HINWEIS: Blauer Text kann gelöscht werden, beziehungsweise soll ersetzt werden

- Systemanalyse -

…hier Name des Systems eintragen…

Version: …eintragen…

|  |  |  |
| --- | --- | --- |
| Projektbezeichnung | PRODUKT.PROJEKTBEZEICHNUNG | |
| Projektleiter |  | |
| Verantwortlich | Anforderungsanalytiker (AG) | |
| Erstellt am |  | |
| Zuletzt geändert | 31.03.2022 10:59 | |
| Bearbeitungszustand | X | in Bearbeitung |
|  | vorgelegt |
|  | fertig gestellt |
| Dokumentablage | C:\V-Modell-Editor\eclipse\workspace\weiter\Teilprojekte\TP14\templates\MasterTemplate.doc | |

Änderungsverzeichnis

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Änderung | | | Geänderte Kapitel | Beschreibung der Änderung | Autor | Zustand |
| Nr. | Datum | Version |
| 1 |  | 1.1 | Alle | Initiale Produkterstellung |  |  |

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](#_Toc34239361)

[2 Ausgangssituation und Zielsetzung 4](#_Toc34239362)

[3 Funktionale Anforderungen 5](#_Toc34239363)

[3.1 Use-Case Übersicht 5](#_Toc34239364)

[3.2 Use-Beschreibungen 5](#_Toc34239365)

[3.3 (Sonstige) Funktionalität 7](#_Toc34239366)

[3.4 Modell des Problembereichs (Konzeptionelles Datenmodell) 7](#_Toc34239367)

[4 Nicht-Funktionale Anforderungen 7](#_Toc34239368)

[4.1 Benutzbarkeit (Usability) 7](#_Toc34239369)

[4.2 Zuverlässigkeit (Reliability) 8](#_Toc34239370)

[4.3 Leistung (Performance) 8](#_Toc34239371)

[4.4 Unterstützbarkeit (Supportability) 9](#_Toc34239372)

[4.5 Sonstige Einschränkungen 9](#_Toc34239373)

[5 Risikoakzeptanz 10](#_Toc34239374)

[6 Skizze der Gesamtsystemarchitektur 10](#_Toc34239375)

[7 Lieferumfang 11](#_Toc34239376)

[8 Abnahmekriterien 11](#_Toc34239377)

[9 Glossar 11](#_Toc34239378)

[10 Abkürzungsverzeichnis 12](#_Toc34239379)

[11 Literaturverzeichnis 12](#_Toc34239380)

[12 Abbildungsverzeichnis 12](#_Toc34239381)

# 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 zur Verwendung für die Veranstaltung **„Software-Projekte“** in Informatik-Curricula 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, was ungefähr dem Thema „Ausgangssituation und Zielsetzung“ in diesem Dokument entspricht; die Anforderungen müssen die Teilnehmer dann in enger Abstimmung mit ihrem „Auftraggeber“ selbst erarbeiten und in diesem Dokument niederlegen. Dadurch sollen sie Gelegenheit erhalten, auch Tätigkeiten der System-Analyse intensiver zu üben. Die „Auftraggeberseite“ liefert also nicht – wie im V-Modell-XT vorgesehen - das komplette Lastenheft, aus dem die „Auftragnehmerseite“ ein separates Pflichtenheft ableitet; stattdessen wird das hier vorliegende Dokument vom studentischen Entwicklerteam zur Dokumentation der Analyse-Ergebnisse erstellt und zugleich als Ersatz für die im V-Modell-XT vorgesehenen Dokumente Lasten- und Pflichtenheft verwendet.

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

In diesem Thema werden die Ausgangssituation und der Anlass zur Durchführung des Projekts anschaulich dargestellt. Es wird beschrieben, welche Defizite bzw. Probleme existierender Systeme oder auch der aktuellen Situation zur Entscheidung geführt haben, das Projekt durchzuführen, und welche Vorteile durch den Einsatz des neuen Systems erwartet werden.

Es werden zusätzlich alle relevanten Stakeholder des Projekts benannt und die technische und fachliche Einbettung des zu entwickelnden Systems in seine Umgebung skizziert. Zusätzlich werden erste Rahmenbedingungen für die Entwicklung identifiziert und beschrieben. Rahmenbedingungen können beispielsweise technische Vorgaben oder Vorgaben zur Sicherheit sein.

