- Anforderungen -

GraphFrameWork und GraphVisualizer

Version: 1.3

|  |  |  |
| --- | --- | --- |
| Projektbezeichnung | GraphFrameWork und GraphVisualizer | |
| Projektleiter | Benjamin Weigl | |
| Verantwortlich | Benjamin Weigl | |
| Erstellt am | 18.03.2015 | |
| Zuletzt geändert | 25.06.2015 13:15 | |
| Bearbeitungszustand | X | in Bearbeitung |
|  | vorgelegt |
|  | fertig gestellt |
| Dokumentablage | OTH\_SWP\_SS15/Basisverzeichnis/trunk/ 01\_Analyse | |

Weitere Produktinformationen

|  |  |
| --- | --- |
| Mitwirkend | Benjamin Weigl Projektleiter  [nicht beteiligt] Projektmanager  Prof. Dr. Kurt Hoffmann Anwender |

Änderungsverzeichnis

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Änderung | | | Gegeänderte Kapitel | Beschreibung der Änderung | Autor | Zustand |
| Nr. | Datum | Version |
| 1 | 18.03.2015 | 1.0 | Alle | Initiale Produkterstellung | Benjamin Weigl | 1.0 |
| 2 | 21.04.2015 | 1.1 | 1, 3 | Umsetzung Reviewergebnis | Benjamin Weigl | 1.1 |
| 3 | 31.05.2015 | 1.11 | 3 | Layout UC überarbeitet | Mathias Schneider | 1.11 |
| 4 | 16.06.2015 | 1.12 | 3 | UseCase export Szenario Matrizen | Pascal Betting | 1.12 |
| 5 | 17.06.2015 | 1.13 | 3 | Use Cases hinzugefügt (Graph neu anordnen & Knoten verschieben & Knoten selektieren) | Patrick Schiessl | 1.13 |
| 6 | 17.06.2015 | 1.131 | 3 | Use Case Knoten verschieben erweitert | Mathias Schneider | 1.131 |
| 7 | 21.06.2015 | 1.2 | Alle | Rechtschreibfehler einzelne Use Case Beschreibungen überarbeitet | Manuel Schlemelch | 1.2 |
| 8 | 23.06.2015 | 1.21 | 3 | Use Case Beschreibung für Szenarios parametrieren hinzugefügt | Pascal Betting | 1.21 |
| 9 | 23.06.2015 | 1.22 | 3 | Variant Index Beschreibung überarbeitet | Mathias Schneider | 1.22 |
| 10 | 23.06.2015 | 1.23 | 3 | Änderungen in UseCase 1 | Manuel Schlemelch | 1.23 |
| 11 | 24.6.2015 | 1.24 | 3 | Überarbeitung des UseCase 5 | Manuel Schlemelch | 1.24 |
| 12 | 25.6.2015 | 1.3 | Alle | Überarbeitung von Format und Rechtschreibfehler sowie Ergänzung von Beschreibungen | Patrick Schiessl | 1.3 |

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 |
| 14.04.15 | 1.0 |  | Schlemelch Manuel | 1.1 |

Inhalt

[1. Einleitung 4](#_Toc422999800)

[2. Ausgangssituation und Zielsetzung 5](#_Toc422999801)

[3. Funktionale Anforderungen 6](#_Toc422999802)

[3.1 Use-Case Übersicht 6](#_Toc422999803)

[3.1.1 Use-Case-Beschreibungen Framework 6](#_Toc422999804)

[3.1.2 UseCase-Beschreibungen Tool 10](#_Toc422999805)

[3.2 (Sonstige) Funktionalität 16](#_Toc422999806)

[3.3 Modell des Problembereichs (Konzeptionelles Datenmodell) 17](#_Toc422999807)

[4. Nicht-Funktionale Anforderungen 18](#_Toc422999808)

[4.1 Benutzbarkeit (Usability) 18](#_Toc422999809)

[4.2 Zuverlässigkeit (Reliability) 18](#_Toc422999810)

[4.3 Leistung (Performance) 19](#_Toc422999811)

[4.4 Unterstützbarkeit (Supportability) 19](#_Toc422999812)

[4.5 Sonstige Einschränkungen 20](#_Toc422999813)

[4.5.1 Schnittstellen 20](#_Toc422999814)

[4.5.2 Implementierung 20](#_Toc422999815)

[4.5.3 Entwurf 20](#_Toc422999816)

[5. Risikoakzeptanz 20](#_Toc422999817)

[6. Skizze der Gesamtsystemarchitektur 20](#_Toc422999818)

[7. Lieferumfang 20](#_Toc422999819)

[8. Abnahmekriterien 21](#_Toc422999820)

[9. Abkürzungsverzeichnis 21](#_Toc422999821)

[10. Literaturverzeichnis 21](#_Toc422999822)

[11. Abbildungsverzeichnis 22](#_Toc422999823)

# 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

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.

