- Anforderungen-

Network-Simulator-and-Analyzer (NSA)

Version: 1.1

|  |  |  |
| --- | --- | --- |
| Projektbezeichnung | Network-Simulator-and-Analyzer (NSA) | |
| Projektleiter | Tamara Graf, Theresa Schuster | |
| Verantwortlich | Tamara Graf, Theresa Schuster | |
| Erstellt am | 04.04.16 | |
| Zuletzt geändert | 27.05.2016 10:39 | |
| Bearbeitungszustand | X | in Bearbeitung |
|  | vorgelegt |
|  | fertig gestellt |

Weitere Produktinformationen

|  |  |
| --- | --- |
| Mitwirkend | Tamara Graf, Theresa Schuster Projektleiter  [nicht beteiligt] Projektmanager  Prof. Dr. Andreas Aßmuth Anwender |

Änderungsverzeichnis

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Änderung | | | Geänderte Kapitel | Beschreibung der Änderung | Autor | Zustand |
| Nr. | Datum | Version |
| 1 | 21.04.16 | 1.0 | Alle | Initiale Produkterstellung | Tamara Graf, Theresa Schuster | 1.0 |
| 2 | 25.4.16 | 1.1 | 3.2, 3.4 | Use Cases und Datenmodell nach Besprechung angepasst / aktualisiert | Tamara Graf | 1.1 |
| 3 | 03.05.16 | 1.2 | 3.2 | Use Cases überarbeitet | Theresa Schuster | 1.2 |
| 4 | 27.5.16 | 1.3 | 3.2 | Use Cases überarbeitet |  |  |

Prüfverzeichnis

Die folgende Tabelle zeigt einen Überblick über alle Prüfungen – sowohl Eigenprüfungen wie auch Prüfungen durch eigenständige Qualitätssicherung – des vorliegenden Dokumentes.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Datum | Geprüfte Version | Anmerkungen | Prüfer | Neuer Produktzustand |
|  |  |  |  |  |

Inhalt

[1 Einleitung 4](#_Toc454104563)

[2 Ausgangssituation und Zielsetzung 4](#_Toc454104564)

[3 Funktionale Anforderungen 6](#_Toc454104565)

[3.1 Use-Case Übersicht 6](#_Toc454104566)

[3.2 Use-Beschreibungen 7](#_Toc454104567)

[3.3 (Sonstige) Funktionalität 21](#_Toc454104568)

[3.4 Modell des Problembereichs (Konzeptionelles Datenmodell) 22](#_Toc454104569)

[4 Nicht-Funktionale Anforderungen 22](#_Toc454104570)

[4.1 Benutzbarkeit (Usability) 22](#_Toc454104571)

[4.2 Zuverlässigkeit (Reliability) 24](#_Toc454104572)

[4.3 Leistung (Performance) 24](#_Toc454104573)

[4.4 Unterstützbarkeit (Supportability) 24](#_Toc454104574)

[5 Skizze der Gesamtsystemarchitektur 25](#_Toc454104575)

[6 Lieferumfang 26](#_Toc454104576)

[7 Abnahmekriterien 26](#_Toc454104577)

[8 Abkürzungsverzeichnis 26](#_Toc454104578)

[9 Literaturverzeichnis 27](#_Toc454104579)

[10 Abbildungsverzeichnis 27](#_Toc454104580)

# Einleitung

Dieses Dokument enthält alle an das zu entwickelnde System gestellten Anforderungen. Die Gliederung orientiert sich am Aufbau des V-Modell-XT®[[1]](#footnote-1)-Produkts „Anforderungen (Lastenheft)“, ist jedoch für die Verwendung in der Veranstaltung **„Software-Projekte“** im Studiengang **„Angewandte Informatik“** der **OTH-Amberg-Weiden** angepasst worden (und nicht konform zum V-Modell-XT): Teilnehmer dieser Veranstaltung erhalten von ihrem „Auftraggeber“ lediglich einen Überblick über das gewünschte System (siehe das Thema „Ausgangssituation und Zielsetzung“ in diesem Dokument); die Anforderungen müssen die Teilnehmer dann in enger Abstimmung mit ihrem „Auftraggeber“ selbst erarbeiten und in diesem Dokument niederlegen. Dadurch sollen sie Gelegenheit erhalten, sich in der Herausarbeitung von Anforderungen intensiver zu üben.

Weil der „Auftraggeber“ also nicht wirklich ein Lastenheft liefert, aus dem die „Auftragnehmerseite“ (=Teilnehmer am Software-Projekt) dann ein separates Pflichtenheft ableitet, dient dieses Dokument als Ersatz für Lasten- und Pflichtenheft im Sinne des V-Modell-XT.

Kern dieses Dokuments sind die funktionalen und nicht-funktionalen Anforderungen an das System, sowie eine Skizze des Gesamtsystementwurfs. Der Entwurf berücksichtigt die zukünftige Umgebung und Infrastruktur, in der das System später betrieben wird, und gibt Richtlinien für Technologieentscheidungen. Ebenfalls Teil der Anforderungen ist die Festlegung von Lieferbedingungen und Abnahmekriterien.

Die funktionalen und nicht-funktionalen Anforderungen dienen nicht nur als Vorgaben für die Entwicklung, sondern sind zusätzlich Grundlage der Anforderungsverfolgung und des Änderungsmanagements. Die Anforderungen sollten so aufbereitet sein, dass die Verfolgbarkeit (Traceability) sowie ein geeignetes Änderungsmanagement für den gesamten Lebenszyklus eines Systems möglich sind.

Im Allgemeinen sollten keine technischen Lösungen vorgegeben werden, um Architekten und Entwickler bei der Suche nach optimalen technischen Lösungen nicht einzuschränken.

# Ausgangssituation und Zielsetzung

Im Rahmen des Moduls *Software-Projekt*, an welchem die Studenten der angewandten Informatik laut Studienplan der OTH Amberg-Weiden im 6. Semester teilnehmen, entsteht ein kleines Software-Entwicklungsprojekt. Die Durchführung erfolgt eigenständig in einem kleinen, studentischen Team.

Das Ziel des Projektes in diesem Sommersemester ist die Erstellung einer Software für das Modul Rechnernetze. Dieses Modul findet im 3. Semester statt und besteht neben dem seminaristischen Unterricht unter anderem auch aus einem Praktikum. Eine erfolgreiche Teilnahme an diesem ist Voraussetzung für die Teilnahme an der Prüfung am Ende des Semesters.

Aus diesem Grund liegt der Schwerpunkt des Software-Projektes auf der Erstellung einer Lernsoftware für das Labor Netzte und Systeme um Studenten während des Praktikums eine wertvolle Unterstützung zu bieten.

Die Studenten sollen die Möglichkeit bekommen mit Rechnernetzkonfigurationen auf möglichst einfache Weise zu experimentieren und auf diese Art die komplexen Zusammenhänge möglichst schnell und grundlegend erfassen. Sollten Probleme auftreten, soll die Software nur unterstützende Hinweise geben, sodass die Fehlerursache eigenständig erkannt und behoben wird.