…hier Ihren Text einfügen…

# Funktionale Anforderungen

Funktionale Anforderungen beschreiben die Fähigkeiten eines Systems, die ein Anwender erwartet, um mit Hilfe des Systems ein fachliches Problem zu lösen. Die Anforderungen werden aus den zu unterstützenden Geschäftsprozessen und den Ablaufbeschreibungen zur Nutzung des Systems abgeleitet.

Die Beschreibung der funktionalen Anforderungen erfolgt beispielsweise in Form von Anwendungsfällen (Use Cases). Ein Anwendungsfall beschreibt dabei einen konkreten, fachlich in sich geschlossenen Teilvorgang. Die Gesamtheit der Anwendungsfälle definiert das Systemverhalten. Ein Anwendungsfall kann in einfachem Textformat beschrieben werden, häufig stehen jedoch organisationsspezifische Muster zur Beschreibung zur Verfügung. Für datenzentrierte Systeme wird im Rahmen der funktionalen Anforderungen ein erstes fachliches Datenmodell erstellt, das als Grundlage des späteren Datenbankentwurfs dient. Das fachliche Datenmodell des Systems wird aus den Entitäten des Domänenmodells abgeleitet.

Die funktionalen Anforderungen sind die zentralen Vorgaben für die Systementwicklung.

## Use-Case Übersicht

…hier Use-Case-Diagramm(e) einfügen…

## Use-Beschreibungen

Beispiel:

|  |  |  |
| --- | --- | --- |
| **Warenkorb abrechnen** | | |
| **Kennung** | | UC-1 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
|  | | |
| **Vorbedingung(en):** | | |
|  | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der Kassierer den nächsten an der Kasse wartenden Kunden bedient. 2. Der Kassierer gibt die Kennung des ersten bzw. nächsten Artikels ein. 3. Das System registriert den Artikel …. 4. …. 5. Der Kunde bezahlt. 6. ….   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 2a | Ungültige Kennung | |
|  | 1. Das System signalisiert, dass die Kennung ungültig ist   **Rückkehr nach**: 2 | |
| 2b | Mehrere Artikel der gleichen Artikelkategorie sollen erfasst werden | |
|  | 1. Der Kassierer gibt die Stückzahl des nächsten zu erfassenden Artikels ein   **Rückkehr nach**: 2 | |
| 5a | Der Kunde kann nicht bezahlen | |
|  | 1. Der Kassierer storniert die Abrechnung.   **Ende.** | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Eingabewort eintragen** | | |
| **Kennung** | | UC-1 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Der Benutzer trägt ein Eingabewort in das dafür vorgesehene Eingabefeld auf der Oberfläche ein. Das System überprüft ob alle Symbole des Eingabeworts im Eingabealphabet enthalten sind. | | |
| **Vorbedingung(en):** | | |
| * Es liegt eine vollständig definierte Turingmaschine vor * Die Simulation ist beendet | | |
| **Nachbedingung(en):** | | |
| * Auf dem Band der Turingmaschine befindet sich ein gültiges Eingabewort | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der Benutzer in der Anwendung ein Eingabewort eingibt. 2. …   **Ende**. | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Simulation durchlaufen/starten** | | |
| **Kennung** | | UC-2 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Der Benutzer gibt dem System den Startbefehl. Das System durchläuft die Simulation der gültigen Definition. Das System signalisiert ob das Eingabewort zu einem akzeptierenden Endzustand geführt hat. | | |
| **Vorbedingung(en):** | | |
| * Die Turingmaschine befindet sich in einem gültigen Zustand * Die Definition der Turingmaschine ist gültig * Auf dem Band der Turingmaschine befindet sich ein gültiges Eingabewort | | |
| **Nachbedingung(en):** | | |
| * Die Definition der Turingmaschine befindet sich im gleichen Zustand wie in der Vorbedingung | | |
| **Normaler Ablauf:** | | |
|  | 1. … 2. …   **Ende**. | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Definition der Turingmaschine als Tabelle** | | |
| **Kennung** | | UC-3 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Der Benutzer definiert alle Zustände, Zustandsübergänge sowie Anfangszustand und die akzeptierenden Zustände in einer Tabelle. Es kann dabei entweder eine neue Tabelle erstellt oder eine bestehende bearbeitet werden. Das System ermittelt aus der Tabelle das Bandalphabet und das Eingabealphabet. | | |
| **Vorbedingung(en):** | | |
| * Die Simulation ist beendet | | |
| **Nachbedingung(en):** | | |
| * Die Definition der Turingmaschine ist gültig | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Benutzer wählt die Tabellenansicht aus 2. Das System zeigt eine Spalte, die bereits als Anfangszustand eingestellt ist 3. Der Benutzer definiert einen Zustand und die zugehörigen Zustandsübergänge durch Texteingabe 4. Nach jeder fertigen Zeile erneuert das System das Modell der Turingmaschine 5. Wiederhole 3 bis die Maschine fertig definiert ist. 6. Das System ermittelt aus der Tabelle das Bandalphabet und das Eingabealphabet.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 3a | Benutzereingabe ist invalide | |
|  | 1. Programm weist den Benutzer auf das Problem hin   **Rückkehr nach**: 3 | |
| 3b | Der Benutzer redefiniert einen Anfangszustand | |
|  | **Weiter bei**: 4 | |
| 3c | Der Benutzer definiert akzeptierende Zustände in der Tabelle | |
|  | **Weiter bei**: 4 | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Definition der Turingmaschine als Zustandsdiagramm** | | |
| **Kennung** | | UC-4 |
| **Priorität** | | Mittel |
| **Kurzbeschreibung:** | | |
| Der Benutzer definiert alle Zustände, Zustandsübergänge sowie Anfangszustand und die akzeptierenden Zustände in einem Zustandsdiagramm. Es kann dabei entweder ein neues Zustandsdiagramm erstellt oder ein bestehendes bearbeitet werden. Das System ermittelt aus dem Zustandsdiagramm das Bandalphabet und das Eingabealphabet. | | |
| **Vorbedingung(en):** | | |
| * Die Simulation ist beendet | | |
| **Nachbedingung(en):** | | |
| * Die Definition der Turingmaschine ist gültig | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Benutzer wählt die Diagrammansicht aus 2. Das System zeigt einen Knoten, der bereits als Anfangszustand eingestellt ist 3. Der Benutzer definiert einen Zustand und die zugehörigen Zustandsübergänge mit Mausklicks 4. Nach jeder fertigen Interaktion erneuert das System das Modell der Turingmaschine 5. Wiederhole 3 bis die Maschine fertig definiert ist. 6. Das System ermittelt aus dem Diagramm das Bandalphabet und das Eingabealphabet.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 3a | Benutzereingabe ist invalide | |
|  | 1. Programm weist den Benutzer auf das Problem hin   **Rückkehr nach**: 3 | |
| 3b | Der Benutzer redefiniert einen Anfangszustand | |
|  | **Weiter bei**: 4 | |
| 3c | Der Benutzer definiert akzeptierende Zustände im Diagramm | |
|  | **Weiter bei**: 4 | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Aktuellen Zustandsübergang hervorheben** | | |
| **Kennung** | | UC-5 |
| **Priorität** | | Mittel |
| **Kurzbeschreibung:** | | |
| Das System hebt dem Benutzer in der Tabelle oder dem Zustandsdiagramm den aktuellen Zustandsübergang hervor. | | |
| **Vorbedingung(en):** | | |
| * Die Option Tabelle anzeigen oder Zustandsdiagramm anzeigen wurde ausgewählt | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. … 2. …   **Ende**. | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
| Auch beide Ansichten gleichzeitig / keine möglich??? | | |

