*Todo list

Wahlpflichtfach KI

Projekt: Schiffe Versenken

Autoren: Victor Apostel, Stefan Bogdanski, Sibille Ritter
Studiengang: (Internationaler) Studiengang Technische Informatik B.Sc.

Inhaltsverzeichnis

1 Einleitung **3**

2 Spielregeln **3**

3 Kommunikationsprotokoll **3**

4 Softwarearchitektur **5**

4.1 Spielserver 7

4.2 Client - Java 8

4.3 Client - Prolog 8

5 Evaluation **8**

Abbildungsverzeichnis

1 Darstellung der möglichen Kommunikationsteilnehmer 5
2 Zustandsübergänge des Clients 5
3 Interne Zustandsübergänge während des laufenden Spiels 6
4 Sequenzdiagramm der Kommunikation 6
5 Klassendiagramm des Servers 7
6 Klassendiagramm des Java Clients 8

Tabellenverzeichnis

1 Kommunikationsprotokoll 4
2 Ergebniskodierung 4

1 Einleitung

Hier Aufgabenbeschreibung

2 Spielregeln

3 Kommunikationsprotokoll

Die Kommunikation zweier Spieler erfolgt auf Basis von Textnachrichten, den sogenannten Kommandos, die einer fest definierten Codierung genügen. Das Ziel dieses Kapitels ist die Erläuterung des Nachrichtenaufbaus und deren Interpretation.

Jedes Kommando, das zu interpretieren gilt, ist aus Gründen der Prologkompatibilität mit runden Klammern umgeben und endet mit einem ".". Da zudem stets ganze Zeilen verarbeitet werden sollen, wird der Nachricht das nicht druckbare Zeichen "\n" angehängt. Damit wird signalisiert, dass die gesamte Nachricht übermittelt wurde.

Innerhalb der umschließenden runden Klammern werden zwei Angaben erwartet. Die erste Angabe wird als *Opcode* bezeichnet und dient der eindeutigen Bestimmung des Kommandotyps. Die zweite Angabe des Kommandos ist eine in eckigen Klammerun umgebene Parameterliste und kann null bis drei durch Kommata getrennte nummerische Elemente beinhalten. Anhand des Opcodes richtet sich die Anzahl der zu erwartenden Parameter.

Die formale Beschreibung des Kommandoaufbaus liegt im Folgenden in Form von regulären Ausdrücken vor.

$$
\begin{aligned}
COMMAND &:= \backslash(OPCODE, LIST\backslash)\backslash.NEWLINE \\
LIST &:= \backslash[[PARAMS]?\backslash] \\
NEWLINE &:= \backslash\backslash r \\
OPCODE &:= [1-5] \\
PARAMS &:= [0-9] + [, [0-9]+]\{0,2\}
\end{aligned}
$$

Der Spielablauf durchläuft mehrere Zustände, die durch das Kommunikationsprotokoll

abgedeckt werden müssen. So startet jeder Spieler in einem Initialisierungszustand. In diesem Zustand wird festgelegt, welcher der beiden Spieler den ersten Angriff ausführt.

Stellt der Server eine gültige Anzahl an verbundenen Spielteilnehmern fest, sendet dieser ein Startsignal an alle Teilnehmer und die Clients wechseln ihrerseits vom Initialisierungszustand in den Angriffs-, bzw. Verteidigungszustand. Weitere Nachrichten die es zu übertragen gilt, sind zum Einen der Angriff auf eine Feldkoordinate, sowie zum Anderen das Ergebnis des Angriffs.

Die Abbildung der genannten Aktionen in Opcodes, sowie die erwarteten Parameter können der Tabelle 1 entnommen werden.

Opcode	**Bedeutung**	**Param 1**	**Param 2**	**Param 3**
1	Angriff	X	Y	-
2	Ergebnis des Angriffs	X	Y	Ergebnis
3	Spieler startet im Verteidigungszustand	-	-	-
4	Spieler startet im Angriffszustand	-	-	-
5	Startsignal	-	-	-

Tabelle 1: Kommunikationsprotokoll

Das Ergebnis eines Angriffs wird ebenfalls kodiert übertragen und kann den Zustand des Clients beeinflussen. Im Regelfall sendet bzw. empfängt dieser die Ereignisse “Wasser“, “Schiff wurde getroffen“, oder “Schiff wurde versenkt“. Des Weiteren kann als Reaktion auf einen Angriff die Nachricht eintreffen, dass das letzte Schiff versenkt wurde. In diesem Fall wechseln die Clientzustände in Abhängigkeit vom Sender und Empfänder dieser Nachricht in “gewonnen“ bzw. “verloren“.

Eine Auflistung der möglichen Ergebnisse eines Angriffs werden in der Tabelle 2 präsentiert.

