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

- Anforderungen-

Sisyphus

Version: 0.1

|  |  |  |
| --- | --- | --- |
| Projektbezeichnung | Sisyphus | |
| Projektleiter | Tilo Fischer | |
| Verantwortlich | Thomas Bartmann | |
| Erstellt am | 18.5.14 | |
| Zuletzt geändert | 20.05.2014 23:35 | |
| Bearbeitungszustand | X | in Bearbeitung |
|  | vorgelegt |
|  | fertig gestellt |
| Dokumentablage | C:\Users\[User]\Documents\GitHub\Sisyphus\\_source\\_analysis\Anforderungen(XYZ).docx | |

Weitere Produktinformationen

|  |  |
| --- | --- |
| Mitwirkend | [Tilo Fischer] Projektleiter  [---] Projektmanager  [Kurt Hoffmann] Anwender |

Änderungsverzeichnis

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Änderung | | | Geänderte Kapitel | Beschreibung der Änderung | Autor | Zustand |
| Nr. | Datum | Version |
| 1 | 20.5.14 | 0.1 | 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](#_Toc382574146)

[2 Ausgangssituation und Zielsetzung 4](#_Toc382574147)

[3 Funktionale Anforderungen 7](#_Toc382574148)

[3.1 Use-Case Übersicht 7](#_Toc382574149)

[3.2 Use-Beschreibungen 7](#_Toc382574150)

[3.3 (Sonstige) Funktionalität 9](#_Toc382574151)

[3.4 Modell des Problembereichs (Konzeptionelles Datenmodell) 9](#_Toc382574152)

[3.5 Klassenmodell 9](#_Toc382574152)

[4 Nicht-Funktionale Anforderungen 9](#_Toc382574153)

[4.1 Benutzbarkeit (Usability) 10](#_Toc382574154)

[4.2 Zuverlässigkeit (Reliability) 10](#_Toc382574155)

[4.3 Leistung (Performance) 11](#_Toc382574156)

[4.4 Unterstützbarkeit (Supportability) 11](#_Toc382574157)

[4.5 Sonstige Einschränkungen 12](#_Toc382574158)

[5 Risikoakzeptanz 13](#_Toc382574159)

[6 Skizze der Gesamtsystemarchitektur 13](#_Toc382574160)

[7 Lieferumfang 14](#_Toc382574161)

[8 Abnahmekriterien 14](#_Toc382574162)

[9 Abkürzungsverzeichnis 14](#_Toc382574163)

[10 Literaturverzeichnis 15](#_Toc382574164)

[11 Abbildungsverzeichnis 15](#_Toc382574165)

# 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

Mathematik-Dozenten stehen oft vor dem Problem, für Studierende aus anwendungsbezogenen Studiengängen einen Zugang zu bestimmten mathematischen Teilgebieten finden zu müssen, der möglichst gut das Interesse wecken kann. Der gewählte Zugang soll allerdings nicht bloß oberflächliches Edutainment bedeuten, sondern ernsthaft das Verständnis für jeweils relevante mathematische Konzepte und Zusammenhänge fördern.

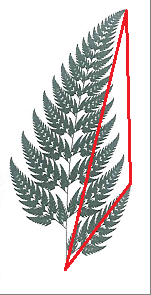
In vielen Mathematik-Vorlesungen stellen die Themen „lineare Algebra“ und „komplexe Zahlen“ einen wichtigen Teil des Stoffs dar. In einer speziell für Studierende der Medientechnik gedachten Mathematik-Vorlesung an der OTH-Amberg-Weiden soll zu diesen beiden Themen im Rahmen eines didaktischen Experiments ein Zugang über die Erzeugung und Manipulation „interessant aussehender“ digitaler Bilder ausprobiert werden – genauer ist dabei an Folgendes gedacht:

Für beide Themen scheint die Erzeugung von Fraktalen ein schöner „Aufhänger“ zu sein. Fraktale faszinieren häufig schon beim bloßen Betrachten und für ihre Erzeugung gibt es ein Verfahren, bei dem mathematische Konzepte wie „lineare Abbildung“ bzw. „affine Abbildung“ eine zentrale Rolle spielen – zugleich lassen sich sehr viele der für das Thema „Lineare Algebra“ zu lehrenden Inhalte rund um diese beiden Begriffe herum aufziehen.

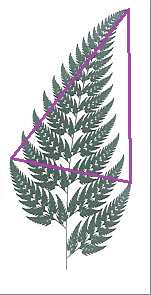
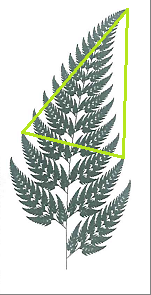
Die zu entwickelnde Software soll deswegen Studierenden (und Dozenten) ermöglichen, die Bedeutung und grundlegende Eigenschaften affiner Abbildungen im Zusammenhang mit der Fraktal-Erzeugung erkunden und verstehen zu können. Dafür ist folgende Funktionalität besonders wichtig:

* Mit Hilfe eines komfortablen Editors können sog. „iterierte Funktionensysteme“ aus (kontrahierenden) affinen Abbildungen zusammengestellt bzw. verändert werden: Man kann affine Abbildungen zu einem iterierten Funktionensystem hinzufügen oder daraus entfernen. Ebenso soll es möglich sein, eine sog. „Kondensationsabbildung“ mitaufzunehmen.
* Für jede einzelne affine Abbildung eines iterierten Funktionensystems erlaubt der Editor die direkte Eingabe der Zahlenwerte für die Koeffizienten von Matrix und Verschiebevektor.
* Zur noch bequemeren Eingabe affiner Abbildungen soll es möglich sein, ein digitales Bild einer fraktalen Struktur einzulesen und dann durch Mausclicks die Punkte  eines Originaldreiecks und die Punkte  des zugehörigen Bilddreiecks anzugeben. Das System berechnet dann die affine Abbildung, die  auf ,  auf  und  auf  abbildet (wenn die Punkte des Originaldreiecks  nicht kollinear sind, dann ist auf diese Weise stets eine affine Abbildung eindeutig festgelegt). Hierzu ein Beispiel:

Auf einem digitalen Bild des sog „Black-Spleenwort“-Farns (auch als „Barnsleys Farn“ bekannt) gibt der Benutzer zunächst das Originaldreieck (rot) und dafür zwei Bilddreicke (grün bzw. blau) an:

Dies ergibt zwei affine Abbildungen. Durch Angabe eines weiteren Originaldreiecks (lila) mit zugehörigem Bilddreick (hellgrün) wird eine dritte affine Abbildung definiert:

 .

Die vierte und letzte affine Abbildung bildet das pinkfarbene Dreieck auf ein zur Strecke entartetes Bilddreieck (orange) ab:

* Das Programm kann zu einem itererierten Funktionensystem das zugehörige Fraktal (oder genauer gesagt eine „gute“ Näherung davon) erzeugen, damit der Anwender erkennen kann, welches Ergebnis „sein“ iteriertes Funktionensystem hervorbringt – dabei ist es durchaus reizvoll, zu einem Fraktal, welches als digitales Bild vorliegt (z.B. der obige Farn), selbst nach einem passenden iterierten Funktionensystem zu suchen und zu schauen, ob das „eigene“ Funktionensystem wirklich das gegebene Fraktal liefert. Das Programm soll die folgenden beiden Erzeugungsverfahren anbieten:
  + Der übliche zufallsgesteuerte Algorithmus erzeugt ausgehend von einem gegebenen Punkt in jedem Schritt einen zusätzlichen Punkt, indem eine zufällig aus dem iterierten Funktionensystem gewählte Abbildung auf den jeweils letzten Punkt angewendet wird. Für jede einzelne Abbildung des iterierten Funktionensystem soll der Benutzer die Wahrscheinlichkeit, mit der sie ausgewählt wird, selbst festlegen können. Nach einer vorgegebenen Anzahl von Schritten bricht der Algorithmus ab – die so entstandene Punktmenge zeigt im Allgemeinen eine gute Näherung des Fraktals.
  + Der andere -ebenfalls übliche- Algorithmus geht von einer vorgegebenen Figur aus (z.B. ein Dreieck oder Rechteck) und wendet in jedem Schritt alle Abbildungen des iterierten Funktionensystems auf diese Figur an; die dabei entstehenden Bildmengen werden in einer Art „Collage“ zu einer neuen Figur zusammengefügt, mit der dann im nächsten Schritt das Gleiche wiederholt wird. Nach genügend vielen Wiederholungen erhält man ebenfalls eine brauchbare Näherung für das Fraktal. Hier ein Beispiel zu diesem Verfahren:







Schritt 1







Schritt 2







Schritt 2

Schritt 3







Für das Thema „Komplexe Zahlen“ sollte man mit dem Programm die Wirkung bestimmter Transformationen  erkunden können, um so das Verständnis für bestimmte grundlegende Tatsachen über komplexe Zahlen zu fördern. Zu diesen grundlegenden Tatsachen gehört in jedem Falle:

* Wirkung der Multiplikation in  („Beträge multiplizieren sich, Winkel addieren sich“)
* Multiplikation mit einer festen komplexen Zahl  (d.h. die Abbildung ) wirkt wie eine Drehstreckung
* Wirkung des Potenzierens bzw. „Wurzelziehens“ (wie wirkt z.B.  ?).

Denkbar wäre es hierzu, mit Hilfe eines Editors bestimmte Transformationen eingeben zu können, und deren Wirkung zu verdeutlichen, indem man ein digitales Bild (z.B. ein Foto) mit Hilfe der Transformation verzerrt. Sofern die Transformationen kontrahierend wirken, könnten diese auch bei der Fraktalerzeugung als Bestandteile eines iterierten Funktionensystems verwendbar sein – man hätte damit eine Brücke zum Bereich „Fraktale“ geschlagen.

