



Abschlusspräsentation

Man don't get angry

Ablauf

- Demonstration
- Analysephase
- Designphase
- Implementierungsphase

Neubiberg, den 17. Dezember 2024

Demonstration

Neubiberg, den 17. Dezember 2024

Analysephase

Verstehen und Durchdringen der Problemstellung

Neubiberg, den 17. Dezember 2024



Erstellte Artefakte:

- Erweiterte Aufgabenstellung
- Use Cases
- Zustandsdiagramme und Klassendiagramme
- Testhandbuch
- Gestaltungsrichtlinien
- GUI Skizzen
- Benutzerhandbuch
- Glossar

- Zweck des Glossars:
 - Einheitliche Definition zentraler Begriffe im Projekt.
 - Vermeidung von Missverständnissen zwischen Teammitgliedern.

Bonuskarte (PowerCard)	Eine Karte, wenn ausgespielt, hat einen Effekt auf das Spiel. Diese dürfen nur direkt vor dem Würfeln ausgespielt werden. Es stehen Turbo, Schild und Tausch zur Verfügung.
Bewegung (Move) <i>“eine Figur ziehen” entspricht der Bewegung einer Figur durch einen Spieler</i>	Die Bewegung ist das Voranschreiten einer Figur nach einmaligem Würfeln, um die gewürfelte Augenzahl.
Schlagen	Eine Figur wird geschlagen, wenn eine Figur einer anderen Farbe auf ihr Feld zieht und sie keinen Schild besitzt, wenn eine Figur geschlagen wurde, kehrt sie in ihr Wartebereich zurück.

- Use Cases beschreiben spezifische Interaktionen zwischen Nutzern und dem System.
- Kategorien:
 - *UC-Game*: Spielabläufe.
 - *UC-Player*: Spieleraktionen.
 - *UC-Piece*: Figureninteraktionen.
 - *UC-Server* und *UC-Client*: Netzwerkkommunikation zwischen Server und Clients.
 - *UC-Settings*: Einstellungen und Anpassungen.
 - *UC-Dialog*: Interaktionen des Spielers mit verschiedenen Dialogfenstern.
 - *UC-Media*: Darstellung und Medienelemente des Spiels.
 - *UC-Lobby*: Aktionen in der Lobby.

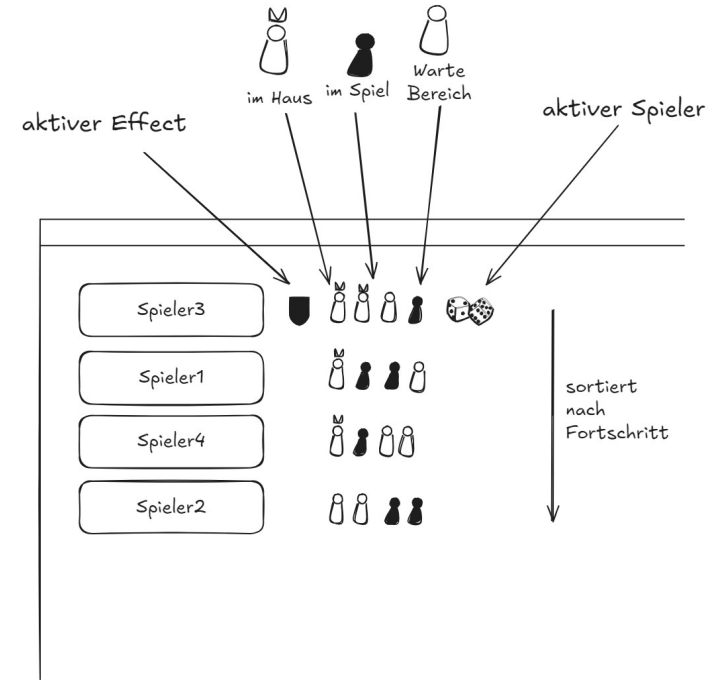
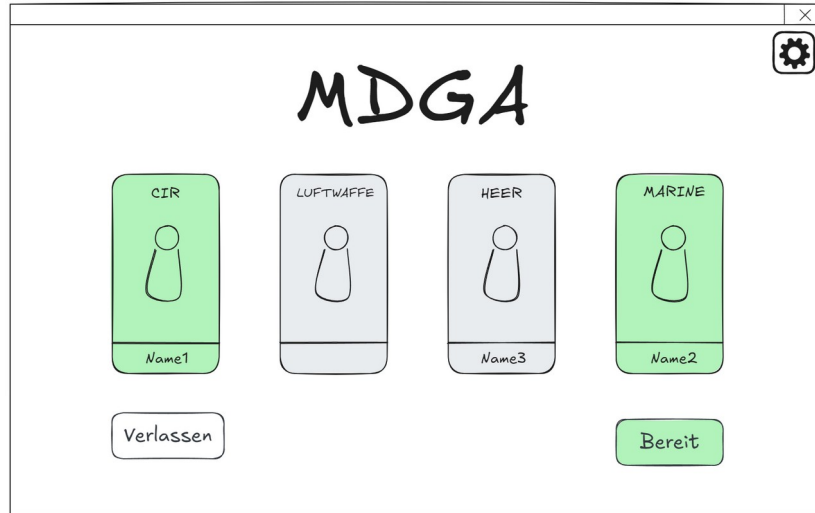
ID	Name	Priorität	Impl.-Status
-----Use-Cases GAME-----			
UC-Game-01	Spiel beginnt mit Startaufstellung	1	-
UC-Game-02	Spiel endet	1	-
UC-Game-03	Spieler hat alle Figuren im Haus	1	-
UC-Game-04	Spieler hat alle Figuren im Wartebereich	1	-
UC-Game-05	Spiel wird beendet	1	-
UC-Game-06	Reihenfolge wird ausgewürfelt	1	-