Um die Darstellung von Use-Case Diagrammen und die Erzeugung einer Szenariomatrix zu vereinfachen, soll ein Graph Framework entwickelt und auf Basis diesem ein entsprechendes Tool erstellt werden. Dazu muss das Tool ein im Microsoft Office Word erstelltes Dokument, mit einem entsprechend formatierten Use-Case (oder mehrere), verstehen können, diesen visuell Darstellen und als Szenariomatrix in eine gültigen Microsoft Office Excel Datei exportieren können.

# 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

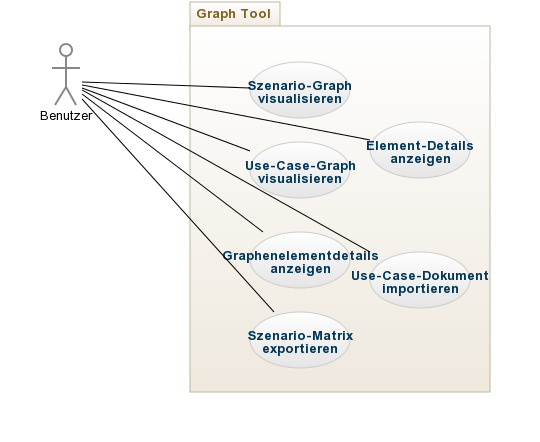
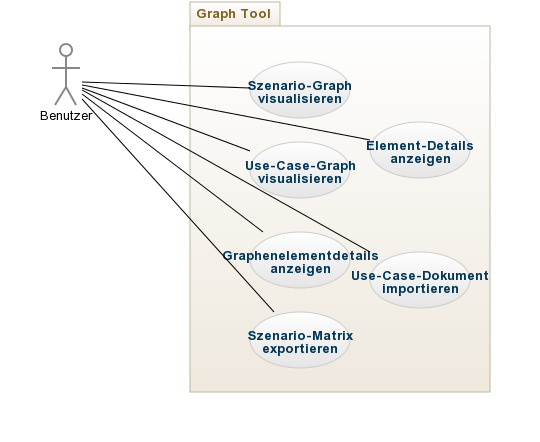
.

Abbildung Use Cases Graphen Tool

Abbildung Use Cases Graphen Framework

1. 1. Use-Case Übersicht
      1. Use-Case-Beschreibungen Framework

|  |  |
| --- | --- |
| **Graph erstellen** | |
| **Kennung** | G-1 |
| **Priorität** | hoch |
| **Kurzbeschreibung:** | |
| Der Aufruf dieser Funktion erstellt ein neues Objekt vom Typ Graph | |
| **Parameter:** | |
| Keine Parameter | |
| **Nachbedingung(en)/Rückgabe:** | |
| Ein gültiges Graph Objekt wurde erstellt. | |

|  |  |
| --- | --- |
| **Graph löschen** | |
| **Kennung** | G-2 |
| **Priorität** | niedrig |
| **Kurzbeschreibung:** | |
| Der Aufruf dieser Funktion entfernt alle Graphelemente vom Graph Objekt und versetzt dieses in einen gültigen aber leeren Ausgangszustand. | |
| **Parameter:** | |
| Keine Parameter | |
| **Nachbedingung(en)/Rückgabe:** | |
| Das Graph Objekt besitzt keine Graphelemente mehr und jeglicher Speicher ist zur Freigabe markiert. | |

|  |  |
| --- | --- |
| **Knoten hinzufügen** | |
| **Kennung** | K-1 |
| **Priorität** | hoch |
| **Kurzbeschreibung:** | |
| Der Aufruf dieser Funktion fügt dem Graphen ein Knoten Objekt hinzu. | |
| **Parameter:** | |
| Der hinzuzufügende Knoten. | |
| **Nachbedingung(en)/Rückgabe:** | |
| Der Graph besitzt das übergebene Knoten Objekt. | |