Schöne wäre es auch, wenn man mit dem Programm auch andere Verfahren der Fraktalerzeugung (für Mandelbrot- und Julia-Mengen) und deren Verbindung zum Thema „Komplexe Zahlen“ studieren könnte.

Gegenüber vielen gängigen Programmen zur Fraktalerzeugung und –betrachtung bietet dieses Programm dem Benutzer sehr weit gehende Möglichkeiten zur Beeinflussung der Erzeugungsverfahren, und die Chance zu verstehen, dass die komplex aussehenden Strukturen der Fraktale ohne (auf den ersten Blick trocken anmutende) mathematische Konzepte wie affine Abbildungen und komplexe Zahlen nicht beherrscht werden könnten.

Der Fokus liegt bei diesem Programm auf der Didaktik: überall, wo es bei mathematischen Konzepten wie affinen Abbildungen, Transformationen der Zahlenebene , etc. etwas zu verstehen gibt, das im Zusammenhang der Fraktalerzeugung besonders wichtig ist, sollte der Benutzer zum Nachdenken angeregt werden und die Chance auf „Aha-Erlebnisse“ geboten bekommen.

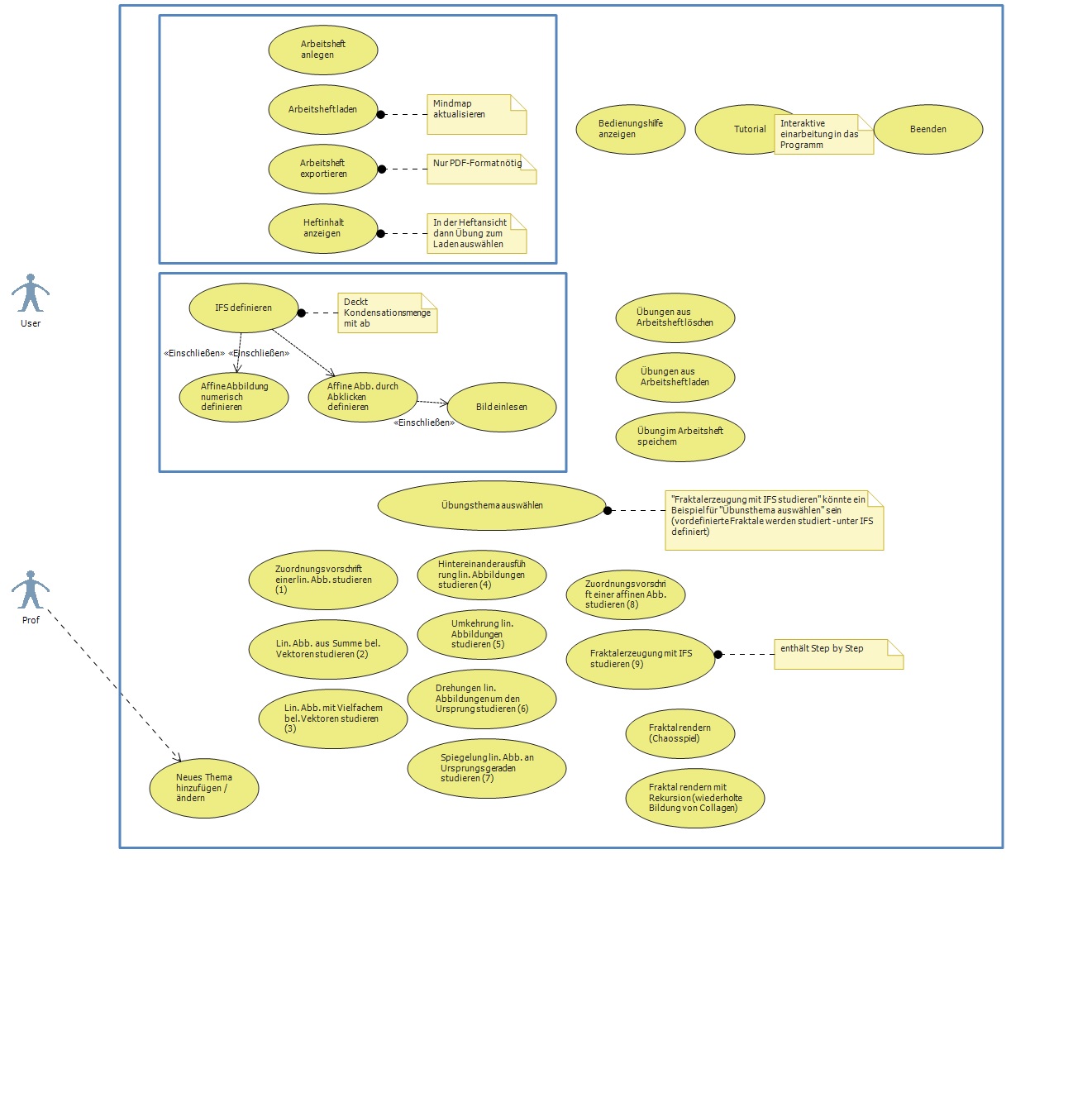
# Funktionale Anforderungen

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



## Use-Beschreibungen

|  |  |  |
| --- | --- | --- |
| **Arbeitsheft anlegen** | | |
| **Kennung** | | UC-A1 |
| **Priorität** | | 10 |
| **Kurzbeschreibung:** | | |
| Dieser Use-Case beschreibt das Anlegen eines Arbeitsheftes. | | |
| **Vorbedingung(en):** | | |
|  | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Use-Case beginnt, nachdem der Benutzer in der Mindmap auf eine Übung geklickt hat und danach “Arbeitsheft anlegen“ gewählt hat. 2. Der Benutzer gibt seinen Vor- und Nachnamen ein. 3. Das System legt ein Arbeitsheft mit diesem Namen an. 4. Der Use-Case endet. | |
| **Ablauf-Varianten:** | | |
| 2a | Es ist bereits ein Arbeitsheft mit diesem Namen vorhanden. | |
|  | 1. Der Benutzer kann das vorhandene Arbeitsheft überschreiben oder abbrechen.  2. Der Use-Case endet. | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Arbeitsheft laden** | | |
| **Kennung** | | UC-A2 |
| **Priorität** | | 8 |
| **Kurzbeschreibung:** | | |
| Dieser Use-Case beschreibt das Laden eines vorhandenen Arbeitsheftes. | | |
| **Vorbedingung(en):** | | |
|  | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, nachdem der Benutzer in der Mindmap auf eine Übung geklickt hat und danach “Arbeitsheft laden“ gewählt hat. 2. Das System zeigt dem Benutzer die Arbeitshefte an, die geladen werden können. 3. Der Benutzer wählt ein beliebiges Arbeitsheft aus. 4. Das System aktualisiert die Mindmap, in der die bereits bearbeiteten Übungen aus diesem Arbeitsheft kenntlich gemacht werden. 5. Die ausgewählte Übung wird gestartet. 6. Der Use-Case endet. | |
| **Ablauf-Varianten:** | | |
| 2a | Es gibt kein vorhandenes Arbeitsheft, welches geladen werden kann. | |
|  | 1. Das System teilt dem Benutzer dies mit.  2. Der Use-Case endet. | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Arbeitsheft als PDF exportieren** | | |
| **Kennung** | | UC-A3 |
| **Priorität** | | 8 |
| **Kurzbeschreibung:** | | |
| Dieser Use-Case beschreibt das Exportieren eines Arbeitsheftes als PDF. | | |
| **Vorbedingung(en):** | | |
| Es muss ein Arbeitsheft geöffnet sein. | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der Benutzer dem System signalisiert, dass er das aktuelle Arbeitsheft als PDF exportieren möchte. 2. Vom System wird ein “Speichern unter“ Dialog-Fenster angezeigt. 3. Der Benutzer wählt den gewünschten Speicherort. 4. Der Benutzer gibt einen Namen für das PDF ein. 5. Das System speichert alle Übungen des Arbeitsheftes in einer PDF-Datei. 6. Der Use-Case endet. | |
| **Ablauf-Varianten:** | | |
| 4a | Es gibt bereits ein PDF mit diesem Namen. | |
|  | 1. Der Benutzer kann die Datei überschreiben, oder Abbrechen.  2. Der Use-Case endet. | |
| 5a | Auf den Speicher kann nicht schreibend zugegriffen werden. | |
|  | 1. Das System teilt dem Benutzer dies mit.  2. Der Use-Case endet. | |
| **Spezielle Anforderungen:** | | |
| PDF Library für die Implementierung | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Heftinhalt anzeigen** | | |
| **Kennung** | | UC-A4 |
| **Priorität** | | 8 |
| **Kurzbeschreibung:** | | |
| Dieser Use-Case beschreibt das Anzeigen des Inhalts eines Arbeitsheftes | | |
| **Vorbedingung(en):** | | |
| Es ist bereits ein Arbeitsheft angelegt oder geladen worden. | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der Benutzer dem System signalisiert, dass er den Heftinhalt des angelegten oder geladenen Arbeitsheftes angezeigt haben möchte. 2. Das System zeigt den Heftinhalt mit allen Übungen an. 3. Der Use-Case endet. | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Bild einlesen** | | |
| **Kennung** | | UC-B1 |
| **Priorität** | | 6 |
| **Kurzbeschreibung:** | | |
| Einlesen eines Bildes das dem Benutzer beim Definieren einer Affinen Abbildung hilft. | | |
| **Vorbedingung(en):** | | |
| Die Übung „IFS definieren“ ist geöffnet | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Anwendungsfall beginnt, wenn der Benutzer „Bild öffnen“ klickt. 2. Das System gibt dem Benutzer die Möglichkeit eine Bilddatei aus dem Dateisystem auszuwählen 3. Der Benutzer wählt ein Bild aus. 4. Das System überprüft das Bild. 5. System zeigt das Bild innerhalb des Übungsblattes an 6. Der Anwendungsfall endet | |
| **Ablauf-Varianten:** | | |
| 2b | Abbruch | |
|  | 1. Benutzer bricht Bildauswahl ab. 2. Weiter bei Schritt 6 im normalen Ablauf. | |
| 4a | Ungültiges Format | |
|  | 1. Das System weist den Benutzer darauf hin, dass das Bildformat nicht unterstützt wird 2. Weiter mit Schritt 6 im normalen Ablauf. | |
| 4b | Ungültige Größe | |
|  | 1. Das System weist den Benutzer darauf hin, dass das Bild eine ungültige Größe hat. 2. Weiter mit Schritt 6 im normalen Ablauf. | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
| - bei der Bildauswahl, soll es möglich sein den Pfad des Dateisystems anzupassen, dabei werden die jeweils im Pfad verfügbaren Bilder aufgelistet | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Bedienungshilfe anzeigen** | | |
| **Kennung** | | UC-B2 |
| **Priorität** | | 1 |
| **Kurzbeschreibung:** | | |
| Die Bedienungshilfe hilft dem Benutzer das Programm zu verstehen. | | |
| **Vorbedingung(en):** | | |
| Es ist eine Übung geöffnet. | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der User die "Bedienungshilfe" auswählt.  2. Das System zeigt das Bedienungshilfe-Fenster an.  3. Der Benutzer navigiert in der Bedienungshilfe vor und zurück.  4. Der Use-Case endet. | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Beenden** | | |
| **Kennung** | | UC-B3 |
| **Priorität** | | 10 |
| **Kurzbeschreibung:** | | |
| Dieser Use-Case beschreibt, wie das Programm beendet wird. | | |
| **Vorbedingung(en):** | | |
|  | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der Benutzer das Schließen des Programmes anfordert. 2. Das System prüft ob alle Daten gespeichert sind. 3. Das System beendet sich. 4. Der Anwendungsfall endet. | |
| **Ablauf-Varianten:** | | |
| 2.a | Alles ist gespeichert. | |
|  | 1. Weiter mit Schritt 4 im normalen Ablauf. | |
| 2.b | Es gibt nicht gespeicherte Daten. | |
|  | 1. Das System teilt dem Benutzer mit das nicht alle Daten gespeichert wurden und gibt dem Benutzer die Möglichkeit entweder das Beenden abzubrechen oder ohne Speicherung zu Beenden. 2. Der Benutzer wählt. 3. Weiter mit Schritt 4 im normalen Ablauf wenn der Benutzer das Beenden bestätigt, ansonsten weiter mit Schritt 5. | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Neues Thema hinzufügen/ändern** | | |
| **Kennung** | | UC-N1 |
| **Priorität** | | 2 |
| **Kurzbeschreibung:** | | |
| Dieser Use-Case beschreibt den Ablauf, wie der Administrator neue Themen hinzufügen oder bestehende ändern kann. | | |
| **Vorbedingung(en):** | | |
|  | | |
| **Nachbedingung(en):** | | |
| Änderungen wurden gesichert. | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der Administrator ein neues Thema hinzufügen oder ein bestehendes ändern möchte. 2. Der Administrator muss sich authentifizieren. 3. Der Administrator legt die Eigenschaften des Themas fest. (Name, Tags (z.B. Fraktal, Affine Abbildung, Mandelbrot), Bilder, …) 4. Das System formatiert und speichert das Thema in einer Datei. 5. Das System liest erneut alle Themen ein. 6. Der Anwendungsfall endet. | |
| **Ablauf-Varianten:** | | |
| 1a | Der Administrator möchte ein bestehendes Thema ändern. | |
|  | 1. Laden der Eigenschaften.  2. Weiter mit 2 im normalen Ablauf. | |
| 3a | Der Administrator möchte das Thema löschen. | |
|  | 1. Das System prüft, ob noch eine Aufgabe mit dem Thema verknüpft ist.  2. Wenn keine Aufgabe mit dem Thema verknüpft ist, wird die Datei gelöscht und mit Schritt 4 im normalen Ablauf fortgefahren.  3. Wenn noch eine Aufgabe mit dem Thema verknüpft ist, wird eine Fehlermeldung ausgegeben und mit Schritt 3 im normalen Ablauf fortgefahren. | |
| 3a | Das System kann nicht speichern, da z.B. kein Schreibzugriff. | |
|  | 1. Das System signalisiert dies mit einer Fehlermeldung.  2. Weiter mit Schritt 3 im normalen Ablauf. | |
| **Spezielle Anforderungen:** | | |
| Der Administrator kann jederzeit abbrechen. | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Fraktal rendern mit Chaosspiel** | | |
| **Kennung** | | UC-R1 |
| **Priorität** | | 9 |
| **Kurzbeschreibung:** | | |
| Dem Benutzer soll ein Fraktal mit dem Algorithmus "Chaosspiel" rendern können. | | |
| **Vorbedingung(en):** | | |
| * Ein IFS ist definiert. | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der Benutzer signalisiert, dass er ein Fraktal mit dem Algorithmus "Chaosspiel" rendern möchte 2. Das System stellt eine Plattform zur Verfügung auf der dem Benutzer  * ein Bereich zur Anzeige des IFS, * eine Interaktionsmöglichkeit zur Eingabe der Wahrscheinlichkeit mit der die Funktionen des IFS bei der Berechnung des Fraktals benutzt werden, * eine Interaktionsmöglichkeit zur Eingabe der Anzahl der durchzuführenden Iterationen, * ein Bereich zur Anzeige des Fraktals, * eine Interaktionsmöglichkeit zum signalisieren dass das Fraktal gespeichert werden soll, * eine Interaktionsmöglichkeit zum Einfügen von Kommentaren/Notizen,   dargestellt werden.   1. Das System lädt das IFS und zeigt dieses im vorgesehenen Bereich an. 2. Der Benutzer gibt die Wahrscheinlichkeit jeder einzelnen Funktion im IFS ein. 3. Das System prüft ob die Wahrscheinlichkeiten ordnungsgemäß eingegeben sind. 4. Der Benutzer wählt die Anzahl der durchzuführenden Iterationen. 5. Das System berechnet mittels des IFS und der Anzahl der Iterationen das Fraktal und zeigt dieses im vorgesehenen Bereich an. 6. Der Anwendungsfall endet. | |
| **Ablauf-Varianten:** | | |
| 5a | Die Summe der Wahrscheinlichkeiten ist ungleich 1. | |
|  | 1. Das System zeigt einen entsprechenden Hinweis an. 2. Weiter mit Schritt 4 im normalen Ablauf. | |