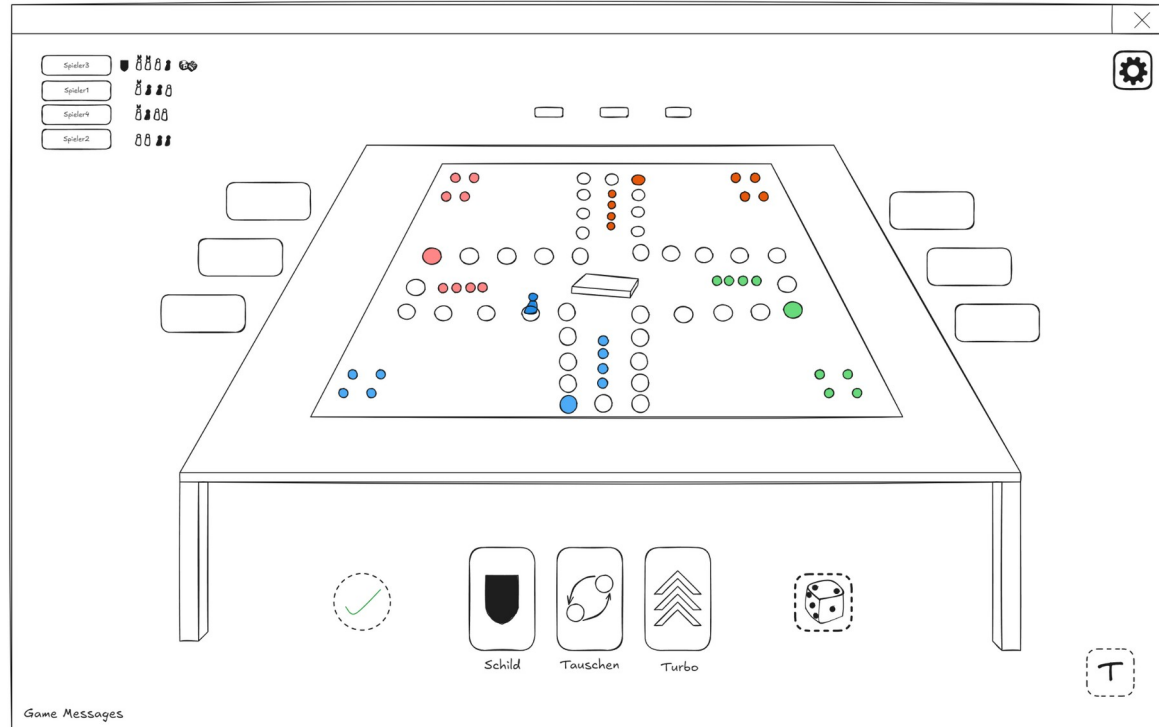
UC-Piece-06 Figur muss Startfeld verlassen	
Übersicht	
Name	Figur muss Startfeld verlassen
ID	UC-Piece-06
Zweck	Der Spieler muss aufgrund weiterer Figuren im Wartebereich das Startfeld räumen mit seiner Figur.
Aktoren	Spieler
Vorbedingungen	1. Figur steht auf dem Startfeld. 2. Spieler hat mindestens eine Figur noch im Wartebereich.
Kurzbeschreibung	Der Spieler zieht seine Figur vom Wartebereich weg, da mindestens noch eine Figur im Wartebereich steht.
Nachbedingung	1. Spieler hat die Figur vom Startfeld wegbewegt. 2. Startfeld ist nun wieder frei.
Ablauf	
Aktor	System
1. Spieler wählt Figur auf dem Startfeld aus.	2. System prüft, ob der Zug möglich ist. 3. Figur wird wegbewegt.