Ein wichtiger Bestandteil der Planungsphase ist die Identifikation aller Stakeholder. Stakeholder sind Personengruppen, deren Erwartungen und Interessen bei der Entwicklung eines Produktes berücksichtig werden müssen. Sie sind von der Software gegenwärtig oder in Zukunft direkt oder indirekt betroffen. Man unterscheidet teilweise auch in interne (z.B. Management, Arbeitnehmer) und externe Stakeholder wie Banken oder Lieferanten.[[2]](#footnote-2)

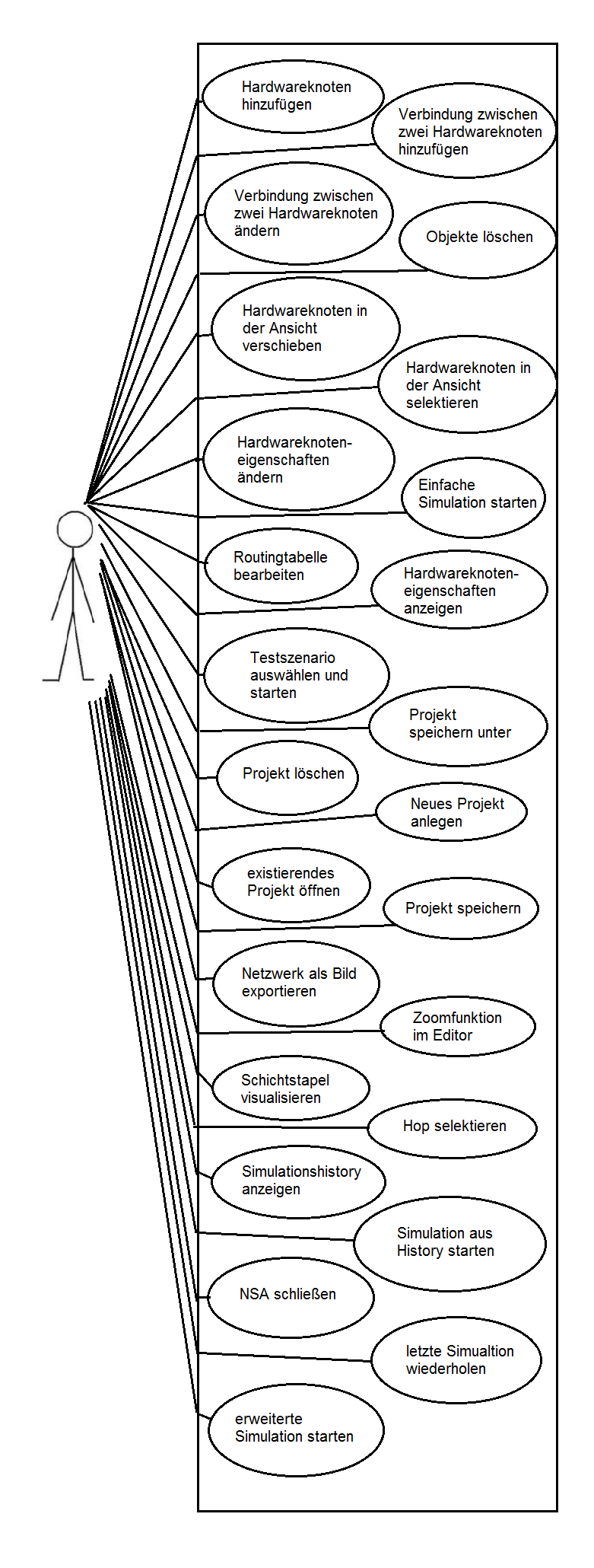
Da es sich hierbei um ein nichtkommerzielles Projekt handelt, gibt es keine externen Stakeholder. Bei den internen Stakeholdern handelt es sich um die Professoren Dr. Aßmuth und Dr. rer. nat. Hoffmann, die Praktikumsbetreuer in Form der Laboringenieure sowie um die Entwickler-Studenten und die Anwender-Studenten (Praktikumsteilnehmer).

Während des Praktikums müssen vorgegebene Übungsaufgaben gelöst werden. Die Praktikumsbetreuer erhalten dafür geeignete Testszenarien, die sie ohne großen Aufwand direkt an den einzelnen Projekten der Studenten durchspielen und so die Übungsaufgaben überprüfen und auswerten können.

Neben den fachlichen Anforderungen gibt es noch einige technische Rahmenbedingen die berücksichtigt werden müssen. Dazu gehört, dass die Software als Stand-Alone Anwendung laufen muss. Die Implementierung soll in Java oder C# erfolgen, da eine Unabhängigkeit von Microsoft-Windows zwingend gefordert wird.

# Funktionale Anforderungen

## Use-Case Übersicht



## Use-Beschreibungen

|  |  |  |
| --- | --- | --- |
| **Hardwareknoten hinzufügen** | | |
| **Kennung** | | UC-1 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
| Der Benutzer wählt einen Hardwareknoten aus, der anschließend auf der Designfläche platziert wird. | | |
| **Vorbedingung(en):** | | |
|  | | |
| **Nachbedingung(en):** | | |
| Ein neuer Hardwareknoten befindet sich auf der Designfläche. | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der Benutzer per Drag & Drop einen Hardwareknoten (Icon) auf die Designfläche zieht. 2. Der Hardwareknoten wird an der gewünschten Stelle platziert.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 1a | Benutzer bricht ab | |
|  | **Ende.** | |
| 1b | Hardwareknoten außerhalb der Designfläche platziert | |
|  | **Ende.** | |

|  |  |  |
| --- | --- | --- |
| **Verbindung zwischen zwei Hardwareknoten hinzufügen** | | |
| **Kennung** | | UC-2 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
| Der Benutzer wählt zwei Hardwareknoten aus, die verbunden werden sollen. Zwischen den gewählten Hardwareknoten wird eine Verbindungslinie gesetzt. | | |
| **Vorbedingung(en):** | | |
| Es sind mindestens zwei noch nicht verbundene Hardwareknoten vorhanden. | | |
| **Nachbedingung(en):** | | |
| Die zwei gewählten Hardwareknoten sind verbunden. | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der Benutzer das Icon für die Verbindungskabel anklickt. 2. Der Benutzer wählt einen Hardwareknoten aus. 3. Danach wählt er den Hardwareknoten aus, der mit dem ersten Hardwareknoten verbunden werden soll. 4. Das System setzt eine Verbindungslinie zwischen den beiden Hardwareknoten.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 3a | Bereits bestehende Verbindung zwischen den beiden Hardwareknoten | |
|  | 1. Der Benutzer wählt einen anderen Hardwareknoten aus.   **Rückkehr nach**: 3 | |

