Fachkonzept zum Programmierprojekt WS12/13

Zusammenstellung

von

Jonne Hass, Benjamin Held, Richard Pump,
Alexander Sirotin, Julian Haack

8. Oktober 2012

Betreuer:
Michael Antemann

Version 1.0

Hochschule Hannover, Abt. Informatik

Inhaltsverzeichnis

1 Einleitung **3**
1.1 Systemzweck 3
1.2 Ziele und Erfolgskriterien des Projekts 3

2 Anforderungen **5**
2.1 Use-Case Diagramme 5
2.2 Use-Case Beschreibungen 5
2.3 Testfälle 5
2.4 Nicht-funktionale Anforderungen 11

3 Domänenmodell **12**

4 GUI-Prototyp **13**

5 Projektplan **14**

Abbildungsverzeichnis **16**

Tabellenverzeichnis **17**

Kapitel 1

Einleitung

1.1 Systemzweck

Der Zweck dieses Projekt ist es im Rahmen der Veranstaltung „Softwareprojekt im 3.Semester“ ein größeres Prorammierprojekt in kleinen Teams durchzuführen und erfolgreich umzusetzen. Aufgabe ist es das klassische Brettspiel „Mühle“ als Java-Applikation umzusetzen. Dabei sollen nicht nur zwei menschliche Spieler gegeneinander antreten können, sondern auch ein Spieler gegen eine künstliche Intelligenz oder zwei vom Computer gesteuerte Spieler gegeneinander spielen. Eine „Musterlösung“ für diese Aufgabe ist dabei nicht vorhanden. Viel mehr sollen die Projektteilnehmer die Aufgabe nach eigenen Vorstellungen planen und umsetzten.
Das Softwareprojekt soll dabei einen Einblick in die spätere Arbeit an größeren Projekten bieten und dafür die Grundlagen bereitstellen, die über das Semester hinüber vermittelt werden sollen.

1.2 Ziele und Erfolgskriterien des Projekts

Die erfolgreiche Bearbeitung des Softwareprojektes ist dabei an mehrere Ziele und Kriterien geknüpft. Wir möchten ein leicht verständliches, gut bedienbares Programm erstellen. Es soll die optionale Möglichkeit haben, nicht gegen menschliche Spieler anzutreten, sondern auch gegen den Computer, der wiederum über verschiedenen Schwierigkeitsstufen verfügen soll.
Das ganze Projekt soll termingerecht zum Ende des Wintersemester 2012/2013 fertig sein, die durch den Dozenten aufgestellten Meilensteine eingehalten werden. Das diese Softwareprojekt im Rahmen des Studiums der angewandten Informatik abläuft steht neben dem Erlernen der Grundfertigkeiten der Projektarbeit auch das Sammeln von Erfahrung, sowie die Erweiterung bereits vorhandener Fertigkeiten im Vordergrund.
Um diese Ziele erfolgreich umzusetzen ist es notwendig, dass alle Teilnehmer gut und effektiv zusammenarbeiten. Eigene Schwächen und Fehler werden dabei sicherlich vorkommen, sollten dann aber eingestanden und bewältigt werden. Das es sich um ein Gemeinschaftprojekt handelt, sollte man zusammen mit anderen effektive Lösungmöglichkeiten erarbeiten und nicht ausschließlich seine eigenen Lösungen verwenden (egolose Programmierung).

Die Einhaltung des vorgegebenen Zeitrahmens ist dabei unerlässlich. Die Aufgaben müssen zügig, aber qualitätiv hochwertig bearbeitet werden, damit ausreichend Zeit zur Qualitätskontrolle und Nachbesserung vorhanden sind. Um dies erfolgreich zu bewältigen ist ein gewisses Maß an Eigeninitiative notwendig, die nicht nur das machen lässt, was unbedingt erforderlich ist, sondern jeden dazu ermuntern sollte neue Ansatzpunkte und Ideen mit in des Projekt einzubringen. Dabei sollten alle ihre erworbenen Fähigkeiten einbringen.

