

Implementierung eines Jass-Server mit Jassbots basierend auf maschinellern Lernen

Themenbereiche:	Webentwicklung, SW-Entwicklung mit Python, Jass-Simulation
Studierende:	Kevin Rickenbach, Yanick Bisang
Dozent:	Prof. Dr. Ruedi Arnold
Experte:	Urs Gehrig, Roche Diagnostics International
Wirtschaftspartner:	Prof. Dr. Thomas Koller, Forschungsteam ABIZ, HSLU
Keywords:	Web-Entwicklung (Python), Angular, Jassen

1. Aufgabenstellung

Das Forschungsteam "Algorithmic Business" (ABIZ) der HSLU I entwickelt in Zusammenarbeit mit der Firma Swisslos Jassbots, also vom Computer simulierte Spieler eines Jass-Spiels. Damit diese getestet werden können, muss ein Jass-Server erstellt werden, der ein Jass-Spiel mit diesen Jassbots simulieren kann. In diesem Projekt gilt es, eine Jass-Server-Infrastruktur sowie nach Möglichkeit eigene Jassbots zu implementieren. Auf dem zu erstellenden Jass-Server sollen verschiedene Jassbots gegeneinander antreten können. Dabei soll es einen Selbstspiel-Modus geben, bei welchem beliebig viele Partien, automatisch mit beliebigen Bots, gespielt und in einem passenden Datenformat persistiert werden können. Zu gespielten Partien soll zudem nach Möglichkeit automatisch eine statistische Auswertung erstellt werden. Als Referenz-Jassbot kann ein bereits entwickelter Bot verwendet werden. Weiter sollen Partien manuell spielbar sein. Es soll also ein einfaches Jass-Client-GUI geben. Zusätzlich soll, basierend auf den Vorarbeiten vom Auftraggeber Thomas Koller, ein eigener Jassbot mit neuronalen Netzen trainiert werden. Dieser Bot soll dann ebenfalls gegen sich selber und gegen andere Bots antreten können und nach Möglichkeit natürlich den Referenz-Jassbot schlagen. Die verschiedenen Teilaufgaben werden vom Auftraggeber wie folgt priorisiert:

1. Jass-Server: Basis-Spiel-Logik für 4 Bots
2. Jass-Server: Spiel-Persistierung
3. Jass-Server: Turnier-Modus
4. Lernen eigener Bot
5. GUI-Jassbot für "manuelles" Spielen
6. Automatische statistische Auswertung zu Partien

Alle relevanten Entscheide (wie Schnittstellen-Definitionen, Daten-Formate, Architektur-Entscheide, usw.) sind in enger Absprache mit Thomas Koller und Ruedi Arnold zu erarbeiten bzw. zu treffen.

2. Ergebnisse

Im Rahmen der Bachelor-Diplomarbeit (BDA) wurden folgende Ergebnisse erarbeitet:

- **Bau eines Jass-Servers:** Im Rahmen der Arbeit wurde ein Jass-Server erstellt, der mehrere Jass-Spiele und Jass-Turniere simulieren kann.
- **Definition eines Turniermodus:** Ein Turniersystem, dass im Kontext eines Modulunterrichts und im Forschungsbereich standhält, wurde definiert. Recherchen an echten Jass-Turnieren wurden durchgeführt.
- **Bau eines Web-Clients:** Zur visuellen Darstellung und als Schnittstelle zwischen Nutzer und Server wurde ein Web-Client erstellt. Die Anforderungen sind für die zwei Benutzergruppen (Forschung, Unterricht) angepasst worden.
- **Persistieren der gespielten Turniere/Spiele:** Jedes gespielte Spiel wird auf dem Server persistiert werden. Das Format ist nahe am Original des Jass-Simulators der Firma Swisslos.
- **Definition der statistische Auswertung:** Die durchgeführten Turniere werden statistisch ausgewertet.
- **Replay-Funktion:** Die persistierten Spiele werden ausgelesen und können wiedergegeben werden.

3. Lösungskonzept

Das Lösungskonzept wurde in mehrere Teile aufgeteilt. Um Turniere, Persistenz der Daten und Web-Client zu konzeptionieren, muss als erstes der Jass-Server aufgebaut werden.

Jass-Server

Die Grundstruktur des Jass-Servers wurde bereits zu Beginn des Projektes vorgegeben. Die Regeln eines Jass-Spiels wurden von Thomas Koller bereits vorgängig programmiert und ausführlich getestet. Schon früh in der Konzept-Phase wurde erkannt, dass der grösste Teil der Funktionalität eines Spiels in einer einzelnen Runde durchgeführt wird. Daher wurde entschieden, dass der Server in einen Turnier-, Spiel- und Runden-Manager aufgeteilt wird. Der Rundenmanager verwaltet Runden und die einzelnen Spielzüge. Der Spielmanager verwaltet Rundenmanager und der Turniermanager verwaltet Spielmanager. Der Jass-Server hat eine Jass-Server-Klasse welche die Schnittstelle zum Web-Client ist.

