Technische Universität Braunschweig

Institut für Programmierung und Reaktive Systeme

Capture the Flag

Team 1

Software-Entwicklungspraktikum (SEP)
Sommersemester 2015

Fachentwurf

Auftraggeber
Technische Universität Braunschweig
Institut für Programmierung und Reaktive Systeme
Prof. Dr. Ursula Goltz
Mühlenpfordtstr. 23
38106 Braunschweig

Betreuer: Benjamin Mensing

Auftragnehmer:

Name	E-Mail-Adresse
Lennart Müller	lennart.mueller@tu-braunschweig.de
Pavel Bezwerk	b.pavel@tu-braunschweig.de
Nils-André Forjahn	n.forjahn@tu-braunschweig.de
Robert Drachenberg	r.drachenberg@tu-braunschweig.de
Falk Jacobsen	f.jacobsen@tu-braunschweig.de
Andhika Yopi Setyawan Putera	a.setyawan-putera@tu-braunschweig.de
Nico Scheideler	n.scheideler@tu-braunschweig.de
Ricardo Reck	r.reck@tu-braunschweig.de

Braunschweig, 3. Juni 2015

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Einleitung

Mit Hilfe des Fachentwurfs wird dem Leser mitgeteilt, was das System macht. Dazu wird es in einem Aktivitätsdiagramm dargestellt. Darüber hinaus werden alle Produktfunktionen detailliert analysiert und jeweils mit einem Sequenzdiagramm abgebildet. Die dauerhaft gespeicherten Daten werden in einem Klassendiagramm dargestellt. Im letzten Kapitel wird auf die Konfiguration des Servers eingegangen.

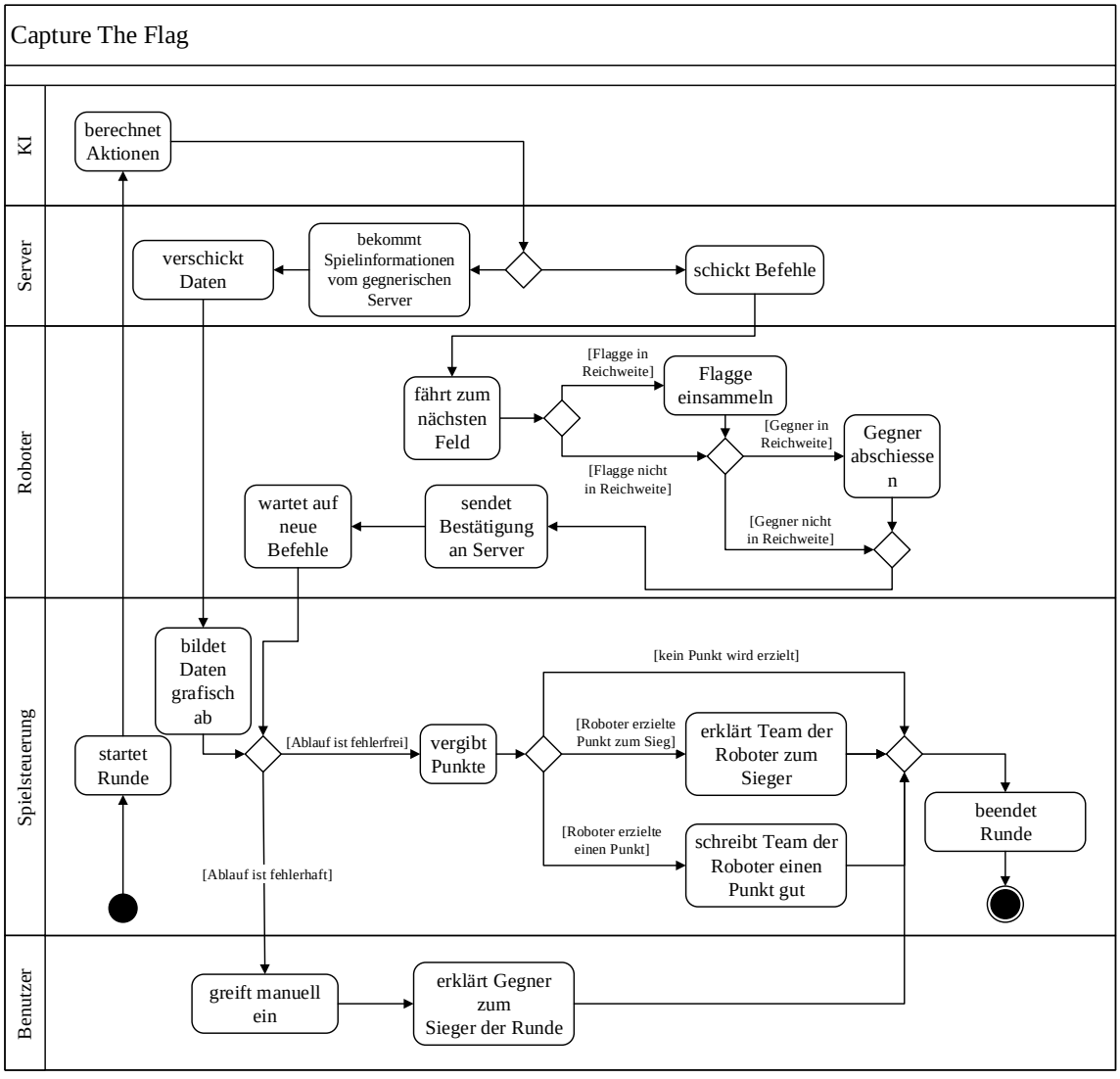

Abbildung 1.1: Aktivitätsdiagramm zum Spielbetrieb

Das Aktivitätsdiagramm zum Spielbetrieb (s. Abbildung 1.1) zeigt den Ablauf einer Spielrunde aus Sicht eines der beiden Teams. Zunächst berechnet die KI Aktionen für die Roboter. Der Server sendet die Befehle an die Roboter weiter, welche diese ausführen. Dabei fährt jeder zunächst ein Feld weiter. Ist eine Flagge in Reichweite wird diese aufgesammelt. Ist ein Gegner in Schussweite wird dieser angegriffen. Nach den ausgeführten Aktionen sendet der Roboter eine Bestätigung und wartet auf neue Befehle. Der eigene Server bekommt vom Gegnerischen Informationen über die Positionen der gegnerischen Roboter und deren Leben. Somit kennt der Server die Positionen aller Roboter, sowie die Anzahl ihrer Leben und die Positionen der Flaggen. Diese Daten werden an die Spielsteuerung geschickt, die sie graphisch darstellt. Darüber hinaus vergibt sie die Punkte.

1.1 Projektdetails

Im folgenden Abschnitt wird näher auf Besonderheiten des Projekts eingegangen.

1.1.1 Spielregeln

Die Spielregeln werden hier näher erläutert.

1. Das Spiel wird rundenbasiert durchgeführt, d.h. die Roboter des einen Teams führen ihre Züge erst dann aus, wenn die Roboter des anderen Teams ihre Züge bereits vollständig abgeschlossen haben.
2. In jedem Zug dürfen beide Roboter vom selben Team Befehle nacheinander in einer vorgegebenen Reihenfolge ausführen.
3. In jedem Zug können sich beide Roboter um jeweils ein Feld bewegen. Sollten sich, nach den wahlweise ausgeführten Zügen, gegnerische Roboter in Schussweite befinden, so können diese angegriffen werden.
4. Der KI sind zu jedem Zeitpunkt das Spielfeld sowie die darauf versehenen Hindernisse, die Startpunkte der Roboter und der Flaggen,Anzahl der Leben, sowie die aktuelle Position der eigenen Roboter und beider Flaggen bekannt.
5. Beide Roboter haben eine „gemeinsame Sicht“. Sollte sich beispielsweise ein gegnerischer Roboter im Sichtfeld nur eines der beiden Roboter des eigenen Teams befinden, so kennt der nicht in Sichtweite befindliche Roboter trotzdem die Position des Feindlichen, da innerhalb des eigenen Teams Informationen geteilt werden dürfen.
6. Bei jedem Treffer verliert der Roboter ein Leben.