| **Spezielle Anforderungen:** | | |
| * Der Benutzer kann jederzeit Kommentare/Notizen im vorgesehenen Bereich einfügen. * Der Benutzer kann die Übung jederzeit über die bereitgestellte Interaktionsmöglichkeit abbrechen. * Der Benutzer kann die Werte der Wahrscheinlichkeiten und die Anzahl der Iterationen ändern. | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Fraktal rendern mit Rekursion** | | |
| **Kennung** | | R2 |
| **Priorität** | | 9 |
| **Kurzbeschreibung:** | | |
| Der Benutzer soll ein Fraktal mit dem Algorithmus "Rekursion" rendern können. | | |
| **Vorbedingung(en):** | | |
| * Ein IFS ist definiert. | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der Benutzer signalisiert, dass er ein Fraktal mit dem Algorithmus "Rekursion" rendern möchte 2. Der Benutzer wählt die Form des Grundlage-Fraktals. 3. Der Benutzer wählt die Anzahl der durchzuführenden Iterationen. 4. Das System lädt das IFS und zeigt dieses im vorgesehenen Bereich an. 5. Das System berechnet mittels der IFS und dem Grundlagen-Fraktal mit der Anzahl der Iterationen das Fraktal und zeigt dieses im vorgesehenen Bereich an. 6. Der Anwendungsfall endet. | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
| * Der Benutzer kann jederzeit Kommentare/Notizen im vorgesehenen Bereich einfügen. * Der Benutzer kann die Übung jederzeit über die bereitgestellte Interaktionsmöglichkeit abbrechen. * Der Benutzer kann die Form des Grundlage-Fraktals und die Anzahl der Iterationen ändern. | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Zuordnungsvorschrift einer linearen Abbildung studieren** | | |
| **Kennung** | | T-1 |
| **Priorität** | | 9 |
| **Kurzbeschreibung:** | | |
| Dem Benutzer soll die Berechnung und Zuordnungsvorschrift einer lin. Abbildung näher gebracht werden. (Siehe unter GUI im Dokument "Zuordnungsvorschrift einer lin. Abb. Verstehen.pdf") | | |
| **Vorbedingung(en):** | | |
| Die Übung "Zuordnungsvorschrift einer linearen Abbildung studieren" wurde ausgewählt und der Anwendungsfall „Übungsthema auswählen“ ist abgeschlossen. | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt 2. Der Benutzer gibt im Eingabekoordinatensystem den Punkt x durch einen Punkt außerhalb des Ursprungs an. 3. Das System zeigt im Eingabekoordinatensystem  * einen Ursprungsvektor x🡪 zum eingegebenen Punkt x, * die beiden Einheitsvektoren e1🡪 und e2🡪, * die Funktion des Vektors x🡪 in Vektorschreibweise und in Abhängigkeit der Einheitsvektoren e1🡪 und e2🡪,   an.   1. Der Benutzer definiert durch die Eingabe der Punkte m1 und m2 die Vektoren m1🡪 und m2🡪 im Ausgabekoordinatensystem. 2. Das System prüft ob die beiden Punkte auf einer Ursprungsgeraden sind. 3. Das System zeigt im Ausgabekoordinatensystem die Vektoren m1🡪 und m2🡪 an. 4. Das System berechnet Anhand der Vektoren m1🡪, m2🡪 die Werte der Funktion der Transformation f(x) und zeigt diese an. 5. Das System berechnet anhand der Funktion der Transformation f(x) den Vektor f(x🡪) und zeigt diesen sowie dessen Funktion im Ausgabekoordinatensystem an. 6. Der Anwendungsfall endet. | |
| **Ablauf-Varianten:** | | |
| 5a | Die beiden Punkte m1 und m2 liegen auf einer Ursprungsgeraden | |
|  | 1. Das System meldet dass die beiden Punkte auf einer Ursprungsgeraden sind. 2. Das System entfernt die beiden Punkte 3. Weiter mit Schritt 4 im normalen Ablauf | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
| * Der Benutzer kann jederzeit Kommentare/Notizen im vorgesehenen Bereich einfügen. * Der Benutzer kann die Determinante der Matrix der Funktion anzeigen lassen. * Eingegebene Vektoren lassen sich ändern | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Linearen Abbildung aus der Summe beliebiger Vektoren studieren** | | |
| **Kennung** | | T-2 |
| **Priorität** | | 9 |
| **Kurzbeschreibung:** | | |
| Dem Benutzer soll die Berechnung und Zuordnungsvorschrift einer Summe beliebiger Vektoren näher gebracht werden. (Siehe unter GUI im Dokument "Verträglichkeit mit Summen von Vektoren.pdf") | | |
| **Vorbedingung(en):** | | |
| Die Übung "Zuordnungsvorschrift einer linearen Abbildung studieren" wurde gewählt und der Anwendungsfall "Übungsthema auswählen" ist abgeschlossen. | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt. 2. Der Benutzer gibt im Eingabekoordinatensystem die Punkte x und y durch Punkte außerhalb des Ursprungs an. 3. Das System prüft ob die beiden Punkte auf einer Ursprungsgeraden sind. 4. Das System zeigt im Eingabekoordinatensystem  * einen Ursprungsvektor x🡪 zum eingegebenen Punkt x, * den Vektor y🡪 von Punkt x zu Punkt y, * einen Ursprungsvektor x+y🡪 zum eingegebenen Punkt y,   an.   1. Der Benutzer gibt die Werte der Matrix in der Funktion der Transformation ein. 2. Das System prüft die Determinante ob die Werte der Matrix gültig sind. 3. Das System berechnet anhand der Funktion der Transformation f(x+y) die Vektoren f(x🡪), f(y🡪), f(x+y🡪) und zeigt diese im Ausgabekoordinatensystem an. 4. Der Anwendungsfall endet. | |
| **Ablauf-Varianten:** | | |
| 3a | Die beiden Punkte liegen auf einer Ursprungsgeraden. | |
|  | 1. Das System teilt dies dem Benutzer mit. 2. Das System entfernt die definierten Punkte. 3. Weiter mit Schritt 2 im normalen Ablauf. | |
| 6a | Die Determinante ist gleich 0. | |
|  | 1. Das System gibt eine Meldung aus dass die Werte der Matrix ungültig sind. 2. Das System entfernt alle Werte aus der Matrix. 3. Weiter mit Schritt 5 im normalen Ablauf. | |
| **Spezielle Anforderungen:** | | |
| * Der Benutzer kann jederzeit Kommentare/Notizen im vorgesehenen Bereich einfügen. * Der Benutzer kann sich die Determinante anzeigen lassen. | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Linearen Abbildung mit vielfachen beliebiger Vektoren studieren** | | |
| **Kennung** | | T-3 |
| **Priorität** | | 9 |
| **Kurzbeschreibung:** | | |
| Dem Benutzer soll die Berechnung und Zuordnungsvorschrift des vielfachen eines beliebigen Vektors näher gebracht werden. (Siehe unter GUI im Dokument "Verträglichkeit mit Vielfachen von Vektoren.pdf") | | |
| **Vorbedingung(en):** | | |
| Die Übung "Linearen Abbildung mit vielfachen beliebiger Vektoren studieren" wurde gewählt und der Anwendungsfall "Übungsthema auswählen" ist abgeschlossen. | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt. 2. Der Benutzer gibt im Eingabekoordinatensystem den Punkt x durch einen Punkt außerhalb des Ursprungs, sowie den Faktor a mit dem der Vektor multipliziert wird ein. 3. Das System zeigt im Eingabekoordinatensystem den Ursprungsvektor x🡪 zum eingegebenen Punkt x und den Ursprungsvektor a\*y🡪 an. 4. Der Benutzer gibt die Werte der Matrix in der Funktion der Transformation f(a\*x) ein. 5. Das System prüft die Determinante ob die Werte der Matrix gültig sind. 6. Das System berechnet anhand der Funktion der Transformation f(a\*x) die Vektoren f(x🡪), f(a\*x🡪) und zeigt diese im Ausgabekoordinatensystem an. 7. Der Anwendungsfall endet. | |
| **Ablauf-Varianten:** | | |
| 5a | Die Determinante = 0. | |
|  | 1. Das System gibt eine Meldung aus dass die Werte der Matrix ungültig sind. 2. Das System entfernt alle Werte aus der Matrix. 3. Weiter mit Schritt 4 im normalen Ablauf. | |
| **Spezielle Anforderungen:** | | |
| * Der Benutzer kann jederzeit Kommentare/Notizen im vorgesehenen Bereich einfügen. * Der Benutzer kann sich die Determinante anzeigen lassen. | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Hintereinanderausführung linearer Abbildungen studieren** | | |
| **Kennung** | | T-4 |
| **Priorität** | | 9 |
| **Kurzbeschreibung:** | | |
| Dem Benutzer soll die Berechnung und Zuordnungsvorschrift der Hintereinanderausführung von Transformationen eines beliebigen Vektors näher gebracht werden. (Siehe unter GUI im Dokument "Hintereinanderausführung.pdf") | | |
| **Vorbedingung(en):** | | |
| Die Übung "Hintereinanderausführung linearer Abbildungen studieren" wurde gewählt und der Anwendungsfall "Übungsthema auswählen" ist abgeschlossen. | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt. 2. Der Benutzer gibt im Eingabekoordinatensystem den Punkt x durch einen Punkt außerhalb des Ursprungs ein. 3. Das System zeigt im Eingabekoordinatensystem den Ursprungsvektor x🡪 zum eingegebenen Punkt x an. 4. Der Benutzer gibt die Werte der Matrix in der ersten Funktion der Transformation f(x) ein. 5. Das System prüft die Determinante, also ob die Werte der Matrix gültig sind. 6. Das System berechnet anhand der Funktion der Transformation f(x) den Vektor f(x🡪) und zeigt diese im ersten Ausgabekoordinatensystem an. 7. Der Benutzer gibt die Werte der Matrix in der zweiten Funktion der Transformation g(x) ein. 8. Das System prüft die Determinante, also ob die Werte der Matrix gültig sind. 9. Das System  * berechnet anhand der zweiten Funktion der Transformation g(x) den Vektor g(f(x🡪)), * zeigt den Vektor g(f(x🡪)) im zweiten Ausgabekoordinatensystem an. * berechnet anhand der zwei Funktionen die resultierende Funktion der Transformation g(f(x)), * zeigt die resultierende Funktion g(f(x)) der Transformation an.  1. Der Anwendungsfall endet. | |
| **Ablauf-Varianten:** | | |
| 5a | Die Determinante = 0. | |
|  | 1. Das System gibt eine Meldung aus dass die Werte der Matrix ungültig sind. 2. Das System entfernt alle Werte aus der Matrix. 3. Weiter mit Schritt 4 im normalen Ablauf. | |
| 8a | Die Determinante = 0. | |
|  | 1. Das System gibt eine Meldung aus dass die Werte der Matrix ungültig sind. 2. Das System entfernt alle Werte aus der Matrix. 3. Weiter mit Schritt 7 im normalen Ablauf. | |
| **Spezielle Anforderungen:** | | |
| * Der Benutzer kann jederzeit Kommentare/Notizen im vorgesehenen Bereich einfügen. * Der Benutzer kann sich die Determinanten der Matrizen anzeigen lassen. * Der Benutzer kann die Werte der Vektoren und Matrizen ändern. | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Umkehrung linearer Abbildungen studieren** | | |
| **Kennung** | | T-5 |
| **Priorität** | | 9 |
| **Kurzbeschreibung:** | | |
| Dem Benutzer soll die Berechnung der Umkehrfunktion eines beliebigen Vektors näher gebracht werden. (Siehe unter GUI im Dokument "Umkehrung.pdf") | | |
| **Vorbedingung(en):** | | |