|  |  |  |
| --- | --- | --- |
| **Verbindung zwischen zwei Hardwareknoten ändern** | | |
| **Kennung** | | UC-3 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
| Der Benutzer verschiebt eine bestehende Verbindung. | | |
| **Vorbedingung(en):** | | |
|  | | |
| **Nachbedingung(en):** | | |
| Die Verbindungslinie ist an veränderter Position. | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der Benutzer eine bestehende Verbindung an einem Ende anklickt und die Maustaste gedrückt hält. 2. Der Benutzer verschiebt die Verbindung an diesem Ende. 3. Das System löst die Verbindung an dieser Stelle. 4. Der Benutzer wählt neuen Hardwareknoten aus. 5. Das System setzt die neue Verbindungslinie.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 3a | Bereits bestehende Verbindung zwischen den beiden Hardwareknoten | |
|  | 1. Der Benutzer wählt einen anderen Hardwareknoten aus.   **Rückkehr nach**: 3 | |

|  |  |  |
| --- | --- | --- |
| **Objekte löschen** | | |
| **Kennung** | | UC-4 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
| Der Benutzer löscht ausgewählte Objekte. | | |
| **Vorbedingung(en):** | | |
| Mindestens ein Hardwareknoten/eine Verbindung ist ausgewählt. | | |
| **Nachbedingung(en):** | | |
| Das/Die ausgewählte/n Objekt/e ist/sind gelöscht. | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der Benutzer ein markiertes Objekt löscht. 2. Das System entfernt die markierten Objekte aus der Designfläche.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 1a | Eingabe des Löschvorgangs mit Hilfe der rechten Maustaste | |
|  | 1. Der Benutzer klickt mit der rechten Maustaste auf die markierten Objekte. 2. Der Benutzer wählt die Option löschen aus.   **Rückkehr nach**: 1 | |
| 1b | Eingabe des Löschvorgangs mit Hilfe der Entf-Taste | |
|  | 1. Der Benutzer drückt die Entf-Taste.   **Rückkehr nach**: 1 | |

|  |  |  |
| --- | --- | --- |
| **Hardwareknoten in der Ansicht verschieben** | | |
| **Kennung** | | UC-5 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
| Der Benutzer verschiebt einen ausgewählten Hardwareknoten. | | |
| **Vorbedingung(en):** | | |
|  | | |
| **Nachbedingung(en):** | | |
| Der Hardwareknoten befindet sich an der neuen Stelle auf der Designfläche. | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der Benutzer einen Hardwareknoten anklickt und die linke Maustaste gedrückt hält. 2. Der Benutzer zieht mit gedrückter linker Maustaste den Hardwareknoten an einen neuen Platz. 3. Das System setzt den Hardwareknoten an den neuen Platz.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 2a | Hardwareknoten mit Verbindungen zu anderen Hardwareknoten | |
|  | 1. Das System verschiebt passend zu dem verschobenen Hardwareknoten auch seine Verbindungen zu anderen Hardwareknoten.   **Rückkehr nach**: 2 | |
| 2b | Hardwareknoten außerhalb der Designfläche platziert | |
|  | 1. Das System behält die ursprüngliche Position bei.   **Ende.** | |
| 2c | Benutzer bricht ab | |
|  | **Ende.** | |

|  |  |  |
| --- | --- | --- |
| **Hardwareknoten in der Ansicht selektieren** | | |
| **Kennung** | | UC-6 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
| Der Benutzer markiert einen Hardwareknoten. | | |
| **Vorbedingung(en):** | | |
| Auf der Designfläche befindet sich mindestens ein Hardwareknoten. | | |
| **Nachbedingung(en):** | | |
| Der Hardwareknoten ist selektiert. | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der Benutzer einen Hardwareknoten anklickt. 2. Das System markiert den Hardwareknoten.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 1a | Mehrere Hardwareknoten markieren mit Strg und linker Maustaste | |
|  | 1. Der Benutzer klickt einen Hardwareknoten an. 2. Danach drückt der Benutzer bei jedem weiteren Knoten noch dazu die Taste Strg.   **Rückkehr nach**: 1 | |
| 1b | Mehrere Hardwareknoten markieren mit linker Maustaste | |
|  | 1. Der Benutzer zieht mit gedrückter linker Maustaste ein Rechteck über den zu markierenden Hardwareknoten auf.   **Rückkehr nach**: 1 | |

|  |  |  |
| --- | --- | --- |
| **Hardwareknoteneigenschaften anzeigen** | | |
| **Kennung** | | UC-7 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
| Der Benutzer selektiert einen Hardwareknoten. Anschließend werden dem Nutzer die Eigenschaften des Knotens in Form einer Tabelle angezeigt | | |
| **Vorbedingung(en):** | | |
| Es existiert ein selektierbarer Hardwareknoten | | |
| **Nachbedingung(en):** | | |
| Die Eigenschaftentabelle des selektierten Knotens wird angezeigt. | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Benutzer selektiert einen Hardwareknoten. 2. Das System zeigt die Eigenschaften des Knotens an.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 2a | Die Eigenschaften werden nicht angezeigt. | |
|  | **Ende.** | |

|  |  |  |
| --- | --- | --- |
| **Hardwareknoteneigenschaften ändern** | | |
| **Kennung** | | UC-8 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
| Der Benutzer kann in der Eigenschaftentabelle eines selektierten Hardwareknotens, eine beliebige Eigenschaft auswählen und den Wert dieser Eigenschaft ändern. Beispiele für diese Eigenschaften wären zum Beispiel der Name des Knotens oder seine IP-Adresse. | | |
| **Vorbedingung(en):** | | |
| Es ist ein Hardwareknoten ausgewählt. | | |
| **Nachbedingung(en):** | | |
| Eine oder mehrere Eigenschaften wurden geändert. | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Benutzer selektiert in der Eigenschaftentabelle das Eingabefeld mit dem Wert der Eigenschaft die er ändern möchte. 2. Der Benutzer ändert den ausgewählten Wert 3. Das System speichert die Änderung.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 1a | Es handelt sich um die Routingtabelle | |
|  | 1. Es wird der Use-Case „Routingtabelle aufstellen/ändern“ ausgeführt.   **Rückkehr nach**: 3 | |

|  |  |  |
| --- | --- | --- |
| **Routingtabelle bearbeiten** | | |
| **Kennung** | | UC-9 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
| Der Benutzer fügt, ändert oder löscht Einträge aus der Routingtabelle. | | |
| **Vorbedingung(en):** | | |
| Der Benutzer hat die Routingtabelle in den Eigenschaften eines Hardwareknotens selektiert. | | |
| **Nachbedingung(en):** | | |
| Die Routingtabelle wurde geändert. | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Benutzer wählt ein Feld der Routingtabelle aus. 2. Der Benutzer kann nun den Wert der Spalte ändern.   Der Benutzer wiederholt Schritt 1 und 2 bis er mit den Änderungen fertig ist.   1. Das System prüft, ob alle Einträge vollständig sind.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 1a | Der Benutzer bricht ab. | |
|  | **Ende.** | |
| 2a | Die Werte sind leer. | |
|  | 1. Der Benutzer kann die Werte befüllen und somit einen neuen Eintrag anlegen.   **Rückkehr nach**: 2 | |
| 3a | Die Werte sind nicht vollständig. | |
|  | 1. Das System fordert den Nutzer auf, die Werte zu vervollständigen.   **Rückkehr nach**: 2 | |
| 3b | Die Werte sind falsch. | |
|  | 1. Das System fordert den Nutzer auf, die Werte zu korrigieren.   **Rückkehr nach**: 2 | |

