1. Top-down versus bottom-up-Strategie beim Testen von Modulen

einfaches Beispiel: Ermittlung von Tag, Monat und Jahr anhand des Systemdatums
Grundsätzlicher Aufbau des Programms

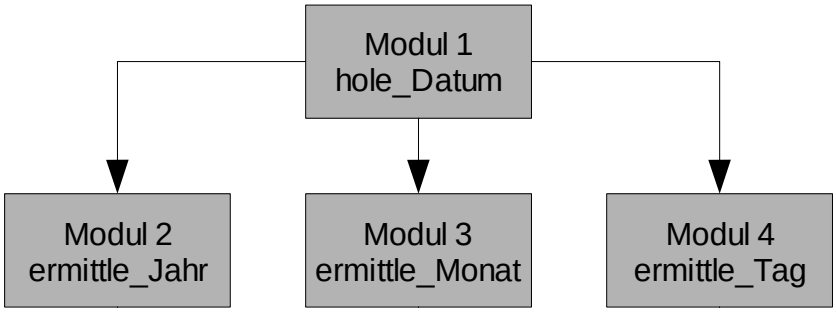

Abb. 1: Das Struktogramm des Beispielprogramms

1.1. Bottom up

- Jedes Modul wird für sich alleine getestet
- Es werden Testprogramme, sogenannte Treiber oder Driver benötigt

Nachteile: - da das Programm nur in Teilen getestet wird können Schnittstellenfehler nicht ausgeschlossen werden
Vorteile: - Beobachtung der Ergebnisse einfach

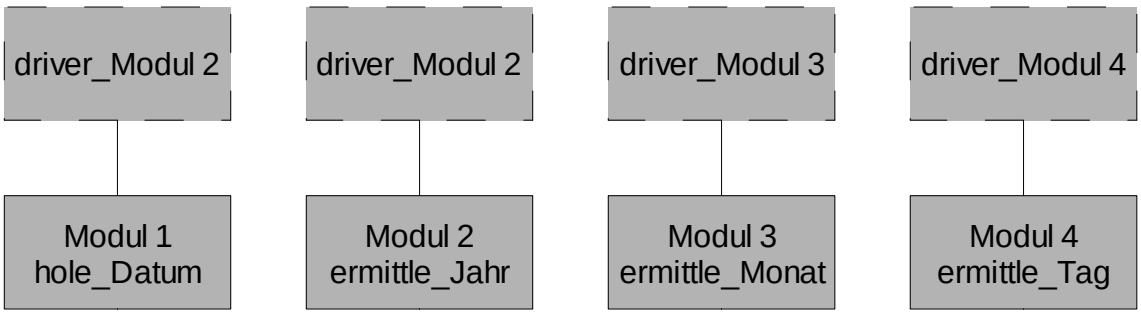

Abb. 2: Treiberprogramme testen die Programmodule

1.2. Top down

- zunächst wird oberstes Modul getestet, die anderen Module werden durch Platzhalter (stubs oder dummies) ersetzt
- nach und nach werden die anderen Module hinzugenommen (Incremental Testing)

Nachteile: - stubs (Platzhalter) sind komplizierter zu programmieren als driver (Treiberprogramme)
- Beobachtung der Ergebnisse schwieriger
Vorteile: - schwerwiegende Fehler in der Spitze der Struktur werden leichter erkannt
- Programm wird mit Schnittstellen als Ganzes getestet

