Software Engineering 1
Abgabedokument
Teilaufgabe 1
(Anforderungsanalyse und Planungsphase)

Persönliche Daten, bitte vollständig ausfüllen:

Nachname, Vorname:	Curanov, Alexandr
Matrikelnummer:	12029720
E-Mail-Adresse:	a12029720@unet.univie.ac.at
Datum:	25.10.2023

Aufgabe 1: Anforderungsanalyse

Analysieren der Spielidee und des Netzwerkprotokolls um 8 Anforderungen (bestehend zumindest aus 3 funktionalen, 3 nichtfunktionalen und einer Designbedingung) nach den folgenden Kriterien zu dokumentieren. Achten Sie darauf den im Skriptum und der Vorlesung behandelten Qualitätsaspekten Genüge zu tun.

Typ der Anforderung: funktional

Anforderung 1

- **Beschreibung:** Spielinitialisierung — Der Server bietet dem Benutzer die Möglichkeit ein neues Spiel zu initialisieren.
- **Bezugsquelle:** Spielidee — «Initial gilt es am Server ein neues Spiel anzufordern. Dieser erste Schritt wird noch von einem Menschen durchgeführt, alles danach erfolgt immer vollautomatisch durch eine Client/KI-Implementierung.»

Anforderung 2

- **Beschreibung:** Kartenerstellung — Die KIs übertragen die von ihnen zufällig generierten Kartenhälften an den Server, welcher diese Karten zu einer grösseren Gesamtspielkarte kombiniert.
- **Bezugsquelle:** Spielidee — «Nach Start des Clients registrieren sich die KIs für das Spiel am Server und erstellen/tauschen danach mit dem Server Kartenhälften aus. Die Karte, auf welcher gespielt wird, ist hierbei nicht fest vorgegeben, sondern entsteht durch die Kombination dieser zufällig erzeugten Kartenhälften durch den Server.»

Anforderung 3

- **Beschreibung:** Schatzplatzierung — Der Server versteckt für jede KI einen Schatz auf ihre jeweiligen Kartenhälfte.
- **Bezugsquelle:** Spielidee — «Hierzu wird je ein Schatz vom Server auf jeder der beiden Kartenhälften versteckt. Der Schatz einer KI ist immer auf der Kartenhälfte zu finden auf welcher die KI startet. »

Anforderung 4

- **Beschreibung:** Benutzerschnittstelle — Das Spiel soll über ein CLI visualisiert werden.
- **Bezugsquelle:** Spielidee — «Während des Spiels müssen die Karte, deren bekannte Eigenschaften und wichtige Spielzustände von den Clients mittels **command-line interface** (CLI) visualisiert werden. »

Typ der Anforderung: nicht funktional

Anforderung 5

- **Beschreibung:** Spieldauer — Ein Spiel darf nicht länger als 320 Spielaktionen dauern.
- **Bezugsquelle:** Spielidee — «Um die Spiele für die Zuschauer spannend zu gestalten, wurde festgelegt, dass ein Spiel insgesamt nicht länger als **320 Spielaktionen (und damit 320 Runden)** dauern darf.»

Anforderung 6

- **Beschreibung:** Bedenkzeit — Jede KI hat maximal 5 Sekunden Bedenkzeit pro Aktion.
- **Bezugsquelle:** Spielidee — «Für jede dieser rundenbasierten Spielaktion hat die KI maximal 5 Sekunden Bedenkzeit.»

Anforderung 7

- **Beschreibung:** Ressourcen Einschränkung — Der Server darf maximal 99 Spiele parallel ausführen.
- **Bezugsquelle:** Netzwerkprotokoll — «Um die Ressourcen des Servers zu schonen gilt die Einschränkung, dass maximal 99 Spiele parallel ausgeführt werden dürfen.»

