Software Engineering 1
Abgabedokument
Teilaufgabe 1
(Anforderungsanalyse und Planungsphase)

Persönliche Daten, bitte vollständig ausfüllen:

Nachname, Vorname:	Micic Dejan
Matrikelnummer:	01549172
E-Mail-Adresse:	a01549172@unet.univie.ac.at
Datum:	20.03.2021

Aufgabe 1: Anforderungsanalyse (2 Punkte)

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:** Kartengenerierung.
  Beide Clients generieren die Kartenhälften die, nachdem die dem Server zugeschickt sind, zusammengestellt werden.
- **Bezugsquelle:** Spielidee, „Die von den KIs bespielte 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 “

Anforderung 2

- **Beschreibung:** Zufällige Kartengenerierung jedes Mal.
  Die Felder auf den Kartenhälften müssen jedes Mal anders generiert werden.
- **Bezugsquelle:** Spielidee, „Die erstellten Karten sollten dabei zufällig generiert und nicht statisch vorgegeben werden“

Anforderung 3

- **Beschreibung:** Regeln zur Generierung der Kartenhälfte.
  Bei der Generierung muss der Client Regeln folgen. Es muss mindestens 3 Berge, 4 Wasser und 15 Wiesenfelder geben. Wenn das nicht der Fall ist verliert die KI.
- **Bezugsquelle:** Spielidee, „Außerdem muss jeder Terraintyp vorkommen, das heißt jede Kartenhälfte muss mindestens 3 Bergfelder, 15 Wiesenfelder und 4 Wasserfelder beinhalten“

Anforderung 4

- **Beschreibung:** Es darf keine Inseln geben.
  Kein Feld darf von allen Seiten vom Wasser begrenzt werden.
- **Bezugsquelle:** Spielidee, „Weiters dürfen keine Inseln generiert werden, daher dürfen Berge oder Wiesen niemals vollständig von Wasser oder Kartengrenzen (oder einer Kombination aus beidem) umschlossen werden.“

Anforderung 5

- **Beschreibung:** Es darf keine Inseln geben.
  Kein Feld darf von allen Seiten vom Wasser begrenzt werden.
- **Bezugsquelle:** Spielidee, „Weiters dürfen keine Inseln generiert werden, daher dürfen Berge oder Wiesen niemals vollständig von Wasser oder Kartengrenzen (oder einer Kombination aus beidem) umschlossen werden.“

Anforderung 6

- **Beschreibung:** KIs können sich nur vertikal und horizontal bewegen.
  Die KIs können sich auf der Karte nur horizontal (Links, Rechts), oder vertikal (Oben, Unten) bewegen.
- **Bezugsquelle:** Spielidee, „Eine Spielfigur kann sich nur horizontal und vertikal zu direkt benachbarten Feldern bewegen.“

Typ der Anforderung: nicht funktional

Anforderung 7

- **Beschreibung:** Das Spiel kann maximal 200 Züge dauern.
  Beide Clients schicken nacheinander die Züge in der Suche nach dem Schatz und der Gegnerischen Burg. Das ganze Spiel darf nicht länger als 200 Züge dauern. Wenn das Spiel länger als 200 Züge dauert verliert die KI die an der Reihe ist automatisch.
- **Bezugsquelle:** Spielidee „Um die Spiele für die Zuschauer spannend zu gestalten, wurde festgelegt, dass ein Spiel insgesamt nicht länger als 200 Spielaktionen dauern darf“

Anforderung 8

- **Beschreibung:** Spielaktion darf nicht länger als 3 Sekunden dauern.
  Wenn die Spielaktion oder Berechnung nicht innerhalb der 3 Sekunden geschickt wird, verliert die KI.
- **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 9

- **Beschreibung:** Verhinderung der Überlastung des Servers.
  Jeder Client kann eine Zweite abfrage zum Server machen, erst wenn 0,4 Sekunden vergehen, damit der Server nicht überlastet wird.
- **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.“

Anforderung 10

- **Beschreibung:** Jedes Spiel wird nach 10 Minuten gelöscht.
  Nach 10 Minuten wird das Spiel vom Server beendet, und die Clients können keine weiteren abfragen schicken.
- **Bezugsquelle:** Netzwerkprotokoll, „Weiters wird jedes Spiel 10 Minuten nach der Spielerzeugung automatisch entfernt/beendet, sodass keine weiteren Nachrichten an dieses gesandt werden können.“

Anforderung 11

- **Beschreibung:** Server kann nicht mehr als 999 Spiele parallel ausführen.
  Nachdem das Maximum von 999 Spielen erreicht wird, löscht der Server die Älteren Speile und ersetzt die mit neuen Spielen.

