**\subsection{Qualitätszielbestimmungen-/anforderungen}**

Das zu entwerfende Programm (UnivIS 2.0) hat folgende Qualitätsziele, die nachfolgend erläutert werden. Zunächst muss die Software eine

/Q10/ Das Programm muss korrekte Funktionalität aufweisen, d. h. sie muss richtige Ergebnisse bei Eingaben und Ausgaben liefern.

/Q20/ Zudem muss gewährleistet sein, dass die Daten sicher und vom Zugriff unberechtigter geschützt sind.

Zudem muss das Programm zuverlässig arbeiten. Es dürfen möglichst keine Fehler passieren, auch die Datenrückgewinnung muss gewährleistet sein, wenn der Server abstürzt.

Eine leichte und intuitive Bedienung von UnivIS 2.0 ist von Nöten, da eine Vielzahl von potenziellen Nutzern auf das System zugreifen soll. Die Heterogenität der Studenten und des Personals macht es erforderlich, dass sich alle Beteiligten schnell an die Bedienung des Programms gewöhnen und keine langen Einarbeitungsphasen oder sogar Kurse notwendig werden.

Damit die Nutzer des Systems effizient arbeiten können, darf im Mittel die Ergebnisausgabe eines Stundenplans zwei Sekunden in Anspruch nehmen.

Die im Lastenheft gemachte Angabe, dass das System eventuell universitätsweit und auf mobilen Plattformen zum Einsatz kommen kann, macht er erforderlich, dass die Änderbarkeit und Anpassung des Programms schnell erfolgen kann. Maximal drei Monate sollten hierfür aufgewendet werden müssen.

Schlussendlich muss das Programm an andere Windowsversionen anpassbar sein. Dies ist gewährleistet, da eine JAVA-Datei verwendet wird, welche plattformunabhängig läuft.

**\subsection{Globale Testszenarien und Testfälle}**

Damit festgestellt werden kann, ob die Software auch einwandfrei funktioniert, müssen verschiedene Testszenarien durchlaufen werden.

Die Tests werden auf verschiedenen Ebenen ausgeführt. Unterschieden wird hierbei zwischen dem Modultest, Integrationstest, Systemtest und letztendlich dem Abnahmetest.

Im Folgenden werden nun diese Testszenarien genauer beschrieben und dargelegt, wie sie am Programm durchgeführt werden.

Im Komponententest, welcher wie auch der nachfolgende Modultest unter White-Box-Tests fällt, beinhaltet das Testen von einzeln abgrenzbaren Teilen (Modulen) des Gesamtprogramms.

Das Testen erfolgt bei White-Box-Tests in der Entwicklungsumgebung Eclipse, wo der Quelltext sichtbar ist. Einzelne Klassen, Funktionen und Unterprogramme werden im Modultest dem Tester sequenziell unterworfen und auf Fehlerfreiheit geprüft. Dies umfasst unter anderem eine korrekte Ausgabe von Uhrzeiten. Auch SQL-Abfragen an die Datenbank ob die Passwortabfrage korrekt abgespeichert wurden, soll der Test beinhalten.

Nachdem im Modultest jedes Modul einzeln überprüft wurde, wird nun im Integrationstest dazu übergegangen, auch die Interaktion einzelner Module mit anderen Modulen zu überprüfen.

Hier soll u. a. getestet werden, ob der Live-Ticker auch alle Lehrveranstaltungen anzeigt, die in Kürze starten. Zudem soll geschaut werden, ob die Stundenplanausgabe die richtigen Inhalte ausgibt und diese anschließend in der PDF korrekt formatiert ausgegeben werden.

Die dritte Teststufe beherbergt nun den Systemtest, bei dem das System gegen sämtliche Anforderungen aus dem Pflichtenheft getestet wird. Ab hier erfolgt das Testen nicht mehr in der Entwicklungsumgebung, sondern wird direkt am lauffähigem Programm ausgeführt. Dies wird auch als Black-Box Test bezeichnet.

Hier werden nun 200 Lehrveranstaltungen zu Testzwecken angelegt. Diese werden vom Systemverwalter manipuliert und auch die Suche nach Alternativterminen für Veranstaltungen wird hierbei überprüft. Auch wird geschaut, ob die persistente Speicherung der Stammdaten

(z. B. Name und Uhrzeit der Lehrveranstaltung) funktioniert. Hierfür wird ein Systemabsturz simuliert, bei dem aufeinanderfolgend erst dem Rechner, dann dem Server der Strom entzogen wird und beide somit neu booten müssen. Anschließend wird eine neue Datenbankabfrage ausgehend des Systems gemacht, um zu sehen, ob die Daten noch vorhanden sind.

Schlussendlich erfolgt der Abnahmetest. Dort dieser wird allerdings nicht von unserem Softwareunternehmen ausgeführt, sondern erfolgt durch den Abnehmer selbst. Durch den tatsächlichen Gebrauch können hierbei noch Fehler entdeckt werden, die bislang nicht auftauchten, da unter Umständen eine andere Reihenfolge der Menüaufrufe, etc. stattfindet. Der Endabnehmer muss die Software akzeptieren, damit die Tests erfolgreich beendet werden können. Besteht das Programm diesen letzten Test nicht, muss es von der SF-GmbH nachgebessert werden.