- Systemanalyse -

Lastenheft für „Geheime Kommunikation mit Quantenkryptographie - eine Lernsoftware zum Verstehen der Grundlagen“

Version: 1.0

|  |  |  |
| --- | --- | --- |
| Projektbezeichnung | QuaKrypto | |
| Projektleiter | Simon Schisslbauer | |
| Verantwortlich | Team | |
| Erstellt am | 30.03.2023 | |
| Zuletzt geändert | 03.04.2023 22:18 | |
| Bearbeitungszustand | X | in Bearbeitung |
|  | vorgelegt |
|  | fertig gestellt |
| Dokumentablage |  | |

Änderungsverzeichnis

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Änderung | | | Geänderte Kapitel | Beschreibung der Änderung | Autor | Zustand |
| Nr. | Datum | Version |
| 1 |  | 1.0 | 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

…hier Ihren Text einfügen…

# Funktionale Anforderungen

## Use-Case Übersicht



## Use-Beschreibungen

|  |  |  |
| --- | --- | --- |
| **Definition eines Übungsszenarios** | | |
| **Kennung** | | UC-1 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Ein Benutzer wählt:   * das Protokoll, das er durch das Programm erlernen will * ob das Übungsszenario lokal oder im Netzwerk durchgeführt werden soll * das gewünschte Ausgangsszenario * die gewünschte(n) Phase(n) * den gewünschten Schwierigkeitsgrad   in einem Startmenü aus. | | |
| **Vorbedingung(en):** | | |
| * Der Benutzer befindet sich im Hauptmenü * Der Benutzer bearbeitet aktuell keine weiteren Übungsszenarien | | |
| **Nachbedingung(en):** | | |
| * Alle Eingaben des Benutzers sind valide * Eine Lobby wurde angelegt | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Anwendungsfall beginnt, wenn der Benutzer das Protokoll auswählt, welches er durch das Programm erlernen will 2. Der Benutzer wählt aus, ob das Übungsszenario lokal oder im Netzwerk bearbeitet werden soll 3. Der Benutzer wählt zunächst das gewünschte Ausgangsszenario aus 4. Der Benutzer wählt die zur Bearbeitung des Übungsszenarios gewünschte(n) Phase(n) aus 5. Der Benutzer wählt den gewünschten Schwierigkeitsgrad aus 6. Der Benutzer bestätigt seine Eingaben 7. Das System prüft, ob die Eingaben des Benutzers gültig sind 8. Das System übernimmt die Eingabedaten und erstellt eine Lobby   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 7a | Ungültige Eingabe | |
|  | 1. Das System weist den Benutzer auf ungültige Eingaben hin   **Ende.** | |
| 8a | Beim gewählten Übungsszenario handelt es sich um eine netzwerkbasierte Sitzung | |
|  | 1. Das System legt eine netzwerkbasierte Lobby an.   **Ende.** | |
| 8b | Beim gewählten Übungsszenario handelt es sich um eine lokale Sitzung | |
|  | 1. Das System legt eine lokale Lobby an.   **Ende.** | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
| * Ist das Startmenü sinnvoll in einem Use Case? Ist das schon zu sehr Realisierung oder passt das so? | | |

|  |  |  |
| --- | --- | --- |
| **Vorhandenem Übungsszenario beitreten** | | |
| **Kennung** | | UC-2 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Der Benutzer oder eine Benutzergruppe tritt einem vorhandenen Übungsszenario bei.  Für ein lokales Übungsszenario legt er   * seine Daten * sein Passwort   fest.  Für ein Netzwerk-basiertes Übungsszenario legt er   * seine Daten * sein Passwort * die Lobby, der er beitreten will,   fest.  Der Benutzer legt zudem seine Rolle fest. Die Rolle (Alice, Bob, Eve) muss genau einem Benutzer zuordnungsbar sein. | | |
| **Vorbedingung(en):** | | |
| Für ein lokales Übungsszenario:   * Ein anderer Benutzer hat den Use Case „Definition eines Übungsszenarios“ einmal abgeschlossen * Ein anderer Benutzer hat eine lokale Übungsszenario Lobby erstellt   Für ein Netzwerk-basiertes Übungsszenario:   * Der Benutzer befindet sich im Hauptmenü * Der Benutzer bearbeitet aktuell keine weiteren Übungsszenarien * Ein anderer Benutzer hat den Use Case „Definition eines Übungsszenarios“ einmal abgeschlossen * Ein anderer Benutzer hat eine netzwerkbasiertes Übungsszenario Lobby erstellt | | |
| **Nachbedingung(en):** | | |
| * Der Benutzer ist der Lobby des Übungsszenarios beigetreten * Alle Eingaben des Benutzers sind valide | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Anwendungsfall beginnt, wenn der Benutzer der Lobby eines Übungsszenarios beitritt 2. Das System prüft, ob die Eingaben gültig sind   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 1a | Der Benutzer fügt bei einer lokalen Lobby eine neue Zeile für einen weiteren Benutzereintrag hinzu | |
|  | 1. Der Benutzer fügt eine leere Zeile hinzu 2. Der Benutzer gibt seine Daten (wie z.B. seinen Namen) ein 3. Der Benutzer legt sein Passwort fest 4. Der Benutzer wählt eine/mehrere noch nicht durch einen anderen Benutzer gewählte(n) Rolle(n) aus   **Rückkehr nach**: 2 | |
| 1b | Der Benutzer wählt im Startmenü die Option aus einem Übungsszenario netzwerkbasiert beizutreten | |
|  | 1. Der Benutzer gibt die Lobby an, der er beitreten will 2. Der Benutzer fügt eine leere Zeile hinzu 3. Der Benutzer gibt seine Daten (wie z.B. seinen Namen) ein 4. Der Benutzer legt sein Passwort fest 5. Der Benutzer wählt eine/mehrere noch nicht durch einen anderen Benutzer gewählte(n) Rolle(n) aus   **Rückkehr nach**: 2 | |
| 1b1a | Ungültige Eingabe | |
|  | 1. Das System weist den Benutzer darauf hin, dass die angegebene Lobby nicht gefunden werden konnte   **Ende.** | |
| 2a | Ungültige Eingabe | |
|  | 1. Das System weist den Benutzer auf ungültige Eingaben hin   **Ende.** | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
| * Wir hätten uns vorgestellt, dass für ein netzwerkbasiertes Übungsszenario jede Rolle (Eve, Alice & Bob) genau einem Rechner zugeordnet werden kann * Lokal und netzwerkbasiert vermischbar? (Beispiel: Eve und Alice üben an Rechner 1, Bob an Rechner 2) | | |