7. Liegt die Flagge auf ihrem Startpunkt, so kann diese in einem Radius von einem Spielfeld um sie herum von einem gegnerischen Roboter aufgenommen werden, was also je nach Position der Flagge ein Gebiet von bis zu 3*3 Knoten umfasst.
8. Verliert ein Roboter bei einem gegnerischen Angriff sein letztes Leben, so hat er direkt im Anschluss automatisch an seinen Startpunkt zurückzufahren und setzt, die vor Spielbeginn festgelegte Anzahl an Runden, aus. War der Roboter zuvor im Besitz der gegnerischen Flagge, so lässt er diese auf dem Feld zurück, auf dem er sich zum Zeitpunkt seiner Vernichtung befand. Die Flagge kann nun nicht mehr in einem Radius wiederaufgenommen bzw. zurückgebracht werden, sondern nur auf dem Feld, auf dem sie liegen gelassen wurde.
9. Ein Spieler gewinnt eine Runde, wenn beide Flaggen in der eigenen Basis (Startposition der eigenen Flagge) sind.
10. Ein Spieler gewinnt das Spiel, sobald die Punkte zum Sieg erreicht wurden.
11. Spielparameter wie Anzahl der Leben, Schussweite, Sichtweite, die Anzahl der Runden, die ein Roboter bei seiner Vernichtung auszusetzen hat, und benötigte Punkte zum Sieg werden vor Beginn des Spiels im Karteneditor festgelegt.

1.1.2 Spielablauf

Der Spielablauf wird hier näher erläutert.

1. Beim Start der Software wird der Karteneditor geöffnet. Dieser bietet die Möglichkeit eine neue Karte zu erstellen oder eine bereits vorhandene Karte zu laden.
2. Nachdem die Spielkarte erstellt oder geladen wurde, muss der Spieler die IP-Adresse für die eigenen Roboter eingeben.
3. Danach kann der Spieler ein neues Spiel bereitstellen oder an einem bestehenden Spiel teilnehmen. Um an einem Spiel teilzunehmen, wird die IP-Adresse des anderen Servers benötigt.
4. Die Spielsteuerung ermöglicht es dem Spieler das Spielgeschehen auf dem Bildschirm zu verfolgen.
5. Sofern der Spieler möchte, kann er in der Spielsteuerung die manuelle Steuerung der Roboter aktivieren und somit selbst in das Spielgeschehen eingreifen.

1.1.3 Die Künstliche Intelligenz

Ein Hauptteil dieses Projektes ist die Entwicklung von einer künstlichen Intelligenz, die die Fähigkeit hat *Capture the Flag* selbständig zu spielen. Besonders wichtig ist die Entwicklung einer Spielstrategie. Die in der KI integrierte Strategie soll variable Reaktionen auf das Spielgeschehen ermöglichen, die zum Sieg führen. Die KI soll dabei verschiedene Aktionen abwägen, wie zum Beispiel Gegner anzugreifen oder nicht.
Der KI stehen verschiedene Spielstrategien zur Verfügung. Diese Strategien greifen auf Algorithmen, wie beispielsweise einen zur Bestimmung des kürzesten Weges, zurück.

2 Analyse der Produktfunktionen

In diesem Kapitel werden die einzelnen Funktionen aus dem Pflichtenheft näher erläutert und analysiert. Hierzu werden die jeweiligen Funktionen in Sequenzdiagrammen dargestellt. Außerdem werden die nichtfunktionalen Anforderungen, sofern möglich, ebenfalls erläutert, analysiert und in Sequenzdiagrammen dargestellt.