|  |  |
| --- | --- |
| **Knoten löschen** | |
| **Kennung** | K-2 |
| **Priorität** | niedrig |
| **Kurzbeschreibung:** | |
| Der Aufruf dieser Funktion entfernt einen im Graph befindliches Knoten Objekt und alle zugehörigen Kanten. | |
| **Parameter:** | |
| Ein Index, der den zu entfernenden Knoten eindeutig identifiziert. | |
| **Nachbedingung(en)/Rückgabe:** | |
| Der Graph besitzt nicht länger den identifizierten Knoten, alle zugehörigen Kanten sind entfernt. | |

|  |  |
| --- | --- |
| **Knoten suchen** | |
| **Kennung** | K-3 |
| **Priorität** | mittel |
| **Kurzbeschreibung:** | |
| Der Aufruf dieser Funktion sucht einen Knoten im Graph und liefert im Erfolgsfall dieses zurück. | |
| **Parameter:** | |
| Ein Index oder ein Attribut, der den zu findenden Knoten eindeutig identifiziert. | |
| **Nachbedingung(en)/Rückgabe:** | |
| Eine Liste mit Knoten, die die Suchkriterien erfüllen. Falls kein Knoten gefunden werden konnte, liefert die Funktion einen Null-Wert zurück. | |

|  |  |
| --- | --- |
| **Knoten erstellen** | |
| **Kennung** | K-4 |
| **Priorität** | hoch |
| **Kurzbeschreibung:** | |
| Der Aufruf dieser Funktion erstellt ein neues Objekt vom Typ Knoten. | |
| **Parameter:** | |
| Keine Parameter | |
| **Nachbedingung(en)/Rückgabe:** | |
| Ein gültiges Knoten Objekt wurde erstellt. | |

|  |  |
| --- | --- |
| **Attribut erstellen** | |
| **Kennung** | A-1 |
| **Priorität** | hoch |
| **Kurzbeschreibung:** | |
| Der Aufruf dieser Funktion erstellt ein neues Objekt vom Typ Attribut. | |
| **Parameter:** | |
| Der Typ, den das Attribut repräsentieren soll. | |
| **Nachbedingung(en)/Rückgabe:** | |
| Ein gültiges Attribut Objekt wurde erstellt. | |

|  |  |
| --- | --- |
| **Attribut hinzufügen** | |
| **Kennung** | A-2 |
| **Priorität** | mittel |
| **Kurzbeschreibung:** | |
| Der Aufruf dieser Funktion fügt einem Graphenelement ein Attribut hinzu. | |
| **Parameter:** | |
| Das zu hinzuzufügende Attribut. | |
| **Nachbedingung(en)/Rückgabe:** | |
| Das Graphenelement besitzt das Attribut. | |

|  |  |
| --- | --- |
| **Attribut löschen** | |
| **Kennung** | A-3 |
| **Priorität** | mittel |
| **Kurzbeschreibung:** | |
| Der Aufruf dieser Funktion löscht einen Wert des Attributs | |
| **Parameter:** | |
| Keine Parameter | |
| **Nachbedingung(en)/Rückgabe:** | |
| Das Graphenelement besitzt nicht länger das Attribut | |

|  |  |
| --- | --- |
| **Attribut ändern** | |
| **Kennung** | A-4 |
| **Priorität** | mittel |
| **Kurzbeschreibung:** | |
| Der Aufruf dieser Funktion ändert ein Attribut eines Graphenelements. | |
| **Parameter:** | |
| Keine Parameter | |
| **Nachbedingung(en)/Rückgabe:** | |
| Der Wert des Attributs ist geändert. | |

|  |  |
| --- | --- |
| **Attribut abfragen** | |
| **Kennung** | A-5 |
| **Priorität** | mittel |
| **Kurzbeschreibung:** | |
| Der Aufruf dieser Funktion sucht ein Attribut eines Graphenelements und gibt dieses im Erfolgsfall zurück. | |
| **Parameter:** | |
| Der Typ, den das Attribut besitzen muss. | |
| **Nachbedingung(en)/Rückgabe:** | |
| Eine Liste mit gefundenen Attributen. Falls kein Attribut gefunden werden konnte, liefert die Funktion einen Null-Wert zurück. | |

