Software Engineering 1
Abgabedokument
Teilaufgabe 1
(Anforderungsanalyse und Planungsphase)

Persönliche Daten, bitte vollständig ausfüllen:

Nachname, Vorname:	Vultur Ilinca
Matrikelnummer:	11925311
E-Mail-Adresse:	a11925311@unet.univie.ac.at
Datum:	März 2022

Aufgabe 1: Anforderungsanalyse (2 Punkte)

Typ der Anforderung: funktional

Anforderung 1

- **Beschreibung:** Aufteilung der Burgen auf die Karte **-** Der Client muss während der Generierung eigener Kartenhälfte ein Burg auf einen Wiesenfeld platzieren.
- **Bezugsquelle:** Spielidee: “können die Burg und der Schatz nur auf Wiesenfeldern […] platziert werden. […] Schätze werden, anders als Burgen, nicht während der Generierung der Kartenhälften von den Spielern (genauer, den künstlichen Intelligenzen) platziert, sondern automatisch und zufällig von einer dritten unabhängigen Instanz (dem Server)“

Anforderung 2

- **Beschreibung:** Die Aufteilung der Schätze auf die Karte - Der Server muss auf je Hälfte der Karte genau ein Schatz auf einem Wiesenfeld verstecken.
- **Bezugsquelle:** Spielidee: “Hierzu wird vom Server jeweils ein Schatz auf jeder Hälfte der Karte versteckt. […] können die Burg und der Schatz nur auf Wiesenfeldern […] platziert werden.“

Anforderung 3

- **Beschreibung:** Sicht aus einem Bergfeld - Wenn der Client sich auf einem Bergfeld befindet, sieht er, zusätzlich zum Bergfeld selbst, auch alle Felder mit einer Entfernung von eins (auch diagonal).
- **Bezugsquelle:** Spielidee: “[…] decken Bergfelder dafür Schätze und Burgen in bis zu einem Feld Entfernung rund um den Berg auf. Daher zusätzlich zum Bergfeld selbst werden alle Felder mit einer Entfernung von eins (auch diagonal) aufgedeckt die sich rund um die Spielfigur bzw. den gerade betretenen Berg befinden […] “

Anforderung 4

- **Beschreibung:** Übertragung einer Kartenhälfte - Der Client muss nach der Registrierung eigenes Spielers eine 4x8 Feldern Kartenhälfte generieren und übertragen.
- **Bezugsquelle:** Spielidee: “Karte wird bei Beginn des Spiels von beiden beteiligten künstlichen Intelligenzen kooperativ erstellt. Hierzu erstellt jede der beiden KIs zufällig eine Hälfte der finalen Spielkarte (mit je 4 x 8 Feldern […] “

Typ der Anforderung: nicht funktional

Anforderung 5

- **Beschreibung:** Zeichenlänge eines SpielIDs - Der Server muss bei einer Anfrage des Clients zur neues Spielgenerierung ein SpielID mit exakt 5 Zeichen generieren.
- **Bezugsquelle:** Netzwerkprotokoll: “Die in XML definierte Nachricht beinhaltet als Hauptbestandteil ein Element namens uniqueGameID die eine eindeutige SpielID beinhaltet (diese ID muss aus exakt 5 Zeichen bestehen). ”

Anforderung 6

- **Beschreibung:** Maximale Denkzeit eines KIs - Der Client kann pro Runde maximal 3 Sekunden zur Berechnung und Durchführung der Spielaktion (Bewegung oder Kartengenerierung) verwenden.
- **Bezugsquelle:** Spielidee: “eine KI für jede dieser rundenbasierten Spielaktion nicht mehr als 3 Sekunden Bedenkzeit zur Berechnung erhält (relevant sind hierbei Spielerbewegung und Kartengenerierung) ”

Anforderung 7

- **Beschreibung:** Zeitspanne zwischen 2 Spielstatus Abfragen - Der Client muss mindestens 0,4 Sekunden zwischen zwei Abfragen zum Spielstatus warten.
- **Bezugsquelle:** Netzwerkprotokoll: “Um zu verhindern, dass der Server überlastet wird sollte zwischen zwei vom gleichen Client durchgeführten Abfragen zum Spielstatus mindestens eine Zeitspanne von 0,4 Sekunden vergehen.”