| Die Übung "Umkehrung linearer Abbildungen studieren" wurde gewählt und der Anwendungsfall "Übungsthema auswählen" ist abgeschlossen. | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt. 2. Der Benutzer gibt im Eingabekoordinatensystem den Punkt x durch einen Punkt außerhalb des Ursprungs ein. 3. Das System zeigt im Eingabekoordinatensystem den Ursprungsvektor x🡪 zum eingegebenen Punkt x an. 4. Der Benutzer gibt die Werte der Matrix in der ersten Funktion der Transformation f(x) ein. 5. Das System prüft die Determinante ob die Werte der Matrix gültig sind. 6. Das System berechnet anhand der Funktion der Transformation den Vektor f(x🡪) und zeigt diesen im Ausgabekoordinatensystem an. 7. Das System berechnet die inverse Transformationsgleichung f-1(f(x)) und zeigt diese als zweite an. 8. Der Benutzer gibt im Ausgabekoordinatensystem den Punkt y durch einen Punkt außerhalb des Ursprungs ein. 9. Das System zeigt im Ausgabekoordinatensystem den Ursprungsvektor y🡪 zum eingegebenen Punkt y an. 10. Das System  * berechnet anhand der zweiten Funktion der Transformation f-1(f(x)) * die dritte Funktion der Transformation f-1(f(y)) * berechnet anhand der dritten Funktion der Transformation f-1(f(y)) den Vektor f-1(f(y🡪)), * zeigt den Vektor f-1(f(y🡪)) im Ausgabekoordinatensystem an. * berechnet anhand der dritten Funktionen die resultierende Funktion f(f(y)), * zeigt die resultierende vierte Funktion der Transformation f(f(y) an.  1. Der Anwendungsfall endet. | |
| **Ablauf-Varianten:** | | |
| 5a | Die Determinante = 0. | |
|  | 1. Das System gibt eine Meldung aus dass die Werte der Matrix ungültig sind. 2. Das System entfernt alle Werte aus der Matrix. 3. Weiter mit Schritt 4 im normalen Ablauf. | |
| **Spezielle Anforderungen:** | | |
| * Der Benutzer kann jederzeit Kommentare/Notizen im vorgesehenen Bereich einfügen. * Der Benutzer kann sich die Determinante anzeigen lassen. * Der Benutzer kann die Werte der Vektoren und der Matrix ändern. | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Drehung einer linearen Abbildung um den Ursprung studieren** | | |
| **Kennung** | | T-6 |
| **Priorität** | | 9 |
| **Kurzbeschreibung:** | | |
| Dem Benutzer soll die Berechnung einer Drehung einer lin. Abbildung näher gebracht werden. (Siehe unter GUI im Dokument "Drehung, Spiegelung, Streckung.pdf") | | |
| **Vorbedingung(en):** | | |
| Die Übung "Drehung einer linearen Abbildung um den Ursprung studieren" wurde gewählt und der Anwendungsfall "Übungsthema auswählen" ist abgeschlossen. | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt. 2. Der Benutzer gibt im Eingabekoordinatensystem den Punkt x durch einen Punkt außerhalb des Ursprungs an. 3. Das System zeigt im Eingabekoordinatensystem  * einen Ursprungsvektor x🡪 zum eingegebenen Punkt x, * die beiden Einheitsvektoren e1🡪 und e2🡪, * die Funktion des Vektors x🡪 in Vektorschreibweise,   an.   1. Der Benutzer gibt den Winkel ein. 2. Das System berechnet  * anhand des Winkels die Werte der Funktion der Transformation f(x), * anhand der Funktion der Transformation f(x) die Vektoren f(x🡪), m1🡪, m2🡪.  1. Das System zeigt  * den Winkel im Ausgabekoordinatensystem, * die Werte der Funktion der Transformation f(x), * die Vektoren f(x🡪), m1🡪, m2🡪 im Ausgabekoordinatensystem, * die Funktion des Vektors f(x)🡪 im Ausgabekoordinatensystem an.  1. Der Anwendungsfall endet. | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
| * Der Benutzer kann jederzeit Kommentare/Notizen im vorgesehenen Bereich einfügen. * Der Benutzer kann sich die Determinante anzeigen lassen. * Der Benutzer kann die Werte der Vektoren und des Winkels ändern. | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Spiegelung einer linearen Abbildung an einer Ursprungsgeraden studieren** | | |
| **Kennung** | | T-7 |
| **Priorität** | | 9 |
| **Kurzbeschreibung:** | | |
| Dem Benutzer soll die Berechnung einer Spiegelung einer lin. Abbildung näher gebracht werden. (Siehe unter GUI im Dokument "Spiegelungen an Gr. durch den Urspr. sind linear.pdf") | | |
| **Vorbedingung(en):** | | |
| Die Übung "Spiegelung einer linearen Abbildung an einer Ursprungsgeraden studieren" wurde gewählt und der Anwendungsfall "Übungsthema auswählen" ist abgeschlossen. | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt. 2. Der Benutzer gibt im Eingabekoordinatensystem den Punkt x durch einen Punkt außerhalb des Ursprungs an. 3. Das System zeigt im Eingabekoordinatensystem  * einen Ursprungsvektor x🡪 zum eingegebenen Punkt x, * die beiden Einheitsvektoren e1🡪 und e2🡪, * die Funktion des Vektors x🡪 in Vektorschreibweise,   an.   1. Der Benutzer gibt den Winkel ein. 2. Das System zeigt im Eingabekoordinatensystem einen Ursprungsvektor g🡪 mit dem eingegebenen Winkel und den Winkel an. 3. Das System berechnet  * Anhand des Winkels die Werte der Funktion der Transformation f(x), * anhand der Funktion der Transformation f(x) die Vektoren f(x🡪), m1🡪, m2🡪.  1. Das System zeigt  * die Werte der Funktion der Transformation f(x), * die Vektoren g🡪, f(x🡪), m1🡪, m2🡪 im Ausgabekoordinatensystem, * die Funktion des Vektors f(x🡪) im Ausgabekoordinatensystem an.  1. Der Anwendungsfall endet. | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
| * Der Benutzer kann jederzeit Kommentare/Notizen im vorgesehenen Bereich einfügen. * Der Benutzer kann sich die Determinante anzeigen lassen. * Der Benutzer kann die Werte des Vektors und des Winkels ändern. | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Zuordnungsvorschrift einer affinen Abbildungen studieren** | | |
| **Kennung** | | T-8 |
| **Priorität** | | 9 |
| **Kurzbeschreibung:** | | |
| Dem Benutzer soll die Berechnung und Zuordnungsvorschrift einer beiliegen affinen Abbildung näher gebracht werden. (Siehe unter GUI im Dokument "Zuordnungsvorschifft einer affinen Abb. verstehen.pdf") | | |
| **Vorbedingung(en):** | | |
| Die Übung "Zuordnungsvorschrift einer affinen Abbildungen studieren" wurde ausgewählt und der Anwendungsfall „Übungsthema auswählen“ ist abgeschlossen. | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt 2. Der Benutzer gibt im Eingabekoordinatensystem die Punkte a, b, c durch Punkte außerhalb des Ursprungs ein. 3. Das System prüft ob die drei Punkte auf einer Ursprungsgeraden sind. 4. Das System zeigt im Eingabekoordinatensystem  * den Ursprungsvektor a🡪 zum eingegebenen Punkt a, * den Vektor ab🡪, * den Vektor ac🡪, * den Vektor bc🡪,   an.   1. Das System zeigt im ersten Ausgabekoordinatensystem die beiden Einheitsvektoren e1🡪 und e2🡪. 2. Das System berechnet anhand der Vektoren ab🡪, ac🡪, e1🡪, e2🡪 die Funktion der Transformation f1(x) sowie die Funktion der inversen Transformation f1-1(x) und zeigt beide an. 3. Der Benutzer gibt im zweiten Ausgabekoordinatensystem die Punkte a', b', c' durch Punkte außerhalb des Ursprungs ein. 4. Das System prüft ob die drei Punkte auf einer Ursprungsgeraden sind. 5. Das System zeigt im Ausgabekoordinatensystem  * den Ursprungsvektor a'🡪 zum eingegebenen Punkt a', * den Vektor a'b'🡪, * den Vektor a'c'🡪, * den Vektor b'c'🡪,   an.   1. Das System berechnet anhand der Vektoren e1🡪, e2🡪, a'b'🡪, a'c'🡪 die dritte Funktion der Transformation f2(x) und zeigt diese an. 2. Das System berechnet anhand der Funktionen f1(x) und f2(x) die resultierende Funktion der Transformation f(x) und zeigt diese an. 3. Der Anwendungsfall endet. | |
| **Ablauf-Varianten:** | | |
| 3a | Alle drei Punkte sind auf einer Ursprungsgeraden | |
|  | 1. Das System meldet dass die drei Punkte auf einer Ursprungsgeraden sind. 2. Das System entfernt die drei definierten Punkte 3. Weiter mit Schritt 2 im normalen Ablauf. | |
| 8a | Alle drei Punkte sind auf einer Ursprungsgeraden | |
|  | 1. Das System meldet dass die drei Punkte auf einer Ursprungsgeraden sind.  2. Das System entfernt die drei definierten Punkte  3. Weiter mit Schritt 7 im normalen Ablauf. | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
| * Der Benutzer kann jederzeit Kommentare/Notizen im vorgesehenen Bereich einfügen. * Der Benutzer kann die Determinante der Matrix der Funktion anzeigen lassen. * Eingegebene Vektoren bzw. Punkte lassen sich ändern | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Fraktalerzeugung mit IFS studieren** | | |
| **Kennung** | | T-9 (Thema 9) |
| **Priorität** | | 9 |
| **Kurzbeschreibung:** | | |
| Dem Benutzer soll die Erzeugung eines Fraktals mittels IFS näher gebracht werden. (Siehe unter GUI im Dokument "Fraktalerzeugung mit einem IFS.pdf") | | |
| **Vorbedingung(en):** | | |
| * Der Anwendungsfall "Thema auswählen" ist abgeschlossen. * Ein IFS ist definiert. | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn das System die Übung zum Thema "Fraktalerzeugung mit einem IFS studieren" startet. 2. Das System stellt, im vom Anwendungsfall "Thema auswählen" vorgesehenem Bereich, eine Plattform zur Verfügung auf der dem Benutzer  * Eine Interaktionsmöglichkeit zur Auswahl der Form des Fraktals, * zwei Bereiche zur Anzeige der aktuellen Iteration, * ein Bereich zur Anzeige des IFS, * Eine Interaktionsmöglichkeit zum signalisieren dass ein weiterer Schritt durchgeführt werden soll,   dargestellt werden.   1. Der Benutzer wählt die Form des Grundlage-Fraktals. 2. Das System läd das IFS und zeigt dieses im vorgesehenen Bereich an. 3. Das System zeigt das ausgewählte Grundlage-Fraktal im ersten Anzeigebereich an. 4. Das System berechnet mittels einer IFS-Funktion und dem Grundlagen-Fraktal den nächsten Teilschritt des zu erstellenden Fraktals und aktualisiert dessen Bereich im zweiten Anzeigebereich. 5. Der Benutzer kann  * die Übung über die vom Anwendungsfall "Thema auswählen" bereitgestellte Interaktionsmöglichkeit speichern, * dem System signalisieren dass der nächste Schritt ausgeführt werden soll.  1. Der Anwendungsfall endet. | |
| **Ablauf-Varianten:** | | |
| 7.a | Der Benutzer signalisiert dass er speichern möchte. | |
|  | 1. Der Anwendungsfall "Übung eines Arbeitsheft speichern" wird abgearbeitet. 2. Weiter mit Schritt 7 im normalen Ablauf. | |
| 7.b | Der Benutzer signalisiert dass der nächste Schritt ausgeführt werden soll.  Bei der aktuellen Iteration sind noch nicht alle Funktionen der IFS durchlaufen worden. | |
|  | 1. Weiter mit Schritt 6 im normalen Ablauf. | |
| 7.c | Der Benutzer signalisiert dass der nächste Schritt ausgeführt werden soll.  Bei der aktuellen Iteration sind alle Funktionen der IFS durchlaufen. | |
|  | 1. Das erzeugte Fraktal wird zum Grundlage-Fraktal. 2. Weiter mit Schritt 5 im normalen Ablauf. | |
| **Spezielle Anforderungen:** | | |
| * Der Benutzer kann jederzeit Kommentare/Notizen im vorgesehenen Bereich einfügen. * Der Benutzer kann die Übung jederzeit über die vom Anwendungsfall "Thema auswählen" bereitgestellte Interaktionsmöglichkeit abbrechen. | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Tutorial anzeigen** | | |
| **Kennung** | | UC-T10 |
| **Priorität** | | 1 |
| **Kurzbeschreibung:** | | |
| Der Use-Case soll dem Benutzer eine Einführung in das Programm bieten. | | |
| **Vorbedingung(en):** | | |
|  | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Use-Case beginnt, wenn der Benutzer das Tutorial anfordert. 2. Der Benutzer wählt einThemengebiet aus. 3. Das System startet das Tutorial zum jeweiligen Thema. 4. Der Use-Case endet, wenn der Benutzer das Tutorial beendet. | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
| Der Benutzer kann jederzeit abbrechen. | | |
| **Zu klärende Punkte:** | | |
| Wo kann das Tutorial aufgerufen werden? In Mindmap oder in der Übung? | | |