|  |  |  |
| --- | --- | --- |
| **Projekt löschen** | | |
| **Kennung** | | UC-10 |
| **Priorität** | | niedrig |
| **Kurzbeschreibung:** | | |
| Das aktuelle Projekt wird gelöscht. | | |
| **Vorbedingung(en):** | | |
| Der Benutzer hat ein Projekt geöffnet | | |
| **Nachbedingung(en):** | | |
| Das Projekt und alle seine Dateien sind gelöscht worden. | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Benutzer möchte das Projekt löschen. 2. Das System fragt den Nutzer, ob er das Projekt wirklich löschen möchte. 3. Der Benutzer klickt einen Button an. 4. Das System schließt das Projekt und löscht alle seine Dateien.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 3a | Der Benutzer klickt „Ja“ an. | |
|  | **Rückkehr nach**: 1 | |
| 3b | Der Benutzer klickt „Nein“ an. | |
|  | **Ende.** | |

|  |  |  |
| --- | --- | --- |
| **Neues Projekt anlegen** | | |
| **Kennung** | | UC-11 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
| Das System legt ein neues Projekt an und öffnet dieses. | | |
| **Vorbedingung(en):** | | |
|  | | |
| **Nachbedingung(en):** | | |
| Es wurde ein neues Projekt erstellt und geöffnet. | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Benutzer möchte ein neues Projekt anlegen. 2. Das System prüft ob ein Projekt geöffnet ist. 3. Das System legt ein neues Projekt an und öffnet dieses.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 2a | Ein Projekt ist geöffnet. | |
|  | 1. Das System fragt den Nutzer ob er das aktuell geöffnete Projekt vor dem Schließen speichern möchte. 2. Das System schließt das Projekt.   **Rückkehr nach**: 2 | |
| 2a1a | Der Nutzer antwortet mit „Ja“ | |
|  | 1. Das Projekt wird gespeichert.   **Rückkehr nach**: 1a2 | |
| 2a1b | Der Nutzer antwortet mit „Nein“ | |
|  | **Rückkehr nach**: 1a2 | |
| 2a1c | Der Nutzer möchte abbrechen. | |
|  | **Ende.** | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Existierendes Projekt öffnen** | | | |
| **Kennung** | | | UC-12 |
| **Priorität** | | | hoch |
| **Kurzbeschreibung:** | | | |
| Das System schließt gegeben falls das aktuelle Projekt und öffnet ein Bestehendes. | | | |
| **Vorbedingung(en):** | | | |
|  | | | |
| **Nachbedingung(en):** | | | |
| Das Bestehende Projekt wurde geöffnet. | | | |
| **Normaler Ablauf:** | | | |
|  | 1. Der Benutzer möchte ein existierendes Projekt öffnen. 2. Das System prüft ob ein Projekt geöffnet ist. 3. Das System fragt den Nutzer, welches Projekt er öffnen möchte. 4. Der Benutzer wählt das zu öffnende Projekt aus. 5. Das System öffnet das ausgewählte Projekt.   **Ende**. | | |
| **Ablauf-Varianten:** | | | |
| 2a | Ein Projekt ist geöffnet. | | |
|  | 1. Das System fragt den Nutzer, ob er das aktuell geöffnete Projekt vor dem Schließen speichern möchte. 2. Der Benutzer wählt eine Option aus. 3. Das System schließt das Projekt.   **Rückkehr nach**: 2 | | |
| 2a1a | | Der Nutzer antwortet mit „Ja“ | |
|  | | 1. Das Projekt wird gespeichert.   **Rückkehr nach**: 1a2 | |
| 2a1b | | Der Nutzer antwortet mit „Nein“ | |
|  | | **Rückkehr nach**: 1a2 | |
| 2a1c | | Der Nutzer möchte abbrechen. | |
|  | | **Ende.** | |

|  |  |  |
| --- | --- | --- |
| **Projekt speichern** | | |
| **Kennung** | | UC-13 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
| Das aktuell geöffnete Projekt wird gespeichert. | | |
| **Vorbedingung(en):** | | |
| Es ist ein Projekt geöffnet. | | |
| **Nachbedingung(en):** | | |
| Das Projekt wurde gespeichert. | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Benutzer möchte das Projekt speichern. 2. Das System prüft, ob es bereits einen Speicherort für das Projekt gibt. 3. Das System speichert das Projekt an den gewünschten Speicherort.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 2a | Es gibt keinen Speicherort. | |
|  | 1. Das System fordert den Nutzer auf den Speicherort anzugeben. 2. Der Benutzer wählt den Speicherort aus.   **Rückkehr nach**: 2 | |
| 2a | Benutzer bricht ab. | |
|  | **Ende.** | |

|  |  |  |
| --- | --- | --- |
| **Projekt speichern unter** | | |
| **Kennung** | | UC-14 |
| **Priorität** | | mittel |
| **Kurzbeschreibung:** | | |
| Das System speichert das aktuell geöffnete Projekt an einen vom Nutzer angegebenen Speicherort. | | |
| **Vorbedingung(en):** | | |
| Ein Projekt ist geöffnet. | | |
| **Nachbedingung(en):** | | |
| Das Projekt wurde an den Ort gespeichert. | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Benutzer möchte das Projekt an einem bestimmten Speicherort speichern. 2. Das System fordert den Nutzer auf den gewünschten Speicherort anzugeben. 3. Der Benutzer wählt einen Speicherort aus. 4. Das System speichert das Projekt an den Ort.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 2a | Der Nutzer möchte abbrechen. | |
|  | **Ende.** | |

|  |  |  |
| --- | --- | --- |
| **Netzwerk als Bild exportieren** | | |
| **Kennung** | | UC-15 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
| Das System erstellt einen Screenshot des Netzwerkaufbaus und speichert diesen an einen vom Nutzer angegebenen Pfad. | | |
| **Vorbedingung(en):** | | |
| Es ist ein Projekt geöffnet. | | |
| **Nachbedingung(en):** | | |
| Der Aufbau des Netzwerkes wurde als Bild gespeichert. | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Benutzer möchte das Netzwerk als Bild exportieren. 2. Das System fordert den Nutzer auf den Speicherort anzugeben. 3. Der Benutzer wählt einen Speicherort aus. 4. Der Benutzer gibt den Dateinamen ein. 5. Das System überprüft, ob der Dateiname und der Speicherort gültig sind. 6. Das System erstellt einen Screenshot des Netzwerks und speichert diesen am gewünschten Ort ab.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 1a | Der Nutzer möchte abbrechen. | |
|  | **Ende.** | |
| 5a | Ungültiger Dateiname oder Speicherort | |
|  | 1. Das System fordert den Benutzer die Eingaben zu korrigieren.   **Rückkehr nach:** 3 | |
| 6a | Exportieren schlägt fehl | |
|  | 1. Das System meldet einen Fehler.   **Ende.** | |

