2017

Hackermans – Projekt Dokumentation

PHILIPP LEHMANN
PASCAL MOLL
PHILIP KÄPPELI
MARIO ALLEMANN

Inhaltsverzeichnis

1 Einleitung

Das Spiel Dominion ist in Java mit Hilfe von JavaFX programmiert und folgt einer MVC-Struktur. Es läuft in einem klassischen Client/Server Modell. Der Client prüft nur grundlegende Szenarien (Thin-Client). Die komplette Programmlogik läuft auf dem Server. Der Server ist mit einer MySQL-Datenbank verbunden.
Hier sind alle minimalen Anforderungen des Spieles, welche wir natürlich alle erfüllen:

- Client-Server Implementation
- Spielzüge können gemacht werden
- Mindestens 5 verschiedene Königreichkarten, 1 Schatzkarte und 1 Siegeskarte
- Mindestens 10 Runden oder ähnliches Spielende
- Implementation für 2 Spieler
- Gewinner auswerten
- Gewinner-Message anzeigen
- Grundlegendes Design

Zu den minimalen Anforderungen haben wir noch zusätzlich folgende Features in unser Spiel implementiert:

Menü

- Connection Fenster
    - Das ist das erste Fenster, das geöffnet wird, bevor man zum Login gelangt
    - Hier kann man die Verbindungsdaten angeben (IP und Port-Nummer)
- Login
    - Anforderungen: mindestens 2 und maximal 14 Zeichen
- Lobby
    - Die Aktionskarten können selektiert werden
- Datenbankanbindung mit:
    - Highscore-Liste, Kartendaten, automatische Client-Update-Funktion anhand einer Versionsnummer
- Chat in der Lobby, sowie im Spiel
- Schönes Design
    - Unterschiedliche Bilder zu unterschiedlichen Views (Mittelalterliche Bilder)
    - Texts in mittelalterlichem Stil
    - Dynamische GUIs (Texte, Knöpfe, Karten etc.)
- 2 verschiedene Sprachen, die man in den Einstellungen ändern kann
    - Englisch und Deutsch
- Musik, die im Hintergrund läuft
    - kann man in den Einstellungen an und abschalten
- Button-Sound, wenn man auf die Knöpfe drückt
    - Der Sound kann in den Einstellungen ein- und ausgeschaltet werden
- Alle Einstellungen, die man vornimmt, werden lokal in einem File gespeichert
    - Beim nächsten Ausführen vom Spiel werden dieselben Einstellungen übernommen.

Im Spiel

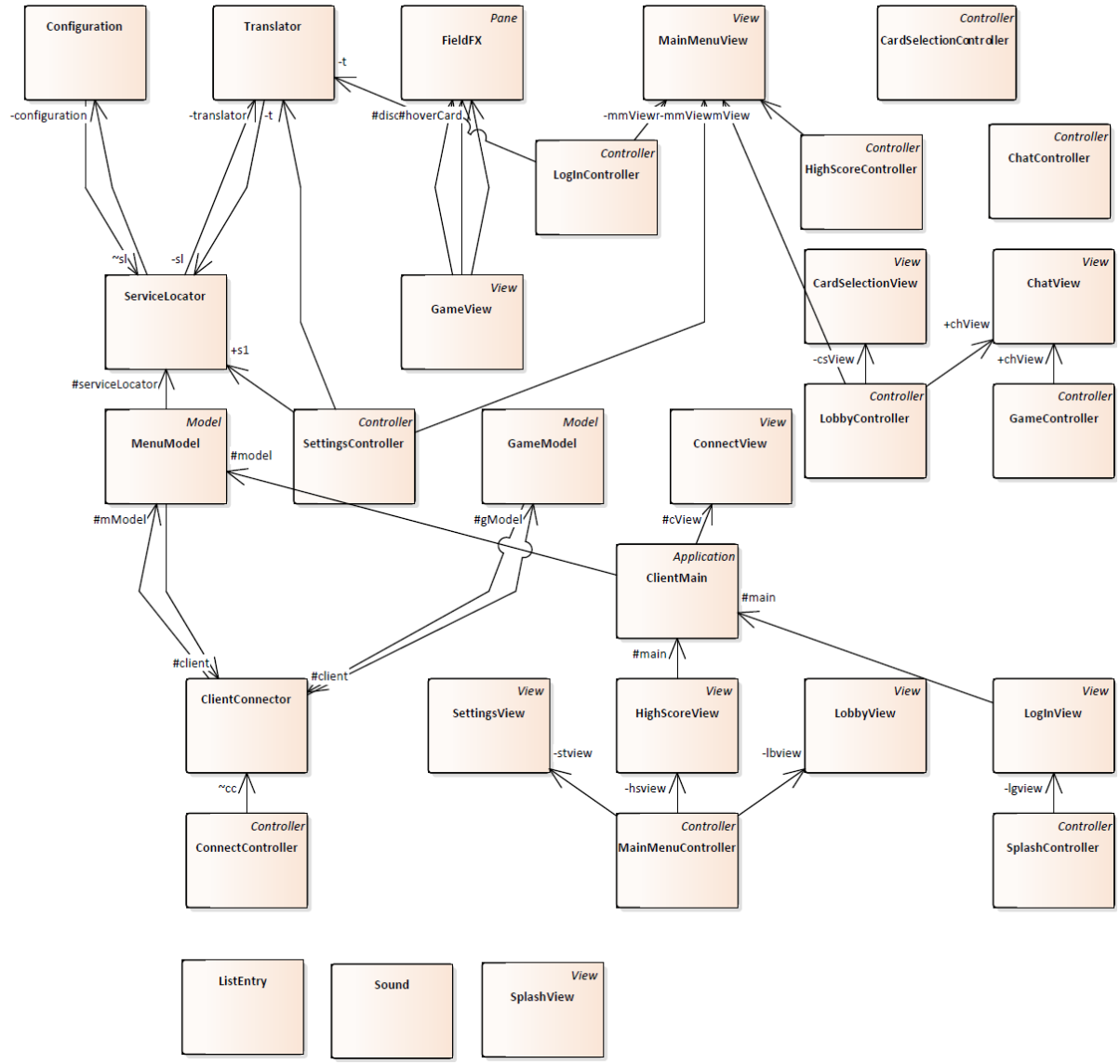
- Das Spiel kann mit 2 bis 4 Spielern gespielt werden
- Die Züge der Gegner werden angezeigt (Letzte gespielte Karte, oberste Karte des Abwurfstapels)
- 6 zusätzliche Spielkarten
- Karten werden beim Mouseover vergrössert angezeigt
- Spieler können jederzeit das Spiel verlassen, ohne dass das Spiel beeinträchtigt wird
    - Ist nur noch ein Spieler übrig, bekommt dieser eine Meldung und wird ins Hauptmenü zurückgeführt, da das Spiel nicht fortgesetzt werden kann
- Spielendmeldung hat einen Timer. Wenn dieser abgelaufen ist, schliesst sich das Fenster und die Teilnehmer sind zurück im Hauptmenu.