Typ der Anforderung: Designbedingung

Anforderung 8

- **Beschreibung:** Die Protokoll des Nachrichtenaustauschs - Der Austausch von Nachrichten zwischen Client und Server wird mittels den HTTP Protokoll und den zugehörigen Operationen GET und POST durchgeführt.
- **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 (2 Punkte)

Dokumentation Anforderung

- **Name:** Generierung und Übertragung einer Kartenhälfte

- **Beschreibung und Priorität:** Der Client muss nach der Registrierung eigenes Spielers eine 4x8 Feldern Kartenhälfte generieren und übertragen.

- **Priorität:** Hoch

- **Relevante Anforderungen:**

1. Aufteilung der Burgen auf die Karte **-** Der Client muss während der Generierung eigener Kartenhälfte ein Burg auf einen Wiesenfeld platzieren.

2. Die Aufteilung der Schätze auf die Karte - Der Server muss auf je Hälfte der Karte genau ein Schatz auf einem Wiesenfeld verstecken.

3. Maximale Denkzeit eines KIs - Der Client kann pro Runde maximal 3 Sekunden zur Berechnung und Durchführung der Spielaktion (Bewegung oder Kartengenerierung) verwenden.

4. Die Protokoll des Nachrichtenaustauschs - Der Austausch von Nachrichten zwischen Client und Server wird mittels den HTTP Protokoll und den zugehörigen Operationen GET und POST durchgeführt.

- **Relevante Business Rules**:

1. Der Server muss sicherstellen, dass alle Regeln bezüglich der Generierung der Karten eingehalten wurden. Dazu müssen beiden Server und Client den HTTP Protokoll implementieren.

2. Im Fall von einem fehlerhaftes Anfrage an der Server, wird der Client informiert, indem die exceptionName and exceptionMessage Felder der ResponseEnvelope gefüllt werden. Der “state” Element wird auch entsprechend mit “Error” gefüllt.

3. Die Datenaustausch wird im Form von XML durchgeführt.

4. Eine Karte muss so generiert werden, dass es mindestens 3 Bergfelder, 15 Wiesenfelder und 4 Wasserfelder beinhaltet. Dabei darf jede Kartengrenze maximal 3 Wasserfelder an den langen Seite und 1 Wasserfeld an den kurzen Seite beinhalten.

- **Impuls/Ergebnis - Typisches Szenario:**

***Vorbedingungen*:**

- o Es muss ein Spiel auf dem Server bereits erstellt worden sodass der eindeutige SpielID dem Client bekannt ist .
- o Es muss sein, dass der jeweilige Client der die Kartenhälfte momentan generieren und übertragen will, schon registriert ist und die entsprechende SpielerID verfügbar ist.
- o Der Client über den wir gerade sprechen ist dran.

***Hauptsächlicher* Ablauf**:

- o Impuls: Der Client sendet eine korrekte Anfrage an den Server, die alle notwendigen Informationen für das Generieren der Kartenhälfte enthält.
- o Ergebnis: Der Sender antwortet mit einem responseEnvelope.
- o Impuls: Der Client fragt nach dem Spielstatus ab.
- o Ergebnis: Der Server antwortet nochmals mit einem responseEnvelope. Der responseEnvelope beinhaltet unter anderen das Element “map” die alle Informationen über die Kartenhälfte.

Nachbedingungen:

- o Die Kartenhälfte müssen übertragen worden.
- o Der andere Client muss seine Kartenhälfte generieren und übertragen bevor es zu anderen Befehlen kommen kann.