|  |  |  |
| --- | --- | --- |
| **Bearbeitung eines Übungsszenarios** | | |
| **Kennung** | | UC-3 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Ein oder mehrere Benutzer durchlaufen die ihnen zugeordneten Handlungsschritte. | | |
| **Vorbedingung(en):** | | |
| * Use Case „Vorhandenem Übungsszenario beitreten“ wurde abgeschlossen * Jede vorhandene Rolle ist durch genau einem Benutzer/Benutzergruppe belegt | | |
| **Nachbedingung(en):** | | |
| * Ein Protokoll des Verlaufs wurde erstellt | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Anwendungsfall beginnt, wenn ein Benutzer, welcher die Sitzung erstellt hat, das Übungsszenario startet 2. Das System initialisiert das Übungsszenario (inkl. Protokoll) 3. Der Rolle wird der Phase entsprechende Aktionen und Informationen angezeigt 4. Die Rolle führt eine beliebige Anzahl an Aktionen durch 5. Die Rolle bestätigt das Ende ihres Zuges durch Verstecken der Daten 6. Das System speichert die durchgeführten Aktionen im Protokoll ab 7. Das System gibt die Steuerung für die nächste Rolle frei 8. Dieser Vorgang (Schritte 3 bis 8) wird so lange wiederholt, bis die aktuelle Phase abgeschlossen wurde 9. Dieser Vorgang (Schritte 3 bis 9) wird so lange wiederholt, bis die letzte durch den Benutzer ausgewählte Phase abgeschlossen wurde   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 2a | Beginn ab zweiter Phase des Protokolls oder in einer späteren Phase | |
|  | 1. Das System generiert die bis zu der initial ausgewählten Phase benötigten Daten   **Rückkehr nach**: 3 | |
| 5a | Es handelt sich um ein Übungsszenario, das über das Netzwerk abläuft | |
|  | 1. Die Rolle bestätigt das Ende ihres Zuges durch eine entsprechende Aktion   **Rückkehr nach**: 6 | |
| 5a1a | Zwei Rollen benutzen den gleichen Rechner | |
|  | 1. Die Rolle bestätigt das Ende ihres Zuges durch zusätzlich Sperren durch ihr jeweiliges Passwort   **Rückkehr nach**: 6 | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
| * Reicht der Detailierungsgrad?!! | | |