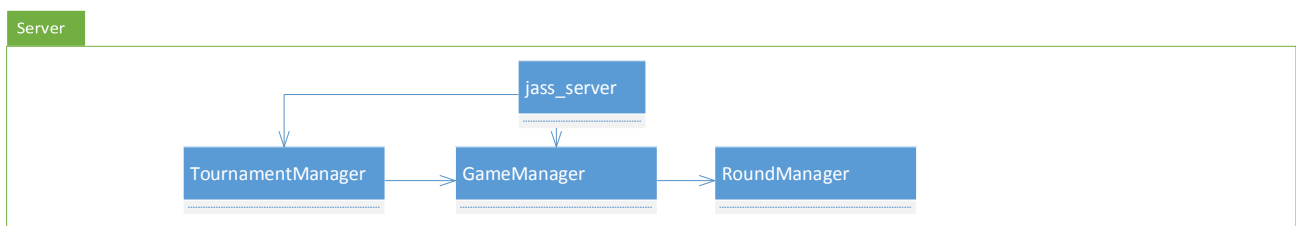


Abbildung 1 - Server-Architektur

Turniere

Es wurden mehrere Konzeptvorschläge definiert, wie ein Turnier durchgeführt werden kann. Zwei Turnier-Modi kamen in die engere Auswahl. Das eine Konzept ist sehr realitätsnah, das andere Konzept will ein Turnier durchführen, das die einzelnen Stärken und Schwächen eines Jassbots ohne Einfluss von anderen Jassbots auswertet. Das realitätsnahe Konzept paart zufällige Spieler zu 4er Gruppen zusammen und lässt sie mit und gegeneinander spielen. Sobald alle Spieler die gewünschte Anzahl Spiele erreicht haben wird das Turnier beendet.

Das zweite Konzept, das Fairness als wichtigste Priorität hatte, basierte darauf, dass Spieler immer mit einer Kopie ihres eigenen Jassbots zusammenspielen. Der Vorteil dieses Konzepts ist, dass die Leistung des Bots nie durch einen schlechteren/besseren Bot als Partner verändert wird.

Aufgrund der einfachen Betrugsmöglichkeit des zweiten Konzepts wurde das erste Konzept umgesetzt.

Web-Client

Der Web-Client sollte folgende Funktionen beinhalten:

- Erstellen, Löschen, Mutieren, Starten von Spielen
- Erstellen, Löschen, Mutieren, Starten von Turnieren
- Darstellung der Turnierstatistiken
- Anschauen von bereit gespielten Jass-Runden.

Das Projekt wurde in Komponenten und Services aufgeteilt. Die Services beinhalten die Funktionalität der Seiten, während die Komponenten die Elemente darstellen. Der Request-Service ist die Schnittstelle zwischen dem Jass-Server und dem Web-Client. Die Kommunikation zwischen dem UI und dem Server basiert auf simplen http-Requests. Die Replay-Funktion des UIs erlaubt es bereits gespielte Spiele anzuschauen. Mit der Implementierung von async/await-Funktionen wurde ein Spielablauf simuliert.

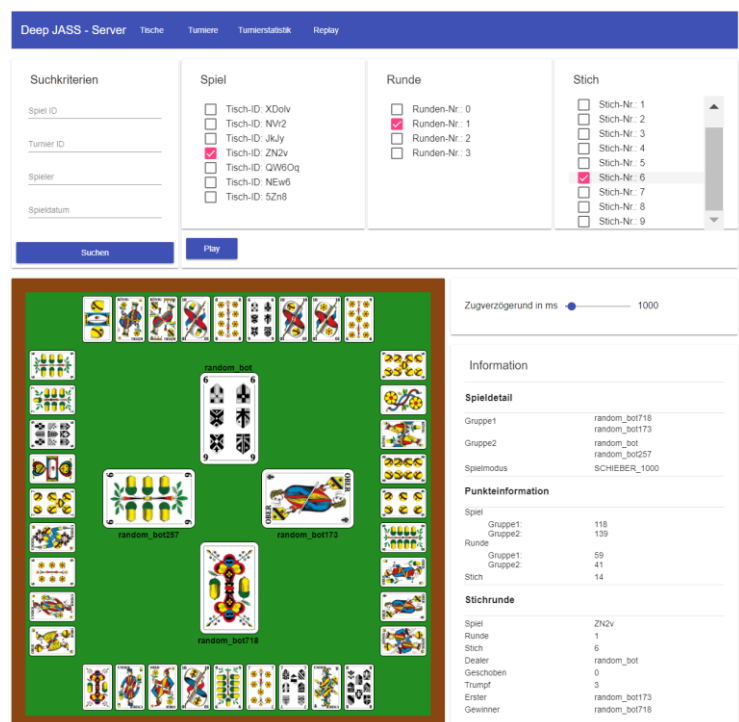


Abbildung 2 - User Interface Replay

4. Spezielle Herausforderungen

Eine der grössten Herausforderungen dieses Projekts, waren die sich ändernden Prioritäten und die genaue Definition der Projektanforderungen. Da das Projekt nicht nur für den Forschungszweig der HSLU gedacht war, sondern auch in naher Zukunft im Unterricht eines neuen Moduls benutzt werden soll, mussten während der Planung der Arbeit immer wieder die verschiedenen Sichten der Kunden einbezogen werden. Die Herausforderung lag darin, keinen Kunden zu vernachlässigen und konstant beide Interpretationen der Anforderungen umzusetzen.