Typ der Anforderung: Designbedingung

Anforderung 8

- **Beschreibung:** Nachrichtenaustausch — Der Nachrichtenaustausch basiert auf einer Restschnittstelle.
- **Bezugsquelle:** Netzwerkprotokoll — «Die technologische Basis des Nachrichtenaustauschs stellt eine Restschnittstelle dar, daher es wird das HTTP Protokoll verwendet sowie die zugehörigen Operationen GET und POST. »

Aufgabe 2: Anforderungsdokumentation

Dokumentation einer zum relevanten Bereich passenden Anforderung nach dem vorgegebenen Schema. Ziehen Sie eine Anforderung heran, für die alle Bestandteile der Vorlage mit relevantem Inhalt befüllt werden können. Wir empfehlen hierzu eine **funktionale** Anforderung auszuwählen.

Dokumentation Anforderung

- **Name:** Schatzplatzierung
- **Beschreibung und Priorität:** Der Server versteckt für jede KI einen Schatz auf ihre jeweiligen Kartenhälfte. Es braucht ein Mechanismus zur Aktualisierung und Speicherung des Status des Schatzes.

  Priorität: Hoch
- **Relevante Anforderungen:**
  - Das Spiel muss durch eine Person initiiert werden.
  - Die KIs generieren zufällige Kartenhälften.
  - Die Karten werden an den Server geschickt, wo sie zu einer gemeinsamen Gesamtspielkarte kombiniert werden.
- **Relevante Business Rules**:
  - Der Schatz des Spielers ist nur auf einer Kartenhälfte zu finden.
  - Es gibt kein Schatzdiebstahl.
  - Die Kartenhälften besitzen keine nicht erreichbaren Stellen sodass Schatz kann gefunden und genommen werden
  - Ein bereits gefundene Schatz kann nicht mehr erneut gefunden werden.
  - Schatz kann nur auf Wiesenfeldern platziert werden.
  - Es muss vermerkt werden, dass der Schatz gefunden wurde.

- **Impuls/Ergebnis - Typisches Szenario:** [Welche Schritte werden durchlaufen, um das der Anforderung zugehörige Verhalten auszulösen bzw. zu nützen und was ist das unmittelbare Ergebnis jedes Schrittes.]

  ***Vorbedingungen*:**

  o Die Initialisierung des Spiels wurde erfolgreich abgeschlossen.

  o Der Client wird am Server registriert.

  o Der Client bekommt eindeutige SpielerID.

  o Kartenhälften werden generiert/validiert und an Server geschickt.

  o Server retourniert Gesamtspielkarten mit versteckten Schatz.

  o Das Spiel läuft und der Schatz wurde noch nicht gefunden

  ***Hauptsächlicher* Ablauf**:

  o Impuls**:** KI erreicht die Position des Schatzes.

  o Ergebnis: Der Server aktualisiert den Spielstatus, markiert den Schatz als gefunden und benachrichtigt den Spieler.

  o Impuls: Neues Spiel starten

  o Ergebnis: Schatzstatus wird zurück auf nicht gefunden gesetzt.

  o Impuls: KI beginnt mit der Suche nach dem Schatz.

  oErgebnis: Der Server verfolgt die Bewegung des KI und überprüft, ob sich der KI in der Nähe des Schatzes befindet.

  o Impuls: KI erreicht die Position des Bergs

  o Ergebnis: Der Server aktualisiert den Spielstatus des Schatzes, ob es Sichtbar ist oder nicht.

  Nachbedingungen:

  o KI wird benachrichtigt, dass der Schatz gefunden ist.

- **Impuls/Ergebnis - Alternativszenario:** [Ein Alternativszenario, das vergleichbare Vorbedingungen aufweist, jedoch sich im Ablauf und Nachbedingungen unterscheidet].

  Vorbedingungen:

  o Die Initialisierung des Spiels wurde erfolgreich abgeschlossen.

  o Der Client wird am Server registriert.

- Der Client bekommt eindeutige SpielerID.
- Kartenhälften werden generiert/validiert und an Server geschickt.
- Server retourniert Gesamtspielkarten mit versteckten Schatz.
- Das Spiel läuft und zwei KI nähern sich dem Schatz.