2.1 Analyse von Funktionalität F10: Karte erstellen

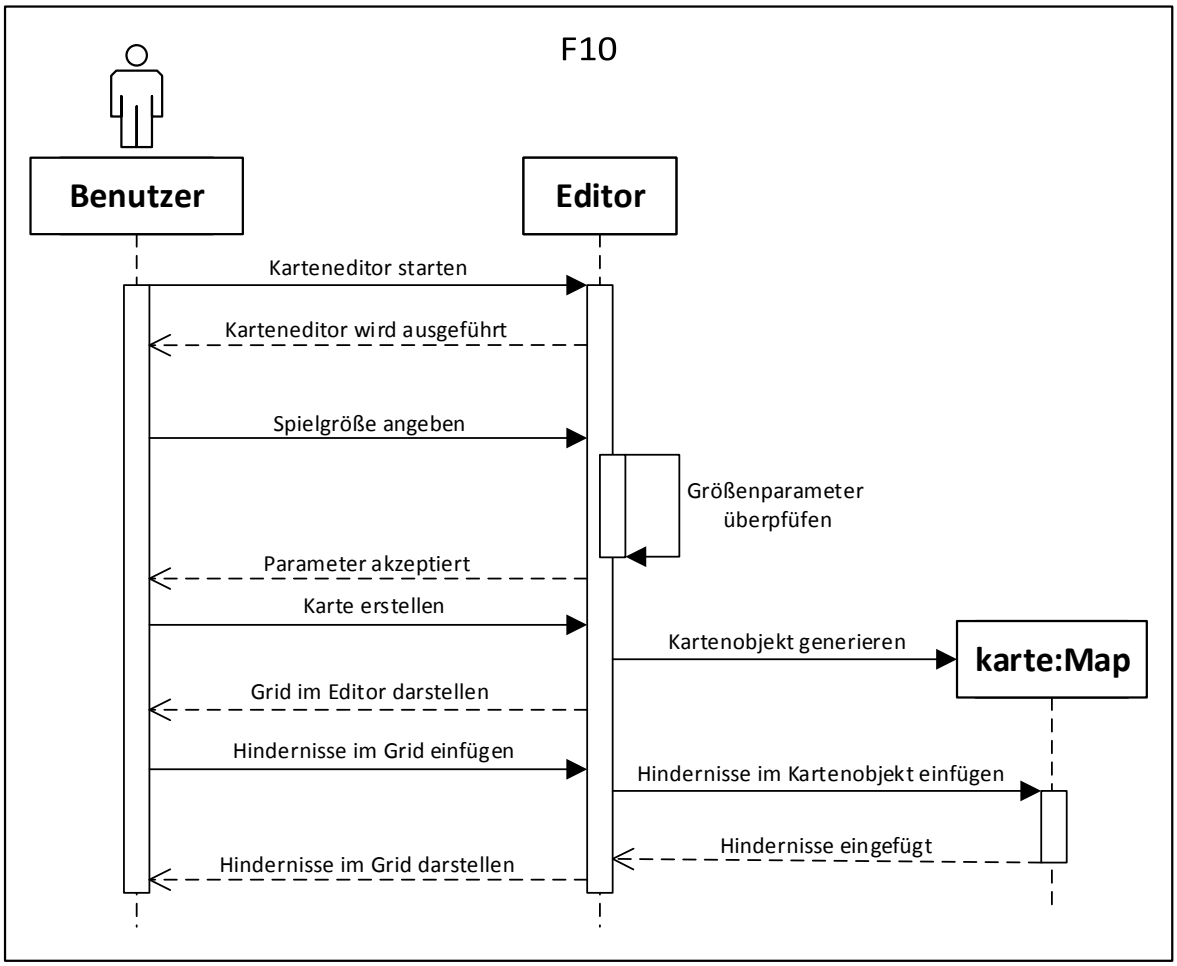

Abbildung 2.1: Sequenzdiagramm zu F10: *Karte erstellen*

Die Funktion F10 *Karte erstellen* ermöglicht dem Benutzer eine Spielkarte zu erstellen. Um eine Spielkarte zu erstellen, muss der Benutzer zunächst den Karteneditor öffnen und die Kartenparameter in den Abfragefeldern *Höhe* und *Breite* eingeben und anschließend mit *Karte erstellen* bestätigen, um das leere Gitternetz im Karteneditor zu erzeugen. Der Benutzer hat die Möglichkeit auf dem Gitternetz Hindernisse hinzuzufügen und muss anschließend die Startpunkte der Roboter und Flaggen definieren. Nach Eingabe der Kartenparameter muss der Benutzer die Spielparameter *Sichtweite*, *Schussweite*, *Roboter-Timeout*, *Leben* und *Punkte zum Sieg* in die vorgesehenen Abfragefelder eintragen. Der Editor prüft in Echtzeit die Eingabe der Parameter und gibt gegebenenfalls einen Fehlerhinweis zurück.

2.2 Analyse von Funktionalität F20: Karte speichern

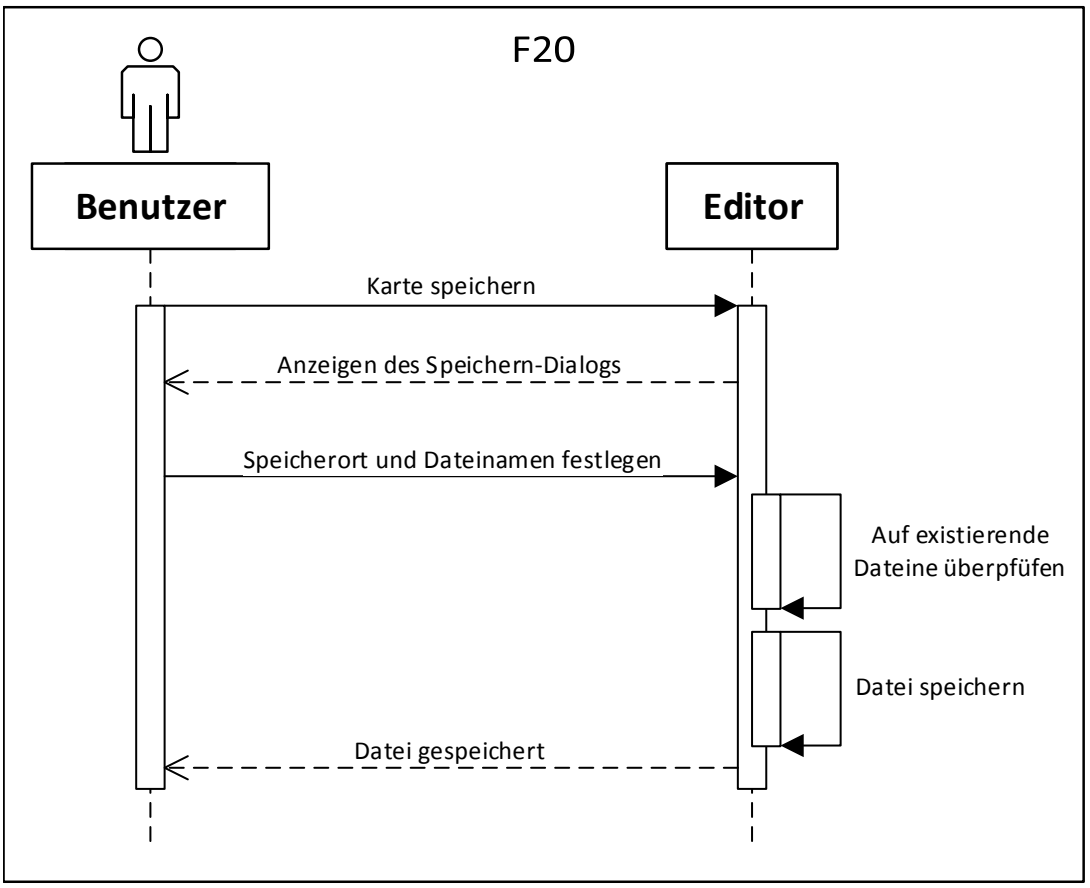

Abbildung 2.2: Sequenzdiagramm zu F20: *Karte speichern*

Die Funktion F20 *Karte speichern* ermöglicht dem Benutzer eine zuvor erstellte Spielkarte zu speichern. Um eine Spielkarte zu speichern, muss der Benutzer die Funktion F10 *Karte erstellen* ausgeführt haben und den Knopf *Karte speichern* drücken. Anschließend kommt der Benutzer in sein Dateisystem, in welchem er den Speicherort und den Namen der Spielkartendatei angibt und seine Eingaben bestätigt. Falls eine Datei schon vorhanden ist, wird der Benutzer darüber informiert. Deswegen muss der Benutzer auswählen, ob er die Datei überschreiben möchte oder nicht. Wenn er sie nicht überschreiben möchte, muss der Benutzer einen alternativen Pfad, bzw. einen anderen Namen zur Speicherung der Datei wählen.