Kapitel 2

Anforderungen

2.1 Use-Case Diagramme

Text...

2.2 Use-Case Beschreibungen

Das Projekt umfasst 6 Use-Cases. Use-Case 1 beschreibt die Initialisierung des Programmes, während Use-Case 2 und 3 die beiden resultierenden Möglichkeiten eines Neuen Spiels, sowie dem Laden eines vorhandenen Spielstandes beschreiben.
Use-Case 4 beschreibt den Ablauf eines Spielzuges, der mittel Use-Case 5 dem Notieren eines Spielzuges geloggt wird. Use-Case 6 befasst sich abschließend mit dem Beenden des Programm durch den Benutzer.

Use-Case 1: Initialisieren
Kurzbeschreibung: Use-Case zur Beschreibung der Initilisierung
Vorbedingung: Das Spiel ist gestartet, Initialmenü wird angezeigt
Szenarion: Verschiedene Auswahlmethoden -Neues Spiel: Erstellen eines neuen Spiels -Spiel laden: Laden eines gespeicherten Spiels -Spiel beenden: Das Spiel beenden
Nachbedingung: Abhängig von der Auswahl wird mit dem Folgeschritt fortgefahren
Ausnahme: -

Tabelle 2.1: Use-Case 1

2.3 Testfälle

Aus den eben beschriebenen Use-Cases werden nun verschiedene Testfälle abgeleitet, die während der Ausführung des Programmes auftreten können.

Use-Case 2: Neues Spiel
Kurzbeschreibung: Use-Case zum Erstellen eines neuen Spiels
Vorbedingung: Spiel gestartet, Wahl von „Neues Spiel“ in Use-Case 1
Szenarion: -Eingabe der Spielernamen -Festlegung von menschlichen Spielern oder Computerspielern -Bei Auswahl eines Computerspielers wird des Denkzeit/ Schwierigkeit festgelegt
Nachbedingung: Das Spiel kann gestartet werden
Ausnahme: Die KI-Komponente kann nicht gefunden werden, eine entsprechende Fehlermeldung wird ausgegeben

Tabelle 2.2: Use-Case 2

Use-Case 3: Spiel laden
Kurzbeschreibung: Use-Case zum Laden eines gespeicherten Spielstandes
Vorbedingung: Spiel gestartet, Wahl von „Spiel laden“ in Use-Case 1
Szenarion: -Datei mit gespeicherten Spielinformationen wird geladen -Das Programm ließt die Informationen ein und setzt den gespeicherten Spielzeitpunkt um
Nachbedingung: Das Spiel kann fortgesetzt werden
Ausnahme: Es kann kein gültiger Spielstand gefunden werden, eine entsprechende Fehlermeldung wird ausgegeben

Tabelle 2.3: Use-Case 3

Use-Case 4: Spielzug durchführen
Kurzbeschreibung: Use-Case zur Beschreibung des Ablaufs eines Spielzuges
Vorbedingung: Das Spiel wurde erfolgreich gestartet, es wurde erfolgreich ein Spielstand geladen
Szenarion: -Der aktuelle Spieler setzt (Phase 1) oder bewegt (Phase 2/3) einen seiner Spielsteine -Das System prüft die Korrektheit des Zuges -Der aktuelle Spieler entfernt einen Spielstein, sollte er eine Mühle geschlossen haben -Das System überprüft, ob ein zulässiger Stein entfernt wurde -Das System überprüft die Siegbedingungen
Nachbedingung: Loggen des Spielzuges
Ausnahme: Ein falscher Spielzug fordert den Spieler zur Durchführung eines regelkonformen Spielzuges auf

Tabelle 2.4: Use-Case 4

Use-Case 5: Spielzug notieren (Loggen)
Kurzbeschreibung: Use-Case zum Notieren des ausgeführten Spielzuges
Vorbedingung: Regelkonformer Spielzug wurde durchgeführt
Szenarion: -Das System notiert das Feld, in das ein Stein gesetzt wurde (Phase 1) -Das System notiert das Feld, aus dem und in das gezogen worden ist
Nachbedingung: Der nächste Spieler ist am Zug
Ausnahme: -