Code	**Bedeutung**
1	Wasser
2	Schiff wurde getroffen
3	Schiff wurde getroffen und versenkt
4	Letztes Schiff wurde versenkt. Das Spiel ist beendet

Tabelle 2: Ergebniskodierung

4 Softwarearchitektur

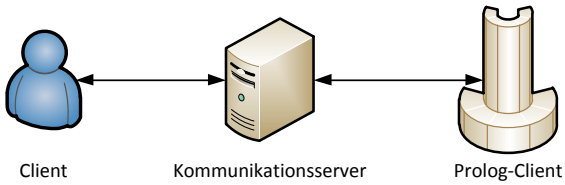

Abbildung 1: Darstellung der möglichen Kommunikationsteilnehmer

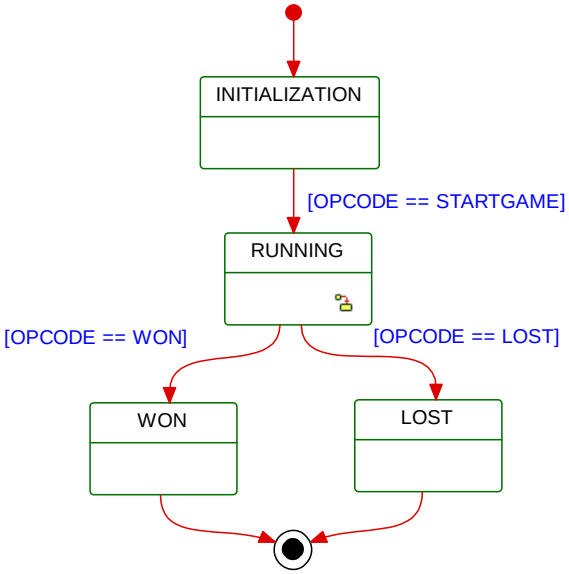

Abbildung 2: Zustandsübergänge des Clients

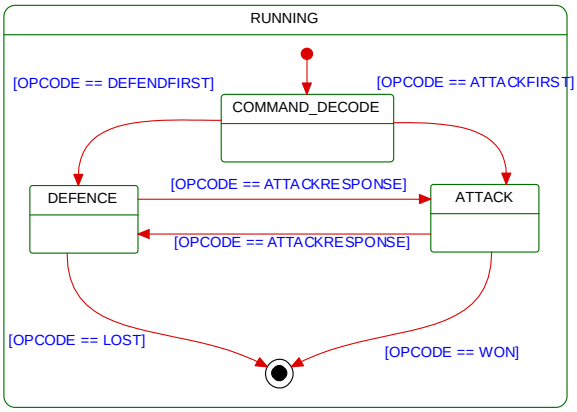