|  |  |
| --- | --- |
| **Kante erstellen** | |
| **Kennung** | KA-1 |
| **Priorität** | hoch |
| **Kurzbeschreibung:** | |
| Der Aufruf dieser Funktion erstellt ein neues Objekt vom Typ Kante. | |
| **Parameter:** | |
| Zwei Knoten. | |
| **Nachbedingung(en)/Rückgabe:** | |
| Ein gültiges Kanten Objekt wurde erstellt. | |

|  |  |
| --- | --- |
| **Kante löschen** | |
| **Kennung** | KA-2 |
| **Priorität** | mittel |
| **Kurzbeschreibung:** | |
| Der Aufruf dieser Funktion entfernt die Kante. | |
| **Parameter:** | |
| Keine Parameter | |
| **Nachbedingung(en)/Rückgabe:** | |
| Die Kante ist gelöscht. | |

|  |  |
| --- | --- |
| **Kante ändern** | |
| **Kennung** | KA-3 |
| **Priorität** | hoch |
| **Kurzbeschreibung:** | |
| Der Aufruf dieser Funktion ändert die mit der Kante verbundenen Knoten | |
| **Parameter:** | |
| Keine Parameter | |
| **Nachbedingung(en)/Rückgabe:** | |
| Die Knoten der Kante sind geändert. Der Zustand der Kante ist weiterhin gültig. | |

* + 1. Use-Case-Beschreibungen Tool

|  |  |  |
| --- | --- | --- |
| **Use-Case-Dokument Importieren** | | |
| **Kennung** | | UC-1 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
| Der Anwender möchte ein vorliegendes Word Dokument, welches UseCases beinhaltet in das Tool importieren. | | |
| **Vorbedingung(en):** | | |
| Das Dokument (.docx) hat das richtige Format und ist nicht beschädigt. | | |
| **Nachbedingung(en):** | | |
| Die Use-Cases existieren als Datenstruktur und können weiterverarbeitet werden. Es liegt eine Liste aller erfolgreich eingelesenen Use-Cases vor. | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Anwender betätigt den Button zum Dokument einlesen 2. Der Anwender wählt den Pfad zu seinem Use-Case-Dokument 3. Der Anwender wählt sein Dokument aus und bestätigt den Dialog 4. Das Tool gibt eine Rückmeldung, ob Importvorgang erfolgreich 5. Das Tool zeigt eine Liste von eingelesenen Use-Cases   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 2a | Der Anwender bricht den Dialog zur Dateiauswahl ab | |
|  | **Ende.** | |
| 3a | Der Anwender bricht den Dialog zur Dateiauswahl ab | |
|  | **Ende.** | |
| 4a | UseCase-Bezeichnung (Name) tritt mehrmals auf | |
|  | 1. Der Anwender erhält negatives Feedback – Das Tool informiert über die Ursache des Fehlers 2. Das Tool überspringt den UseCase und fährt mit dem nächsten UseCase fort   **Ende.** | |
| 4b | UseCase-Kennung (z.B. UC-1, UC-2, …) tritt mehrmals auf | |
|  | 1. Der Anwender erhält negatives Feedback – Das Tool informiert über die Ursache des Fehlers 2. Das Tool überspringt den UseCase und fährt mit dem nächsten UseCase fort   **Ende.** | |
| 4c | Fehler in der Dokumentstruktur der Word-Datei (.docx) | |
|  | 1. Der Anwender erhält negatives Feedback – Das Tool informiert über die Ursache des Fehlers (kann WORD-Dokument nicht öffnen, Fehler in der Dokumentenstruktur oder einfach nur leeres Dokument)   **Ende.** | |
| **Spezielle Anforderungen:** | | |
| keine | | |
| **Zu klärende Punkte:** | | |
| Soll der Anwender mehrere Dateien auswählen können, die eingelesen werden sollen? | | |