- Zweck:
 - Sicherstellung der korrekten Funktionalität gemäß den definierten Use Cases.
 - Identifikation und Behebung von Fehlern vor der Auslieferung.
- Struktur:
 - Tests basieren auf den Use Cases.

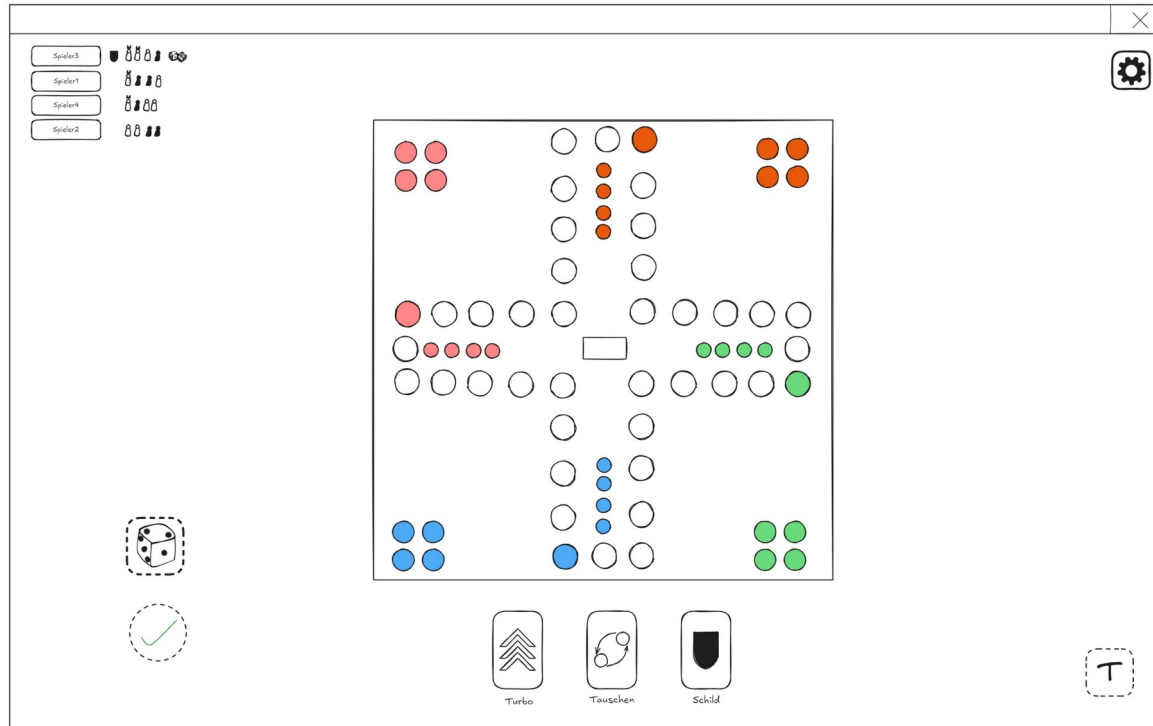
ID	Use-Case	Test-Name	Beschreibung	Beteiligte Klassen	Art	Status
T001	UC-Game-01	testStartLineUp	Nachdem das Spiel durch den Host gestartet wurde, hat jeder Spieler drei Figuren im Wartebereich, eine Figur auf dem Startfeld und eine zufällige Sonderkarte auf der Hand.			
T002	UC-Game-02	testAwardCeremony	Alle außer ein Spieler haben ihre vier Figuren in das Haus gebracht. Das Spiel wird beendet und die Siegerehrung beginnt, wobei zuerst alle Spieler der Gewinnreihenfolge nach angezeigt und danach noch Spielstatistiken wie "Würfe/Geworfen", "meiste Sonderkarten verwendet" und "meiste 6en gewürfelt" bekannt gegeben werden.			
T003	UC-Game-03	testPlayerFinishes	Ein Spieler hat alle Figuren ins Haus gebracht und ist somit fertig. Mit den Figuren kann nun nicht mehr interagiert (ziehen, schlagen, tauschen, etc.) werden und der Spieler wird in den folgenden Runden übersprungen.			

- Ziel: übersichtliche und intuitive Benutzeroberfläche gestalten.