Abbildung 3: Interne Zustandsübergänge während des laufenden Spiels

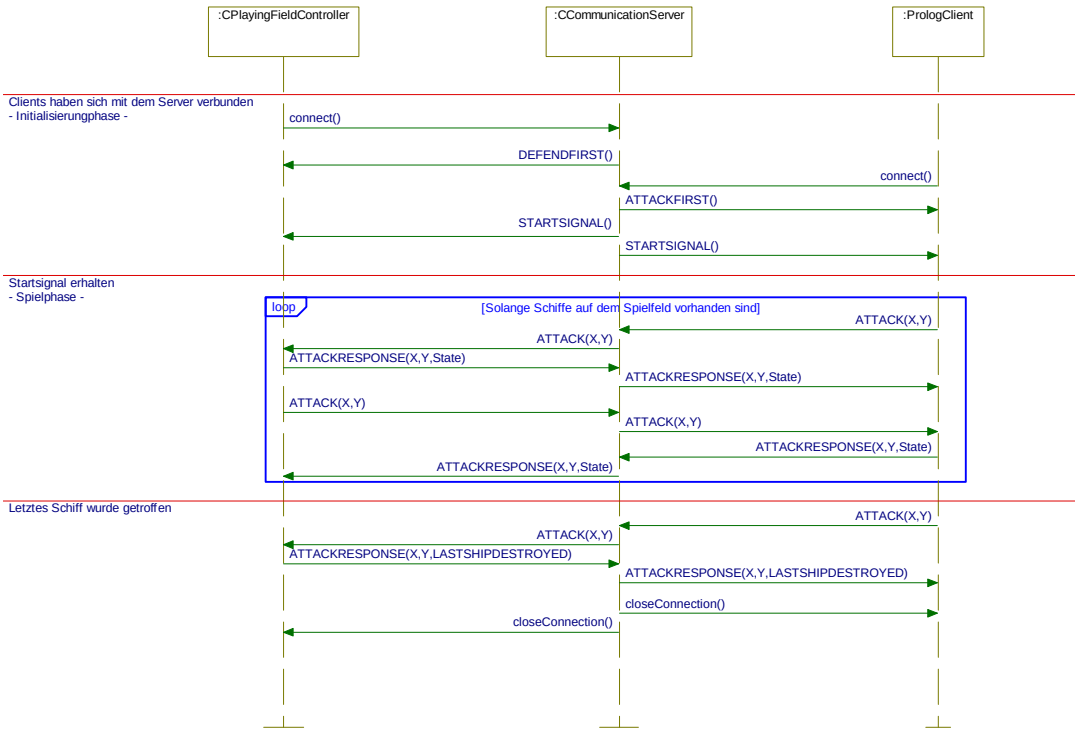

Abbildung 4: Sequenzdiagramm der Kommunikation

4.1 Spielserver

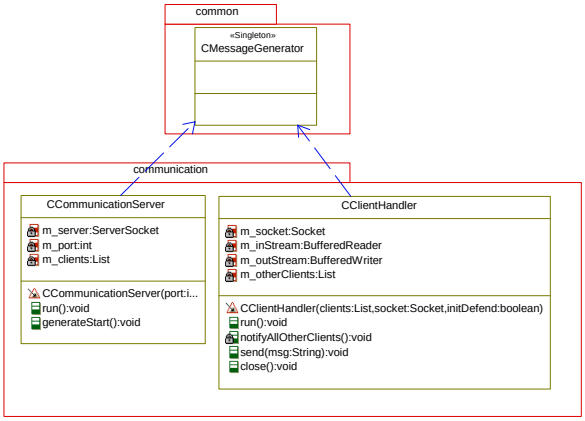

Abbildung 5: Klassendiagramm des Servers

4.2 Client - Java

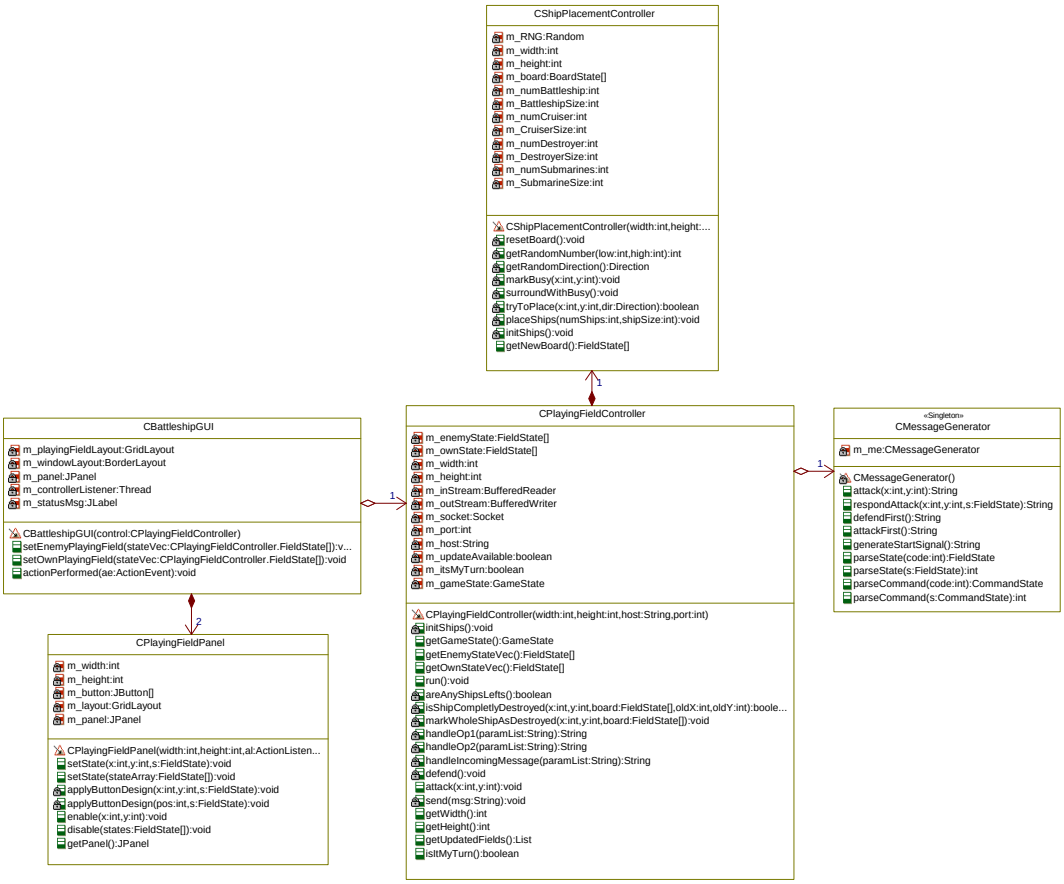

Abbildung 6: Klassendiagramm des Java Clients

4.3 Client - Prolog

5 Evaluation