|  |  |  |
| --- | --- | --- |
| **Use-Case-Graph visualisieren** | | |
| **Kennung** | | UC-2 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Das Programm visualisiert mithilfe des Systems und vorher importierten Use-Cases, den Graphen. | | |
| **Vorbedingung(en):** | | |
| Mindestens ein Use-Case muss importiert sein | | |
| **Nachbedingung(en):** | | |
| Ein Use-Case Graph wurde visualisiert.  Use-Case in der Auswahlliste bleibt selektiert. | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, sobald der User ein Use-Case aus der Liste selektiert. 2. Die Zeichenfläche des Use-Cases wird angezeigt 3. Das Tool analysiert den Graphen des entsprechenden Use-Cases und zeichnet alle Knoten und Kanten auf die Zeichenfläche   **Ende**. | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Szenario-Graph visualisieren** | | |
| **Kennung** | | UC-3 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Das Programm visualisiert mithilfe des Systems und vorher importierten UseCases, die Szenariographen. | | |
| **Vorbedingung(en):** | | |
| Mindestens ein UseCase muss importiert sein und mindestens ein Szenario wurde vom System aus diesem analysiert. Ein UseCase muss aus einer Liste selektiert sein. | | |
| **Nachbedingung(en):** | | |
| Die Details des selektierten Graphenelements werden angezeigt | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, sobald der User ein Szenario aus einer Liste selektiert 2. Alle Farben des UseCase Graphen werden zurückgesetzt 3. Alle Knoten und Kanten des Szenariographen werden in der Farbe Rot auf der Zeichenfläche eingefärbt   **Ende**. | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
| Ist kein UseCase aus der Liste der eingelesenen UseCases ausgewählt (z.B. direkt nach dem Einlesen des Word-Dokuments), so ist die Lasche „Scenarios“ deaktiviert | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Graphenelementdetails anzeigen** | | |
| **Kennung** | | UC-4 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Das Programm visualisiert mithilfe des Systems alle Details des zuvor ausgewählten Graphenelements. | | |
| **Vorbedingung(en):** | | |
| Mindestens ein Graphenelement muss selektiert sein | | |
| **Nachbedingung(en):** | | |
| Die Details des selektierten Graphenelements werden angezeigt | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, sobald der User ein Graphenelement (UseCase oder Sequenz aus der Liste oder ein Graphenelement auf der Zeichenfläche) auswählt 2. Alle Eigenschaften des Graphenelement werden dem User in einem separaten Bereich des Tool angezeigt und visualisiert 3. Alle Attribute werden angezeigt   **Ende**. | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Szenario-Matrix exportieren** | | |
| **Kennung** | | UC-5 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
| Das System exportiert, die für den selektierten Use-Case vorhanden Scenario-Matrix als .xlsx Datei. (Die Scenario-Matrizen der einzelnen Use-Cases werden schon beim Selektieren des Use-Cases erstellt.) | | |
| **Vorbedingung(en):** | | |
| Ein Use-Case ist selektiert, welcher mindestens ein Scenario enthält. | | |
| **Nachbedingung(en):** | | |
| Eine Excel-Datei für die Scenario-Matrix des Use-Cases wurde erstellt. | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der User über die Benutzer-System-Schnittstelle die Option „Export scenario matrix“ auswählt. 2. Der Benutzer wählt über einen entsprechenden Dialog einen Pfad und Dateinamen aus, in dem die exportierte Excel-Datei abgelegt werden soll. 3. Das System erzeugt nun anhand der im Speicher liegenden Scenario-Matrix eine .xlsx Datei mit den Informationen der Szenarien und speichert diese an dem in Schritt 2 angegebenem Pfad ab.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 3a | Das Schreiben der Excel-Datei ergab einen Fehler | |
|  | 1. Das System gibt eine Fehlermeldung aus, daraufhin endet der Use-Case.  **Ende.** | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Alle Szenario-Matrizen exportieren** | | |
| **Kennung** | | UC-6 |
| **Priorität** | | hoch |
| **Kurzbeschreibung:** | | |
| Das System exportiert die Scenario-Matrizen aller eingelesenen Use-Cases als .xlsx Datei. (Die Scenario-Matrizen der einzelnen Use-Cases werden schon beim Selektieren des Use-Cases erstellt.) | | |
| **Vorbedingung(en):** | | |
| Mindestens ein Use-Case ist eingelesen, welcher Szenarien enthält. | | |
| **Nachbedingung(en):** | | |
| Eine Excel-Datei für die Scenario-Matrizen der Use-Cases wurde erstellt. | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der User über die Benutzer-System-Schnittstelle die Option „Export all scenarios“ auswählt. 2. Der Benutzer wählt über einen entsprechenden Dialog einen Pfad und Dateinamen aus, in dem die exportierte Excel-Datei abgelegt werden soll. 3. Das System erzeugt nun anhand der im Speicher liegenden Scenario-Matrizen der eingelesenen Use-Cases eine .xlsx Datei mit den Informationen der Szenarien und speichert diese an dem in Schritt 2 angegebenem Pfad ab.   **Ende**. | |