|  |  |  |
| --- | --- | --- |
| **Übungsthema auswählen** | | |
| **Kennung** | | T-11 |
| **Priorität** | | 10 |
| **Kurzbeschreibung:** | | |
| Ein Übungsthema wird ausgewählt, um eine neue Übung zu Beginnen. | | |
| **Vorbedingung(en):** | | |
| Ein Arbeitsheft muss geöffnet sein! | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der Benutzer dem System signalisiert, dass er eine neue Übung starten möchte. 2. Das System stellt eine Plattform zur Verfügung auf der dem Benutzer   - der Name des ausgewählten Themas,  - das Lernziel des ausgewählten Themas,  - Fragen zur Bearbeitung zum ausgewählten Thema,  - ein Bereich zur Bearbeitung des ausgewählten Themas,  - ein Bereich zum Einfügen von Kommentaren/Notizen zur bearbeiteten Übung,  - eine Interaktionsmöglichkeit zum signalisieren das gespeichert werden soll,  - eine Interaktionsmöglichkeit zum signalisieren das abgebrochen werden soll,  dargestellt werden.   1. Das System startet den Anwendungsfall der dem ausgewählten Thema entspricht und stellt diesen im Bereich der zur Bearbeitung vorgesehenen ist dar. 2. Der Anwendungsfall endet. | |
| **Ablauf-Varianten:** | | |
|  | | |
| **Spezielle Anforderungen:** | | |
|  | | |
| **Zu klärende Punkte:** | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Übung aus Arbeitsheft löschen** | | |
| **Kennung** | | UC-U1 |
| **Priorität** | | 7 |
| **Kurzbeschreibung:** | | |
| Im Arbeitsheft wird eine Übung gewählt, um sie zu Löschen | | |
| **Vorbedingung(en):** | | |
| Der Heftinhalt wird angezeigt | | |
| **Nachbedingung(en):** | | |
| Der Heftinhalt bleibt angezeigt – wobei die gewählte Übung entfernt wurde | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Use-Case beginnt, wenn der Benutzer signalisiert, dass er eine der Übungen aus dem Arbeitsheft löschen will 2. Der Benutzer wählt die Übung, die er gelöscht haben möchte 3. Das System löscht die ausgewählte Übung 4. Der Use-Case endet | |
| **Ablauf-Varianten:** | | |
| 5a | Die letzte Übung wurde entfernt | |
|  | 1. Weiter mit Schritt 6 | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
| - der Benutzer hat vor Schritt 2 die Möglichkeit abzubrechen | | |
| **Zu klärende Punkte:** | | |
|  | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Übung des Arbeitsheft laden** | | |
| **Kennung** | | UC-U2 |
| **Priorität** | | 8 |
| **Kurzbeschreibung:** | | |
| Im Arbeitsheft wird eine Übung gewählt, um sie zu Laden | | |
| **Vorbedingung(en):** | | |
| Der Heftinhalt wird angezeigt | | |
| **Nachbedingung(en):** | | |
| Dem Benutzer wird die gewählte Übung angezeigt und er kann daran Änderungen vornehmen | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Use-Case beginnt, wenn der Benutzer signalisiert, dass er eine Übung aus dem Arbeitsheft laden will 2. Der Benutzer wählt die zu ladende Übung aus 3. Das System wechselt in die ausgewählte Übung 4. Der Use-Case endet | |
| **Ablauf-Varianten:** | | |
|  |  | |
|  |  | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
| - der Benutzer hat die Möglichkeit jederzeit | | |
| **Zu klärende Punkte:** | | |
| * statt in die ausgewählte Übung zu Wechseln, wäre eine Alternative, dass die Übung in der Heftansicht bearbeitbar wird; vermutlich wäre das bei der Implementierung schwieriger | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Übung eines Arbeitsheft speichern** | | |
| **Kennung** | | UC-U3 |
| **Priorität** | | 9 |
| **Kurzbeschreibung:** | | |
| Eine geöffnete Übung wird gespeichert | | |
| **Vorbedingung(en):** | | |
| Die jeweilige Übung ist geöffnet | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Use-Case beginnt, wenn der Benutzer signalisiert dass er die aktuell geöffnete Übung speichern will 2. Das System speichert die geöffnete Übung 3. Das System aktualisiert die Ansicht des Arbeitsheft 4. Der Use-Case endet | |
| **Ablauf-Varianten:** | | |
| 2a | Das Speichern schlägt fehl | |
|  | 1. Das System stellt fest, dass die Übung nicht gespeichert werden kann 2. Das System weist den Benutzer darauf hin, dass das Speichern fehlschlug 3. Weiter mit Schritt 4 | |
| 3a | Das Arbeitsheft wird nicht angezeigt | |
|  | 1. Weiter mit Schritt 4 | |
| **Spezielle Anforderungen:** | | |
| * Das Speichern der Übung soll nur dann fehlschlagen können, wenn Zugriffsprobleme oder ein Speicherplatzmangel am Speicherort vorliegt | | |
| **Zu klärende Punkte:** | | |
|  | | |
|  | | |