Normale Ansicht



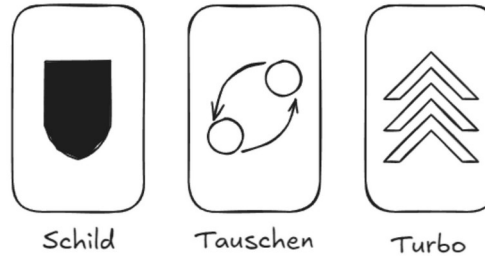
Strategische Ansicht

- Anleitung für Spieler, um das Spiel "Man don't get angry" zu verstehen und nutzen zu können.

1. Gesundheitshinweis.....	4
2. Systemanforderungen.....	4
3. Sprachunterstützung des Spiels.....	5
4. Spielprinzip.....	5
5. Spielregeln.....	6
a. Aufbau des Spielfeldes.....	6
b. Grundlegende Regeln.....	7
i. Start des Spiels.....	7
ii. Spielzug.....	7
iii. Ziel des Spiels.....	8
iv. Spielende.....	9
v. Siegerehrung.....	9
c. Bonuskarten.....	9

c. Bonuskarten

Während des Spiels können auf dem Bonusfeld folgende Bonuskarten gezogen werden:

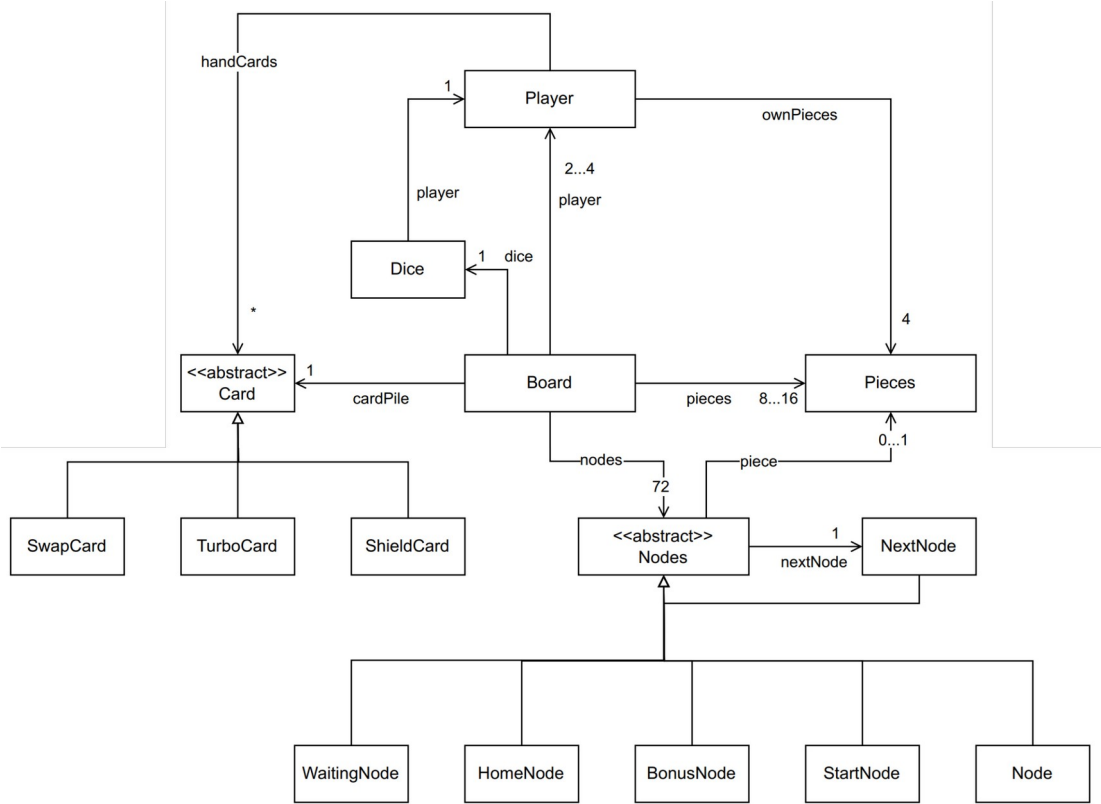


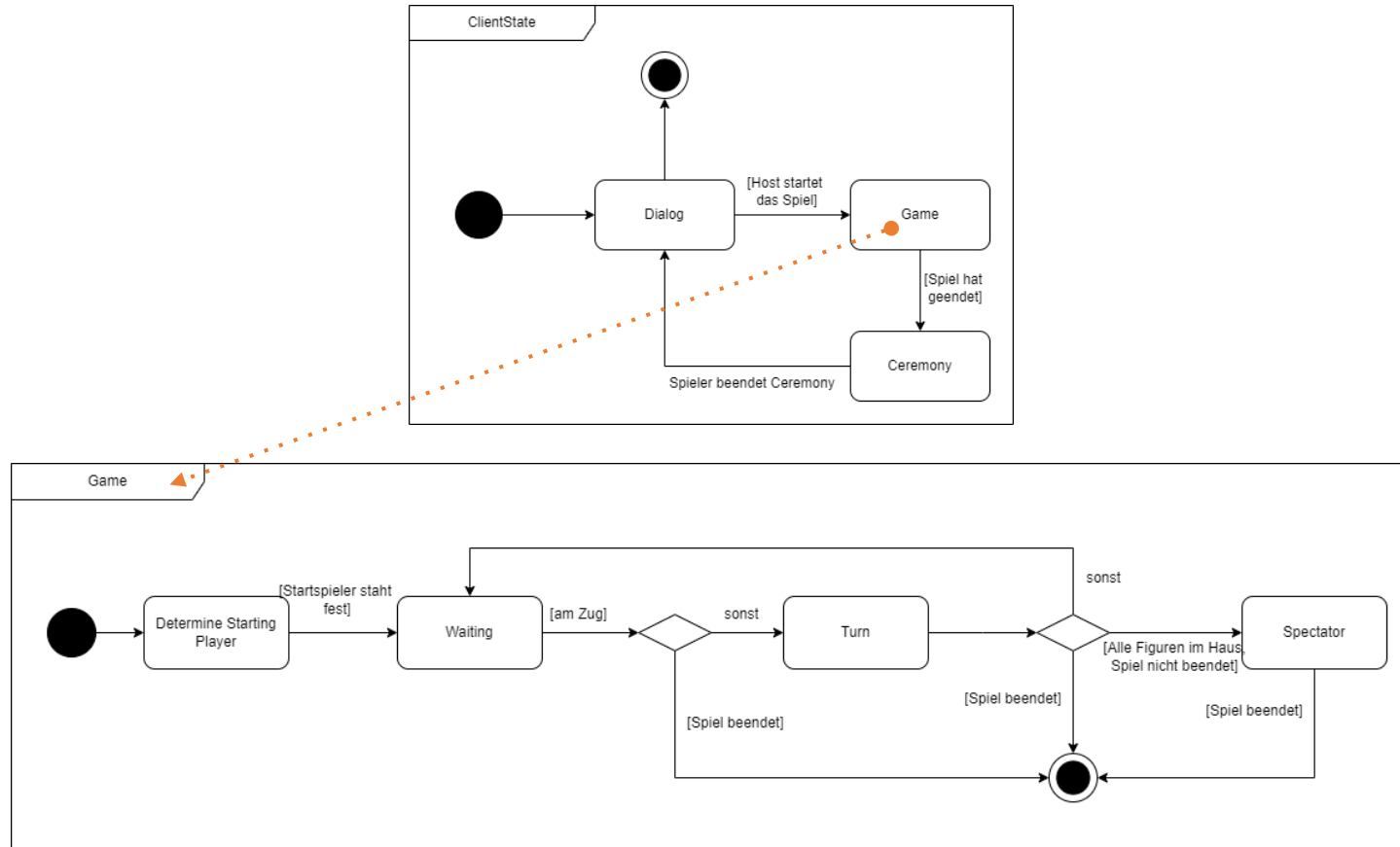
Das Bonuskarten-Deck besteht aus 40 Karten, davon sind 16 Turbo-Karten, 12 Schild-Karten und 12 Tausch-Karten.

Man kann eine Bonuskarte **nur direkt vor dem Würfeln** einsetzen.

Mit der Turbo-Karte wird die Augenzahl zu einer gewissen Wahrscheinlichkeit verdoppelt.

- 60%: Die Augenzahl wird verdoppelt.
- 20%: Die Augenzahl bleibt unverändert.
- 20% Die Figur bewegt sich nicht.





Designphase

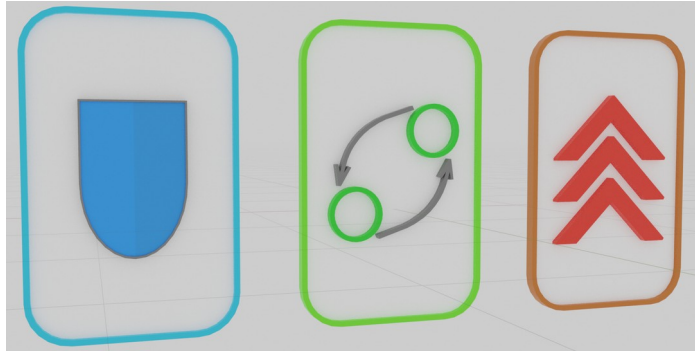
Das vorliegende, domänenspezifische Analysemodell verfeinert und dann in eine technische Sicht übertragen.