|  |  |  |
| --- | --- | --- |
| **Zoomfunktion im Editor** | | |
| **Kennung** | | UC-16 |
| **Priorität** | | Niedrig |
| **Kurzbeschreibung:** | | |
| Der User kann die Ansicht eines erstellten oder geöffneten Projekts vergrößern und verkleinern. | | |
| **Vorbedingung(en):** | | |
| Es muss ein Projekt geöffnet sein, das visuelle Elemente enthält. Die Netzwerkvisualisierung muss sichtbar sein. | | |
| **Nachbedingung(en):** | | |
| Elemente in der Ansicht werden nun vergrößert bzw. verkleinert dargestellt. | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der User auf einen Button zum vergrößern/verkleinern der Ansicht klickt. 2. Die Ansicht wird vergrößert/verkleinert. Als Mitte für die Größenänderung wird die Mitte der aktuellen Ansicht der Netzwerkvisualisierung verwendet.   **Ende.** | |
| **Ablauf-Varianten:** | | |
| 1a. |  | |
|  | 1. Der Anwender fährt mit der Maus über die Netzwerkvisualisierung. 2. Der Anwender bewegt das Mausrad nach vorne/hinten. 3. Die Ansicht wird vergrößert/verkleinert. Als Mitte für die Größenänderung wird die Position der Maus verwendet.   **Ende.** | |

|  |  |  |
| --- | --- | --- |
| **Testszenario auswählen und starten** | | |
| **Kennung** | | UC-17 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Der Nutzer kann ein bestehendes Testszenario zu Netzwerkzusammenstellungen aus einer Liste auswählen und eine Simulation starten. | | |
| **Vorbedingung(en):** | | |
| Das zu testende Netzwerk muss geladen sein. | | |
| **Nachbedingung(en):** | | |
| Das Testszenario wurde ausgeführt. Die Simulation für das Testszenario wurde ausgeführt. Es wird angezeigt ob das Simulationsergebnis mit dem erwarteten Ergebnis übereinstimmt. | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der Nutzer die Liste der Testszenarien anzeigt. 2. Der Nutzer wählt ein Testszenario aus der Liste aus. 3. Der Nutzer startet das Testszenario. 4. Das Tool führt die Simulation des Testszenarios aus.   **Ende.** | |
| **Ablauf-Varianten:** | | |
| 1-4a |  | |
|  | 1. Der Nutzer bricht den Vorgang ab.   **Ende.** | |

|  |  |  |
| --- | --- | --- |
| **Einfache Simulation starten** | | |
| **Kennung** | | UC-18 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Der Anwender kann grundlegende Kommunikationsszenarien in einem Rechnernetz simulieren. Der Weg des Datenpakets wird visualisiert und kann so genau verfolgt werden. | | |
| **Vorbedingung(en):** | | |
| Es sind mindestens 2 Rechner und / oder Router vorhanden, die verbunden sind. | | |
| **Nachbedingung(en):** | | |
| Die Simulationsergebnisse werden angezeigt. | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Use-Case beginnt, wenn der Anwender das Kommando zum Starten der Simulation gibt. 2. Der Anwender legt im Editor fest, von welchem Startpunkt (Client, Rechner, Hardwareknoten?) aus das Datenpaket versendet werden sollen. 3. Der Anwender legt im Editor den Zielort des Datenpakets fest. 4. Das System speichert die Simulation in der History. 5. Das System führt die Simulation durch.   **Ende.** | |
| **Ablauf-Varianten:** | | |
| 2a | Der Anwender möchte abbrechen. | |
|  | **Ende.** | |
| 3a | Der Anwender möchte abbrechen. | |
|  | **Ende.** | |

|  |  |  |
| --- | --- | --- |
| **Schichtstapel visualisieren (Weg eines Datenpaketes durch alle Schichten)** | | |
| **Kennung** | | UC-19 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
| Das Programm zeigt für einen ausgewählten Hardwareknoten die Schichten an. Jede Schicht wird mit grün gekennzeichnet, sofern das Datenpaket ohne Probleme diese Schicht durchlaufen hat. Rot falls nicht. | | |
| **Vorbedingung(en):** | | |
| Ein Hardwareknoten ist ausgewählt und eine Simulation wurde durchgeführt. | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Use-Case beginnt, wenn der Anwender einen Hardwareknoten ausgewählt hat. 2. Das System zeigt alle Schichten des Schichtstapels an und kennzeichnet alle durchlaufenen mit grün, alle nicht erfolgreich durchlaufenen mit rot.   **Ende**. | |

|  |  |  |
| --- | --- | --- |
| **Hop selektieren** | | |
| **Kennung** | | UC-20 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Der Nutzer selektiert einen Hop des Datenpaketes in der Liste aller Hops. Das System zeigt Informationen zu diesem an. | | |
| **Vorbedingung(en):** | | |
| Es wurde eine Simulation durchgeführt. | | |
| **Nachbedingung(en):** | | |
| Informationen zum Hop werden angezeigt. | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Nutzer selektiert einen Hop in der Liste aller Hops der zuvor ausgeführten Simulation. 2. Das System zeigt alle relevanten Nodes, Schichten und Verbindung zum ausgewählten Hop an.   **Ende**. | |

|  |  |  |
| --- | --- | --- |
| **Simulationshistory anzeigen** | | |
| **Kennung** | | UC-21 |
| **Priorität** | | mittel |
| **Kurzbeschreibung:** | | |
| Das System zeigt alle Simulationen an, die während der aktuellen Sitzung gestartet wurden. | | |
| **Vorbedingung(en):** | | |
| Es ist ein Projekt geöffnet. | | |
| **Nachbedingung(en):** | | |
| Die Simulationshistory wird angezeigt. | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Use-Case beginnt, wenn der Nutzer die Simulationshistory anzeigen lassen möchte. 2. Das System zeigt die Simulationshistory an.   **Ende**. | |

|  |  |  |
| --- | --- | --- |
| **Simulation aus History starten** | | |
| **Kennung** | | UC-22 |
| **Priorität** | | mittel |
| **Kurzbeschreibung:** | | |
| Das System startet eine neue Simulation mit den gleichen Werten, wie die Simulation aus der History, die vom Nutzer ausgewählt wurde. | | |
| **Vorbedingung(en):** | | |
| Die Simulationshistory wird angezeigt. | | |
| **Nachbedingung(en):** | | |
| Eine neue Simulation wurde gestartet. | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Nutzer wählt eine Simulation aus der History aus, die er wiederholen möchte. 2. Das System führt eine neue Simulation, mit den gleichen Parametern wie die Ausgewählte, durch.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 1a | Der Nutzer möchte abbrechen. | |
|  | **Ende.** | |

|  |  |  |
| --- | --- | --- |
| **Letzte Simulation wiederholen** | | |
| **Kennung** | | UC-23 |
| **Priorität** | | Niedrig |
| **Kurzbeschreibung:** | | |
| Das System führt die letzte ausgeführte Simulation erneut durch. | | |
| **Vorbedingung(en):** | | |
| Es ist ein Projekt geöffnet und es wurde bereits eine Simulation durchgeführt. | | |
| **Nachbedingung(en):** | | |
| Es wurde die letzte Simulation wiederholt. | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Use-Case beginnt, wenn der Nutzer das Kommando zum Starten der letzten Simulation gibt. 2. Das System führt eine neue Simulation, mit den gleichen Parametern wie die letzte Simulation, durch.   **Ende**. | |