|  |  |  |
| --- | --- | --- |
| **Iteriertes Funktionensystem definieren** | | |
| **Kennung** | | UC-Z1 |
| **Priorität** | | 9 |
| **Kurzbeschreibung:** | | |
| Ein Iteriertes Funktionensystem (IFS) wird durch mehrere Abbildungen definiert. | | |
| **Vorbedingung(en):** | | |
| - die Übung „IFS definieren“ wurde geöffnet | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Der Use-Case beginnt. 2. Das System gibt dem Benutzer die Möglichkeit eine neue Abbildung hinzuzufügen; dabei stehen mindestens folgende verschiedene Modi zur Verfügung:   - affine Abbildung hinzufügen und diese numerisch definieren  - affine Abbildung hinzufügen und diese grafisch definieren  - (mögliche Weitere zu klären)   1. Der Benutzer wählt aus den Modi, eine Möglichkeit aus. 2. Der jeweilige Use-Case um die Abbildung zu definieren wird ausgeführt. 3. Das System fügt die Werte der gerade definierten Abbildung der Anzeige hinzu. 4. Das System gibt dem Benutzer die Möglichkeit die Definition des IFS abzuschließen 5. Der Benutzer verdeutlicht, ob er entweder weitere Abbildungen hinzufügen oder die Definition des IFS abschließen möchte 6. Der Benutzer signalisiert, dass er die Definition des IFS abschließen möchte 7. Das System speichert die Werte des IFS 8. Der Use-Case endet | |
| **Ablauf-Varianten:** | | |
| 8a | Der Benutzer möchte weitere Abbildungen hinzufügen | |
|  | 1. Der Benutzer signalisiert, dass er eine weitere Transformation hinzufügen möchte  2. Weiter mit Schritt 2 im normalen Ablauf. | |
| **Spezielle Anforderungen:** | | |
| - Der Benutzer hat die Möglichkeit hinzugefügte Transformationen, einzeln auszuwählen und zu Entfernen. | | |
| **Zu klärende Punkte:** | | |
| - soll es eine Obergrenze für die maximale Anzahl von Abbildungen geben  - sollen mehrere verschiedene IFS definiert werden können, aus denen man dann in „IFS studieren“ eine auswählt? Oder genügt eine einzige IFS? Wenn es mehrere geben soll, wird es vielleicht Sinn machen, dass man einem IFS einen Namen geben kann.  - welche weiteren Abbildungsmodi sind wichtig oder zumindest Nice-To-Have: vielleicht Julia Abbildung, Mandelbrot Abbildung, | | |

|  |  |  |
| --- | --- | --- |
| **Affine Abbildung numerisch definieren** | | |
| **Kennung** | | UC-Z2 |
| **Priorität** | | 9 |
| **Kurzbeschreibung:** | | |
| Eine affine Abbildung wird durch Eingabe einer Matrix und eines Verschiebevektors definiert. | | |
| **Vorbedingung(en):** | | |
| - die Übung „IFS definieren“ ist geöffnet | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Use-Case beginnt, wenn der Benutzer dem System signalisiert, dass er eine affine Abbildung numerisch definieren möchte. 2. Der Benutzer gibt eine Matrix und einen Verschiebevektor ein. 3. Das System prüft, ob die Matrix kontrahierend ist. 4. Das System erkennt, dass die Matrix kontrahierend ist. 5. Das System speichert die Werte der affinen Abbildung. 6. Der Use-Case endet. | |
| **Ablauf-Varianten:** | | |
| 4a | Das System erkennt, dass die Matrix nicht kontrahierend ist | |
|  | 1. Das System weist den Benutzer darauf hin, dass die Matrix nicht kontrahierend ist und daher eine andere Matrix angegeben werden muss.  2. Das System löscht die vom Benutzer eingegebenen Werte der Matrix.  3. Der Benutzer gibt eine Matrix ein.  4. weiter mit Schritt 3 | |
|  |  | |
|  |  | |
| **Spezielle Anforderungen:** | | |
| Das System stellt sicher, das keine ungültigen Werte eingegeben werden können.  Der Benutzer kann jederzeit abbrechen. | | |
| **Zu klärende Punkte:** | | |
| - Kann in anderen Stellen als in der Übung „IFS definieren“ dieser Use-Case aufgerufen werden? | | |