| **Ablauf-Varianten:** | | |
| 3a | Das Schreiben der Excel-Datei ergab einen Fehler | |
|  | 1. Das System gibt eine Fehlermeldung aus, daraufhin endet der UseCase.  **Ende.** | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Knoten auf Zeichenfläche verschieben** | | |
| **Kennung** | | UC-7 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Durch die gedrückte linke Maustaste auf ein Elements in der Zeichenfläche kann dieses mithilfe der Mausbewegung neu positioniert werden. Hinweis: Die neue Position wird gespeichert und bei erneuter Selektion des Use Case Graphens wieder geladen. Wird ein neues Dokument oder der Use-Case "Use-Case-Graphen neu anordnen auf Zeichenfläche" (UC-9) ausgeführt, wird der Knoten wieder an der Default Position gezeichnet. | | |
| **Vorbedingung(en):** | | |
| Mindestens ein Use Case muss importiert und ausgewählt sein. | | |
| **Nachbedingung(en):** | | |
| Das Element befindet sich auf der neuen Position proportional zum Mauszeiger. | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, sobald der User ein Element auf der Zeichenfläche auswählt und die linke Maustaste gedrückt hält 2. Durch die anschließende Mausbewegung, verschiebt sich das Element auf der Zeichenfläche proportional zur Maus. 3. Sobald der User die linke Maustaste loslässt, wird das Element auf die Position auf der Zeichenfläche platziert und der Use Case endet | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Zeichenfläche zoomen** | | |
| **Kennung** | | UC-8 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Durch die gedrückte linke Steuerungstaste (Strg) und gleichzeitiges drehen des Mausrades kann die Ansicht der Zeichenfläche vergrößert oder verkleinert werden. | | |
| **Vorbedingung(en):** | | |
| Mindestens ein Use Case muss importiert und ausgewählt sein. Auch muss die Zeichenfläche oder ein Element als letztes selektiert werden, sodass der Fokus des Fensters auf der Zeichenfläche bzw. auf einem deren Elemente ist. | | |
| **Nachbedingung(en):** | | |
| Die Zeichenfläche wurde vergrößert bzw. verkleinert proportional zur Drehung des Mausrades | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, sobald der User die linke Steuerungstaste (Strg) gedrückt hält 2. Durch die anschließende Drehung des Mausrades nach unten verkleinert sich die Ansicht der Zeichenfläche (herauszoomen aus der Zeichenfläche) 3. Sobald der User die linke Steuerungstaste loslässt endet der Use Case | |
| **Ablauf-Varianten:** | | |
| 2a | Der User dreht das Mausrad nach oben (hineinzoomen in die Zeichenfläche) | |
|  | 1. Durch die anschließende Drehung des Mausrades nach oben vergrößert sich die Ansicht der Zeichenfläche (hineinzoomen in die Zeichenfläche)   Rückkehr nach: 2 | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Use-Case-Graphen neu anordnen auf Zeichenfläche** | | |
| **Kennung** | | UC-9 |
| **Priorität** | | Hoch |
| **Kurzbeschreibung:** | | |
| Sobald die Schaltfläche neben dem ausgewählte Use Case gedrückt wird, werden alle Knoten neu angeordnet und neu platziert mit einer internen Logik und alle Kanten werden neu gezeichnet. | | |
| **Vorbedingung(en):** | | |
| Mindestens ein Use Case muss importiert und ausgewählt sein | | |
| **Nachbedingung(en):** | | |
| Ein Use Case Graph wurde neu in der Zeichenfläche angeordnet | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, sobald der User ein Use-Case aus der Liste selektiert und auf die zugehörige Schaltfläche für die Anordnung drückt. 2. Der Use-Case Graph wird neu angeordnet (Kanten werden nach interner Logik platziert und Kanten werden neu gezeichnet) 3. Sobald der Graph fertig gezeichnet wurde, endet der Use-Case | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Szenarios parametrieren** | | |
| **Kennung** | | UC-10 |
| **Priorität** | | Mittel |
| **Kurzbeschreibung:** | | |
| Über den Szenario-Tab können die Parameter für die Szenario-Erstellung modifiziert werden. Dabei kann man folgendes festlegen:   * Wie viele Varianten pro Szenario maximal durchlaufen werden sollen * Wie oft eine Schleife in den Szenarien durchlaufen werden soll | | |
| **Vorbedingung(en):** | | |
| Mindestens ein Use- Case muss selektiert sein. | | |
| **Nachbedingung(en):** | | |
| Die Szenarios des Use-Cases wurden neu berechnet. | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, sobald der User einen der Szenario-Parameter im Szenario-Tab ändert 2. Das System berechnet die Szenarien neu. Gegebenenfalls ändert sich die Anzahl der insgesamt vorhanden Szenarien für den selektierten Use-Case 3. Sobald die Szenarien neu berechnet wurden, endet der Use-Case | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