|  |  |  |
| --- | --- | --- |
| **Definition importieren (.txt)** | | |
| **Kennung** | | UC-6 |
| **Priorität** | | Mittel |
| **Kurzbeschreibung:** | | |
| Der Benutzer lädt eine Definition für eine Turingmaschine in das System. Das System prüft, ob diese Definition gültig ist. | | |
| **Vorbedingung(en):** | | |
| * Die Simulation ist beendet | | |
| **Nachbedingung(en):** | | |
| * Die Definition der Turingmaschine ist gültig | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Anwendungsfall beginnt, wenn der Benutzer auf die Schaltfläche „Import“ klickt  2. Es erscheint ein Dateiauswahldialog, in dem .txt Dateien gezeigt werden  3. Der Benutzer wählt die Datei aus  4. Das System überprüft, ob die Datei eine syntaktisch valide Definition einer Turingmaschine darstellt  5. Das System erstellt aus der Definition der Datei eine Turingmaschine  **Ende**. | |
| **Ablauf-Varianten:** | | |
| 4a | Datei enthält ungültige Definition | |
|  | 1. Das System erstellt keine Maschine  2. Das System fordert den Benutzer auf, eine andere Datei zu wählen  3. Die Variante endet  **Rückkehr nach**: 2 | |
| 3a | Der Benutzer beendet den Auswahldialog | |
|  | **Der Use-Case endet.** | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Definition exportieren (.txt)** | | |
| **Kennung** | | UC-7 |
| **Priorität** | | Mittel |
| **Kurzbeschreibung:** | | |
| Der Benutzer gibt den Befehl die aktuelle Definition der Turingmaschine zu exportieren. Das System speichert diese in menschenlesbarer Form als Textdatei an dem gewünschten Ablageort. | | |
| **Vorbedingung(en):** | | |
| 1. Die Definition der Turingmaschine ist gültig | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Anwendungsfall beginnt, wenn der Benutzer auf die Schaltfläche „Export“ klickt  2. Es erscheint ein Dateiauswahldialog, in dem alle Dateien gezeigt werden  3. Der Benutzer benennt seine Datei und speichert diese  **Ende**. | |
| **Ablauf-Varianten:** | | |
| 3a | Speichervorgang kann nicht ausgeführt werden (z.B. Nicht genügend Speicher vorhanden, keine Berechtigung) | |
|  | 1. Speicherdialog verweigert verbos das Speichern   **Rückkehr nach**: 2, der Dialog bleibt bestehen | |
| 3b | Der Benutzer beendet den Auswahldialog | |
|  | **Der Use-Case endet.** | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Simulation steuern** | | |
| **Kennung** | | UC-8 |
| **Priorität** | | Niedrig |
| **Kurzbeschreibung:** | | |
| Dem Benutzer stehen zur Steuerung der Simulation verschiedene Steuerungsmöglichkeiten zur Verfügung. Das System reagiert auf die entsprechende Auswahl. | | |
| **Vorbedingung(en):** | | |
| 1. Die Turingmaschine befindet sich in einem gültigen Zustand 2. Die Definition der Turingmaschine ist gültig | | |
| **Nachbedingung(en):** | | |
| 1. Die Turingmaschine befindet sich in einem gültigen Zustand 2. Die Definition der Turingmaschine ist gültig | | |
| **Normaler Ablauf:** | | |
|  | 1. … + Tabelle oder Diagramm wird angezeigt 2. …   **Ende**. | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
|  | | |
| **Kennung** | |  |
| **Priorität** | |  |
| **Kurzbeschreibung:** | | |
|  | | |
| **Vorbedingung(en):** | | |
|  | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. … 2. …   **Ende**. | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
|  | | |
| **Kennung** | |  |
| **Priorität** | |  |
| **Kurzbeschreibung:** | | |
|  | | |
| **Vorbedingung(en):** | | |
|  | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. … 2. …   **Ende**. | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
|  |  | |
|  | 1. … 2. …   **Ende.** | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

## (Sonstige) Funktionalität

Hier werden funktionale Anforderungen erfasst, die sich nicht vernünftig durch einen einzelnen Use-Case beschreiben lassen, weil sie z.B. „Querschnitt“-Funktionalität betreffen, die sich über viele Use-Cases erstreckt. Beispiele könnten sein:

|  |  |  |
| --- | --- | --- |
| **ID** | **Beschreibung** | **Querverweise** |
| FR-001 | Das System soll jede Fehlersituation dauerhaft protokollieren |  |
| FR-002 | Nutzung des Systems ist generell nur nach vorheriger Authentifizierung möglich |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## Modell des Problembereichs (Konzeptionelles Datenmodell)