- **Bezugsquelle:** Netzwerkprotokoll, „Um die Ressourcen des Servers zu schonen gilt die Einschränkung, dass maximal 999 Spiele parallel ausgeführt werden dürfen. Sobald die Zahl der gleichzeitig laufenden Spiele die 999 Spiele übersteigt, werden ältere Spiele automatisch entfernt und mit den neueren danach erstellen Spielen ersetzt“

Typ der Anforderung: Designbedingung

Anforderung 12

- **Beschreibung:** ResponseEnvelope als Antwort vom Server.
  Die Antworten, die der Client vom Server bekommt, sind in ein responseEnvelope gekapselt. Im responseEnvelope steht auch, ob die Anfrage vom Client Korrekt war oder nicht.
- **Bezugsquelle:** Netzwerkprotokoll, **„**Hierbei ist wesentlich, dass die meisten Antworten von einem allgemeinen responseEnvelope gekapselt werden. Dieser nimmt die eigentlichen Daten, die übertragen werden sollen auf und bietet jedoch darüber hinaus auch die Möglichkeit mitzuteilen, ob eine Anfrage vom Client korrekt oder inkorrekt war**“**

Aufgabe 2: Anforderungsdokumentation (2 Punkte)

Dokumentation von *einer* in Aufgabe 1 identifizierten Anforderungen 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:** Kartengenerierung
- **Beschreibung und Priorität:** Beide Clients müssen für sich eine Kartenhälfte(4x8) generieren, die dann im Nachhinein dem Server geschickt wird. Die Kartenhälften müssen basierend auf Regeln aufgebaut werden. Wenn die Regeln nicht beibehaltet sind, verliert die KI das Spiel automatisch.
  Priorität: Hoch.
- **Relevante Anforderungen:**
  - Wie muss die Kartenhälfte ausschauen?
  - Welche Art von Feldern kann es geben?
  - An welchem Feld startet die KI?
  - An welchem Feld kann die Burg platziert werden?
  - In welche Richtungen kann sich die KI bewegen?
  - Wie passiert die Bewegung der KIs?
  - An welchen Feldern kann sich die KI bewegen?
- **Relevante Business Rules**:
  - Die Karte muss mindestens 3 Bergfelder, 15 Wiesenfelder und 4 Wasserfelder beinhalten.
  - Die Burg kann nur an einem Wiesenfeld generiert werden.
  - Die Wasserfelder können keine Insel erzeugen.
  - Es dürfen nicht mehr als 3 Wasserfelder an der langen Seite der Karte und nicht mehr als 1 Wasserfeld an der kurzen Seite.
  - Die Karte muss bei jedem neuen Speil anders generiert sein.
  - KIs können die Kartengrenzen nicht übertreten.
- **Impuls/Ergebnis - Typisches Szenario:**
  ***Vorbedingungen*:**
  - Der Mensch erzeugt ein neues Spiel.
  - Die unuiqueGameID wird den Clients übermittelt.
  - Clients registrieren sich beim Server auf das Spiel mit der uniqueGameID aus dem letzten Schritt.
  - Jeder Client generiert die Kartenhälfte, die dann dem Server abgeschickt wird.

  ***Hauptsächlicher* Ablauf**:
  - Impuls: Die Person erzeugt ein neues Spiel
  - Ergebnis: Eine uniqueGameID wird für das Spiel erzeugt
  - Impuls: KIs kriegen die uniqueGameID übermittelt.
  - Ergebnis: Die KIs registrieren sich für dasselbe Spiel.
  - Impuls: Clients (KIs) schicken die selbst generierten Kartenhälften zum Server und es wird die ganze Karte zusammengestellt bei den langen Kartenseiten(8x8).
  - Ergebnis: Die Beiden KIs haben eine zufällig generierte Karte zum Spielen.
  - Impuls: KIs laufen die Karte durch in der Suche nach einem Schatz.
  - Ergebnis: KI findet Schatz.

