software-System-Integration
--- SW Module ---
Software Modul Entwurfs Spezifikation
Software Modul Test Spezifikation
--- Coding ---
Form des Source Codes (händisch, automatisch)
Software Modul Test Report
Software Code Review Report

--- SW Integration ---
Software Modul Integration Test Report
Software System Integration Test Report
Software Architektur Integration Test Report
--- SW Operation and Maintenance ---
Benutzerhandbuch
Bedienanleitung
Wartungsanleitung
--- SW Validation ---
Validierungsreport
--- SW Modification ---
SW Modifikations-Anleitung
SW Modifikations-Anforderung
SW Modifikations-Auswirkungsanalyse
SW Modifikations-Bericht
--- Phasenübergreifende ---
SW Qualitätsplan
...der Qualität
Bericht über den Qualitätsnachweis

Testbar

--- SW Anforderungen ---
Software Anforderungsspezifikation
Software Validierungsplan
--- SW Architektur ---
Software Architekturbeschreibung
Testspezifikation Software-Architektur-Integration
Bedienanleitung Entwicklungswerkzeuge
Codier Richtlinie
--- SW Syst
Software Sy
Testspezifik
Integratio
--- SW Mod
Software Modul Entwurfs Spezifikation
Software Modul Test Spezifikation
--- Coding ---
Form des Source Codes (händisch, automatisch)
Software Modul Test Report
Software Code Review Report

--- SW Integration ---
Software Modul Integration Test Report
Software System Integration Test Report
Software Architektur Integration Test Report
--- SW Operation and Maintenance ---
Benutzerhandbuch
Bedienanleitung
Wartungsanleitung
--- SW Modifikation ---
SW Modifikations-Auswirkungsanalyse
SW Modifikations-Bericht
--- Phasenübergreifende ---
SW Qualitätsplan
SW Verifikationsplan
Verifikationsbericht
Plan zum Nachweis der Qualität
Bericht über den Qualitätsnachweis

Inspizierbar

Inspektionen

- Finden von **60% bis 90%** aller Fehler in der Spezifikation und im Code allein durch Inspektionen.
- Reduzierung des Problems der "Unersetzbarkeit" von Entwicklern (Verteilung des Wissens auf mehrere Mitarbeiter).
- Erhöhung der Qualität in einer frühen Phase.
- Verringerung des Test- und Korrekturaufwandes ("Gesundtesten" ist nicht nötig).
- **Inspektionen sind auf jede Art von Dokument anwendbar.**

Dynamische Tests

Dynamische Tests

- Der zu testende Code wird ausgeführt.
- Dynamische Tests werden unter Berücksichtigung der inneren Struktur (White-Box-Test) bis hin zur Betrachtung als geschlossenes System durchgeführt (Black-Box-Test).

White-Box-Test

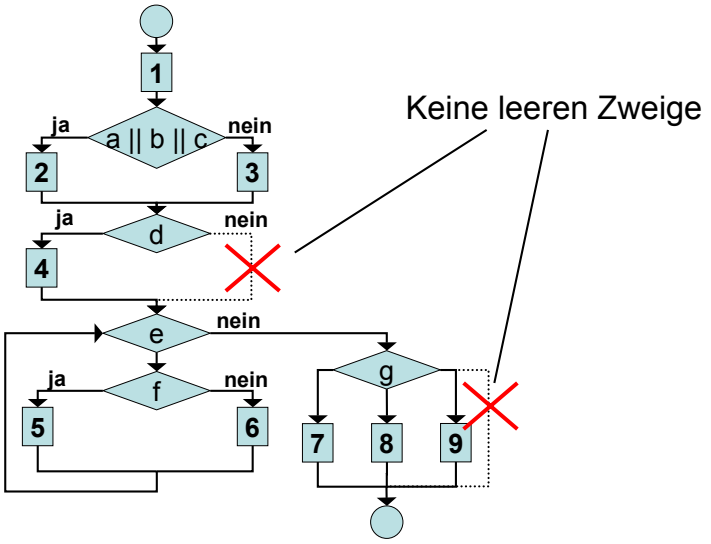
- ***Grundlage ist die innere Struktur des Codes.***

Black-Box-Test

- **Erfolgt ohne Kenntnis der inneren Struktur (Schnittstellentest).**

Gray-Box-Test

- **Kombiniert die Methodik von Black-Box- und White-Box-Tests.**



Tests und Fehleridentifizierungsquote

Kontrollflussorientiert

Anweisungsüberdeckungstest oder C0-Test (18%)

Jeder Befehl im Programm muss mindestens einmal ausgeführt werden.