Tabelle 2.5: Use-Case 5

Use-Case 6: Spiel beenden
Kurzbeschreibung: Use-Case zum Beenden eines laufenden oder abgeschlossenen Spiels
Vorbedingung: Spiel ist im Gange
Szenarion: -Der Spieler möchte das Spiel beenden, obwohl es nicht abgeschlossen ist -Er kann einen Speicherstand anlegen -Das Programm wird beendet
Nachbedingung: Gegebenenfalls wurde ein Speicherstand angelegt
Ausnahme: -

Tabelle 2.6: Use-Case 6

Testfall: ungültiger Spielzug
Situation: Spieler ist am Zug
Erwartetes Ergebnis:

- Spielkontrolle verbietet Spielzug
- Spielstein wird nicht gesetzt
- Spieler wird aufgefordert einen gültigen Spielzug durchzuführen

Testfall: Gültiger Spielzug, (n> 3 Steine, keine Mühle)
Situation:

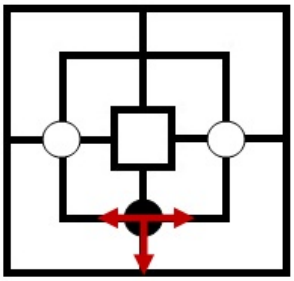

Abbildung 2.1: Testfall X

Erwartetes Ergebnis:

- Spielstein wird in ausgewähltes Feld gezogen

Testfall: Gültiger Spielzug, (n> 3 Steine, Mühle wird geschlossen, gegnerische Steine auch außerhalb einer Mühle vorhanden)
Situation:

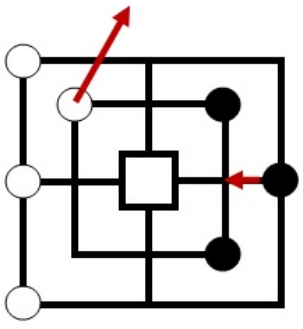

Abbildung 2.2: Testfall X

Erwartetes Ergebnis:

- Spielstein wird in ausgewähltes Feld gezogen
- Eine Mühle wird dabei geschlossen
- Ein gegnerischer Spielstein, der nicht zu einer Mühle gehört, wird entfernt

Testfall: Gültiger Spielzug, (n> 3 Steine, Mühle wird geschlossen, keine gegnerischen Steine außerhalb einer Mühle vorhanden)
Situation:

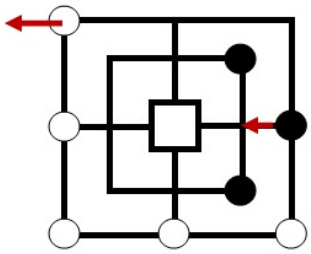

Abbildung 2.3: Testfall X

Erwartetes Ergebnis:

- Spielstein wird in ausgewähltes Feld gezogen
- Eine Mühle wird dabei geschlossen
- Ein gegnerischer Spielstein wird entfernt

Testfall: Gültiger Spielzug, (n= 3 Steine, keine Mühle)
Situation:

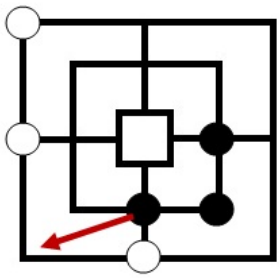

Abbildung 2.4: Testfall X

Erwartetes Ergebnis:

- Spielstein springt in ein auswähltes Feld

Testfall: Gültiger Spielzug, (n= 3 Steine, Mühle wird geschlossen, gegnerischen Steine außerhalb einer Mühle vorhanden)
Situation:

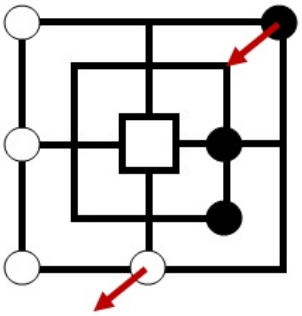

Abbildung 2.5: Testfall X

Erwartetes Ergebnis:

- Spielstein springt in ausgewähltes Feld
- Eine Mühle wird dabei geschlossen
- Ein gegnerischer Spielstein, der nicht zu einer Mühle gehört, wird entfernt

Testfall: Gültiger Spielzug, (n= 3 Steine, Mühle wird geschlossen, keine gegnerischen Steine außerhalb einer Mühle vorhanden)
Situation:

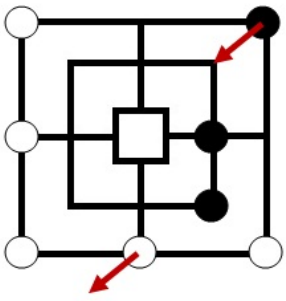

Abbildung 2.6: Testfall X

Erwartetes Ergebnis:

- Spielstein springt in ausgewähltes Feld
- Eine Mühle wird dabei geschlossen
- Ein gegnerischer Spielstein wird entfernt

Testfall: Neues Spiel erfolgreich starten
Situation: Der Benutzer erstellt ein neues Spiel
Erwartetes Ergebnis: Nach Eingabe der relevanten Daten wird ein neues Spiel gestartet

Testfall: Neues Spiel nicht erfolgreich starten
Situation: Der Benutzer erstellt ein neues Spiel, es wird ein KI-Spieler ausgewählt
Erwartetes Ergebnis: Es kann keine KI-Komponente gefunden werden und es wird eine Fehlermeldung ausgegeben

Testfall: Ein gespeichertes Spiel erfolgreich laden
Situation: Der Benutzer möchte einen existierenden Spielstand laden
Erwartetes Ergebnis: Der Spielstand wird geladen, das Spiel kann fortgesetzt werden

Testfall: Ein gespeichertes Spiel nicht erfolgreich laden
Situation: Der Benutzer möchte einen existierenden Spielstand laden, es existiert allerdings kein gültiger Spielstand
Erwartetes Ergebnis: Es kann kein gültiger Spielstand gefunden werden und es wird eine Fehlermeldung ausgegeben

Testfall: Spiel beenden nach einem abgeschlossenen Spiel
Situation: Das Spiel ist abgeschlossen, das Programm soll beendet werden
Erwartetes Ergebnis: Das Programm wird beendet

Testfall: Spiel beenden während das Spiel nicht abgeschlossen ist
Situation: Das Spiel läuft, soll aber durch den Benutzer beendet werden
Erwartetes Ergebnis: Es besteht die Möglichkeit das Spiel zu speichern, das Programm wird beendet

2.4 Nicht-funktionale Anforderungen

Neben einigen funktionalen Anforderungen, soll das Projekt auch mehrere nicht funktionale Anforderungen erfüllen.
Die Softwarearchitektur soll so aufgebaut sein, dass das folgende Bedingungen erfüllt:

- graphische Benutzungsschnittstelle
- mindestens zwei Schichtenarchitektur: (GUI- und Anwendungsschicht)
- ausführbar aus einer jar-Datei
- Spiel-Algorithmen sind als autauschbare Komponenten implementiert
- genutzte Javaversion: jdk7

Das Projekt soll zudem noch einige ergonomische Komponenten berücksichtigen:

- komfortables Setzen der Steine mit der Maus

- Spielablauf durch Spieler/ Beobachter gut verfolgbar
- Verfolgbarkeit gewährleistet durch Loggen der Spielzüge

Kapitel 3

Domänenmodell

Text...

Kapitel 4

GUI-Prototyp

Text...