* 1. (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 |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

* 1. 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 in den Software-Projekten für den Studiengang „Angewandte Informatik“ das konzeptionelle Modell an dieser Stelle zusammen mit den Use-Cases dokumentiert.

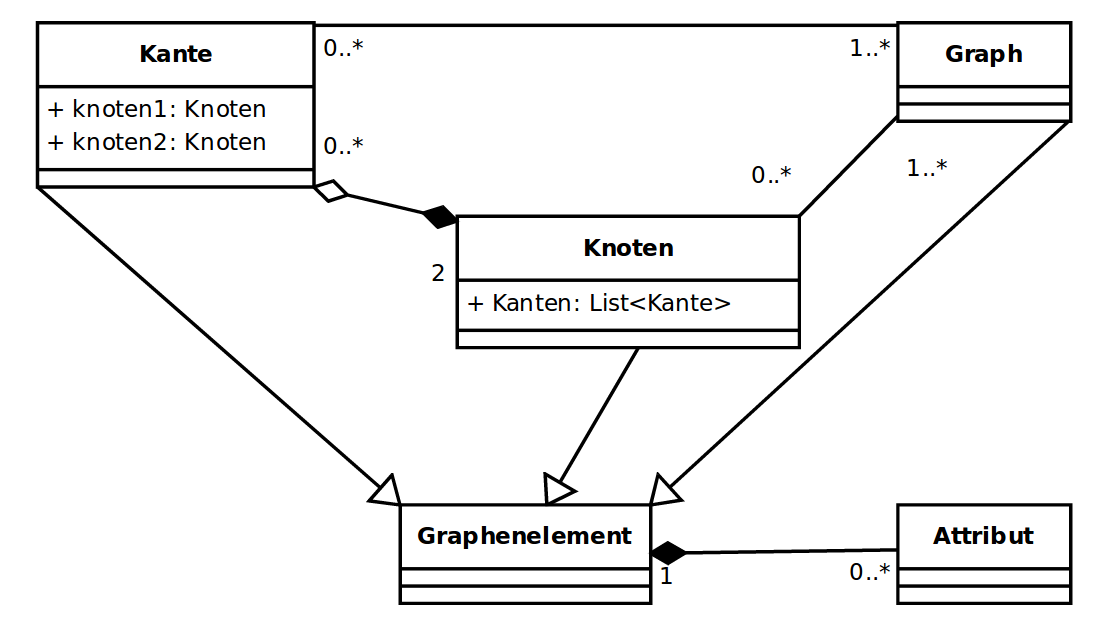


Abbildung Konzeptionelles Datenmodell Graphen Framework

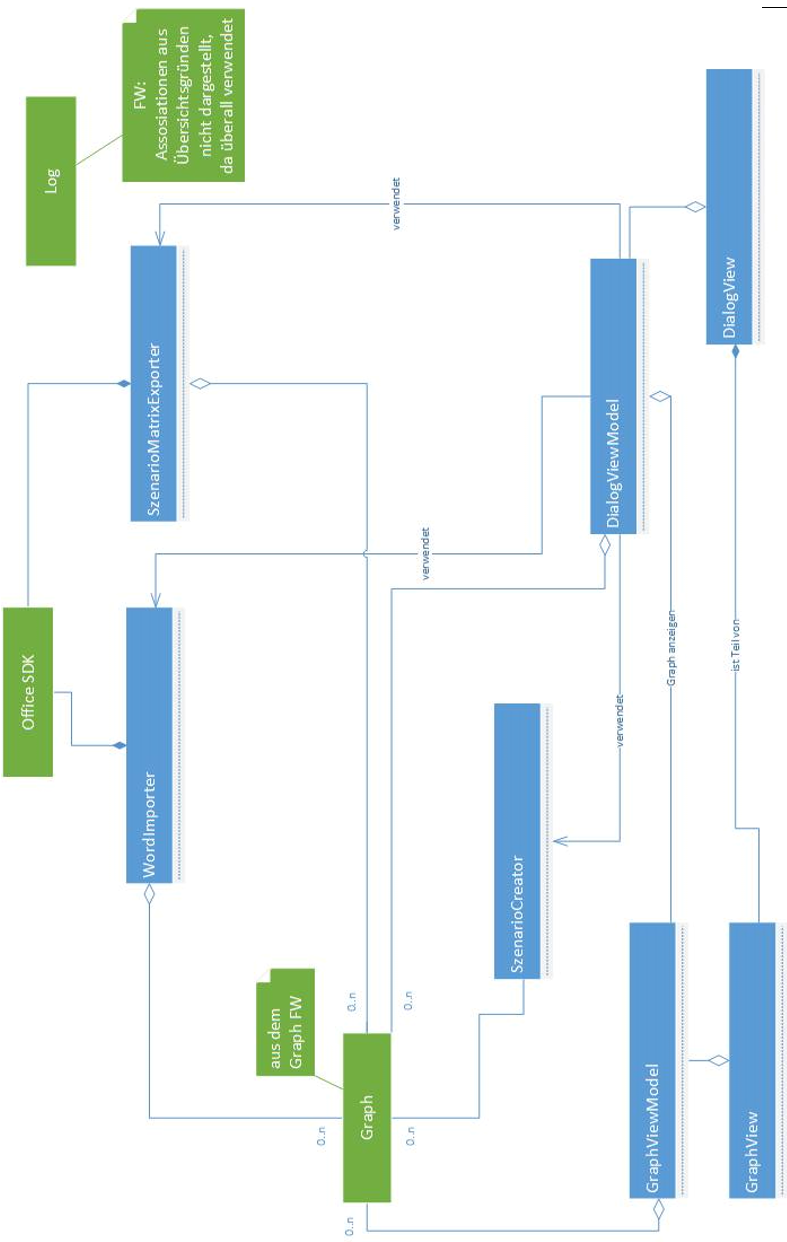


Abbildung Konzeptionelles Datenmodell Tool

# 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.

1. 1. 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** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

* 1. 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… |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

* 1. 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… |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

* 1. 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** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

* 1. 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.

* + 1. Schnittstellen
    2. Implementierung
    3. 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.