2.3 Analyse von Funktionalität F30: Karte laden

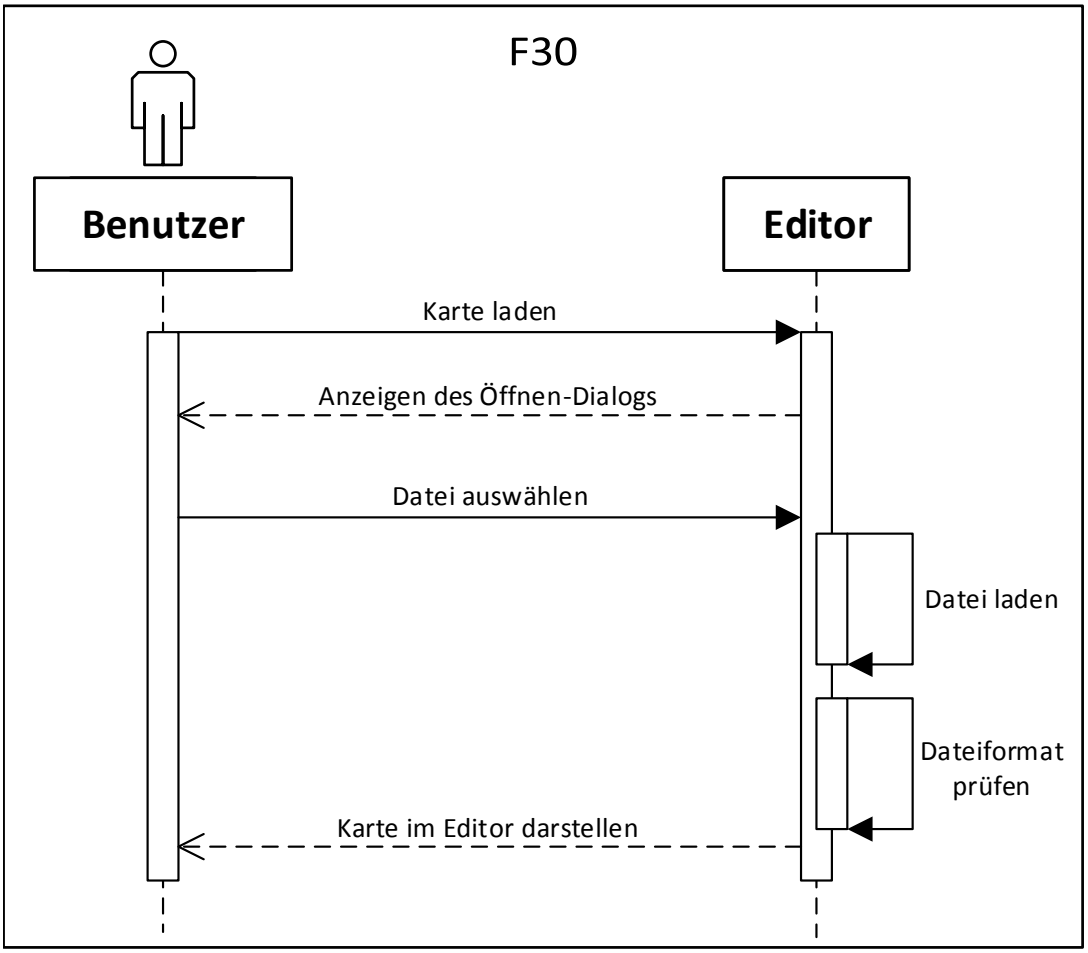

Abbildung 2.3: Sequenzdiagramm zu F30: *Karte laden*

Die Funktion F30 *Karte laden* ermöglicht dem Benutzer eine gespeicherte Spielkarte in den Karteneditor zu laden. Um eine Spielkarte zu laden, muss der Benutzer die Funktion F20 *Karte speichern* ausgeführt haben und den Button *Karte laden* drücken. Anschließend wird das Dateisystem des Benutzers geöffnet und der Benutzer wählt den Dateipfad zu der gewünschten Spielkartendatei aus. Durch das Bestätigen seiner Eingabe wird die Spielkarte in den Karteneditor geladen und alle bereits angegebenen Parameter werden von den Parametern der geladenen Spielkarte überschrieben. Falls das Dateiformat ungültig ist, kann keine Karte angezeigt werden und stattdessen wird im Editor eine Fehlermeldung ausgegeben.

2.4 Analyse von Funktionalität F40: Server bereitstellen

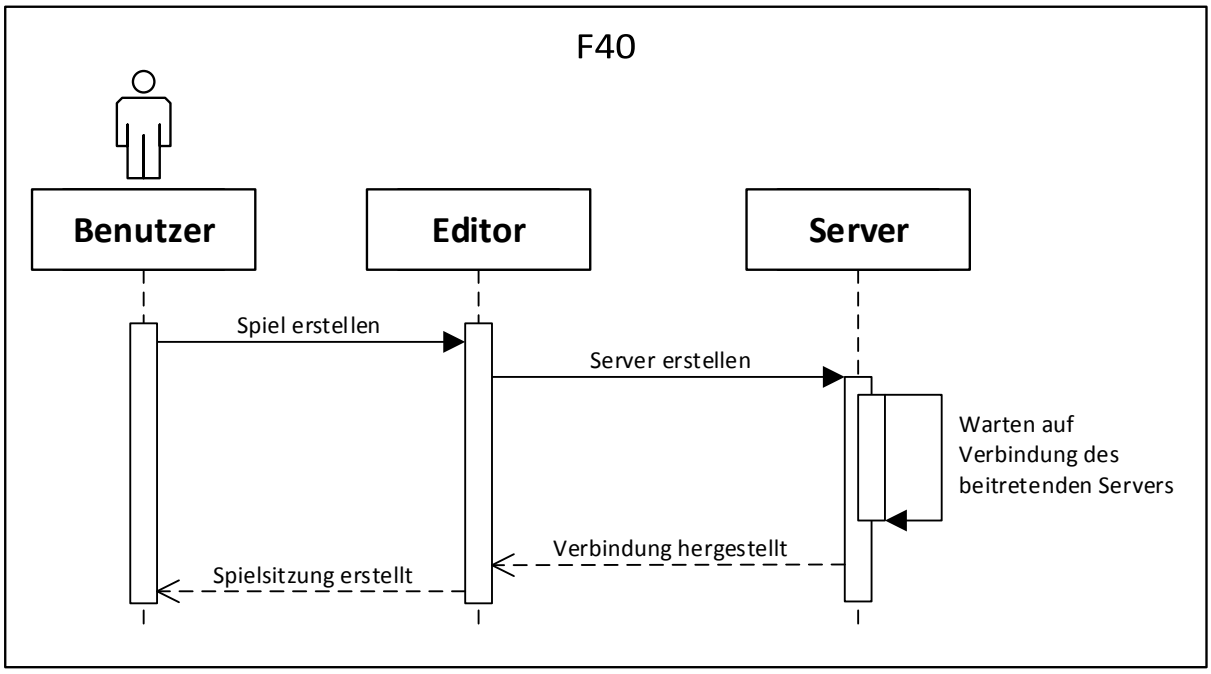

Abbildung 2.4: Sequenzdiagramm zu F40: *Server bereitstellen*

Die Funktion F40 *Server bereitstellen* ermöglicht es eine Schnittstelle für den gegnerischen Server bereitzustellen. Um einen Server bereitzustellen, muss die Funktion F140 *Roboter mit Server verbinden* abgeschlossen sein. Nachdem der Server bereitgestellt wurde, wird auf eine Antwort des anderen Servers gewartet. In diesem Zeitfenster kann das andere Team sich mit dem bereitgestellten Server per Eingabe der IP-Adresse verbinden.