|  |  |  |
| --- | --- | --- |
| **Aufruf des Informationsverzeichnisses** | | |
| **Kennung** | | UC-4 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Um sein Wissen und seine Kenntnisse zu vertiefen, soll der Benutzer ein Verzeichnis mit Informationen über das Protokoll aufrufen können. | | |
| **Vorbedingung(en):** | | |
| * Es existiert ein standardmäßiges Informationsverzeichnis mit den wichtigsten Informationen über das Protokoll * Vom Benutzer bereits erstellte Einträge in das Informationsverzeichnis sind lokal für alle Benutzer eines Rechners gespeichert * Das Verzeichnis ist von jedem Zustand des Programms abrufbar | | |
| **Nachbedingung(en):** | | |
| * Das Informationsverzeichnis, aus dem der Benutzer seine Informationen entnehmen kann, ist geöffnet | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Anwendungsfall beginnt, wenn der Benutzer die Aktion zum Öffnen des Informationsverzeichnisses ausführt. 2. Das System zeigt dem Benutzer das hinterlegte Informationsverzeichnis an. | |
| **Ablauf-Varianten:** | | |
| 2a | Auf dem lokalen Informationsverzeichnis sind keine eigenen Einträge hinterlegt. | |
|  | 1. Das System zeigt dem Benutzer das standardmäßige Informationsverzeichnis an.   **Ende** | |
| 2b | Auf dem lokalen Informationsverzeichnis sind eigene Einträge hinterlegt. | |
|  | 1. Das System zeigt dem Benutzer das erweiterte Informationsverzeichnis an.   **Ende** | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Bearbeitung des Informationsverzeichnisses** | | |
| **Kennung** | | UC-5 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Um sein Informationsverzeichnis zu personalisieren und um Quellen hinzuzufügen und löschen zu können, soll der Benutzer das nach der Installation standardmäßige Informationsverzeichnis editieren können. | | |
| **Vorbedingung(en):** | | |
| * Die Datei des Informationsverzeichnisses ist existent und wurde nicht vom Benutzer gelöscht * Das Informationsverzeichnis ist aufgerufen (Use-Case 4 „Aufruf des Informationsverzeichnisses“) | | |
| **Nachbedingung(en):** | | |
| * Die vom Benutzer getätigten Änderungen an dem Informationsverzeichnis wurden übernommen * Das geänderte Informationsverzeichnis wurden abgespeichert | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Anwendungsfall beginnt, wenn der Benutzer die Aktion zum Editieren des Informationsverzeichnisses ausführt. 2. Der Benutzer wählt aus, ob er einen Eintrag im Informationsverzeichnis bearbeitet, löschen oder hinzufügen will. 3. Das System speichert die vom Benutzer getätigten Änderungen am Informationsverzeichnis.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 2a | Der Benutzer will einen Eintrag im Informationsverzeichnis bearbeiten. | |
|  | 1. Der Benutzer wählt den zu bearbeitenden Eintrag anhand einer Identifikationsnummer aus. 2. Das System lädt den zu bearbeitenden Eintrag. 3. Der Benutzer tätigt seine Änderungen an dem gewählten Eintrag. 4. Der Benutzer bestätigt die getätigten Änderungen am gewählten Eintrag.   **Rückkehr nach**: 3 | |
| 2a4a | Der Benutzer verwirft die Änderungen. | |
|  | **Rückkehr nach**: 2 | |
| 2b | Der Benutzer will einen Eintrag im Informationsverzeichnis löschen. | |
|  | 1. Der Benutzer wählt den zu löschenden Eintrag anhand einer Identifikationsnummer aus. 2. Das System lädt die Überschrift des zu löschenden Eintrags. 3. Der Benutzer bestätigt die Löschung des gewählten Eintrags.   **Rückkehr nach**: 3 | |
| 2b3a | Der Benutzer verwirft die Änderungen. | |
|  | **Rückkehr nach**: 2 | |
| 2c | Der Benutzer will einen Eintrag im Informationsverzeichnis hinzufügen. | |
|  | 1. Das System lädt einen neuen Eintrag im Informationsverzeichnis. 2. Der Benutzer fügt Informationen in den neuen Eintrag ein. 3. Der Benutzer bestätigt die Erstellung des neuen Eintrags.   **Rückkehr nach**: 3 | |
| 2c3a | Der Benutzer verwirft die Änderungen. | |
|  | **Rückkehr nach**: 2 | |
| 2d | Der Benutzer bricht die Bearbeitung des Informationsverzeichnisses ab. | |
|  | 1. Der Benutzer bestätigt den Abbruch des Vorgangs.   **Ende** | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Auswertung des Übungsszenarios** | | |
| **Kennung** | | UC-6 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Um dem Benutzer nach Vollendung eines Übungsszenarios Feedback über sein Protokollverständnis geben zu können, soll das System die Durchführung des Übungsszenarios auswerten. | | |
| **Vorbedingung(en):** | | |
| * Der Benutzer hat die Bearbeitung eines Übungsszenarios vollständig abgeschlossen (Use-Case 3 „Bearbeitung eines Übungsszenarios“) * Das System hat das auszuwertende Übungsszenario protokolliert | | |
| **Nachbedingung(en):** | | |
| * Der Benutzer hat Feedback zu seinen während der Bearbeitung des Übungsszenarios getätigten Aktionen erhalten. | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Anwendungsfall beginnt, wenn der Benutzer die Aktion zur Anzeige der Auswertung des Übungsszenarios ausführt. 2. Das System zeigt dem Benutzer die Auswertung des Übungsszenarios an. 3. Der Benutzer schließt die Auswertung des Übungsszenarios.   **Ende**. | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  | 1. … 2. …   **Rückkehr nach**: | |
| **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 |
|  |  |

# 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)