# 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) * Die Dokumentation ist für die Weiterentwicklung in künftigen SW-Projekten besonders wichtig (Qualität!) |
| **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) * Die Dokumentation ist für die Weiterentwicklung in künftigen SW-Projekten besonders wichtig (Qualität!) |
| **Implementierung** | | | |
| 3 | Lauffähiger und getesteter Quellcode (incl. Entwicklerdokumentation) |  | Abgabe am Semesterende   * Die Dokumentation ist für die Weiterentwicklung in künftigen SW-Projekten besonders wichtig (Qualität!) |
| **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 |
|  |  |

# Literaturverzeichnis

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

# Abbildungsverzeichnis

[Abbildung 1 Use Cases Graphen Tool 6](file:///C:\Users\Luke\Documents\GitHub\OTH_SWP_SS15\Basisverzeichnis\trunk\01_Analyse\Anforderungen.docx#_Toc423001035)

[Abbildung 2 Use Cases Graphen Framework 6](file:///C:\Users\Luke\Documents\GitHub\OTH_SWP_SS15\Basisverzeichnis\trunk\01_Analyse\Anforderungen.docx#_Toc423001036)

[Abbildung 3 Konzeptionelles Datenmodell Graphen Framework 16](#_Toc423001037)

[Abbildung 4 Konzeptionelles Datenmodell Tool 16](#_Toc423001038)

1. V-Modell® ist eine geschützte Marke der Bundesrepublik Deutschland. [↑](#footnote-ref-1)