2.5 Analyse von Funktionalität F50: Server beitreten

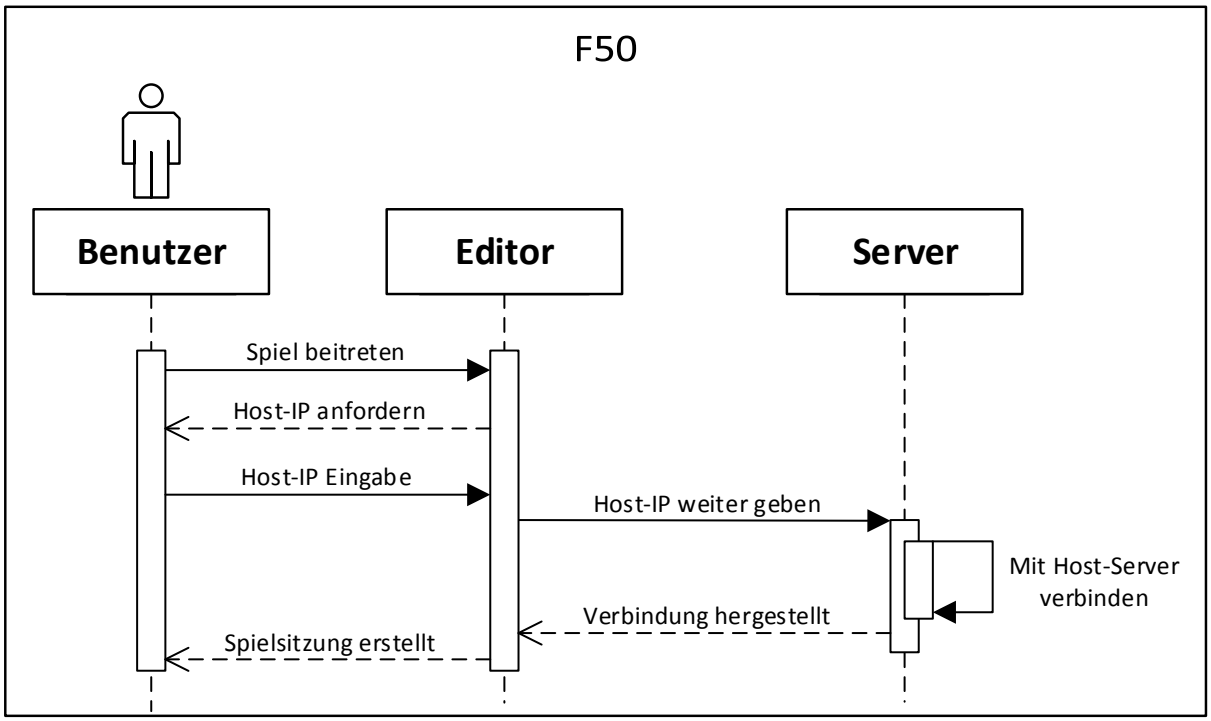

Abbildung 2.5: Sequenzdiagramm zu F50: *Server beitreten*

Die Funktion F50 *Server beitreten* ermöglicht es dem Benutzer einem bestehenden Server beizutreten. Um die Funktion korrekt ausführen zu können, muss das gegnerische Team eine Serverschnittstelle freigeben. Sie bildet die Komponente, mit der der eigene Server kommuniziert. Der Benutzer muss zunächst die IP-Adresse des Host-Servers eingeben. Bei falscher Eingabe wird eine Fehlermeldung ausgegeben. Der Benutzer muss anschließend die IP-Adresse neu eingeben. Bei korrekter Eingabe verbindet sich der eigene Server mit dem Host-Server und tritt der Spielsitzung bei.

2.6 Analyse von Funktionalität F60: Darstellung des Spielgeschehens

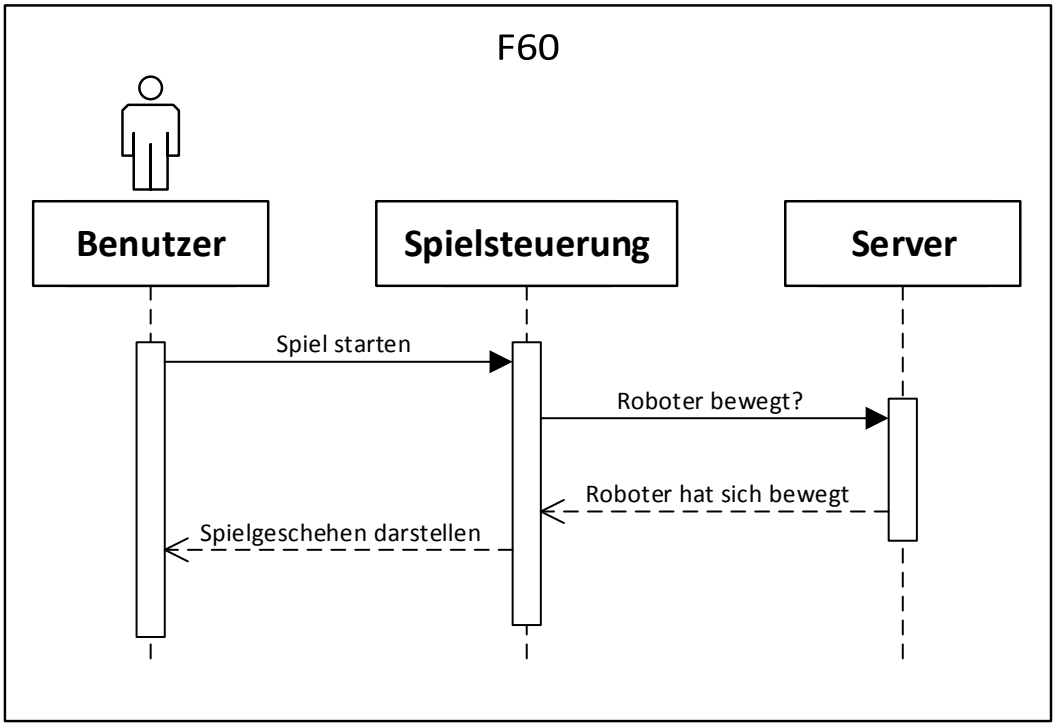

Abbildung 2.6: Sequenzdiagramm zu F60: *Darstellung des Spielgeschehens*

Die Funktion F60 *Darstellung des Spielgeschehens* ermöglicht es dem Benutzer das Spielgeschehen direkt in der GUI der Spielsteuerung zu verfolgen. Während die GUI in Echtzeit das Spielgeschehen wiedergibt, bezieht der eigene Server mit F80 und F90 Spieldaten vom anderen Server, um die Darstellung in der GUI aktuell zu halten.