- **Impuls/Ergebnis - Alternativszenario:**

Vorbedingungen:

- o Es muss ein Spiel auf dem Server bereits erstellt worden sodass der eindeutige SpielID dem Client bekannt ist .
- o Es muss sein, dass der jeweilige Client der die Kartenhälfte momentan generieren und übertragen will, schon registriert ist und die entsprechende SpielerID verfügbar ist.
- o Es muss sein, dass der andere Client bereits seine eigene Kartenhälfte generiert und übertragen hat.
- o Der Client über den wir gerade sprechen ist dran.

***Hauptsächlicher* Ablauf**:

o Impuls: Der Client sendet eine korrekte Anfrage an den Server, die alle notwendigen Informationen für das Generieren der Kartenhälfte enthält.

o Ergebnis: Der Sender antwortet mit einem responseEnvelope.

o Impuls: Der Client fragt nach dem Spielstatus ab.

o Ergebnis: Der Server antwortet nochmals mit einem responseEnvelope. Der responseEnvelope beinhaltet unter anderen das Element “map” die alle Informationen über die vollständige Karte.

Nachbedingungen:

o Der Client der nach dem Spielstatus abfragt wird auch angekündigt, ob er dran ist. Wenn er dran ist, muss er eine Bewegung durchfuhren. Wenn nicht, muss er warten.

- **Impuls/Ergebnis - Fehlerfall:**

Vorbedingungen:

oEs muss ein Spiel auf dem Server bereits erstellt worden sodass der eindeutige SpielID dem Client bekannt ist .

o Es muss sein, dass der jeweilige Client der die Kartenhälfte momentan generieren und übertragen will, schon registriert ist und die entsprechende SpielerID verfügbar ist.

o Es muss sein, dass der andere Client bereits seine eigene Kartenhälfte generiert und übertragen hat.

o Der Client über den wir gerade sprechen ist dran.

***Hauptsächlicher* Ablauf**:

oImpuls: Der Client sendet keine Anfrage an den Server während der Bedenkzeit von 3 Sekunden.

o Ergebnis: Dieser Client hat verloren.

oImpuls: Der andere Client fragt nach dem Spielstatus ab.

o Ergebnis: Der Server informiert diesen Client dass er gewinnen hat.

o Impuls: Der ursprünglicher Client fragt nach dem Spielstatus ab.

o Ergebnis: Der Sender informiert den Client, dass er verloren hat.

Nachbedingungen:

o Das Spiel wird aus dem Server eliminiert.

- **Benutzergeschichten:**

1. Als Anwender möchte ich einen korrekt erstellte Kartenhälfte besitzen um an einen spannendes zwei Spieler Spiel teilnehmen zu können.
2. Als Client möchte ich im Fall von einer fehlerhaftes Anfrage an der Server eine vollständige und intuitive Fehlermeldung bekommen.
3. Als Server möchte ich zwei Kartenhälften bekommen die alle definierten Business Rules umsetzen.

- **Benutzerschnittstelle:**

Sobald beide Clients ihre korrekte Kartenhälften übertragen, wird die vollständige Karte zu beiden Clients beim Abfrage des Spielstatus sichtbar. Das kann im Form von einer Matrix dargestellt werden die per Console ausgegeben wird. Die einzige Matrix Elemente können die entsprechende Informationen beinhalten.

Im Fall von einem fehlerhaftes Kartenhälfte, wird derjenige Client über das Spielverlust informiert und der andere über sein Sieg informiert durch ein entsprechenden CLI Text Ausgabe.