Kapitel 5

Projektplan

Zunächst die Mitarbeiterübersicht:

- MA1: Jonne Haß
- MA2: Benjamin Held
- MA3: Richard Pump
- MA4: Alexander Sirotin
- MA5: Julian Haack

Anschließend folgt der zu diesem Zeitpunkt aktuelle Projektplan:

Nr.	Arbeitspaket	Plan	Ist	Prognose	MA	Start	Ende
1	Planung				Alle	26.06.	08.10.
1.2	Anforderungen/Ziele festlegen	xxx	xxx	xxx	Alle	26.09.	03.10.
1.3	Projektplan erstellen	xxx	xxx	xxx	MA3	26.09.	03.10.
2	*Stufe 1: Grundlegendes*	xxx	xxx	xxx	Alle	26.09.	21.11.
2.1	**Analyse**	xxx	xxx	xxx		26.09.	08.10.
2.1.1	Use-Cases	xxx	xxx	xxx	MA2	26.09.	05.10.
2.1.2	Testfälle	xxx	xxx	xxx	Alle	26.09.	05.10.
2.1.3	GUI-Design	xxx	xxx	xxx	MA5	26.09.	05.10.
2.1.4	Domänenmodell	xxx	xxx	xxx	xxx	26.09.	05.10.
2.2	Design	xxx	xxx	xxx		10.10.	24.10.
2.2.1	UML-Modellierung der Klassen	xxx	xxx	xxx	xxx	start	ende
2.2.2	Programmablaufplan	xxx	xxx	xxx	xxx	start	ende
2.2.3	GUI verfeinern	xxx	xxx	xxx	xxx	start	ende

Tabelle 5.1: Projektplanung/ Arbeitspakete

Nr.	Arbeitspaket	Plan	Ist	Prognose	MA	Start	Ende
2.3	Implementierung	xxx	xxx	xxx		24.10.	19.11.
2.3.1	Logische Architektur/						
	Kontrollklassen	xxx	xxx	xxx	xxx	start	ende
2.3.2	GUI	xxx	xxx	xxx	xxx	start	ende
2.3.3	Einfache KI	xxx	xxx	xxx	xxx	start	ende
2.3.4	Dokumentation pflegen	xxx	xxx	xxx	xxx	start	ende
2.4	Debugging/ Testen	xxx	xxx	xxx		19.11.	21.11.
2.4.1	Testfälle durcharbeiten	xxx	xxx	xxx	Alle	19.11.	21.11.
3	Stufe 2: „Hohe Kunst“	xxx	xxx	xxx		21.11	19.12.
3.1	Analyse 2	xxx	xxx	xxx	xxx	21.11.	24.11.
3.1.1	Überarbeitung von Analayse 1	xxx	xxx	xxx	xxx	start	ende
3.1.2	Anforderungen an KI	xxx	xxx	xxx	xxx	start	ende
3.2	Design 2	xxx	xxx	xxx		24.11.	28.11.
3.2.1	KI-Design	xxx	xxx	xxx	xxx	start	ende
3.2.2	Schnittstellen-Integration	xxx	xxx	xxx	xxx	start	ende
3.3	Implementierung 2	xxx	xxx	xxx		28.11.	16.12.
3.3.1	Implementierung der Schnittstellen						
	Schnittstellen	xxx	xxx	xxx	xxx	start	ende
3.3.2	KI	xxx	xxx	xxx	xxx	start	ende
3.4	Debugging/ Testen	xxx	xxx	xxx		16.12.	19.12.
4	Projektleitung					02.10.	29.01.
5	Qualitätssicherung					02.10.	29.01.
6	Puffer					08.10.	26.10.

Tabelle 5.2: Projektplanung/ Arbeitspakete

Abbildungsverzeichnis

2.1 Testfall X 8
2.2 Testfall X 8
2.3 Testfall X 9
2.4 Testfall X 9
2.5 Testfall X 10
2.6 Testfall X 10

Tabellenverzeichnis

2.1 Use-Case 1 5
2.2 Use-Case 2 6
2.3 Use-Case 3 6
2.4 Use-Case 4 6
2.5 Use-Case 5 7
2.6 Use-Case 6 7

5.1 Projektplanung/ Arbeitspakete 14
5.2 Projektplanung/ Arbeitspakete 15