2.7 Analyse von Funktionalität F70: Manuelle Spielsteuerung

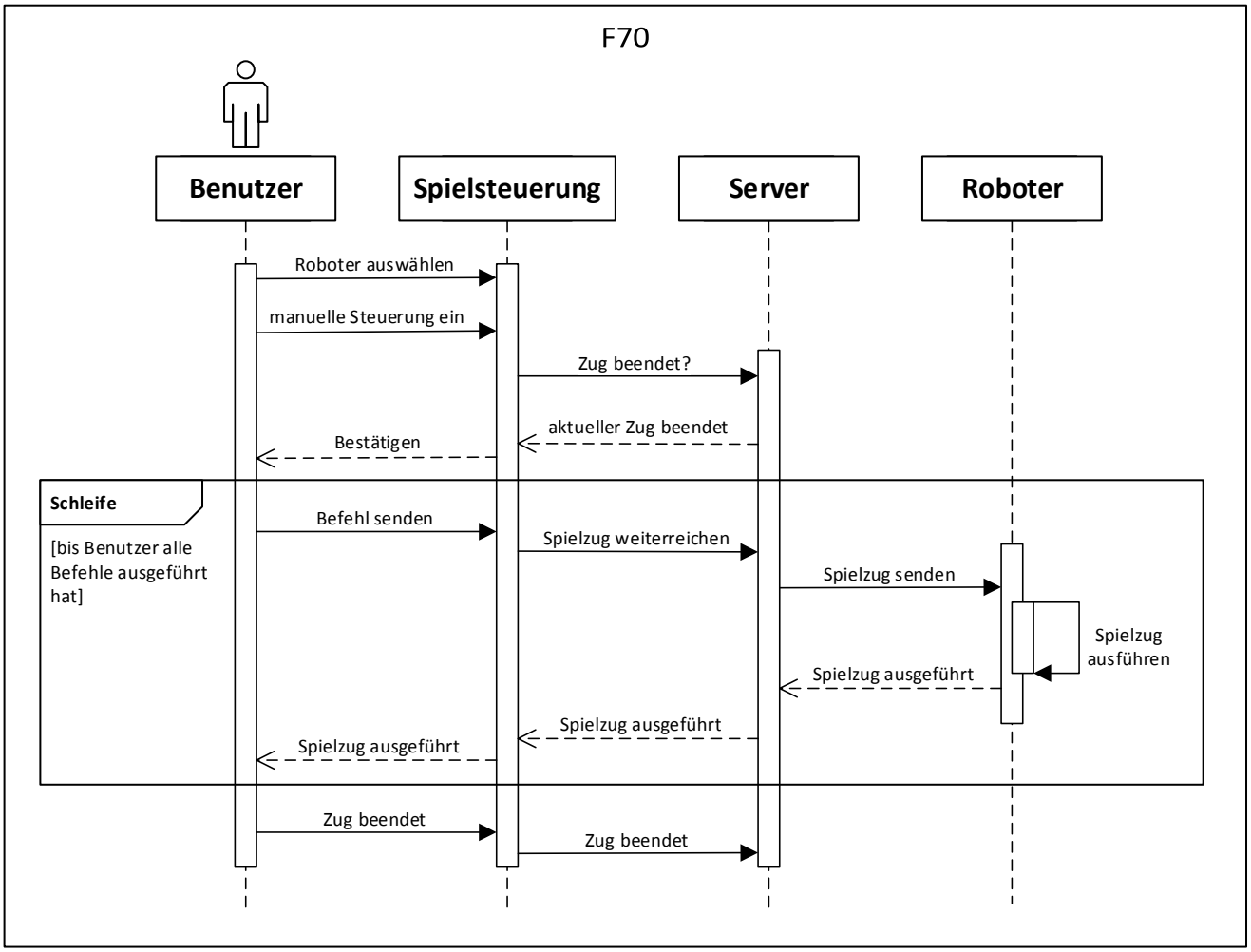

Abbildung 2.7: Sequenzdiagramm zu F70: *Manuelle Spielsteuerung*

Die Funktion F70 *Manuelle Spielsteuerung* ermöglicht es dem Benutzer die Roboter eigenständig zu bewegen und somit aktiv in das Spielgeschehen einzugreifen. Der Benutzer wählt in der GUI den zu bedienenden Roboter aus und aktiviert die manuelle Spielsteuerung. Hiermit wird die KI temporär außer Kraft gesetzt. Um die Spielkontrolle zu übernehmen, muss der aktuelle Spielzug beendet sein. Anschließend kann der Benutzer dem Roboter Befehle übermitteln. Diese führen die Befehle aus und senden eine Bestätigung. Der Benutzer muss am Ende seines Zuges die Spielsteuerung der KI übergeben, indem er seinen Zug für beendet erklärt und die manuelle Steuerung ausschaltet. Dadurch werden dem Server die neuen Spielinformationen übermittelt.

2.8 Analyse von Funktionalität F80: Eigener Server empfängt Datenpakete vom gegnerischen Server

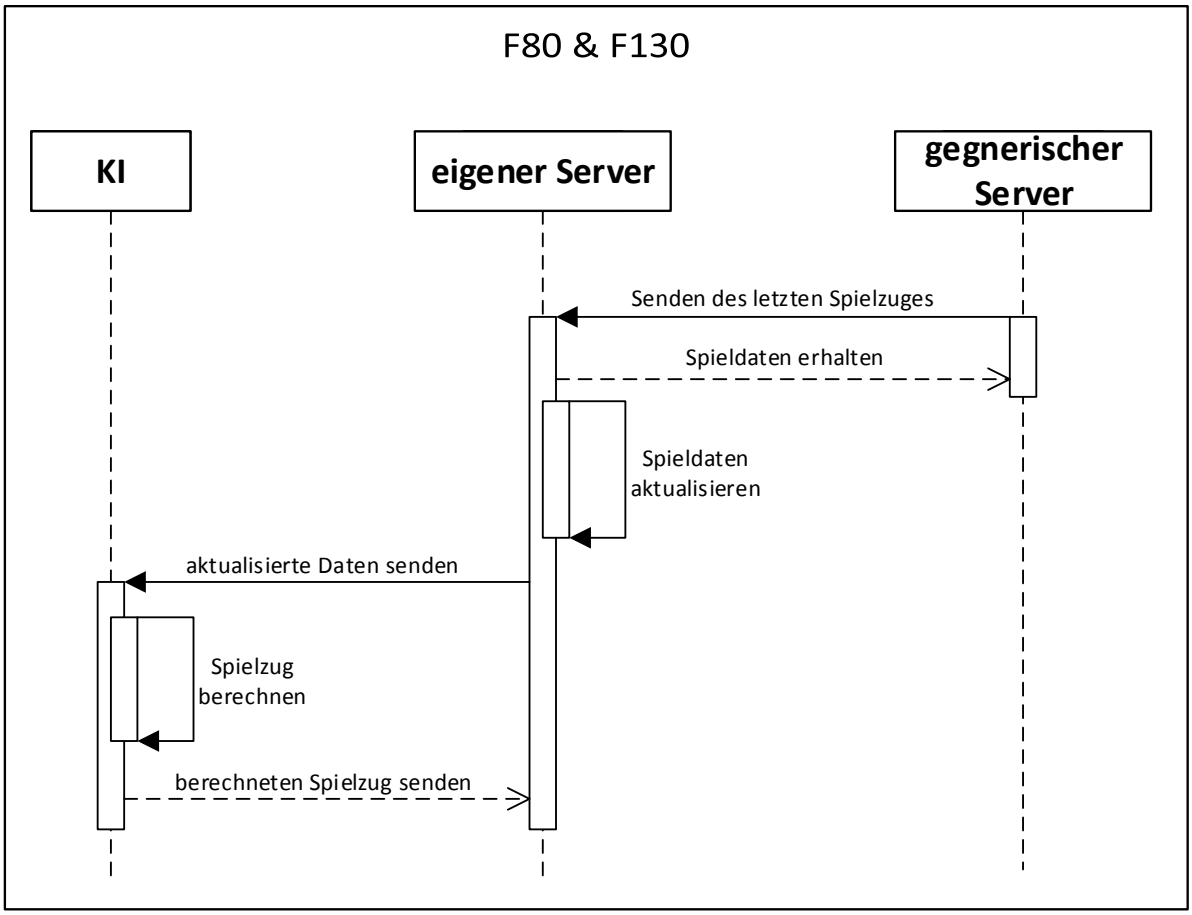

Abbildung 2.8: Sequenzdiagramm zu F80 & F130: *Eigener Server empfängt Datenpakete vom gegnerischen Server* und *KI berechnet Befehle*

Die Funktion F80 *Eigener Server empfängt Datenpakete vom gegnerischen Server* ermöglicht es durch das Empfangen von Spieldaten des gegnerischen Servers das Spielgeschehen auf der GUI zu aktualisieren. Die Spieldaten werden ebenfalls an die KI gesendet, damit diese den nächsten Zug berechnen kann F130.

2.9 Analyse von Funktionalität F130: KI berechnet Befehle

Die Funktion F130 *KI berechnet Befehle* ermöglicht es der KI nach dem Erhalt der Spieldaten aus F80 die Situation zu erkennen, zu analysieren und dementsprechend mit Spielzügen zu reagieren.