***Hauptsächlicher* Ablauf**:

- Impuls: Zwei KI erreichen fast gleichzeitig die Schatzposition.
- Ergebnis: Der Server bestätigt den Fund für den zuerst ankommenden KI und aktualisiert den Spielstatus entsprechend.

Nachbedingungen:

- KI hat den Schatz erfolgreich gefunden
- Zweite KI erreicht die Schatzposition

- **Impuls/Ergebnis - Fehlerfall:** [Erwarteten Fehlerfall beschreiben und wie dieser gehandhabt wird. Dies ist besonders relevant, da ansonsten nur Gutfälle berücksichtigt werden was im Produktiveinsatz schnell nicht mehr ausreichend ist.]

Vorbedingungen:

- Die Initialisierung des Spiels wurde erfolgreich abgeschlossen.
- Der Client wird am Server registriert.
- Der Client bekommt eindeutige SpielerID.
- Kartenhälften werden generiert/validiert und an Server geschickt.
- Server retourniert Gesamtspielkarten mit versteckten Schatz.

***Hauptsächlicher* Ablauf**:

- Impuls**:** KI näherst sich an der Position des Schatzes jedoch landet ins Wasser.
- Ergebnis: Der Server aktualisiert den Spielstatus, markiert den Status des KI als verloren.

Nachbedingungen:

- Das Spiel wird beendet und die Clients werden informiert ob sie gewonnen oder verloren haben.

- **Benutzergeschichten:**
  - o Als KI möchte ich den Aktuellen Spielstatus vom Server abfragen zu können, um die Position des Schatzes zu bekommen, falls es aufgedeckt wird.
  - o Als KI möchte ich, dass der Server die Information speichert, dass der Schatz aufgedeckt wurde und/oder aufgenommen wurde.
  - o Als Anwender möchte ich, dass die Kartenhälften zufällig generiert werden um das Spiel spannender zu machen.
  - o Als Server möchte ich, den Aktuellen Status des Schatzes speichern und verwalten.

- **Benutzerschnittstelle:**

```
… … … … … ~ … ^ …
… [B] … … … ~ … ^ …
… M … … ^ ^ ^ ^ …
… ~ … … $ … … ^ …
~ ~ … … ~ ~ … ^ …
```

- **Externe Schnittstellen:**
  - o Kommunikation zwischen Server und Client: Diese Schnittstelle ermöglicht die Kommunikation und verwendet HTTP-Anfragen (GET, POST).

