# Testkonzept

**Ablauf im Detail**

Während der Entwicklung einzelner Klassen werden für diese Klassen ebenfalls Testklassen geschrieben.Sie werden nach folgender Konvention benannt:
Sei <Name> der Name der zu testenden Klasse.Damit folgt intuitive der Name für die Testklasse:Test<Name>
Wenn ein Softwaremodul fertig gestellt ist,so wird es dem Verantwortlichen für Tests-übergeben.
Der Testverantwortliche hat folgende Aufgaben:

1. Es wird geprüft ob die Funktionalität des Moduls gewährleistet ist.Dazu wird der Testverantwortliche die in der Test Suite für dieses Modul aufgeführten Tests durchführen (JUnit Tests)
2. Der Testverantwortliche führt Integrationstests mit dem Modul durch
3. Der Testverantwortliche wird dann die Funktionalität im Gesamtsystem prüfen
4. Zu letzt werden die Produktleistungen und Qualitätsanforderungen wie sie im Pflichtenheft notiert sind geprüft
5. Die Testergebnisse werden in einem Testprotokoll vermerkt

**Testphasen**

1. Komponententests

Bei diesen Tests handelt es sich um einzelne Klassen.Die verantwortlichen Implementierer der verschiedenen Klassen implementieren diese Tests und führen sie durch.
Die Klassen werden in wachsender Reihenfolge ihrer Komplexität nach getestet.Zuerst werden die Klassen getestet,die von den anderen Klassen unabhängig sind,danach werden komplexere Gebilde getestet,die dann auf die schon getesteten,einfacheren Klassen zugreifen.Am Ende einer Zeitscheibe liegen die geforderten Klassen vor samt Test-Cases,welche die geforderte Funktionalität der Klassen nachweisen.Ein Programmierauftrag ist erfüllt,wenn die geforderten Klassen samt Tests vorliegen und die Tests vom Testbeauftragten als zweckmäßig eingestuft sind.

2. Integrationstests

Ziel des Integrationstests ist es,die Kommunikation und das Zusammenwirken der zu größeren Teilsystemen zusammengefügten Komponenten zu überprüfen.Dabei sollen Schnittstellenfehler aufgedeckt werden,welche durch den Komponententest nicht entdeckt werden können.Vor allem sollte nach jeden hinzugefügten Integrationsschritt ein Testschritt durchgeführt werden,da mit großer Wahrscheinlichkeit eine dabei aufgetretene Fehlerwirkung durch die gerade hinzugenommene Komponente verursacht wurde,und somit die Problemstelle eingegrenzt werden kann.Es lassen sich verschiedene Ansätze beim Testen

unterscheiden,von denen man für sich die geeignete auswählen kann.z.B.
Bottom-up Integration:Integration der Komponenten von den Blättern des Abhängigkeitsbaums zur Wurzel
Top-down Integration:Integration der Komponenten von der Wurzel des Abhängigkeitsbaums zu den Blättern.

3. Systemtests

Der Systemtest stellt die abschließende Stufe des Testens dar.Im Gegensatz zu den ersten beiden Stufen ist dies kein Test,bei dem dem Tester Informationen über den internen Aufbau des Testobjekts zur Verfuegung stehen (White-Box-Test).Vielmehr ist es ein Test ,bei dem das System als schwarzer Kasten betrachtet wird (Black-Box-Test),dessen innere Struktur und Funktionalität dem Tester verborgen bleibt.
Testfälle können nur anhand extern sichtbarer Schnittstellen und der Spezifikation des Testobjekts abgeleitet werden.
Der Systemtest wird unter der Leitung der Verantwortlichen für Test und unter Teilnehme aller Teammitglieder durchgefuehrt.

**Über JUnit**

JUnit ist ein Test-Framework,und stellt dabei Klassen zur Verfügung,welche den automatischen Test,schon während der Entwickung,ermöglichen.Die Software ist frei erhältlich unter WWW.junit.org.Die Tests werden direkt in Java kodiert,sie über-
prüfen sich selbst und sind somit wiederholbar.
Testfälle können mit JUnit einfach organisiert und mit Hilfe von einem Plugin in Eclipse ausgeführt werden.Als Softwareentwickler erhalten wir mit JUnit ein gebrauchsfertiges Werkzeug,um unsere Testprozesse konsistent und konsequent automatisieren zu können.
Dieses Werkzeug bietet zwei Möglichkeiten zur Durchführung von Tests.Die erste Möglichkeit ist,die einzelnen Module fertig zu entwickln,anschließend mittels JUnit Test-Cases zu erzeugen und diese auf die Module anzuwenden.Nach erfolgreichem Ablauf der Tests kann Code Refactoring betrieben werden.Dies ist eine Variante einen Quellcode in strukturierter Form zu bringen.
Eine zweite Möglichkeit ist,mit Hilfe von JUnit erst die Tests zu schreiben,bevor wir den Code entwickeln,der diese erfüllt.Somit können wir sicherstellen,dass unserer Code einfach zu testen ist,aufgrund der inkrementellen Entwicklung des Programms.