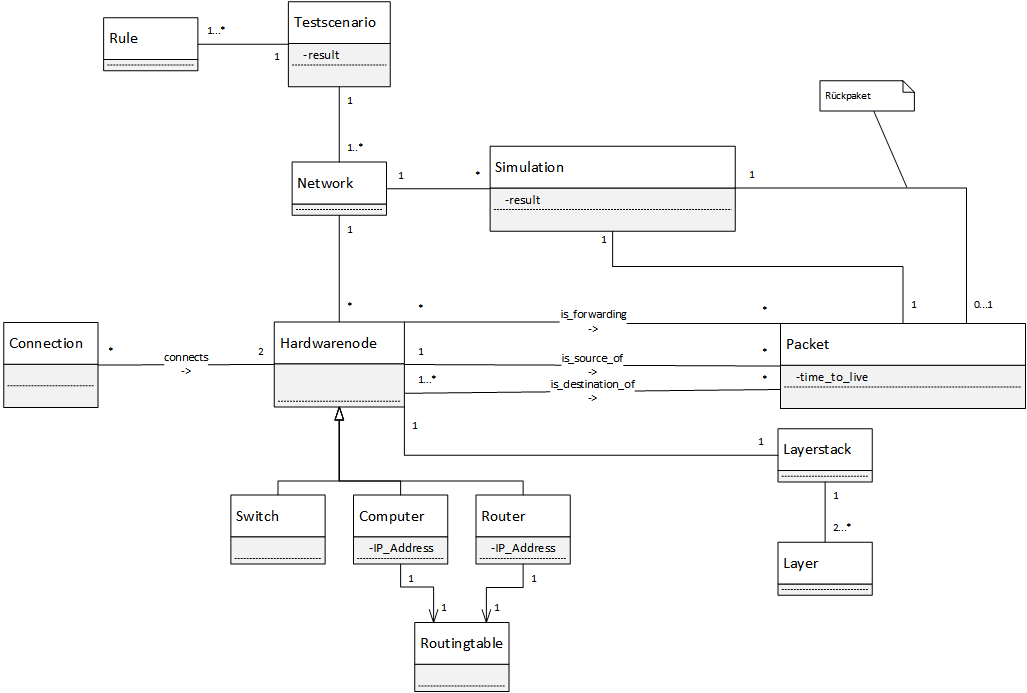
|  |  |  |
| --- | --- | --- |
| **Erweiterte Simulation starten** | | |
| **Kennung** | | UC-24 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Der Anwender kann grundlegende Kommunikationsszenarien in einem Rechnernetz simulieren. Der Weg des Datenpakets wird visualisiert und kann so genau verfolgt werden. | | |
| **Vorbedingung(en):** | | |
| Es sind mindestens 2 Rechner und / oder Router vorhanden, die verbunden sind. | | |
| **Nachbedingung(en):** | | |
| Die Simulationsergebnisse werden angezeigt. | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Use-Case beginnt, wenn der Anwender das Kommando zum Starten der erweiterten Simulation gibt. 2. Der Anwender legt in einem Dialogfenster fest, von welchem Startpunkt (Client, Rechner, Hardwareknoten?) und zu welchem Endpunkt das Datenpaket versendet werden soll und stellt die weiteren Parameter (TTL etc.) ein. 3. Das System führt die Simulation durch. 4. Das System speichert die Simulation in der History.   **Ende.** | |
| **Ablauf-Varianten:** | | |
| 2a | Der Anwender möchte abbrechen. | |
|  | **Ende.** | |
| 3a | Der Anwender möchte abbrechen. | |
|  | **Ende.** | |
| 3b | Die Simulation kann nicht durchgeführt werden. | |
|  | 1. System zeigt Fehler an.   **Ende.** | |
| 4a | Die Simulation kann nicht gespeichert werden. | |
|  | **Ende.** | |

|  |  |  |
| --- | --- | --- |
| **NSA schließen** | | |
| **Kennung** | | UC-25 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
| Das System schließt das Programm. | | |
| **Vorbedingung(en):** | | |
| Das Programm läuft. | | |
| **Nachbedingung(en):** | | |
| Das Programm wurde geschlossen. | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Use-Case beginnt, wenn der Anwender das Programm schließen möchte. 2. Das System fragt den Nutzer, ob er das offene Projekt speichern möchte. 3. Das System speichert das Projekt. 4. Das System beendet das Programm.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 1a | Der Nutzer möchte abbrechen. | |
|  | **Ende.** | |
| 2a | Es ist kein Projekt geöffnet. | |
|  | **Rückkehr nach: 4.** | |
| 2b | Der Nutzer möchte nicht speichern. | |
|  | **Rückkehr nach: 4.** | |

## (Sonstige) Funktionalität

|  |  |  |
| --- | --- | --- |
| **ID** | **Beschreibung** | **Querverweise** |
| FR-001 | Fehler und Exceptions sollen applikationsweit geloggt werden. |  |
| FR-002 | Einstellungen, die der Benutzer vorgenommen hat, sollen persistent sein, also automatisch gespeichert und beim nächsten Start der Applikation geladen werden |  |
| FR-003 | Simulierte Netzwerkprotokolle sollen im Sinne der einfachen Erweiterbarkeit Plugin-basiert eingebunden oder überschrieben/erweitert werden können |  |
| FR-004 | Es soll eine Hilfe-Funktion geben, mit der alle Benutzeroberflächenelemente bei MouseOver mit ausführlichen Tooltips erläutert werden |  |