Hier wird ein Modell des Problembereichs („konzeptuelles Datenmodell“) in Form eines oder mehrerer UML-Klassendiagramme eingefügt. Das konzeptuelle Datenmodell wird oft nicht streng getrennt von den Anforderungen erarbeitet, weil ein genaues Verständnis z.B. der Use-Cases nicht ohne Verständnis der „Konzepte“ aus dem Problembereich möglich ist. Aus diesem Grund wird das konzeptionelle Modell an dieser Stelle zusammen mit den Use-Cases dokumentiert.

# Nicht-Funktionale Anforderungen

Nicht-funktionale Anforderungen beschreiben Anforderungen an das System, die nicht-fachlicher Natur sind, jedoch entscheidend zur Anwendbarkeit des Systems beitragen. Sie definieren beispielsweise Qualitätsanforderungen, Sicherheitsanforderungen oder Performanceanforderungen.

Nicht-funktionale Anforderungen definieren grundlegende Eigenschaften eines Systems, die im Architekturentwurf berücksichtigt werden müssen. Sie können zur Abschätzung der Entwicklungskosten herangezogen werden und sollten, soweit möglich, messbar beschrieben sein.

Zur einfachen Strukturierung der Anforderungen werden diejenigen Anforderungen, die nicht eindeutig zu den funktionalen Anforderungen gehören, den nicht-funktionalen Anforderungen zugeordnet.

Die hier verwendete Einteilung unterscheidet verschiedene Arten von Anforderungen nach dem „FURPS“-Schema (**F**unctionality, **U**sability, **R**eliability, **P**reformance, **S**upportability), das auf Hewlett-Packard zurückgeht. Das FURPS-Schema ist hier noch um die Kategorie „Sonstige Einschränkungen“ erweitert. Anforderungen zur Funktionalität sind bereits im Kapitel 3 dokumentiert, in diesem Kapitel folgen lediglich alle restlichen Anforderungen.

Bei Bedarf kann dieses Schema zur Einteilung der Anforderungen auch durch ein anderes Schema (z.B. nach DIN ISO) ersetzt werden – wichtig ist nur, dass bei der Erfassung der Anforderungen überhaupt ein erprobtes Schema verwendet wird. Dies soll eine bessere Übersicht bieten und dazu beitragen, dass keine Anforderungen vergessen werden.

## Benutzbarkeit (Usability)

Hier werden Anforderungen erfasst, die die Benutzbarkeit („usability“ = Benutzbarkeit / Benutzerfreundlichkeit / Gebrauchstauglichkeit) des Systems betreffen. Hierzu zählen insbesondere Anforderungen zur (Software/Hardware)-Ergonomie („human factors“).

|  |  |  |
| --- | --- | --- |
| **ID** | **Beschreibung** | **Querverweise** |
| UR-001 | Das System soll… |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## Zuverlässigkeit (Reliability)

Hier werden Anforderungen erfasst, die die Zuverlässigkeit („reliability“ = Zuverlässigkeit) des Systems betreffen. Hierunter fallen insbesondere Anforderungen an die Wiederherstellbarkeit („recoverability“) und die Verfügbarkeit („availabilty“) des Systems. Die Wiederherstellbarkeit betrifft die Fähigkeit, bei Ausfall oder Störung das Leistungsniveau wieder zu erreichen und betroffene Daten wieder zu gewinnen. Eine Kenngröße in diesem Zusammenhang ist MTTR („**m**ean **t**ime **t**o **r**epair“), die die mittlere Zeit bis zur Wiederinbetriebnahme des Systems bei einem Ausfall angibt. Die Verfügbarkeit kann als Verhältnis zwischen der Zeit, in der das System funktionsfähig ist, und der Gesamtzeit angegeben werden:

V=MTBF / (MTBF+MTTR)

MTTR: mean time to repair (s.o.)