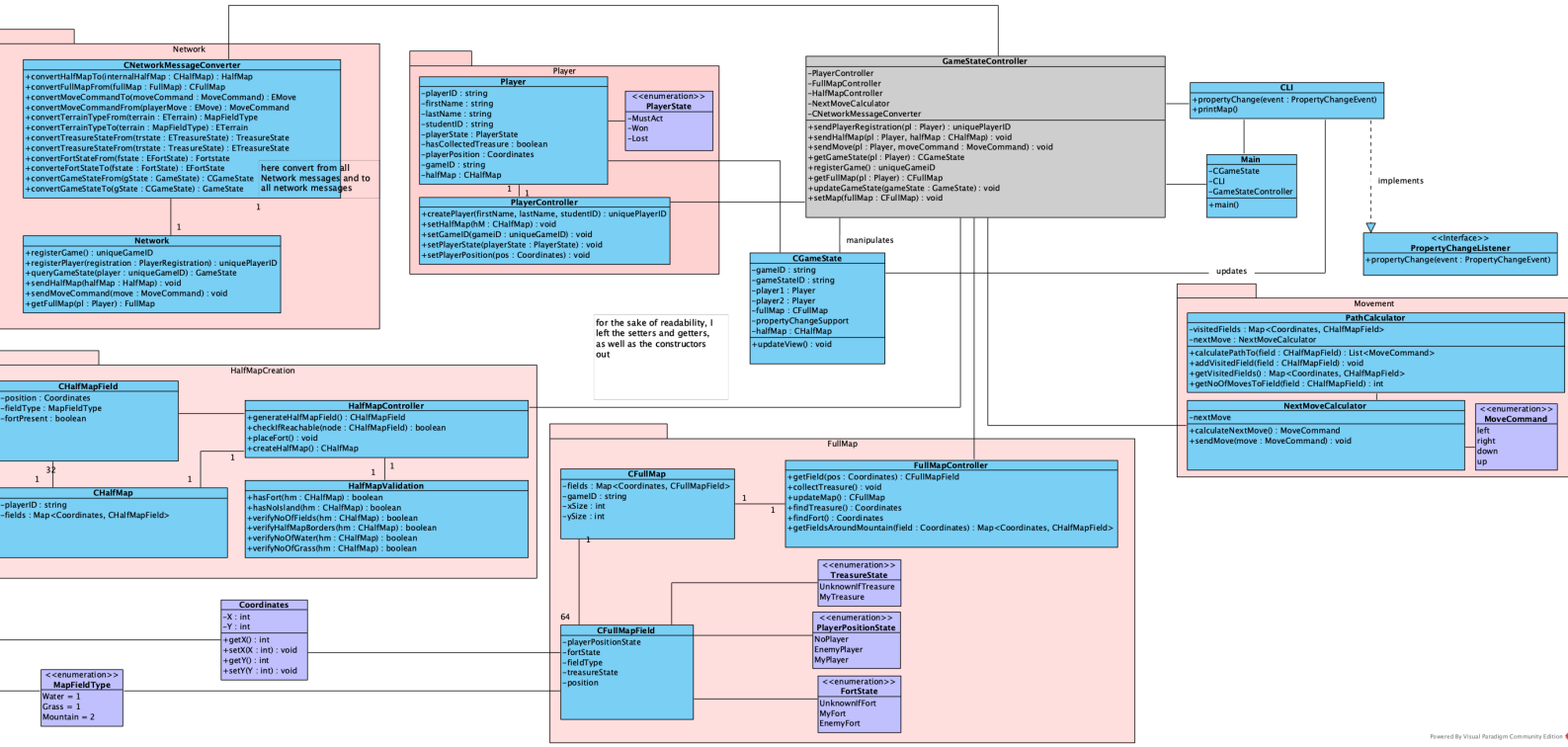
- **Externe Schnittstellen:**

Übertragung einer der beiden Kartenhälften: Der Client muss ein HTTP POST request mit den notwendigen Daten in XML Format an den Endpoint: http(s):// <domain>:<port>/games/<SpielID>/halfmaps senden. Als Antwort von den Server bekommt er einen entsprechenden responseEnvelope. Der Netzwerkprotokoll verwendet daher den REST Schnittstelle.