Server

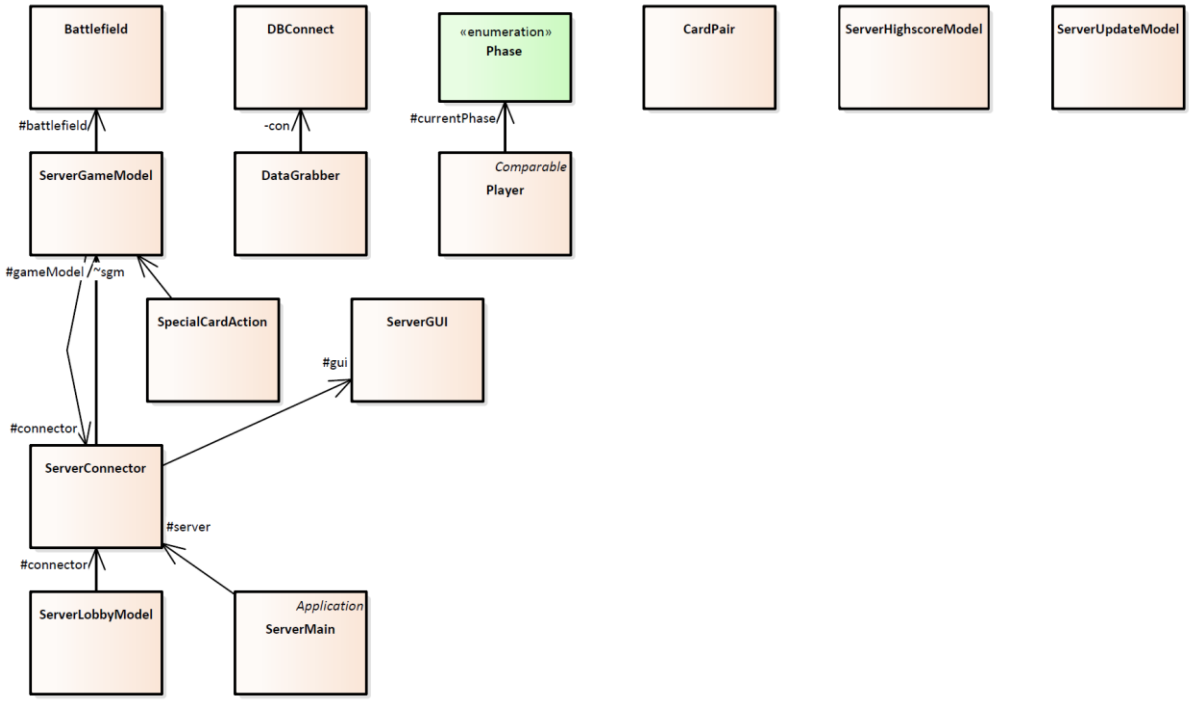
- Ein serverseitiges GUI zeigt alle ein-und ausgehenden Meldungen, sowie die externe IP und den Port an