Zweigüberdeckungstest (34%)

Alle Entscheidungen oder Sprünge werden erfasst (Entwurf entsprechend vieler Testfälle ist nötig). Er enthält den Anweisungsüberdeckungstest vollständig

Bedingungsüberdeckungstest (ca. 50%)

Test aufgrund der Bedingungen von Schleifen und logischen Ausdrücken. Alle Bedingungen, die zum Durchlaufen eines Zweigs führen können, werden getestet.

Pfadüberdeckungstest (>60%)

Ziel ist die Ausführung aller unterschiedlichen Pfade des Programms.

Datenflussorientiert (Statistiken aus Girgis, Woodware `86)

all defs: (24%, keine Kontrollflussfehler)

Jede Definition muss in einer Berechnung oder Bedingung benutzt werden

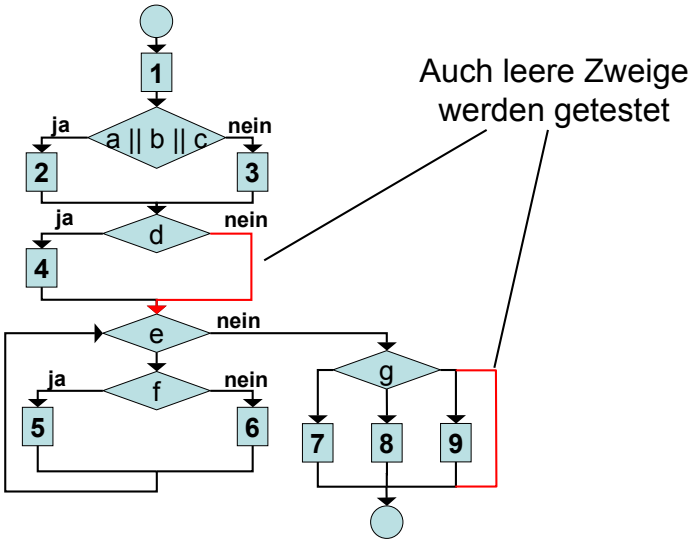
all p-uses: (34%, identifiziert sicher Kontrollflussfehler)

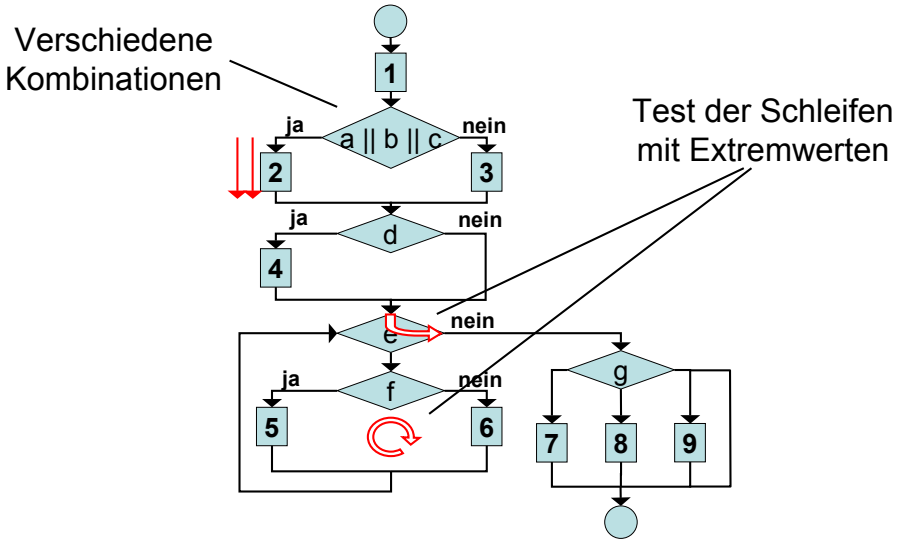
Jede Kombination aus Variablendefinition und deren prädikative Nutzung muß getestet werden

Beinhaltet Zweigüberdeckung

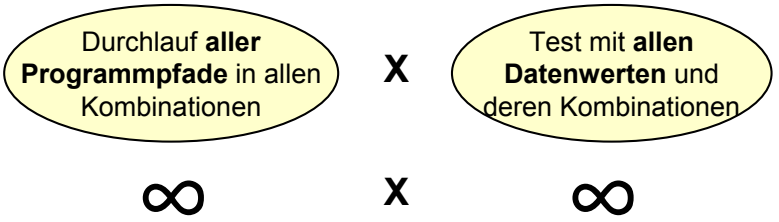
all c-uses: (48%, identifiziert Berechnungsfehler)

Jede Kombination aus Variablendefinition und deren berechnende Benutzung muss getestet werden





Was bedeutet Testabdeckung?