- Impuls: KI die den Schatz gefunden hat sucht die gegnerische Burg.
- KI findet die Burg, und hat damit das Spiel gewonnen.

Nachbedingungen:

- Die KIs spielen je eine Spielrunde, indem sie je einen Schritt machen, in der Suche nach dem versteckten Schatz.
- Das Spiel ist gewonnen, indem die KI den Schatz gefunden hat und zur gegnerischen Burg geht.

Impuls/Ergebnis - Alternativszenario:

Vorbedingungen:

- Spieler erzeugt ein Spiel.
- Die unuiqueGameID wird den Clients übermittelt.
- Clients schicken die Kartenhälften.

Hauptsächlicher Ablauf:

- Impuls: Die Person erzeugt ein neues Spiel
- Ergebnis: Eine uniqueGameID wird für das Spiel erzeugt
- Impuls: KIs kriegen die uniqueGameID übermittelt.
- Ergebnis: Die KIs registrieren sich für dasselbe Spiel.
- Impuls: Clients (KIs) schicken die selbst generierten Kartenhälften zum Server und es wird die ganze Karte zusammengestellt bei den kurzen Kartenseiten(4x16).
- Ergebnis: Die Beiden KIs haben eine zufällig generierte Karte zum spielen.
- Impuls: KIs laufen die Karte durch in der Suche nach einem Schatz.
- Ergebnis: KI findet Schatz.
- Impuls: KI die den Schatz gefunden hat sucht die gegnerische Burg.
- KI findet die Burg, und hat damit das Spiel gewonnen.

Nachbedingungen:

- Die KIs spielen je eine Spielrunde, indem sie je einen Schritt machen, in der Suche nach dem versteckten Schatz.
- Das Spiel ist gewonnen, indem die KI den Schatz gefunden hat und zur gegnerischen Burg geht.

- **Impuls/Ergebnis - Fehlerfall:**

Im Fehlerfall kann es passieren, dass eine KI die generierte Kartenhälfte falsch macht(eine Bussines rule die die Kartengenerierung angeht wurde nicht angehalten)

Vorbedingungen:

- Der Mensch erzeugt ein neues Spiel.
- Die unuiqueGameID wird den Clients übermittelt.
- Clients registrieren sich beim Server auf das Spiel mit der uniqueGameID aus dem letzten Schritt.
- Jeder Client generiert die Kartenhälfte, die dann dem Server abgeschickt wird.

***Hauptsächlicher* Ablauf**:

- Impuls: Die Person erzeugt ein neues Spiel
- Ergebnis: Eine uniqueGameID wird für das Spiel erzeugt
- Impuls: KIs kriegen die uniqueGameID übermittelt.
- Ergebnis: Die KIs registrieren sich für dasselbe Spiel.
- Impuls: Clients (KIs) schicken die selbst generierten Kartenhälften zum Server und es wird die ganze Karte zusammengestellt.

Nachbedingungen:
- Server überprüft die generierten Karten.
- Eine Kartenhälfte, die von einer KI generiert wurde, ist Falsch (Erfüllt nicht die angegebenen Voraussetzungen).
- Die KI die die Falsche Kartenhälfte generiert hat, verliert automatisch.

- **Benutzergeschichten:**
  - Als Anwender möchte ich, dass die Karte immer anders generiert wird, damit das Spiel immer spannend ist.
  - Als Mensch möchte ich immer sehen können, wie das Spiel vorangeht.
  - Als eine KI möchte ich so schnell wie möglich den Schatz bzw. die Gegnerische Burg finden damit ich auch gewinnen kann.
  - Als KI möchte ich nur die begehbaren Felder besuchen, dass ich nicht verliere, wenn ich auf ein Wasserfeld gehe.
  - Als ein Server möchte ich, dass die KIs die Karten gut generiert zu mir schicken, damit das Spiel überhaupt funktioniert.