2 Datenstrukturen

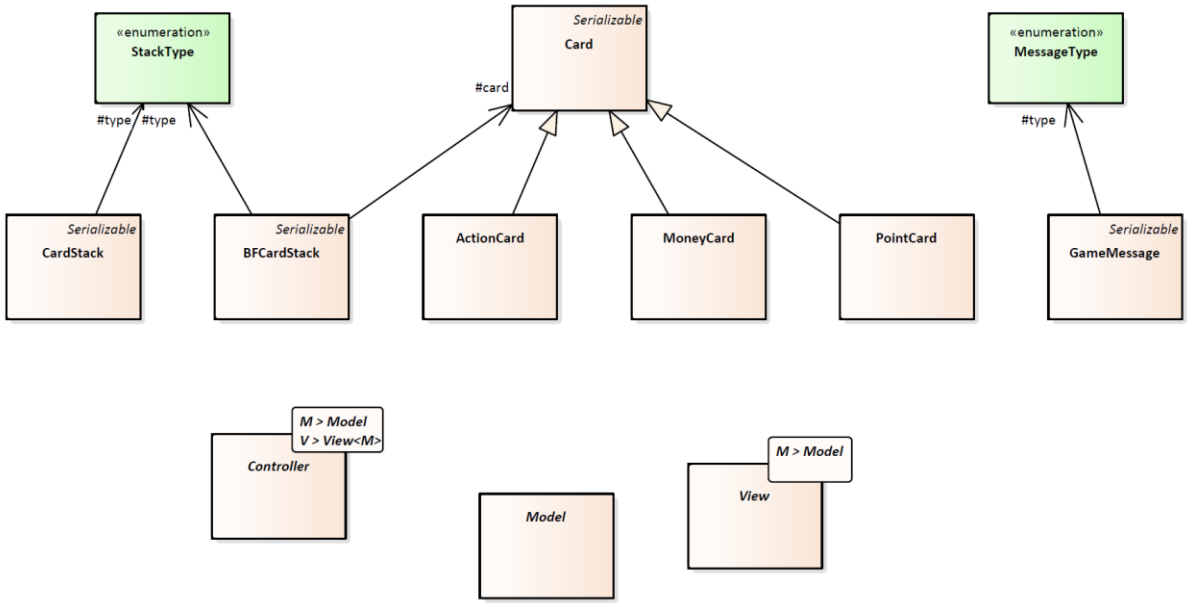
2.1 Klassendiagramm Client



2.2 Klassendiagramm Server



2.3 Klassendiagramm Shared



3 Client / Server Kommunikation

3.1 Server:

Der Server läuft sowohl über den localhost, ein lokales Netzwerk und über das Internet. Die MySQL-Datenbank läuft auf dem Server.

3.2 Client:

Es wird eine Thin-Client Struktur verwendet. Der Client führt dabei nur minimale Logikprüfungen aus (Eingabeprüfungen und Sperrmechanismen). Die restliche Logik wird auf dem Server ausgeführt. Die Kommunikation zwischen Server und Client erfolgt mithilfe von Serialisierung. Zu diesem Zweck wurde eine eigene GameMessage-Klasse definiert, deren Objekte serialisiert und übermittelt werden.

3.3 Sequenzfluss:

Der Client startet die Kommunikation mit dem Server beim Start der Anwendung. In den Spielmenüs sendet der Client bei entsprechendem User-Input (Bsp. Aufrufen der Highscore-Liste) eine Nachricht an den Server und wartet anschliessend auf dessen Antwort. In der Lobby, sowie im Spiel, empfängt der Client permanent Nachrichten des Servers.

3.4 Vorbedingung die erfüllt wurde:

Zur Sicherstellung, dass die Clients nur im richtigen Zeitpunkt Nachrichten schicken können, wurde ein GUI-Sperrmechanismus verwendet.

4 Fazit

Während der Laufzeit dieses Projektes konnten alle Teammitglieder, vor allem Philipp Lehmann, Pascal Moll und Mario Allemann, unserer Gruppe beim Programmieren sehr viel dazulernen. Wir schätzten unsere Gruppe sehr und halfen einander, falls es Fragen und Probleme gab.

Natürlich gab es Momente, an denen einige Teammitglieder am Projekt zweifelten, jedoch haben wir diese Momente als Gruppe überstanden und ein faszinierendes und funktionierendes Spiel zum Laufen gebracht.

Wir hatten jede Woche mindestens eine Sitzung per Skype, da sich 2 Teammitglieder im Ausland befanden. In diesen Sitzungen besprachen wir was erledigt wurde, was momentan zu erledigen ist und was in Zukunft erledigen werden muss.

Die abgemachten Termine wurden fast immer eingehalten. Dies machte es möglich, das Projekt an einem angemessenen Zeitpunkt fertigzustellen.

Wir möchten uns bei der Fachhochschule Nordwestschweiz und bei unseren Dozenten Bradley Richards und Lukas Frey bedanken, damit wir so ein IT-Projekt durchführen konnten. Es war eine spannende und lehrreiche Zeit während diesem Semester.