## Modell des Problembereichs (Konzeptionelles Datenmodell)



# Nicht-Funktionale Anforderungen

## Benutzbarkeit (Usability)

|  |  |  |
| --- | --- | --- |
| **ID** | **Beschreibung** | **Querverweise** |
|  | **Installation/Deinstallation** |  |
| UR-01 | Die Anwendung soll in wenigen Schritten ohne einen Rechnerneustart installiert werden können. |  |
| UR-02 | Ohne Erlaubnis des Benutzers sollen keine Icons auf dem Desktop installiert werden. |  |
| UR-03 | Es soll nicht notwendig sein, vor Benutzung der Anwendung eine Readme-Datei zu lesen. |  |
| UR-04 | Außer den Dateien, die der Benutzer selber erzeugt hat, sollen nach der Deinstallation keine Dateien oder Einträge in der Registry erhalten bleiben. |  |
|  |  |  |
|  | **Arbeiten mit der Anwendung** |  |
| UR-05 | Dateien der Anwendung, die zum Programmlauf benötigt werden, sollen nur in den Installationsverzeichnissen oder in temporären Verzeichnissen abgelegt werden |  |
| UR-06 | Die Namen aller Dateien, die der Benutzer bei seiner Arbeit sieht, sollen lesbar sein. |  |
| UR-07 | Alle Funktionen und Aktionen sollen vom Benutzer ausgelöst werden und nicht von der Software selber. |  |
| UR-08 | Informationen, die verfügbaren Operationen und ihre Auswirkungen sollten so gut wie möglich visualisiert werden. |  |
| UR-09 | **Verständlichkeit:** Das System soll selbsterklärend sein, d.h. verwendete Symbole und Begriffe entsprechen der Wirklichkeit (Metaphorische Übertragung von Begriffen der realen Welt).   * Interface-Elemente wie Router = Bild von einem Router |  |
| UR-10 | Das System soll intuitiv bedienbar sein. |  |
| UR-11 | Das System soll bequem bedienbar sein. |  |
| UR-12 | **Erlernbarkeit:** Die Anwender sollen innerhalb von 1 Stunde den sicheren Umgang mit dem System erlernen können. |  |
| UR-13 | Die Darstellung der Symbole im System soll ausreichend groß sein.  (2cm => damit man es gut lesen kann.) |  |
| UR-14 | Das System soll so bedient werden können, dass innerhalb von 3 Stunden keine Ermüdungserscheinungen auftreten. |  |
| UR-15 | Die Sprache der Anwendung sollte eindeutig und verständlich sein (keine Umgangssprache!) |  |
| UR-16 | Die Anwendung sollte alleine mit Tastatur bedienbar sein |  |
| UR-17 | Die Anwendung sollte alle Standard-Direktzugriffe (Shortcuts) unterstützen, soweit es sinnvoll ist. |  |
| UR-18 | Gelegentliche Anwender können das System nach längerer Unterbrechung benutzen, ohne die Methoden erneut lernen zu müssen. |  |
| UR-19 | Der Anwender soll bei der Benutzung zufrieden sein. |  |
| UR-20 | Die Fehlerkorrektur innerhalb der Rechnernetzkonfiguration soll einfach sein. |  |
| UR-21 | Fehler bei der Rechnernetzkonfiguration haben keine schwerwiegenden Folgen. |  |
| UR-22 | Es gibt wenige Fehlermöglichkeiten bei der Benutzung des Systems. |  |
| UR-23 | Das System reagiert in angemessener Zeit auf Benutzereingaben (Geschwindigkeit). |  |
| UR-24 | Das System bzw. die erhaltenen Ergebnisse sind brauchbar und leicht zu interpretieren. |  |
| UR-25 | "Undo"(Rückgängig)/Redo(Wiederherstellen) soll für die meisten Aktionen verfügbar sein |  |
| UR-26 | Befehle, die nicht rückgängig gemacht werden können, sollen nach Bestätigung fragen => Warnung des Benutzers vor unerwünschten Folgen. |  |
| UR-27 | User interface soll für jeden Benutzer verständlich sein, unabhängig seines "Skill-Levels" |  |
| UR-28 | Alles in einem Mode(wenn man was editieren will, kein mode switch erforderlich, z.b. View-mode oder edit-mode) [Design- und Aktionsmodus] |  |
| UR-29 | Abbrechen einer bereits gestarteten Operation ist möglich. |  |
| UR-30 | Grundfunktionalität sollen von überall in der Anwendung nicht mehr als 3 Klicks entfernt sein. |  |
| UR-31 | Es soll kein horizontales Scrolling möglich sein. |  |
| UR-32 | In Simulationstabellen soll der Header beim Runterscrollen immer angezeigt werden. |  |
| UR-33 | Design und Farbe sollen ansprechend sein(Lesbarkeit, Farbzusammenspiel etc.) |  |
| UR-34 | Kernfaktoren(z.b. Funktion, UI, Userunterstützung, Zuverlässigkeit, ...) sollen mind 85% Zufriedenheit erreichen |  |
|  |  |  |
| UR-36 | Jede Aktion eines Benutzers soll mit einer Systemreaktion beantwortet werden. (Rückmeldung) |  |
| UR-37 | Der Benutzer ist zu einem bestimmten Zeitpunkt jeweils nur mit einem überschaubaren Umfang von Informationen konfrontiert |  |
| UR-38 |  |  |
| UR-39 |  |  |
| UR-40 |  |  |
|  |  |  |
|  | **zentrale Benutzereinstellungen** |  |
| UR-41 | Zentrale Farbeinstellungen (Eigenschaften der Anzeige) sollen übernommen werden |  |
| UR-42 | Das Erscheinungsbild der Anwendung wird in angemessener Form angepasst, wenn sich die Größe einzelner Fenster oder die Bildschirmauflösung verändert |  |
| UR-43 | System soll anpassbar sein (personal preferences) |  |
|  |  |  |
|  |  |  |
|  | **Hilfe** |  |
| UR-44 | Dokumentation und Hilfe sollen vollständig und hilfreich sein |  |
| UR-45 | Hilfe soll kontextsensitiv sein und erklären, wie man gewöhnliche Aufgaben erledigt |  |
| UR-46 | Fehlermeldungen sollen erklären, wie man den Fehler/Ursache behebt und wo man weitere Lösungsmöglichkeiten erhält. |  |
| UR-47 | Für mind. 95% aller Aktionen keine Hilfe nötig |  |
| UR-48 | Der Benutzer muss auf diese Zusatzinformationen ohne großen Aufwand zugreifen können (keine langen zu durchlaufenden Hierarchien) |  |

## Zuverlässigkeit (Reliability)

|  |  |  |
| --- | --- | --- |
| **ID** | **Beschreibung** | **Querverweise** |
| RR-01 | Das System soll bei Stromausfall / Absturz die Daten zum letzten Speicherzeitpunkt wieder herstellen können. |  |
| RR-02 | Robustheit: Fähigkeit eines Systems auch bei falschen Eingabe korrekt zu Arbeiten  => Das System soll bei falschen Eingaben die Fehler erkennen und darauf hinweisen. Das System gibt Lösungsvorschläge. |  |