Aber nicht alle Werte und Programmpfade führen zu einem Fehler!

Kontrollflussorientierte Tests betrachten nur den **Programmablauf**.
Daten werden nur indirekt und sehr begrenzt betrachtet.

Vollständiger Test -> unendlich viele Testfälle.

Datenflussorientierte Tests betrachten die **Initialisierung und Nutzung von Daten**.
Es wird aber nicht auf verschiedene Werte dieser Daten eingegangen.

Ein vollständiger -> unendlich viele Testfälle.

Funktion mit einem gültigen Definitionsbereich von 1..12:

int daysOfMonth(int month);

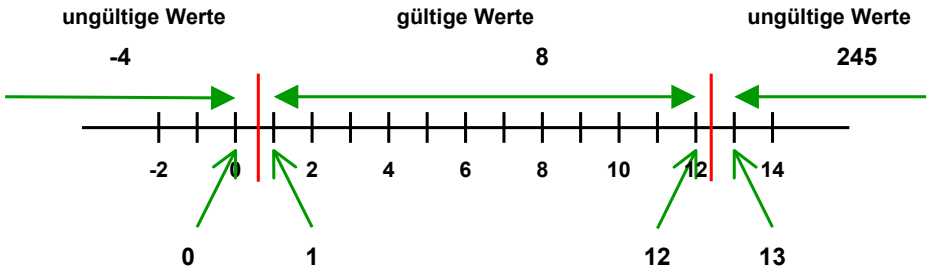
Test mit Zufallszahlen z.B.:

-23, 13, 356

Problem: Kein Wert liegt im gültigen Bereich.

Funktionale Äquivalenzklassenbildung

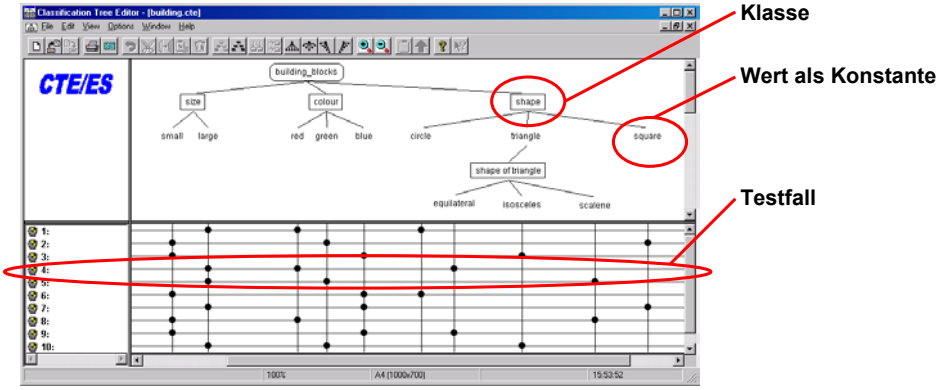
Werte z.B.: -4, 8, 245



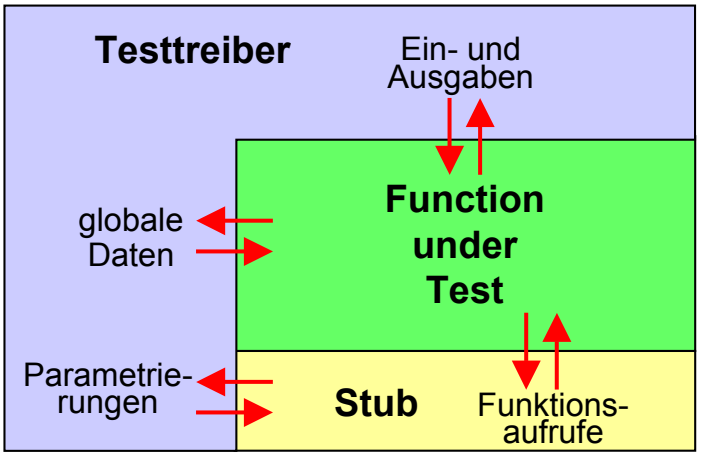
Grenzwertanalyse

Werte: 0, 1, 12, 13

- Grafische Ermittlung von Grenz- und Äquivalenzklassen
- Ausprägung als konkrete Ein-/Ausgaben oder Sequenzen



Die CTE-Methode wurde vom Forschungslabor Softwaretechnologie der DaimlerChrysler AG entwickelt.