Aufgabe 3: Architektur entwerfen, modellieren und validieren

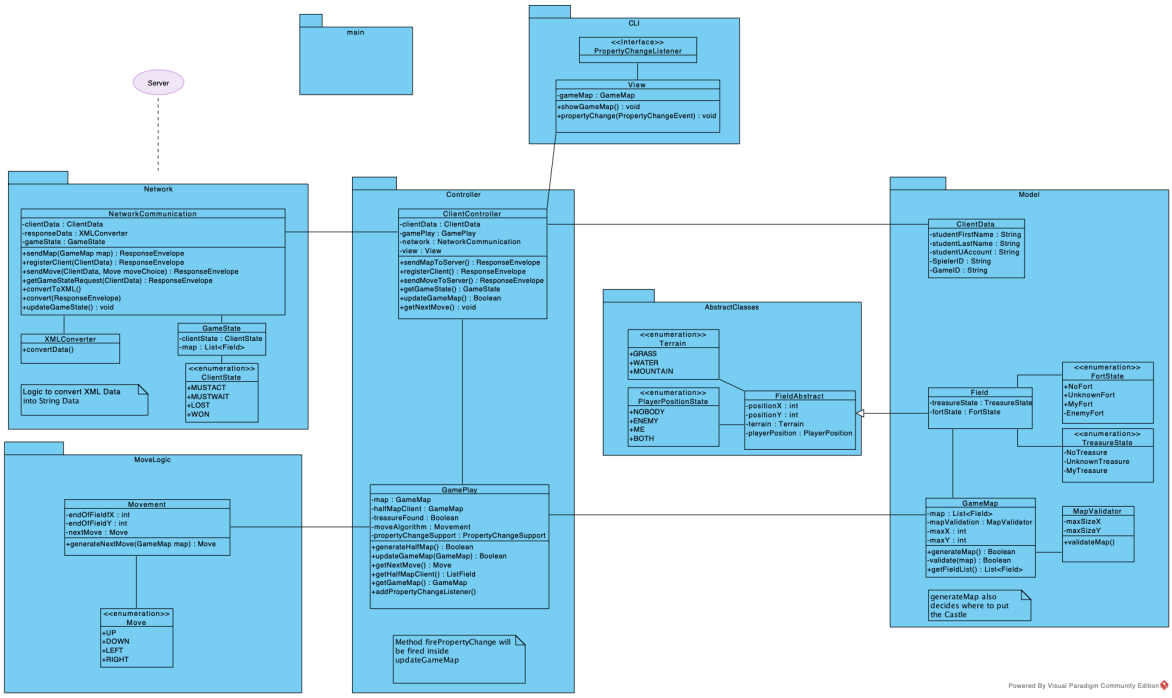
Modellieren Sie händisch alle notwendigen Packages, Klassen und deren Methoden (samt Beziehungen) als zwei UML Klassendiagramme. Achten Sie darauf, dass die Modelle sinnvoll benannte Packages, **Klassen**, **Methoden** (inkl. Parameter und Rückgaben) und **Felder** beinhalten und die Vorgaben der Spielidee bzw. des Netzwerkprotokolls vollständig in sinnvoller Granularität abgedeckt werden.

Basierend auf dem Klassendiagramm: Erstellen Sie zwei Sequenzdiagramme zu den beiden in der Übungsangabe vorgegebenen Aspekten. Alle erstellten Diagramme sollten semantisch und syntaktisch korrekt sowie untereinander konsistent sein.

Wir bieten hierzu Unterstützung:

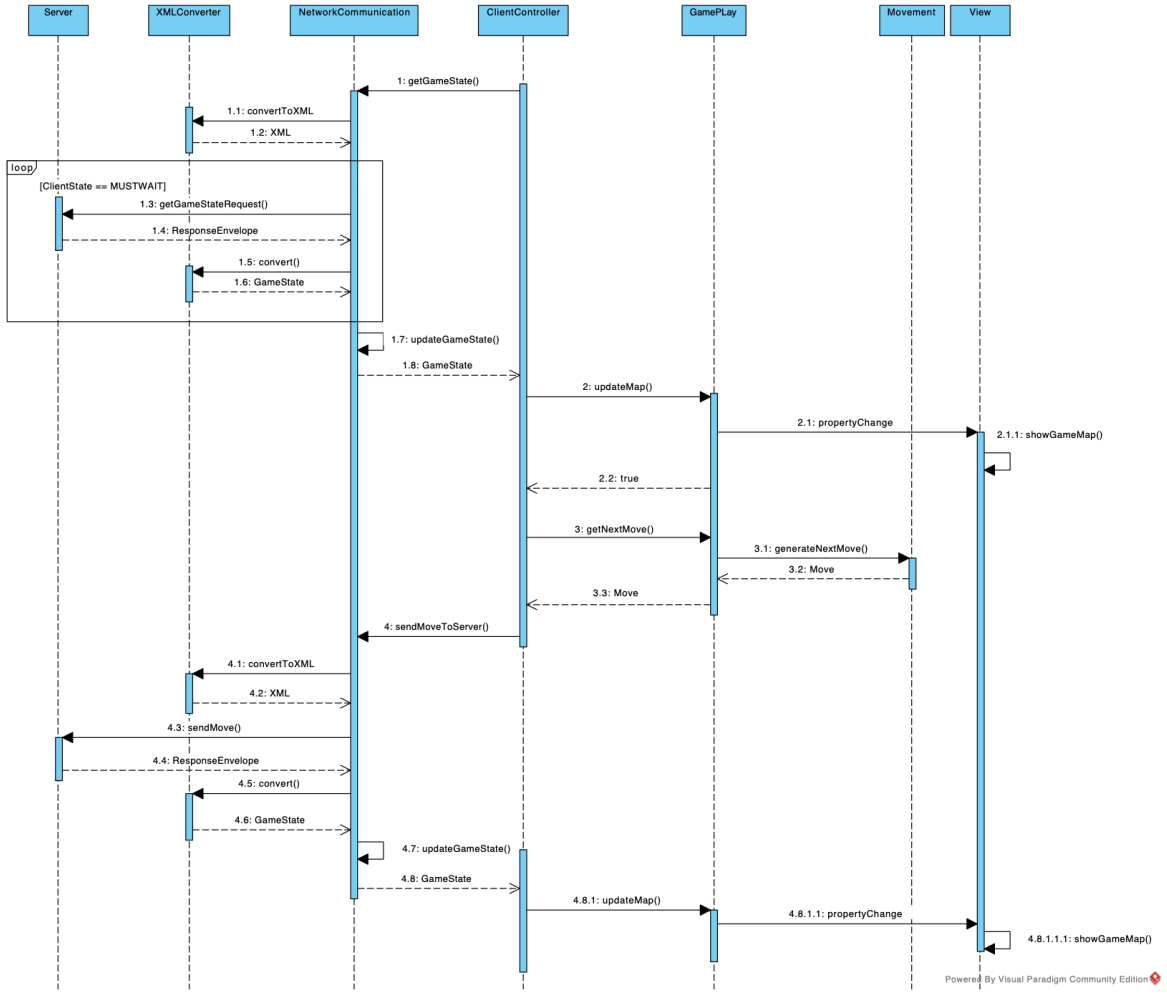
- Wie man die Modellierung und den Entwurf einer Architektur angeht wird im zugehörigen *Tutorial* besprochen samt einer *Ausgangsbasis* für den Client. Zum Nachlesen ist in der Angabe ein passendes Skriptum mit Beispielentwurf hinterlegt. Wenn Sie Ihr UML-Modellierungswissen ausbauen möchten/ müssen wird im Skriptum zusätzlich auch dazu passende Literatur referenziert.
- Beachten Sie zur Ausarbeitung das auf Moodle zur Verfügung gestellte auszugsweise abgebildete Diagrammbeispiel. Dieses sollte als Vorlage dienen, und verdeutlicht welche Erwartungen an das Klassendiagramm beziehungsweise die dazugehörigen Sequenzdiagramme gestellt werden (Darstellung, Inhalte, Detailgrad etc.) und wie diese zusammenhängen.
- Für die Modellierung von Model-View-Controller (nur für Client verpflichtend) gibt es ein fertiges Beispieldiagramm in den Unterlagen. Sie können dieses als Grundlage heranziehen und für Ihre Architektur anpassen/integrieren.
- Für die Modellierung der für das Netzwerk relevanten Klassen gibt es im Netzwerkprotokoll auf Moodle eine Anleitung. Es ist nicht notwendig Netzwerknachrichten als Datenklassen in die Diagramme aufzunehmen. Wenn diese an einer Stelle genützt werden (z.B. als Parameter) kann der Namen der Klassen als Type angeführt werden. *Empfehlung*: Eigene auf konkrete Use Cases spezialisierte (und deshalb besser geeignete) Datenklassen zu erstellen statt nur die bereitgestellten Netzwerkklassen zu „kopieren“ da diese nur auf den Datenaustauch fokussiert sind.
  Im Netzwerkprotokoll wird auch der Ablauf (Wer, Wann, Was) der Interaktion zwischen Clients/Server dargestellt. Nützen Sie das um diesen nachvollziehen und vor allem um diesen Ablauf auch in den Diagrammen zu berücksichtigen und auch nicht wesentliches für die Sequenzdiagrammabläufe zu übersehen.