MTBF: **m**ean **t**ime **b**etween **f**ailures (also die mittlere Zeit zwischen zwei Ausfällen)

|  |  |  |
| --- | --- | --- |
| **ID** | **Beschreibung** | **Querverweise** |
| RR-001 | Das System soll… |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## Leistung (Performance)

Hierunter fallen Anforderungen an die Leistung („performance“) des Systems. Die Anforderungen beziehen sich insbesondere auf das Zeitverhalten (Ausführungsgeschwindigkeit, Antwortzeiten, Durchsatz) sowie auf das Verbrauchsverhalten (Anzahl der belegten Betriebsmittel und Dauer der Betriebsmittelbelegung).

|  |  |  |
| --- | --- | --- |
| **ID** | **Beschreibung** | **Querverweise** |
| PR-001 | Das System soll… |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## Unterstützbarkeit (Supportability)

Hierunter fallen Anforderungen, die Bereiche wie Anpassbarkeit („adaptability“), Testbarkeit („testability“), Wartbarkeit („maintainability“), Erweiterbarkeit („extensibility“), Lokalisierbarkeit („localizability“ = Anpassbarkeit an verschiedene Sprach- und Kulturräume) betreffen.

|  |  |  |
| --- | --- | --- |
| **ID** | **Beschreibung** | **Querverweise** |
| SR-001 | Das System soll… |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## Sonstige Einschränkungen

In manchen Fällen können von vorneherein Einschränkungen („constraints“) für Entwurf, Implementierung, Schnittstellen und Hardware des geplanten Systems bestehen, die ebenfalls als Anforderungen zu berücksichtigen sind und das bisherige „FURPS“-Schema zu „FURPS+“ (vgl. [Lar], S. 88) erweitern.

### Schnittstellen

### Implementierung

### Entwurf

# Risikoakzeptanz

Für sicherheitskritische Systeme werden in diesem Thema Vorgaben für die Behandlung der Systemsicherheit festgelegt. Es wird aufgezeigt, welche Risiken im Rahmen des Systembetriebs bestehen, welche Schäden, oder auch welche Klassen von Schäden, mit welcher Wahrscheinlichkeit auftreten können und inwieweit das Eintreten eines Schadensfalls toleriert wird bzw. nicht mehr akzeptabel ist.

Die Risikoakzeptanz für die identifizierten möglichen Schadensfälle wird beispielsweise in Form einer Risikoakzeptanzmatrix dokumentiert. Die Matrix ist eine Vorgabe des Auftraggebers, in der er festlegt, bei welcher Schadensklasse und welcher Eintrittswahrscheinlichkeit er welche Risikoklasse akzeptiert.

Bei vielen Aufgabenstellungen in der Veranstaltung „Software-Projekte“ kann dieses Thema weggelassen werden.

…hier Ihren Text einfügen…

# Skizze der Gesamtsystemarchitektur

Das reine Aufstellen von Anwenderanforderungen ohne Überlegungen zu möglichen Lösungsräumen birgt die große Gefahr, unrealistische Anwenderanforderungen zu definieren. Für die Einordnung, Systematisierung, Kategorisierung und auch Priorisierung von Anwenderanforderungen ist ein Koordinierungsrahmen hilfreich, um die Visualisierung der Anwenderanforderungen zu erleichtern.

Diese Aufgabe kann eine Gesamtsystemarchitektur leisten, die die Sichtweise des Anwenders repräsentiert und nicht die technische Sichtweise des Systemanalytikers beziehungsweise des Systemarchitekten. Das heißt, es ist eine funktionale Systemarchitektur mit Einbettung in die funktionalen Abläufe von Nachbarsystemen zu erstellen. Eine technische Systemarchitektur ist in dieser frühen Phase kaum möglich.

Des Weiteren sind die Besonderheiten der Einsatzumgebung des neuen Systems zu beschreiben, um vor allem die Anforderungen an die Systemsicherheit berücksichtigen zu können.

…hier Ihren Text einfügen…

# 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 „Systemanalyse(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) |

# 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.

# Glossar

|  |  |
| --- | --- |
| Begriff | Erklärung |
| Eingabealphabet |  |
| Eingabewort |  |
| Gültige Definition |  |
| Gültige Turningmaschine |  |

# Abkürzungsverzeichnis

|  |  |
| --- | --- |
| Abkürzung | Erklärung |
|  |  |

# 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)