- **Benutzerschnittstelle:**
  Den Spielablauf kann der Mensch über die CLI sehen. Eine Kartenhälfte ändert sich nach jedem Schritt der KI und schaut so aus:

W	G	G	G	G	G	M	M
G	G	W	M	G	W	G	G
M	G	G	G	G	G	G	G
G	W	G ->P	M	G	G	G	W

W – Water

G – Grass

M – Mountain

->P – Player position

Wenn die KI auf einem Grassfeld mit dem Schatz ist, wird der Schatz als „*“ angezeigt.

- **Externe Schnittstellen:**
  - Schnittstelle Netzwerkprotokoll: Wir benutzen http, um Nachrichten zu schicken und erhalten.
  - Schnittstelle Kommunikation mi Server: Die Kommunikation mit Server ist durch ResponseEnvelope gekapselt und alle Nachrichten sind als XML Datei erwartet.
  - Jede Nachricht wird an ein geeignetes Endpoint geschickt. Der Mensch erzeugt das Spiel bei folgendem Endpoint: http(s)://<domain>:<port>/games. Zur übertragung der Kartenhälften wird http(s)://<domain>:<port>/games/<SpielID>/halfmaps mit http Post requested.

- Um den Status des Spiels für eine KI mit SpielerID wird ein GET request an http(s)://<domain>:<port>/games/<SpielID>/states/<SpielerID> gesendet.
- Damit eine KI eine Bewegung senden kann, muss ein POST request an http(s)://<domain>:<port>/games/<SpielID>/moves senden

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

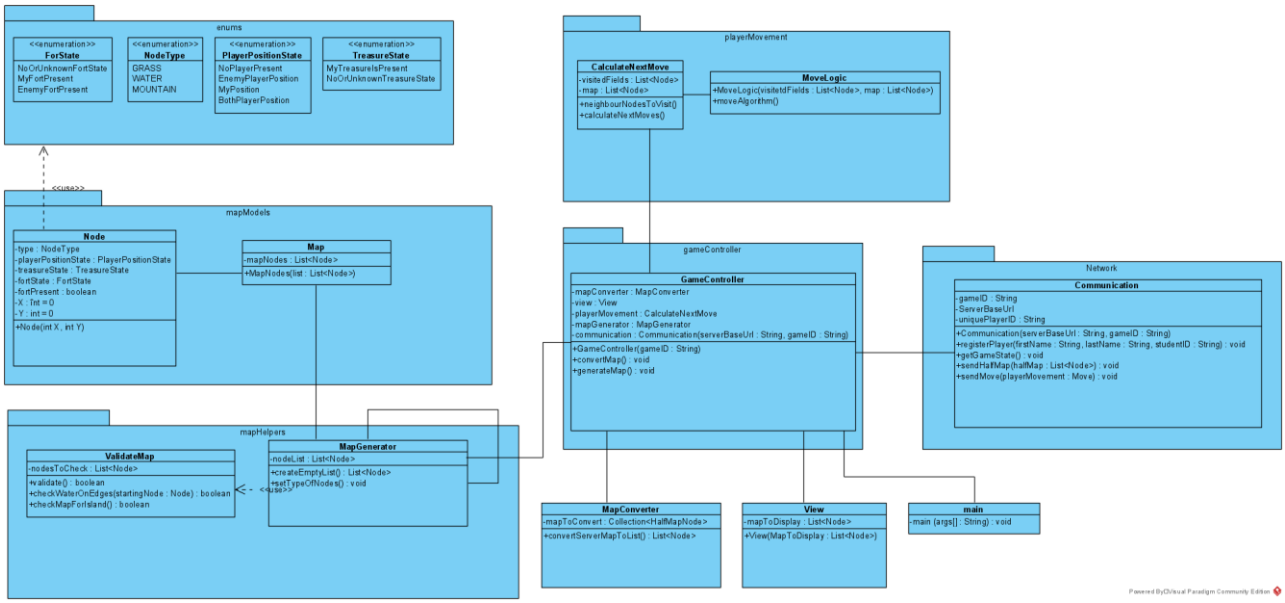
Modellieren Sie händisch alle notwendigen Packages, Klassen und deren Methoden (samt Beziehungen) als zwei UML Klassendiagramme. Achten Sie darauf, dass die Modelle sinnvoll benannte Packages, Klassen, Methoden und Felder beinhalten und die Vorgaben der Spielidee bzw. des Netzwerkprotokolls vollständig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