Client Class Diagramm:



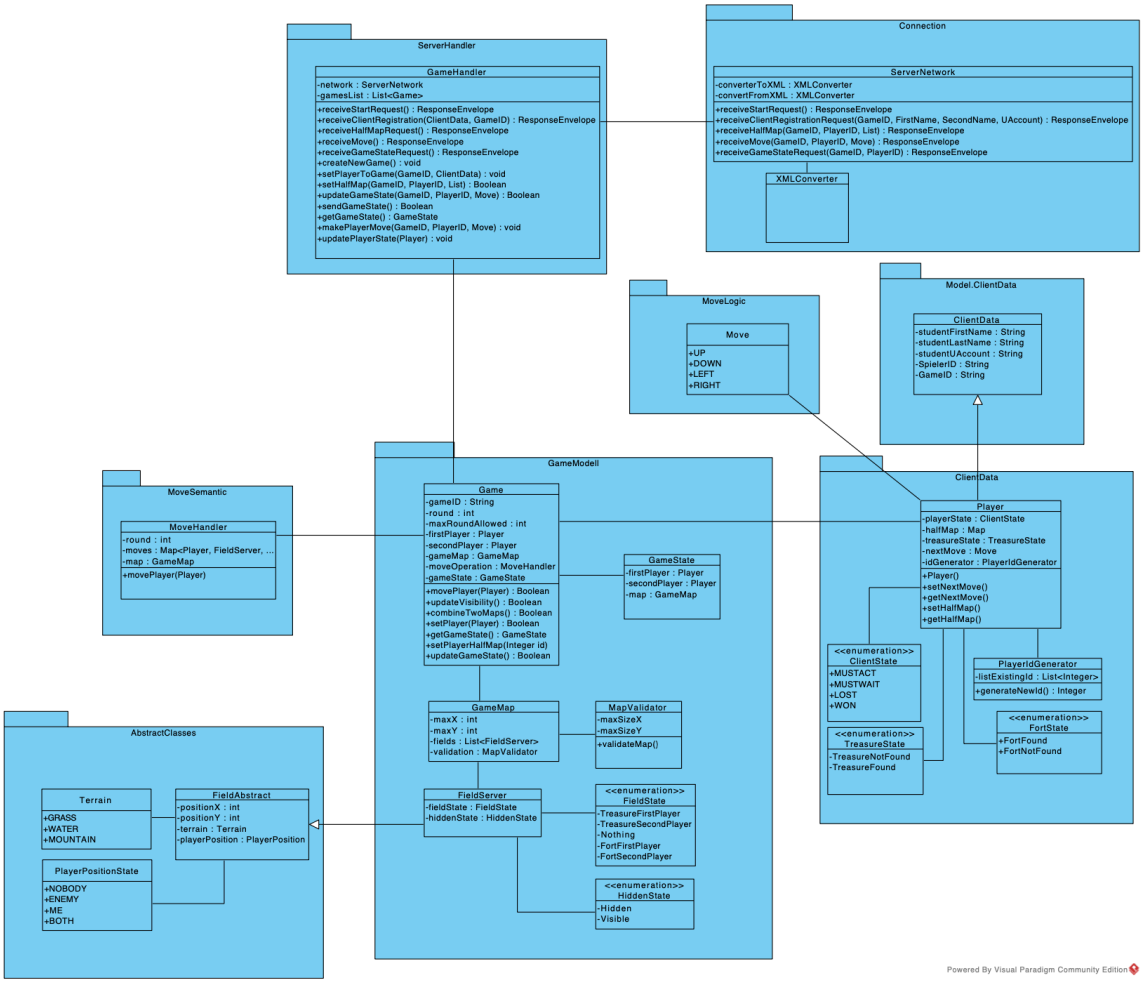
Die ClientController Klasse übernimmt die Steuerung der Kommunikation mit dem Server, die Implementierung der eigentlichen Spiellogik sowie die Darstellung der Benutzeroberfläche über die Command Line Interface. Die NetworkCommunication Klasse ist für die direkte Interaktion mit dem Server verantwortlich. Sie kümmert sich um das Senden und Empfangen von Daten, wobei sie auch die Aufgabe übernimmt, die übertragenen Daten entsprechend zu konvertieren.