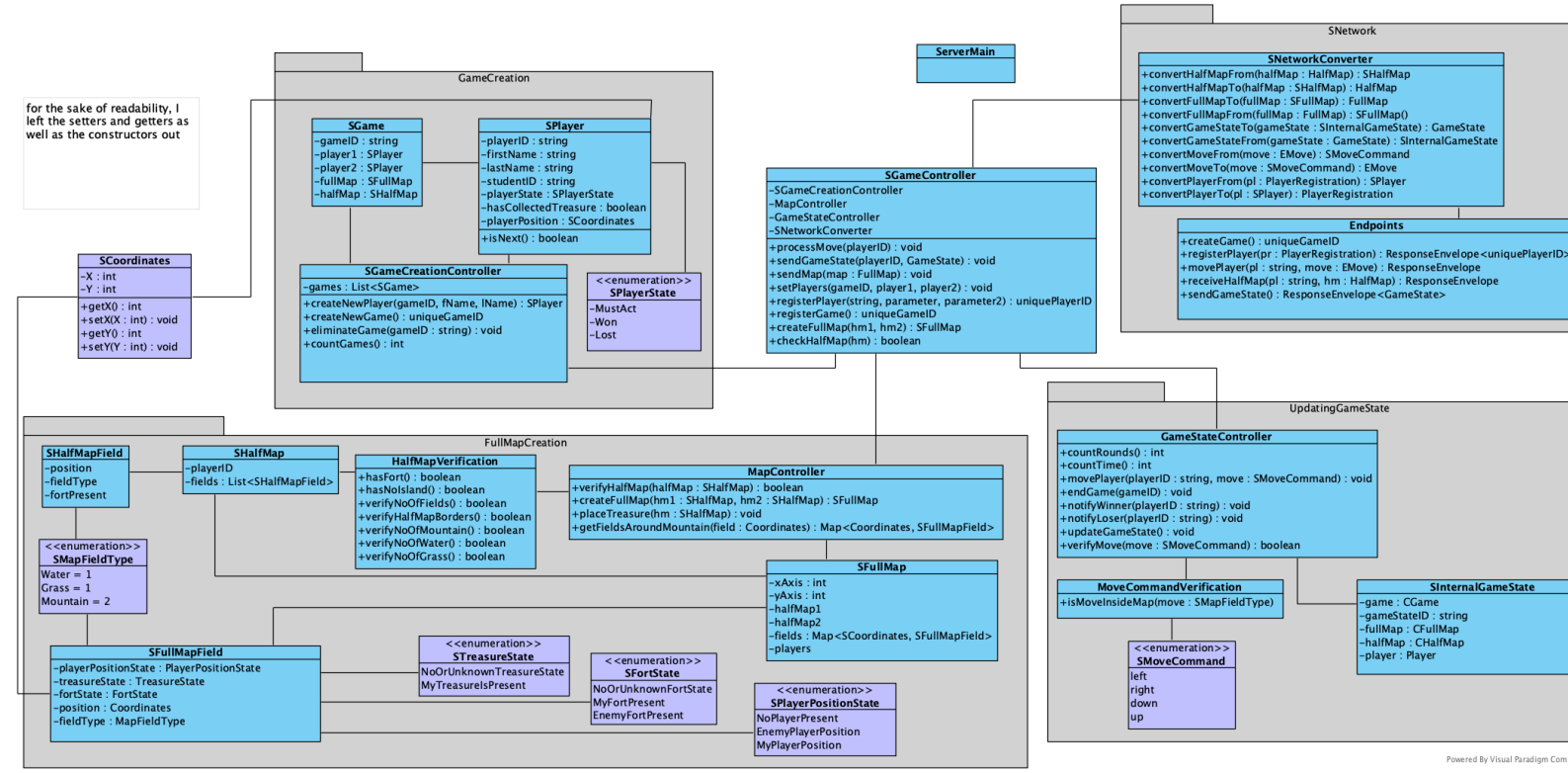
Informationen über die Karte abfragen durch Spielstatus Abfrage: Der Client muss ein HTTP POST request mit den notwendigen Daten in XML Format an den Endpoint: http(s)://<domain>:<port>/games/<SpielID>/states/<SpielerID> senden. Als Antwort von den Server bekommt er einen entsprechenden responseEnvelope. Der Netzwerkprotokoll verwendet daher den REST Schnittstelle.