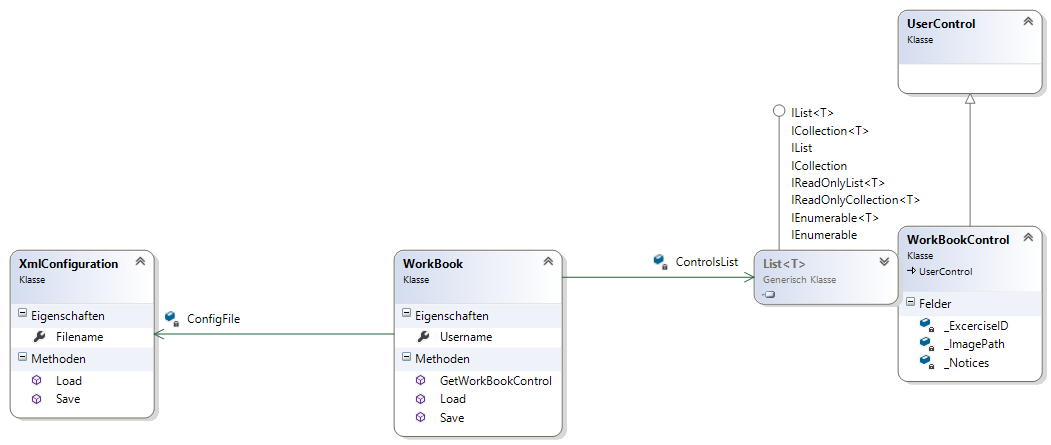
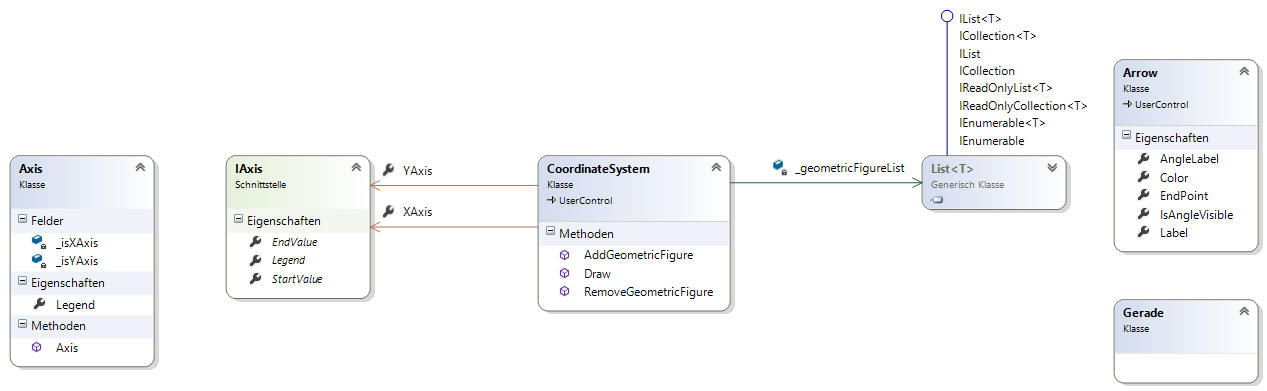
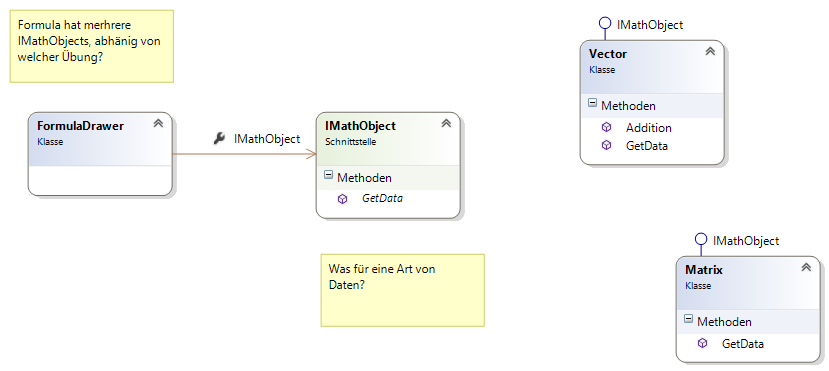
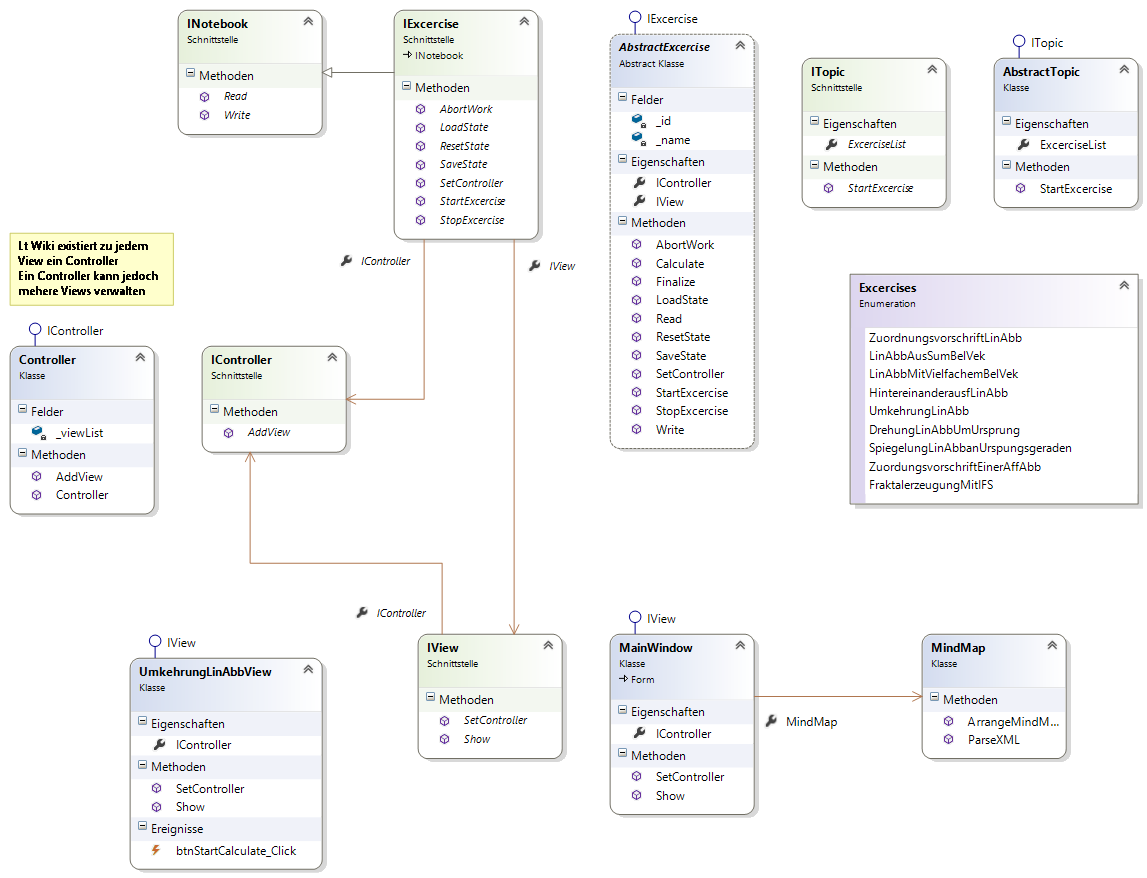
|  |  |  |
| --- | --- | --- |
| **Affine Abbildung durch abklicken definieren** | | |
| **Kennung** | | UC-Z4 |
| **Priorität** | | 9 |
| **Kurzbeschreibung:** | | |
| Dieser Use-Case beschreibt das Definieren einer affinen Abbildung durch abklicken. | | |
| **Vorbedingung(en):** | | |
|  | | |
| **Nachbedingung(en):** | | |
|  | | |
| **Normaler Ablauf:** | | |
|  | 1. Dieser Anwendungsfall beginnt, wenn der Benutzer dem System signalisiert, dass er eine affine Abbildung durch abklicken definieren möchte. 2. Der Benutzer kann ein Bild laden (siehe Use-Case Bild einlesen). 3. Der Benutzer legt die Anforderungen fest (linear oder kontrahierend). 4. Der Benutzer legt durch klicken drei Punkte für das Ausgangsdreieck fest. 5. Das System prüft die Eingabe. 6. Das System zeigt das Ausgangsdreieck an. 7. Der Benutzer legt durch klicken drei Punkte für das Zieldreieck fest. 8. Das System prüft die Eingabe. 9. Das System zeigt das Zieldreieck an. 10. Das System fügt die affine Abbildung dem IFS hinzu. 11. Der Benutzer kann noch ein oder mehrere Zieldreieck/e hinzufügen. 12. Der Anwendungsfall endet. | |
| **Ablauf-Varianten:** | | |
| 5a | Es wurde kein gültiges Dreieck angegeben. | |
|  | 1. Das System signalisiert dies dem Benutzer.  2. Weiter mit Schritt 4 im normalen Ablauf. | |
| 8a | Es wurde kein gültiges Dreieck angegeben. | |
|  | 1. Das System signalisiert dies dem Benutzer.  2. Weiter mit Schritt 7 im normalen Ablauf. | |
| **Spezielle Anforderungen:** | | |
| Der Benutzer kann jederzeit abbrechen.  Der Benutzer kann die Bereiche des Ausgangsdreiecks oder des Zieldreiecks vergrößert darstellen lassen, um die Punkte genauer auswählen zu können. | | |
| **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 Übungs-Editors ist generell nur nach vorheriger Authentifizierung möglich |  |
| FR-003 | Die Darstellungen der Übungen sollen vergrößer- und verkleinerbar sein |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## Modell des Problembereichs (Konzeptionelles Datenmodell)