Aufgaben des Testtreibers

Vorbedingungen für den Ablauf des Tests schaffen

Eingangsumgebung

- globale Variablen
- statische Variablen in der Testfunktion (Code muss instrumentiert werden)
- Funktionsparameter

Zu testende Funktion ausführen

Nachbedingungen erfassen

Ausgangsumgebung

- globale Variablen
- statische Variablen in der Testfunktion
- Funktionsparameter (Pointer, ...)
- Rückgabewert der Funktion

Ausgaben für die Nachbereitung speichern

Function Under Test

```
extern int max(int a, int b);

int setMaxToPort(int a, int b)
{
    int c;

    c= max(a, b);    // Noch nicht
                     // vorhanden
    return c;
}
```

Teststub

```
int g_retMax;
int max(int a, int b)
{
    return g_retMax;
}
```

Testtreiber

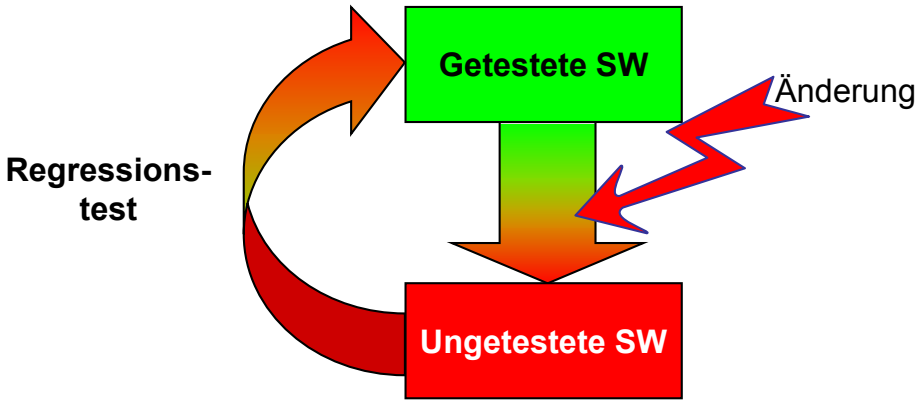
```
int g_retMax;

void main()
{
    int a= 10;
    int b= 20;
    int Result;

    g_retMax= 5;

    Result= setMaxToPort(a, b);

    printf("%d", Result);
}
```



Nach einer Codeänderung darf keine Regression auftreten,
d.h., die geänderten Anteile müssen weiterhin der Spezifikation genügen.
Dies wird durch ***Regressionstests*** geprüft.

Änderungen an existierendem Code können

korrektiv (Spezifikation verletzt),

inkrementell (Spezifikation erweitert),

adaptiv (Spezifikation geändert)

oder *optimierend* (Spezifikation unverändert) sein.

Testauswertung

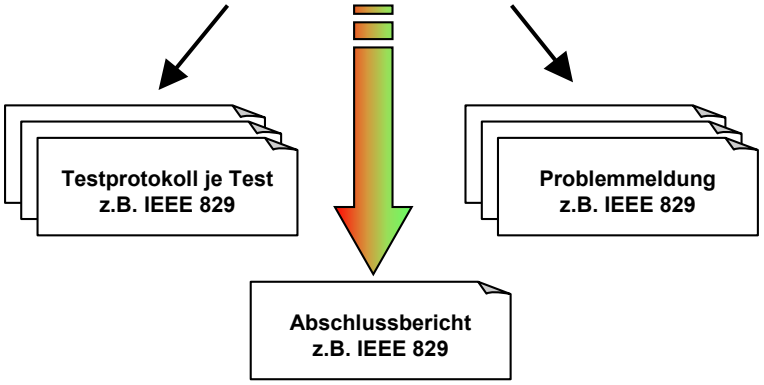
--- SW Anforderungen ---
Software Anforderungsspezifikation
Software Validierungsplan
--- SW Architektur ---
Software Architekturbeschreibung
Testspezifikation Software-Architektur-Integration
Bedienanleitung Entwicklungswerkzeuge
Codier Richtlinie
--- SW Syst
Software Sy
Testspezifik
Integratio
--- SW Module ---
Software Modul Entwurfs Spezifikation
Software Modul Test Spezifikation
--- Coding ---
Form des Source Codes (händisch, automatisch)
Software Modul Test Report
Software Code Review Report