2.10 Analyse von Funktionalität F90: Eigener Server sendet Datenpakete zum gegnerischen Server

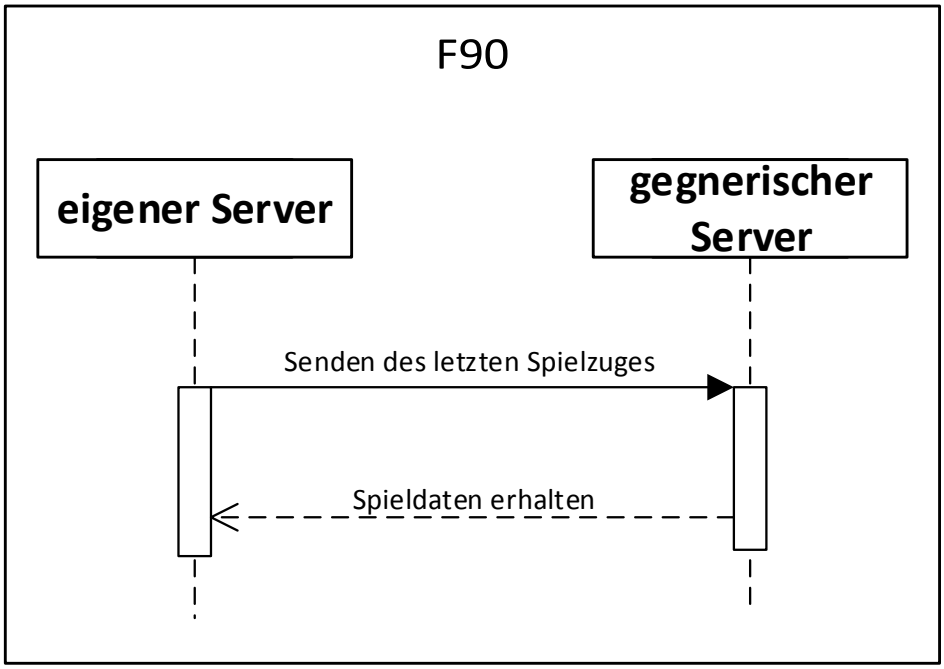

Abbildung 2.9: Sequenzdiagramm zu F90: *Eigener Server sendet Datenpakete zum gegnerischen Server*

Die Funktion F90 *Eigener Server sendet Datenpakete zum gegnerischen Server* ermöglicht es durch das Senden von Spieldaten dem anderen Team den aktuellen Zug zu übermitteln. Die Spieldaten sorgen dafür, dass der gegnerische Server erfährt, was während des Spielzugs geschehen ist. Mit diesen Daten kann die gegnerische KI auf die Situation reagieren. Das Datenpaket besteht pro spielendem Roboter aus jeweils der alten Position eines Roboters, seiner neuen Position nach der Bewegung und seinem Angriffsziel. Am Ende eines Spielzugs wird es an den gegnerischen Server geschickt und ein Paket der selben Form vom gegnerischen Server erwartet, welches sein Rundenende signalisiert.

2.11 Analyse von Funktionalität F100: Befehle an Roboter senden

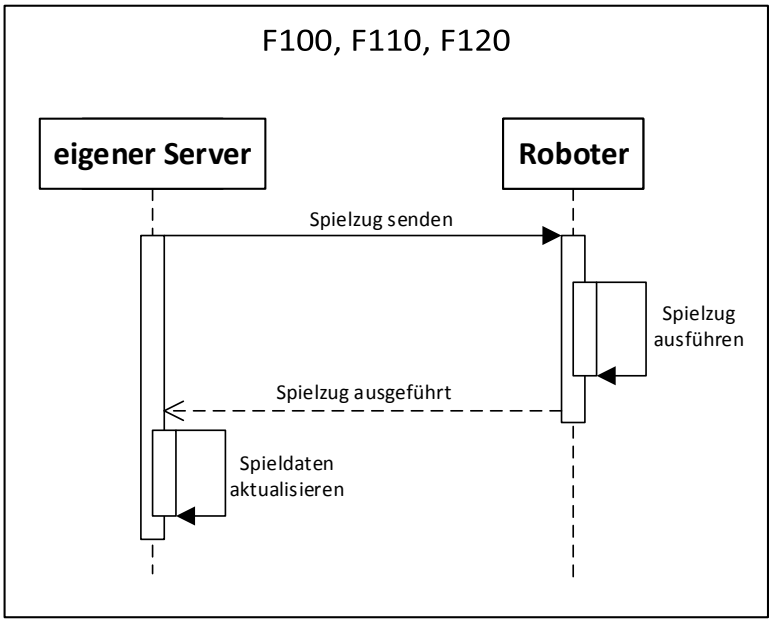

Abbildung 2.10: Sequenzdiagramm zu F100, F110 & F120: *Befehle an Roboter senden*, *Roboter führt Befehle aus* und *Bestätigung an Server senden*

Die Funktion F100 *Befehle an Roboter senden* ermöglicht es die von der KI berechneten Aktionsbefehle an den Roboter zu senden. Der Roboter bekommt alle Aktionen, wie „Fahren“, „Schießen“, „Fahne aufheben“, etc. mitgeteilt, die er in F110 auszuführen hat.

2.12 Analyse von Funktionalität F110: Roboter führt Befehle aus

Die Funktion F110 *Roboter führt Befehle aus* ermöglicht es dem Roboter die in F100 empfangenen Aktionsbefehle auf dem Spielfeld auszuführen. Danach sendet der Roboter eine Bestätigung an den Server (F120).

2.13 Analyse von Funktionalität F120: Bestätigung an Server senden

Die Funktion F120 *Bestätigung an Server senden* ermöglicht es dem Roboter nach Ausführung der Aktionsbefehle aus F110 eine Bestätigung an den Server zu senden, um den Zug zu validieren und zu beenden. Somit wird dem Server die aktuelle Spielsituation mitgeteilt.