# 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 für jede Übung/Notiz ein eigenes Fenster zur Verfügung stellen, um dem Benutzer die höchstmögliche Übersicht zu ermöglichen, je nach Monitornutzung |  |
| UR-002 | Das System soll die die allgemein gültigen Benutzereingaben unterstützen (Steuereingaben [tab; enter;…], Hilfstext [F1;…]) |  |
| UR-003 | Das System soll dem Benutzer Rückmeldung geben über Fehlermeldungen („lesbar“), Bearbeitungszustand (Fortschrittsbalken) |  |
| UR-004 | wenn  das System einen längeren Rechenvorgang durchführt, sollte der Benutzer  die Möglichkeit nicht-rechenbezogene Sachen im Arbeitsheft zu ändern und Funktionen außerhalb des Arbeitsheft nutzen können; z.B. Notizen  hinzufügen, Hilfe aufrufen |  |
| UR-005 | Darstellung von Vektoren und (2-dimensionalen) Matrizen innerhalb von Arbeitsblätter wie in Mathematik üblich -> GUI   * evtl. nur Nice-To-Have |  |
| UR-006 | in  Arbeitsblättern Erkennbarkeit für den Benutzer welche Eingaben erfolgen  sollten |  |
| UR-007 | Popup's von Dialogen gering halten |  |
| UR-008 | farbliche  Hervorhebungen oder farbige Schriften von Vektoren - die Ergebnisse  oder Eingabefelder kennzeichen - sollten änderbar sein |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## 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 | System  frei von kritischen Fehlern (keine Speicherzugriffsfehler, Abstürze...)  -> Das erledigt sich hoffentlich durch C# selbst |  |
| RR-002 | es  sollen vom Benutzer keine Eingaben durchgeführt werden können, durch  die das System dann längere Zeit nicht mehr reagiert -> Hier wieder  das Abbrechen von Berechnungen das Stichwort.   * vgl. hier mit Leistung (Performance) Anforderungen * relevant  für Fraktalberechnungen die einige Iterationen benutzen -> Muessen  Fraktale berechnet werden oder verwenden wir vorberechnete Bilder? |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## 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 | Rechenvorgänge der Anwendung sollten i.d.R. höchstens 3 Sekunden betragen -> s.o. (Abbrechbar)   * evtl. Ausnahmen bei Fraktalberechnung möglich (?) |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## 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 | über  Plugins (DLL-Bibliotheken) sollte es möglich sein die möglichen  Aufgabentypen zu erweitern bzw. zu ändern -> Hier muss auf jeden Fall  die Schnittstelle so frueh wie moeglich geklaert werden. Auch die  Interfaces muessen dafuer stehen. |  |
| SR-002 | kompilierbares Programm mit Visual Studio 2012 unter Windows 7, 8 |  |
| SR-003 | kompilierbar mit Mono unter Mac, Linux   * Nice-To-Have |  |
| SR-004 | die  Mind-Map sollte anhand einer Strukturdatei (z.B. XML) aufgebaut werden  -> Wenn die Aufgaben ueber DLLs eingebunden werden dann koennen wir  auch daraus selber eine XML Datei generieren. Sollte man auch klaeren... |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

## 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.

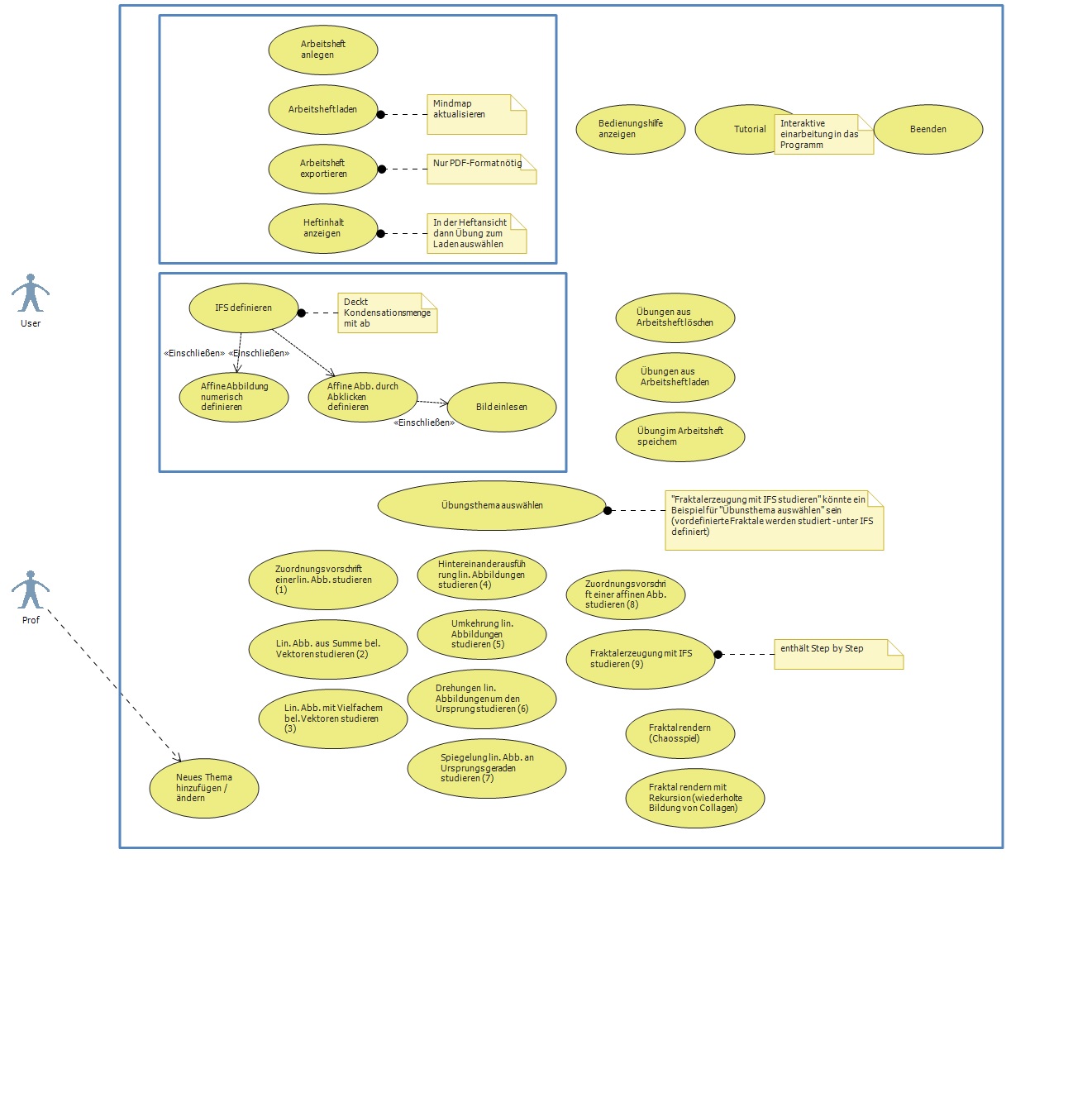
---

# 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.



# Lieferumfang

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

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

# Abnahmekriterien

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

Folgende Testfälle sind für den Abnahme – Test spezifiziert:

|  |  |
| --- | --- |
| Testfall | Datei |
| TestCase\_A1\_Arbeitsheft\_anlegen | \_source\\_analysis\Test Cases\TestCase\_A1\_Arbeitsheft\_anlegen.txt |
| TestCase\_T8\_Zuordnungsvorschrift\_aff\_Abb\_studieren | \_source\\_analysis\Test Cases\TestCase\_T8\_Zuordnungsvorschrift\_aff\_Abb\_studieren.txt |
| TestCase\_Usecase\_T0\_Uebungsthema\_auswaehlen | \_source\\_analysis\Test Cases\TestCase\_T0\_Uebungsthema\_auswaehlen |
| TestCase\_Usecase\_T1\_Zuordnungsvorschrift\_lin\_Abb\_studieren | \_source\\_analysis\Test Cases\TestCase \_T1\_Zuordnungsvorschrift\_lin\_Abb\_studieren |
| TestCase\_Usecase\_U4\_Uebung\_eines\_Arbeitsheft\_speichern | \_source\\_analysis\Test Cases\TestCase \_U4\_Uebung\_eines\_Arbeitsheft\_speichern |
| TestCase\_T9\_Fraktalerzeugung\_mit\_IFS\_studieren | \_source\\_analysis\Test Cases\ TestCase\_T9\_Fraktalerzeugung\_mit\_IFS\_studieren |

# 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)