Reports

--- SW Integration ---
Software Modul Integration Test Report
Software System Integration Test Report
Software Architektur Integration Test Report
--- SW Operation and Maintenance ---
Benutzerhandbuch
Bedienanleitung
Wartungsanleitung
--- SW Validation ---
Validierungsreport
--- SW Modification ---
SW Modifikations-Anleitung
SW Modifikations-Anforderung
SW Modifikations-Auswirkungsanalyse
SW Modifikations-Bericht
--- Phasenübergreifende ---
SW Qualitätsplan
SW Verifikationsplan
Verifikationsbericht
Plan zum Nachweis der Qualität
Bericht über den Qualitätsnachweis

Testergebnisse müssen dokumentiert werden!

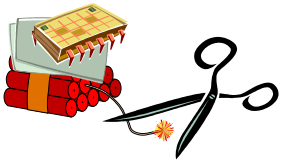
Fehler müssen dokumentiert werden, um das Wissen um sie verfügbar zu machen und sie zu beheben, nicht um Schuldzuweisungen machen zu können.



Testkosten

Werkzeuge und
Testumgebung

Testspezifikation
Testplanung
Testdesign



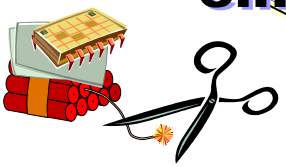
Kosten für die Fehlerbeseitigung

Testdurchführung
Testdokumentation

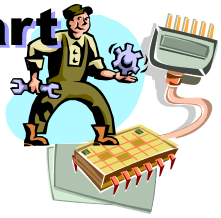
Werkzeuge und
Testumgebung

Testspezifikation
Testplanung
Testdesign

Reviews wurden eingespart

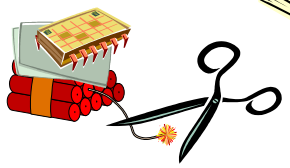


Kosten für die Fehlerbeseitigung



Testdurchführung
Testdokumentation

Werkzeuge und
Testumgebung



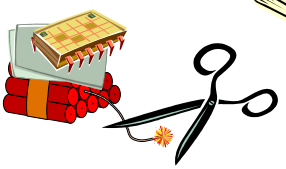
Kosten für die Fehlerbeseitigung

Testdurchführung
Testdokumentation

Werkzeuge und
Testumgebung

Testspezifikation
Testplanung
Testdesign

Detaillierte Planung



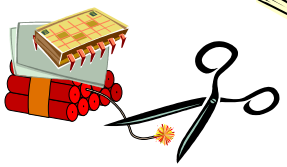
Kosten für die Fehlerbeseitigung

Testdurchführung
Testdokumentation

Werkzeuge und
Testumgebung

Testspezifikation
Testplanung
Testdesign

Design for Test



Kosten für die Fehlerbeseitigung

Testdurchführung
Testdokumentation

Die Prüfung der Software ist ein geplanter Vorgang!

Sie orientiert sich an klar definierten Qualitätsmerkmalen!

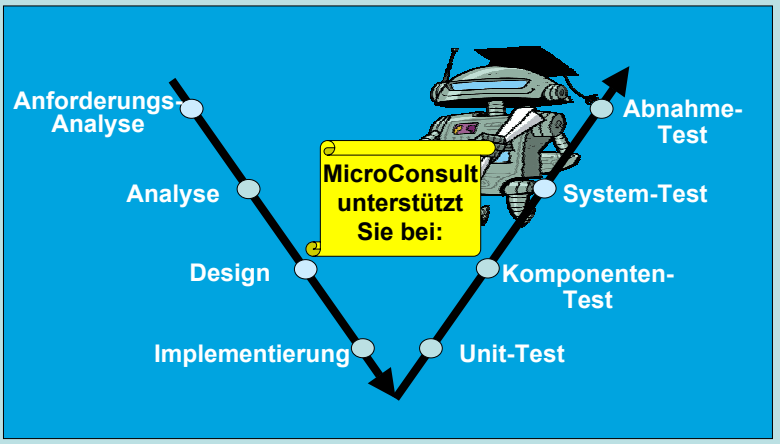
Sie setzt klare Anforderungen (Requirements) voraus!

Prüfen ist allgemein! Nur Ausführbares kann getestet werden!

Entwicklung und Test gehen Hand in Hand (Design for Test)!

Der richtige Mix aus statischer Prüfung und dynamischen Tests führt zum Erfolg!

Training, Coaching, Engineering



HW-/SW-Technologien, Tools, Methoden, Prozess, Team