Aufgabe 3: Architektur entwerfen, modellieren und validieren (10 Punkte)

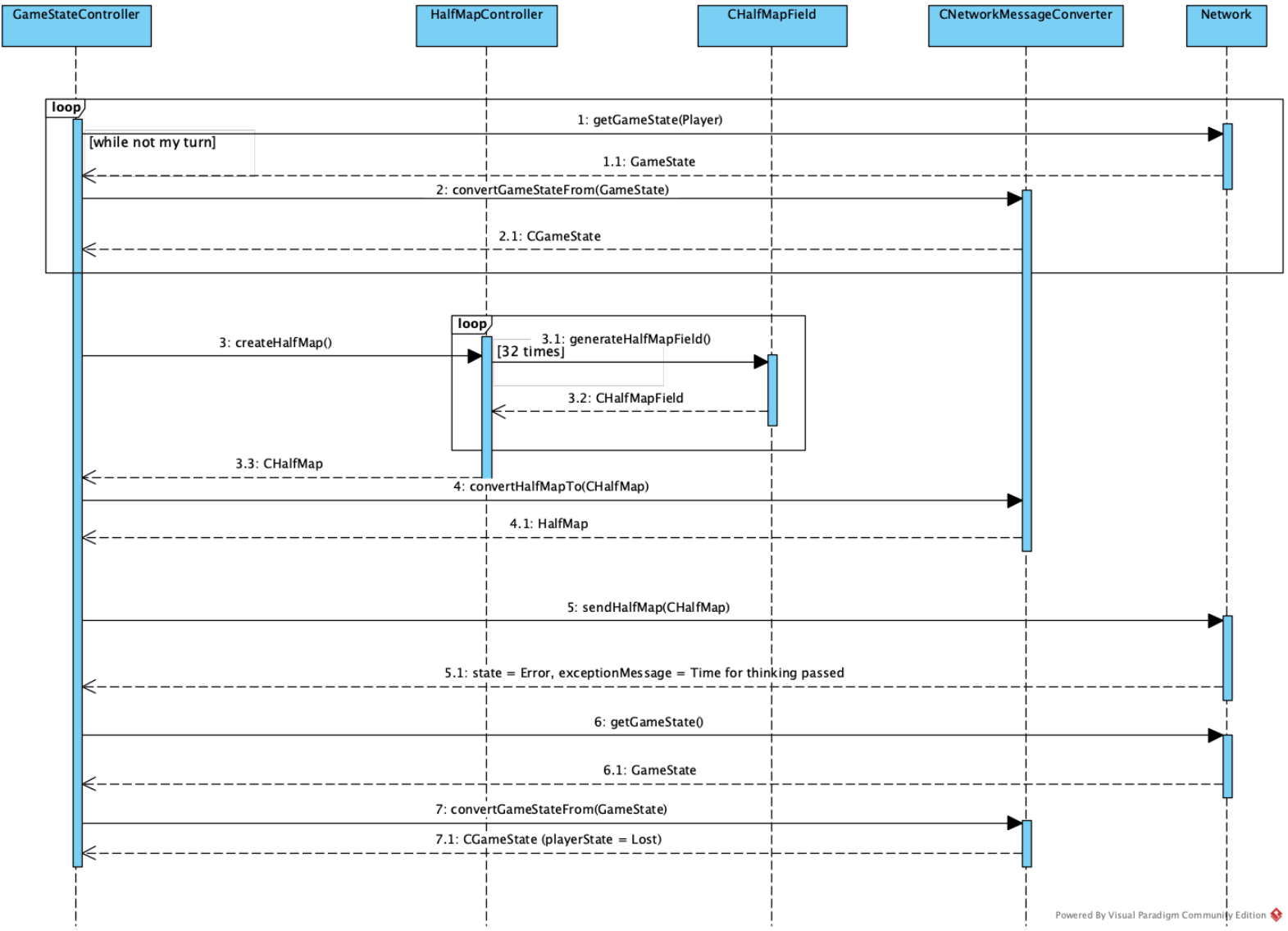
Client Klassendiagramm



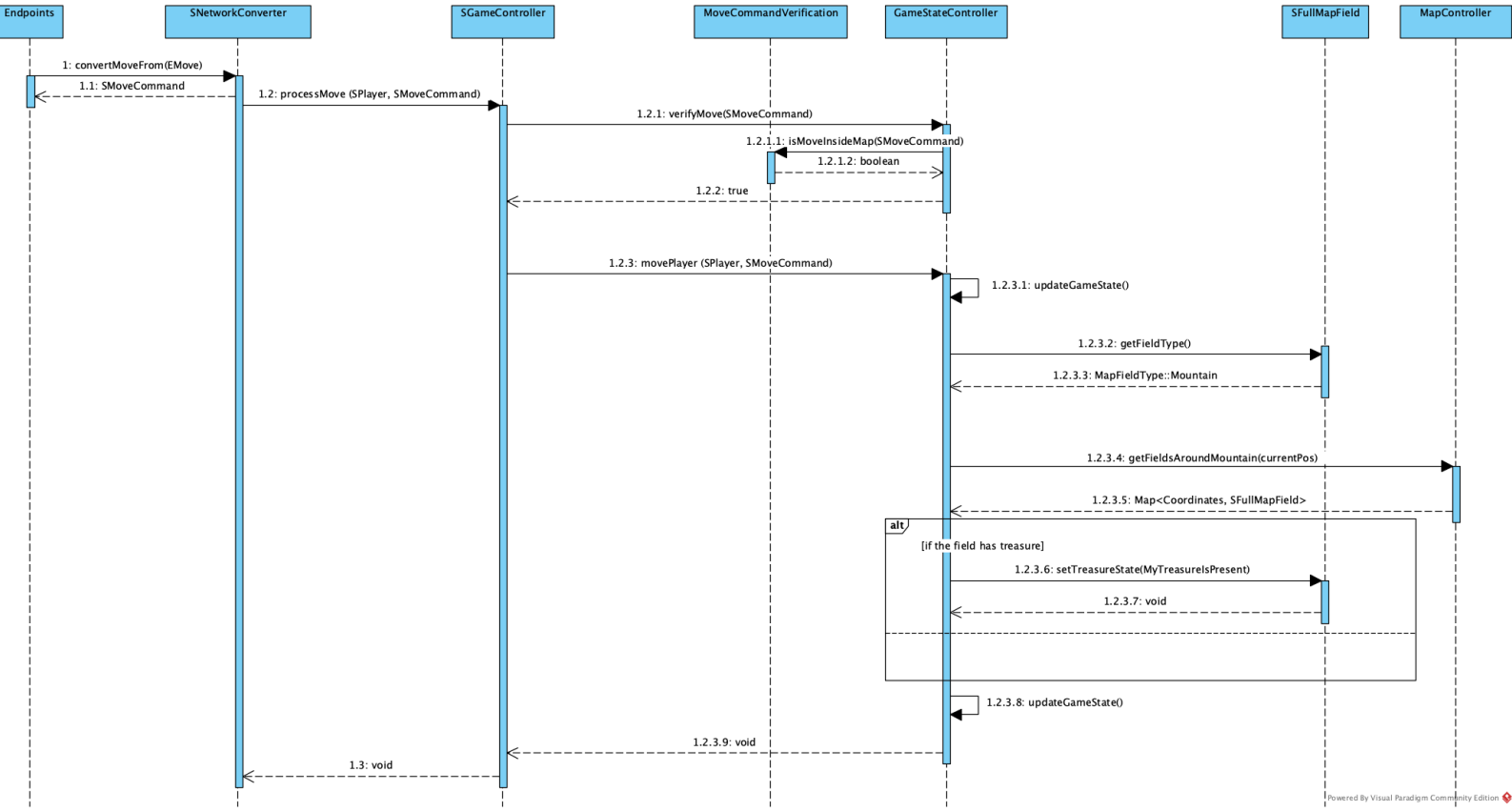
Server Klassendiagramm



Client Sequenzdiagramm



Server Sequenzdiagramm



Note: Bei dem Klassendiagrammen habe ich den setters und getters sowohl Constructors rausgelassen, um das Lesen ein bisschen zu erleichtern.