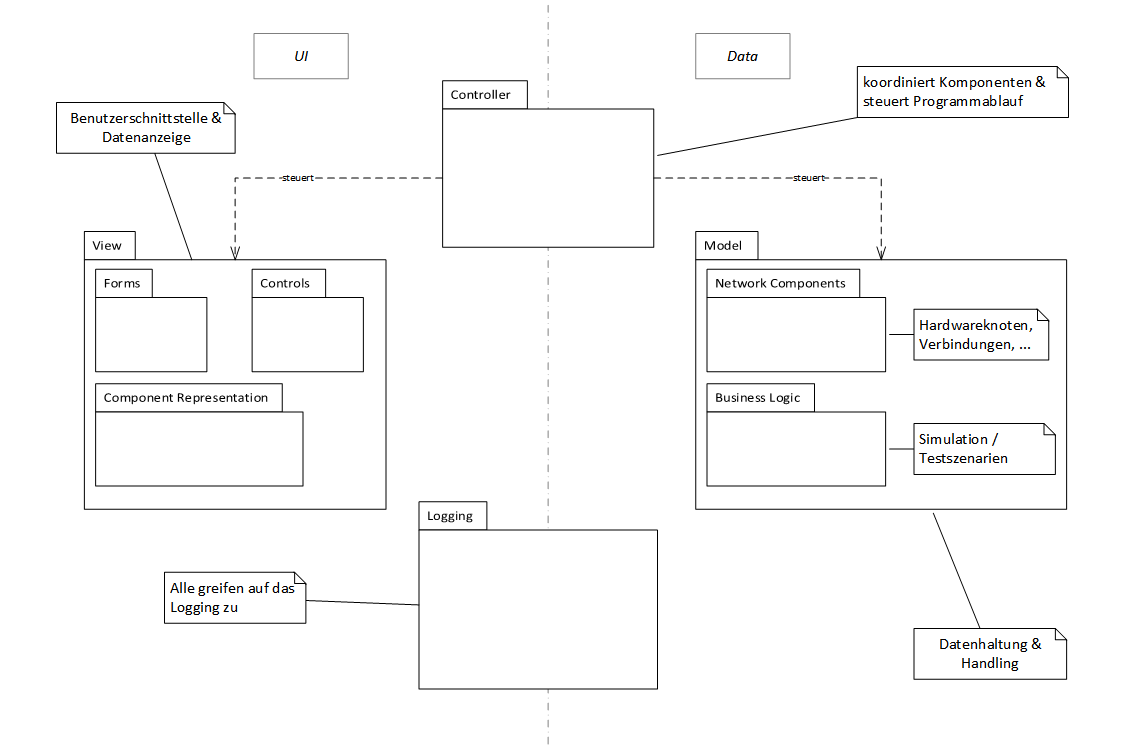
## Leistung (Performance)

|  |  |  |
| --- | --- | --- |
| **ID** | **Beschreibung** | **Querverweise** |
| PR-01 | Das System soll in einer angemessenen Zeit auf Benutzereingaben antworten; nicht länger als 5 Sekunden pro Aktion. |  |
| PR-02 | Betriebsmittel werden nur belegt wenn aktiv durch einen Studenten mit dem System gearbeitet wird. Andernfalls können die Betriebsmittel (Computer) für andere Aufgaben verwendet werden. |  |
| PR-03 | Keine zu großen Schwankungen in der Reaktionszeit |  |
| PR-04 | Fehlerrate definieren, wenn Antwort länger als nötig(Bei Auslastung z.b.) |  |
| PR-05 | Workload festlegen(max. Anzahl an Verbindungen, wie schnell Gesamtzeit?) |  |
| PR-06 | Gleichbleibende Leistung unabhängig von Plattform(Linux, Windows, etc.) |  |

## Unterstützbarkeit (Supportability)

|  |  |  |
| --- | --- | --- |
| **ID** | **Beschreibung** | **Querverweise** |
| SR-01 | Das System soll als Stand-Alone-Anwendung (auf allen Rechner des Labors Netze und Systeme) lauffähig sein. |  |
| SR-02 | Das System soll unanhängig von Microsoft-Windows sein. |  |
| SR-03 | Das System ist wartungsfreundlich. |  |
| SR-04 | Scalability (Erweiterbarkeit) festlegen, hängt mit Workload zusammen |  |
| SR-05 | siehe usability oben (Dokumentationen, Hilfen, etc.) |  |
| SR-06 | Hinzufügen/Erweitern des Codes möglich (von jedem oder nur mit bestimmtem Wissen) |  |

# Skizze der Gesamtsystemarchitektur



# Lieferumfang

Die folgende Tabelle enthält alle Arbeitsergebnisse, die in der Veranstaltung „Software-Projekte“ zu dem vom Team zu liefernden „End-Produkt“ gehören – für die individuell von jedem Projektteilnehmer zu liefernden Ergebnisse lesen Sie bitte im Projektleitfaden bzw. im Projektkalender nach. Die Benotung erfolgt nicht nur auf Grundlage des lauffähigen Programms, sondern bezieht die Qualität der Analyse, des Entwurfs und des Systemtests mit ein.

|  |  |  |  |
| --- | --- | --- | --- |
| **Lfd. Nr.** | **Was?** | **Art des Dokuments** | **Bemerkungen** |
| **Ergebnis der System-Analyse** | | | |
| 1 | Das Dokument „Anforderungen(XYZ)“ (also dieses Dokument) mit funktionalen, nicht-funktionalen Anforderungen und konzeptionellem Datenmodell. | * Siehe Vorlage. * Wird bei Projektbeginn mit einer Überblick gebenden Systembeschreibung an das Team ausgegeben. Das Dokument ist vom Team weiterzuführen und wieder abzugeben. | * Bitte auf Abgabetermin während des Semesters achten (s. Projektkalender). * Rechtzeitig vor Abgabe auf Qualitätssicherung achten (Review) |
| **Dokumentation des Systementwurfs** | | | |
| 2 | Das Dokument „Systementwurf(XYZ)“. | * Siehe Vorlage. | * Bitte auf Abgabetermin während des Semesters achten (s. Projektkalender) * Rechtzeitig vor Abgabe auf Qualitätssicherung achten (Review) |
| **Implementierung** | | | |
| 3 | Lauffähiger und getesteter Quellcode |  | Abgabe am Semesterende |
| **Test** | | | |
| 4 | Testspezifikation Systemtest | * Siehe Vorlage | * Endgültige Abgabe am Semesterende; zur Vorbereitung des Abnahmetests ist die Aufstellung der in den Abnahmetest einbezogenen Testfälle früher vorzulegen (Termin im Projektkalender) |
| 5 | Testprotokoll Systemtest | * Siehe Vorlage | * Abgabe am Semesterende |

# Abnahmekriterien

In der Veranstaltung „Software-Projekte“ werden vom „Auftraggeber“ (in Absprache mit den Teilnehmern) rechtzeitig vor Semesterende Systemtestfälle ausgewählt, die das System dann am Tag der Abnahme ohne Beanstandung „überstehen“ muss.

# Abkürzungsverzeichnis

|  |  |
| --- | --- |
| Abkürzung | Erklärung |
| NSA | Network-Simulator-and-Analyzer |

# Literaturverzeichnis

**[Lar]** Larman Craig, *Applying UML And Patterns. An Introduction to Object-Oriented Analysis And Design,* Prentice Hall, 2nd ed., 2002

# Abbildungsverzeichnis

1. V-Modell® ist eine geschützte Marke der Bundesrepublik Deutschland. [↑](#footnote-ref-1)
2. <http://welt-der-bwl.de/Stakeholder>, <http://wirtschaftslexikon.gabler.de/Definition/anspruchsgruppen.html> [↑](#footnote-ref-2)