2.14 Analyse von Funktionalität F140: Roboter mit Server verbinden

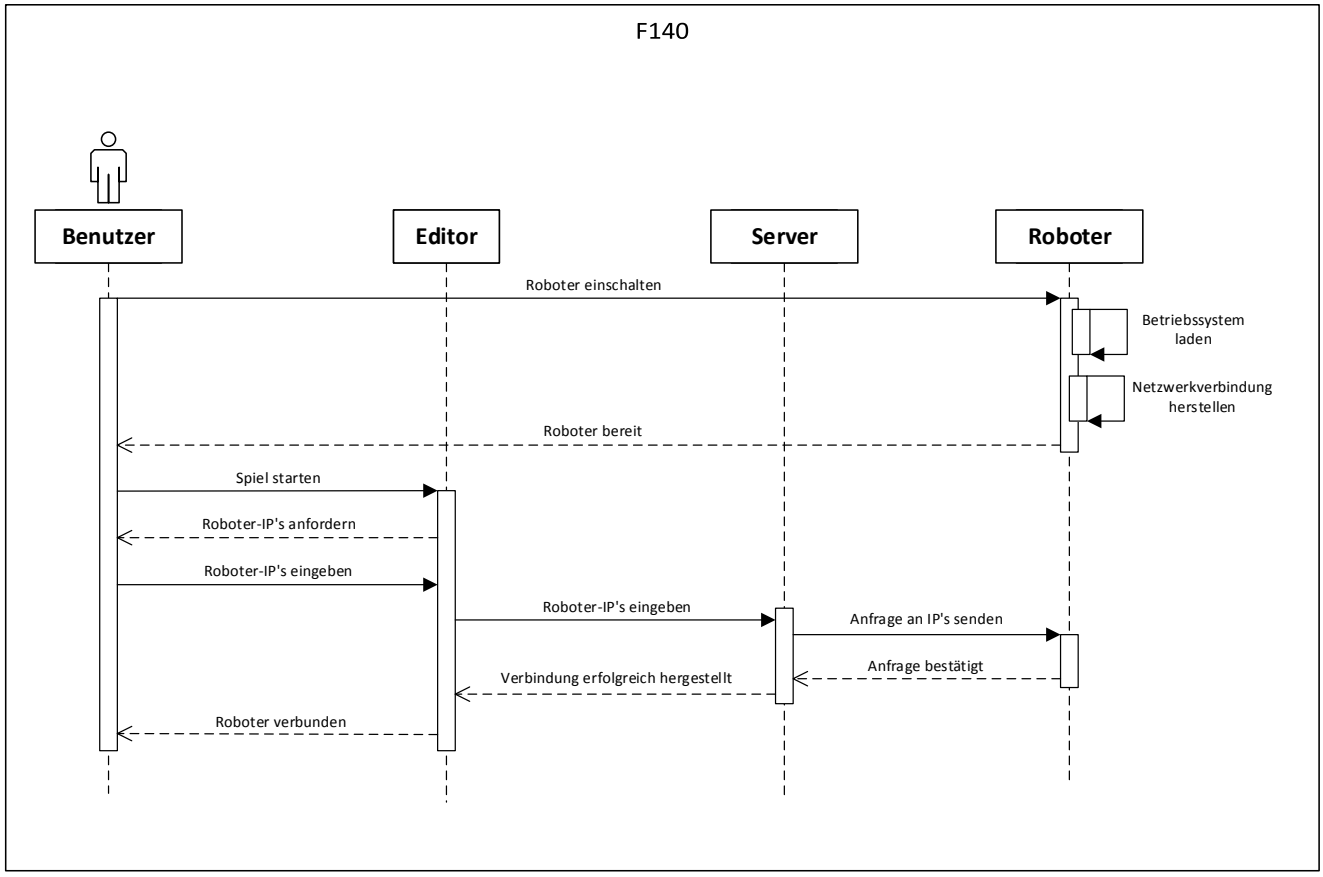

Abbildung 2.11: Sequenzdiagramm zu F140: *Roboter mit Server verbinden*

Die Funktion F140 *Roboter mit Server verbinden* stellt eine Verbindung zwischen Roboter und Server her und initialisiert den Roboter. Dabei wird zunächst der Roboter vom Benutzer eingeschaltet. Der Roboter lädt anschließend das Betriebssystem und verbindet sich automatisch mit dem Netzwerk. Der Benutzer kann nun das Spiel starten, vorausgesetzt die Funktion F10 *Karte erstellen* ist erfüllt, und muss dann die IP-Adresse des Roboters eingeben, welche er von dessen Display ablesen kann. Die IP-Adresse wird an den Server weitergeleitet, der zunächst eine Verbindung mit dem Roboter herstellt, welcher die Anfrage bestätigt. War die Verbindung erfolgreich, so ist das System spielbereit.

3 Datenmodell

Die einzigen Daten, die dauerhaft gespeichert werden und nicht zu den Konfigurationsdateien zählen, sind <E10> Karten.

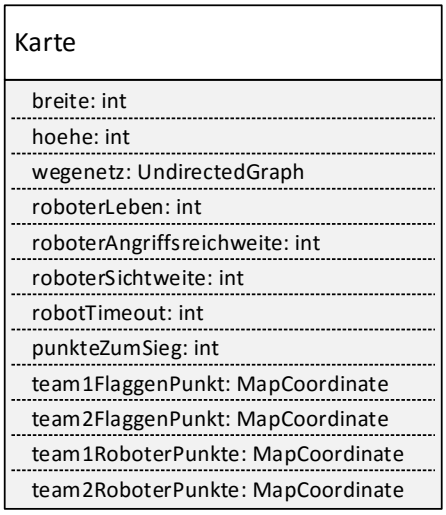

Abbildung 3.1: Klassendiagramm der gespeicherten Karten

In der <E10> Kartendatei werden die Abmessung (maximal 256*256), begehbaren Wege in Form eines ungerichteten Graphen und die Startpunkte der Roboter und Flaggen gespeichert. Außerdem enthält sie die Spielparameter für die Roboter wie Leben, Angriffs- und Sichtweite und Timeout-Rundenzahl, wenn ein Roboter keine Leben mehr hat. Zuletzt wird auch die Zahl der zu gewinnenden Runden bis zum Sieg gespeichert.
Anzumerken ist, dass Wege und Hidnernisse, wie noch in <D10> beschrieben, nun nicht mehr explizit gespeichert werden, sondern implizit aus dem Graphen des Wegenetzes hervorgehen.

4 Konfiguration

Zur Inbetriebnahme der Software müssen die Komponenten Karteneditor.jar und Server.jar im selben Ordner vorliegen. Im Hauptordner müssen sich auch zwei Unterordner befinden, zum einen für die KI's namens "ki" und zum anderen für alle Konfigurationsdateien namens "config". Die KI's, welche mit der KI-API erstellt wurden, müssen in Form von .jar-Dateien in ihrem Unterordner vorliegen.
Dieser spielfähige Zustand des Systems, mit Default-KI und vordefinierten Konfigurationsdateien für das normale Spiel, sind nach Entpacken des Archivs bereits auf dem Rechner vorzufinden.

Folgende Konfigurationsdateien sind im Unterorder "config" zu finden:

- **Roboter.cfg**
  Wird bei Spielstart vom Rechner zu allen Robotern geschickt, die sich damit konfigurieren. Hier ist die Fahrgeschwindigkeit der Roboter, ihre Software-Updateraten und die Anzahl der Querlinien bis zur nächsten Kreuzung veränderbar.

5 Glossar

In diesem Abschnitt werden einige Fachbegriffe erläutert.

Server - Zentrales System, welches auf Verbindungen von Clients reagiert und z.B. Anfagen bearbeitet oder Dateien bereitstellt.

API - Application programming interface Programmierschnittstelle.

Spielsteuerung - Das GUI-Programm zur Visualisierung des Spiels und Steuerung der Roboter Karte Die Karte ist die logische Repräsentation des Spielfeldes im Computer, welches nach dem realen Vorbild durch einen Benutzer modelliert wird.

GUI - Das Graphical User Inteface biete nutzern die Möglichkeit, Software über bspw. Knöpfe und Textfelder zu steuern.

K.I. - Die künstliche Intelligenz ist ein Teilgebiet der Informatik, welches sich mit der Automatisierung intelligenten Verhaltens befasst.

Grid - Das Spielfeld in Form eines Gitters.

IP - Adresse - IP steht für Internetprotokoll. Die IP - Adresse ist eine Adresse die in Computernetzen vergeben wird und ein Gerät adressierbar macht.

Algorithmus - Beschreibt eine eindeutige Handlungsvorschrift. Besteht aus endlich vielen Handlungsschritten die zur Lösung eines Problems beitragen.

Sequenzdiagramm - Ist ein Verhaltensdiagramm. Dieses stellt eine Interaktion grafisch dar. Benutzt wird die Modellierungssprache UML (Unified Modeling Language). Sequenzdiagramme beschreiben den Austausch von Nachrichten zwischen Ausprägungen mittels Lebenslinien.

Host - Server - Ein Host ist ein Dienstrechner, welcher ein in ein Rechnernetz eingebundenes Rechnersystem mit zugehörigem Betriebssystem bezeichnet. Dieser verwaltet die Interaktionen zwischen den einzelnen Klienten und bedient diese. Außerdem beherbergt er Server.

validieren - Die Wichtigkeit, die Gültigkeit, den Wert von etwas feststellen, bestimmen.

Default - Voreinstellung oder Standardwert .

Spieldaten - Geben Informationen über den Spielverlauf. Diese Informationen setzen sich aus den aktuellen Leben der Roboter, der Positionen der Roboter und der Flaggen zusammen.

Kartenparameter - Setzen sich aus den Startpunkten der Flaggen und Roboter, der Sicht- und Schussweite, Lebenspunkte der Roboter, Kartengröße, Hindernisse, Wartezeiten für abgeschossene Roboter und benötigte Punkte bis zum Sieg zusammen.

Basis - Die Startposition der eigenen Flagge.