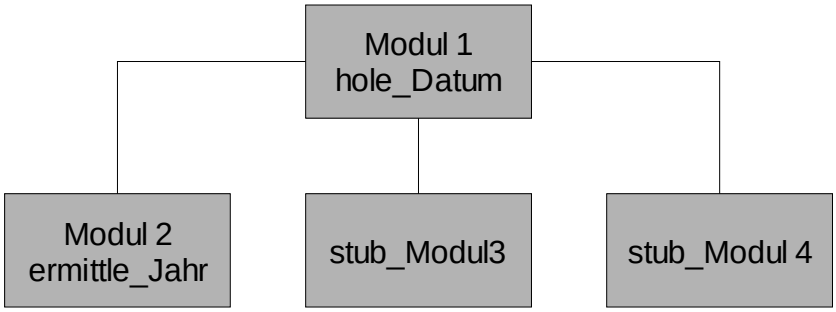

Abb. 3: Module 3 und 4 wurden durch stubs ersetzt

2. Der White Box Test

- der innere Aufbau der Module und der Kontrollfluss ist bekannt
- der Programmcode liegt offen vor, wie in einem gläserner Behälter
- Kriterium für den White Box Test ist die Ausführung möglichst jedes Programmpfades beim Testen

Schwächen:

1. Intensives Prüfen beweist nicht, dass das Programm mit seiner Spezifikation übereinstimmt.
2. Das Programm oder das Modul kann einige spezifizierte Funktionen nicht enthalten.
3. Fehler in den Daten werden häufig nicht aufgedeckt.

-> *externe Testgruppe benötigt*

3. Der Black Box Test

- Tester beurteilt Programmcode lediglich nach Funktion, die eigentliche Implementierung ist nicht bekannt
- Das detaillierte Lastenheft ist ein wichtiger Leitfaden
- Keine Kriterien für die Testabdeckung wie beim White Box Test möglich
- Tester braucht viel Erfahrung und Geschick

Die wichtigsten Grundsätze:

1. *Ein Programmierer sollte nie versuchen, sein eigenes Programm zu testen*
2. *Das Testen von Software ist ein Experiment*
3. *Das Testen ist definiert als die Ausführung eines Programms mit der erklärten Absicht, Fehler zu finden*
4. *Definition der erwarteten Ergebnisse vor dem Beginn des Tests*
5. *Die Wahrscheinlichkeit, in einem bestimmten Segment des Programmcodes in der näheren Umgebung eines bekannten Fehlers weitere zu finden, ist überproportional hoch*

4. Der Gray Box Test

- Mischung aus White und Black Box Test
- Tester kennt in groben Zügen die Implementierung und kann Schwachpunkte in der Entwicklungsphase leichter erkennen
- Vorteile beim Kompatibilitätstest, z.B. von Druckern

5. Die wichtigsten Testausprägungen

5.1. Der Funktionstest
- Funktionen der Software werden auf Modulebene getestet

5.2. Der Volume Test
- Programmcode muss grosse Mengen von Daten bearbeiten

5.3. Der Stress-Test
- Programm wird innerhalb eines kurzen Zeitraums mit hoher Belastung ausgesetzt

5.4. Speicherverbrauch und Auslastung des Prozessors
- besonders wichtig bei Echtzeitsystemen

5.5. Recovery Testing
- Ausfall eines Computers der eines Programms darf nicht zu Datenverlusten oder Leistungsminderung führen

5.6. Der Mutationstest
- die verwandten Testfälle werden selbst einem Test unterzogen

5.7. Benchmarks
- Überprüfung der Leistungsfähigkeit der Software

5.8. Der Test von Prozeduren und Verfahren
- Anwendungsfall wird auf mögliche Missverständnisse/Fehlerquellen und Effizienz geprüft

5.9. Configuration Testing
- Der durchschnittliche Benutzer verfügt nicht über einen ebenso leistungsstarken Computer wie der Entwickler

5.10. Usability Testing
- Programm/Website wird auf seine Bedienbarkeit hin überprüft

5.11. Überprüfung von Dokumenten
- Dokumentation wird auf gültigen und vollständigen Inhalt hin untersucht

5.12. Der Systemtest

6. Test-Automation

- Bei jedem neuen Release muss das Programm wieder getestet werden, Anpassung der alten Testfälle ist leichter als Neuerstellung
- Es können mehrere Testläufe gleichzeitig ausgeführt und somit mehrere Benutzer simuliert werden
- Testläufe lassen sich auf verschiedenen Rechnerkonfigurationen wiederholen

Quelle: Georg Erwin Thaller: Software-Test, Verifikation und Validation, Heise Verlag