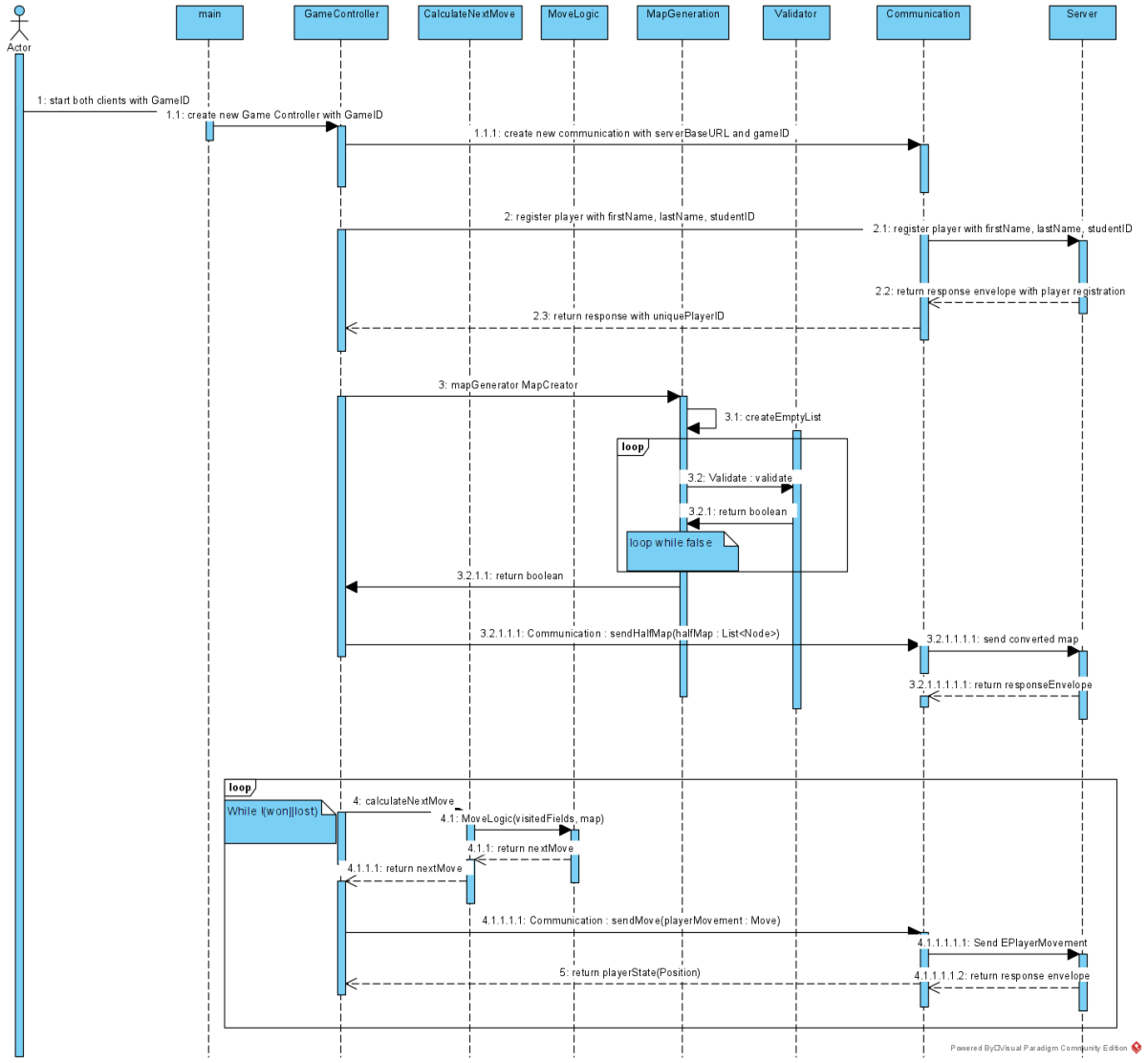
TIPP: Beachten Sie zur Ausarbeitung das auf Moodle zu Verfügung gestellte auszugsweise abgebildete 4+1 Diagrammbeispiel. Dieses sollte als Vorlage dienen, und verdeutlicht welche Erwartungen an das Klassendiagramm beziehungsweise die dazugehörigen Sequenzdiagramme gestellt werden (Darstellung, Inhalte, Detailgrad, etc.) und wie diese zusammenhängen.