Client Sequence Diagramm



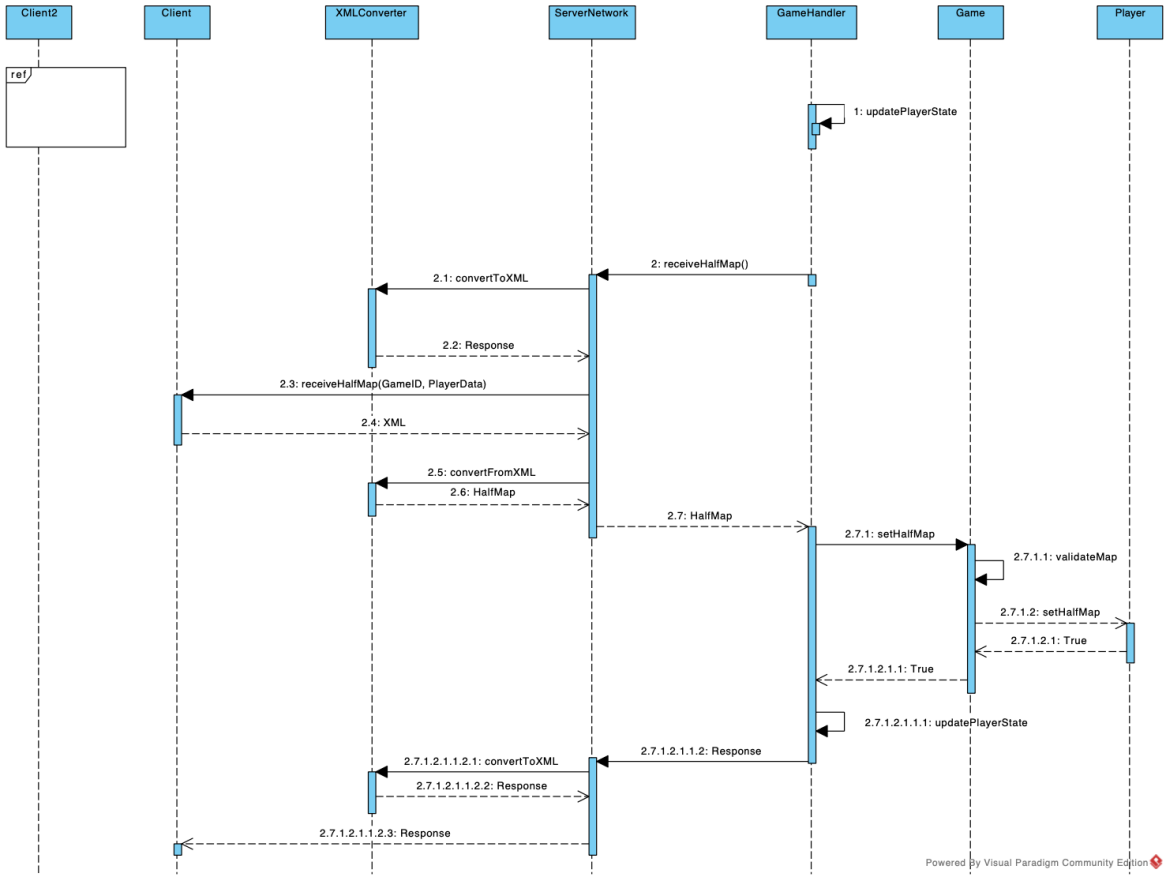
Hier wird genau gezeigt wie ClientController Klasse die ganze Steuerung übernimmt.

Server Class Diagramm



Die „GameHandler“-Klasse übernimmt die zentrale Steuerung des Spiels und bindet verschiedene Client-Klassen durch Imports ein.

Server Sequence Diagramm



Hier wird genau gezeigt wie GameHandler Klasse die ganze Steuerung übernimmt.