[Alle vier Diagramme mit einer kurzen Beschreibung hier einfügen. Sollten die Diagramme zu viel Platz benötigen können Sie diese auch als separate SVG Dateien (empfohlenen) oder PNG Dateien im Dokumentationsordner (siehe GitLab Repo) einfügen. Achten Sie bei der Verwendung von SVG Dateien darauf, dass sich diese fehlerfrei darstellen lassen.]

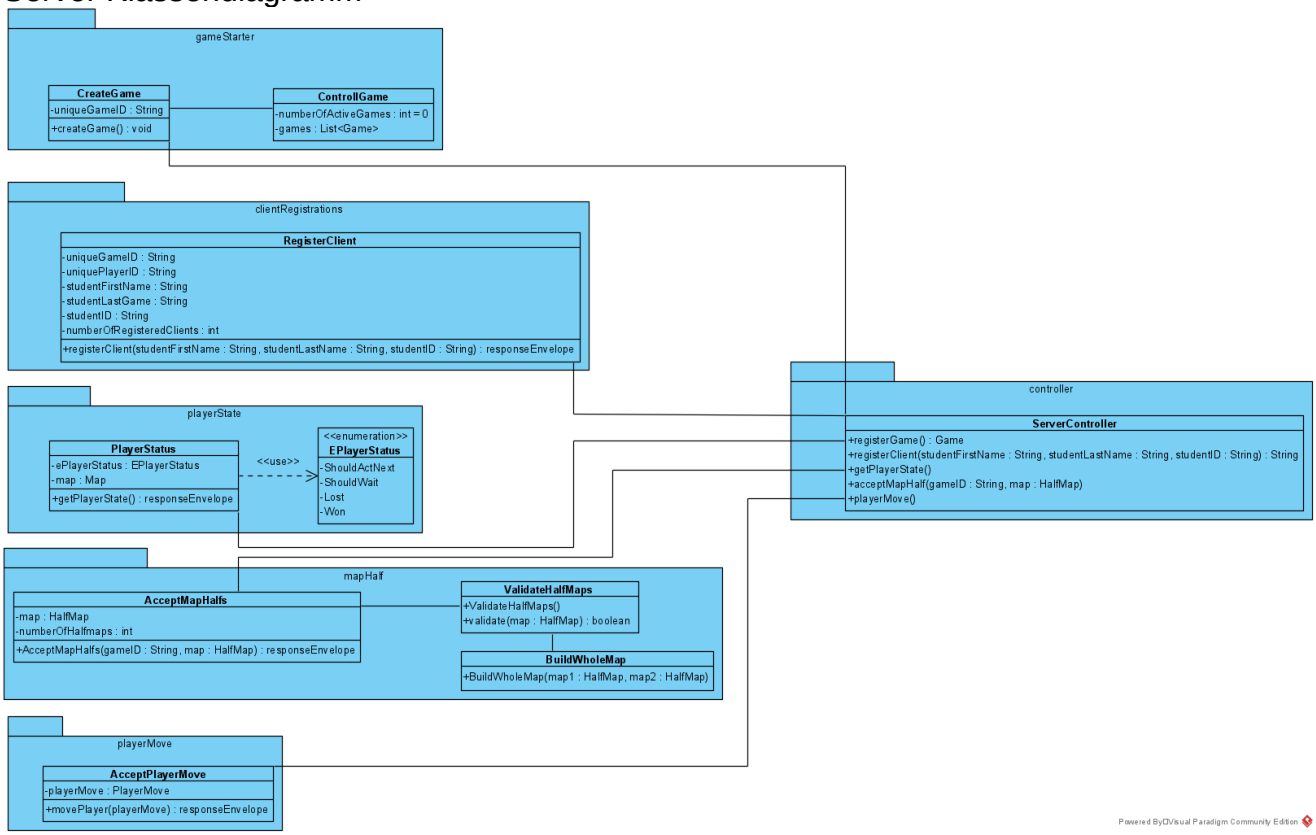
Client Klassendiagramm



Client Sequenzdiagramm



Server Klassendiagramm



Server Sequenzdiagramm