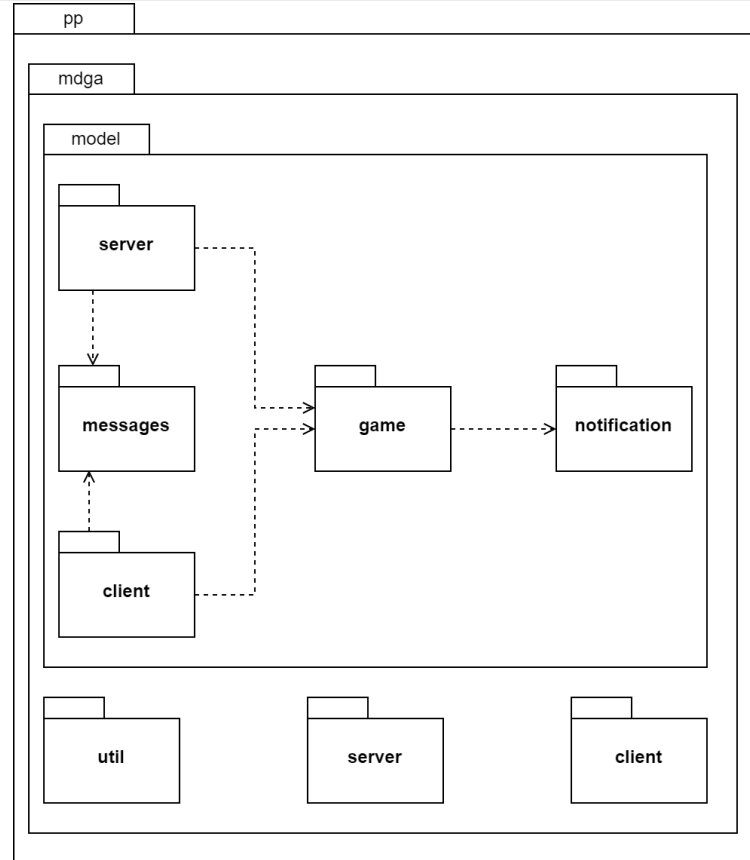
Neubiberg, den 17. Dezember 2024



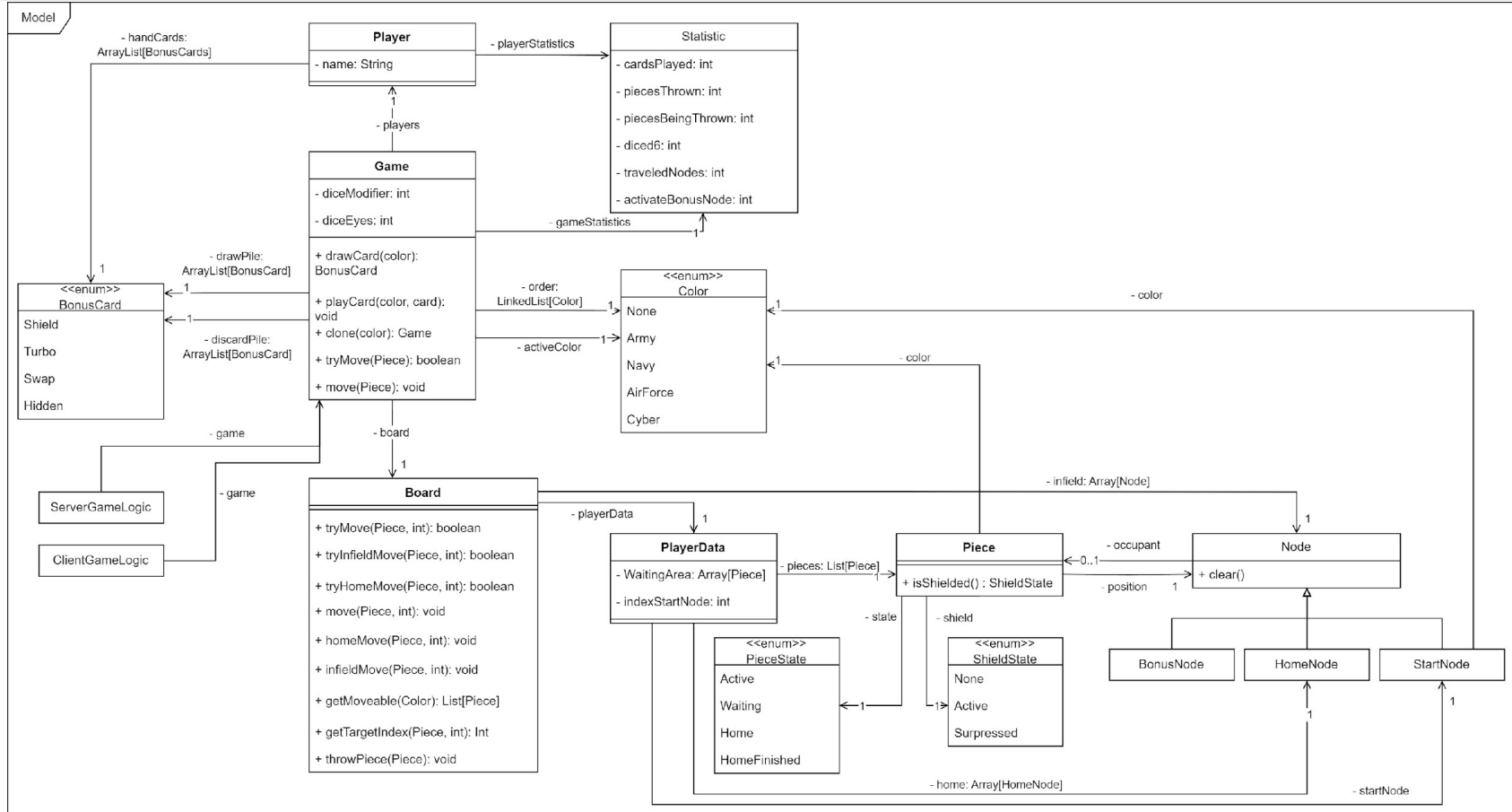
Erstellte Artefakte:

- Diagramme
 - Klassendiagramme
 - Paketdiagramme
 - Sequenzdiagramme
 - Flussdiagramme
 - Zustandsdiagramme
 - BPMN Diagramme
- Miscellaneous
 - Use Cases
 - Testhandbuch
 - Benutzerhandbuch
 - AssetListe
 - MessageList

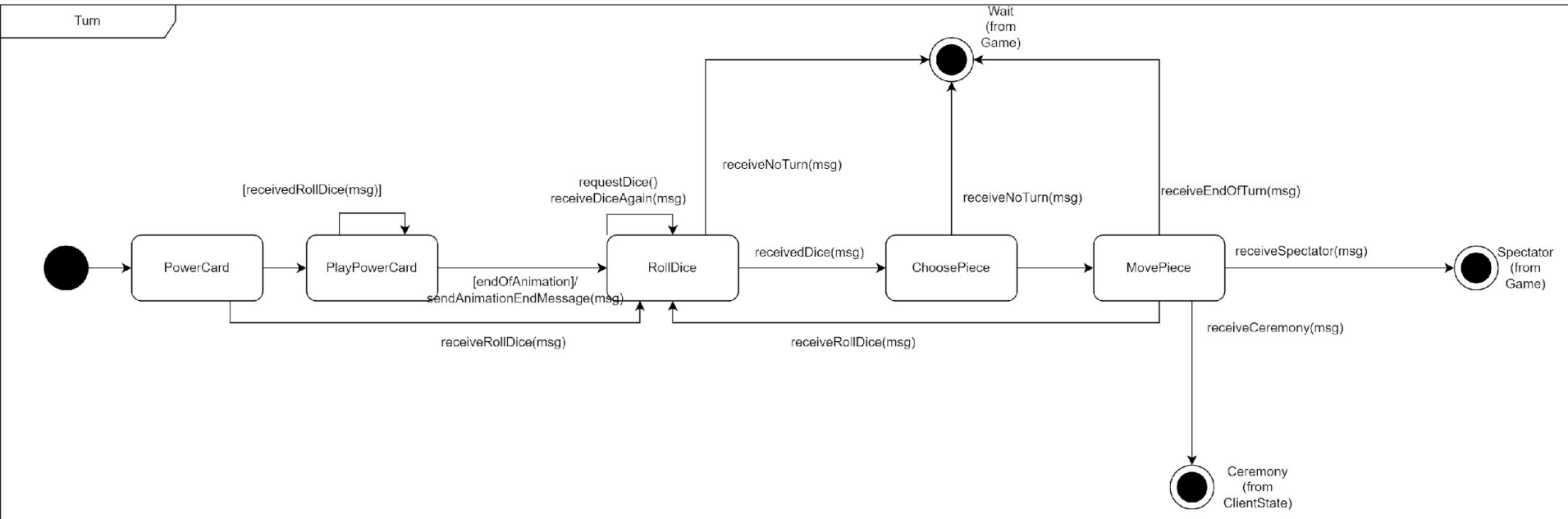


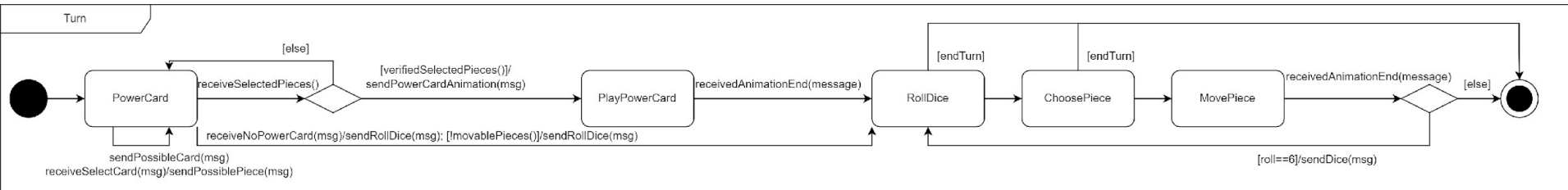


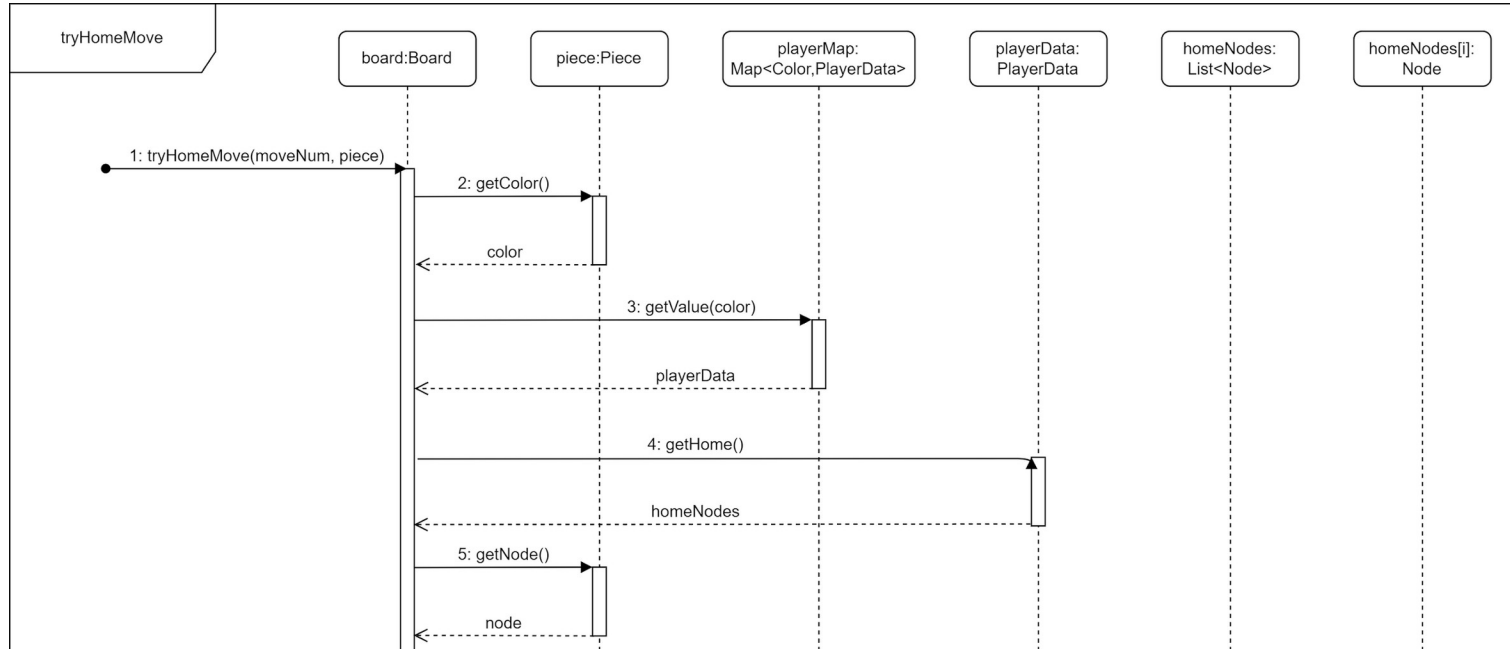


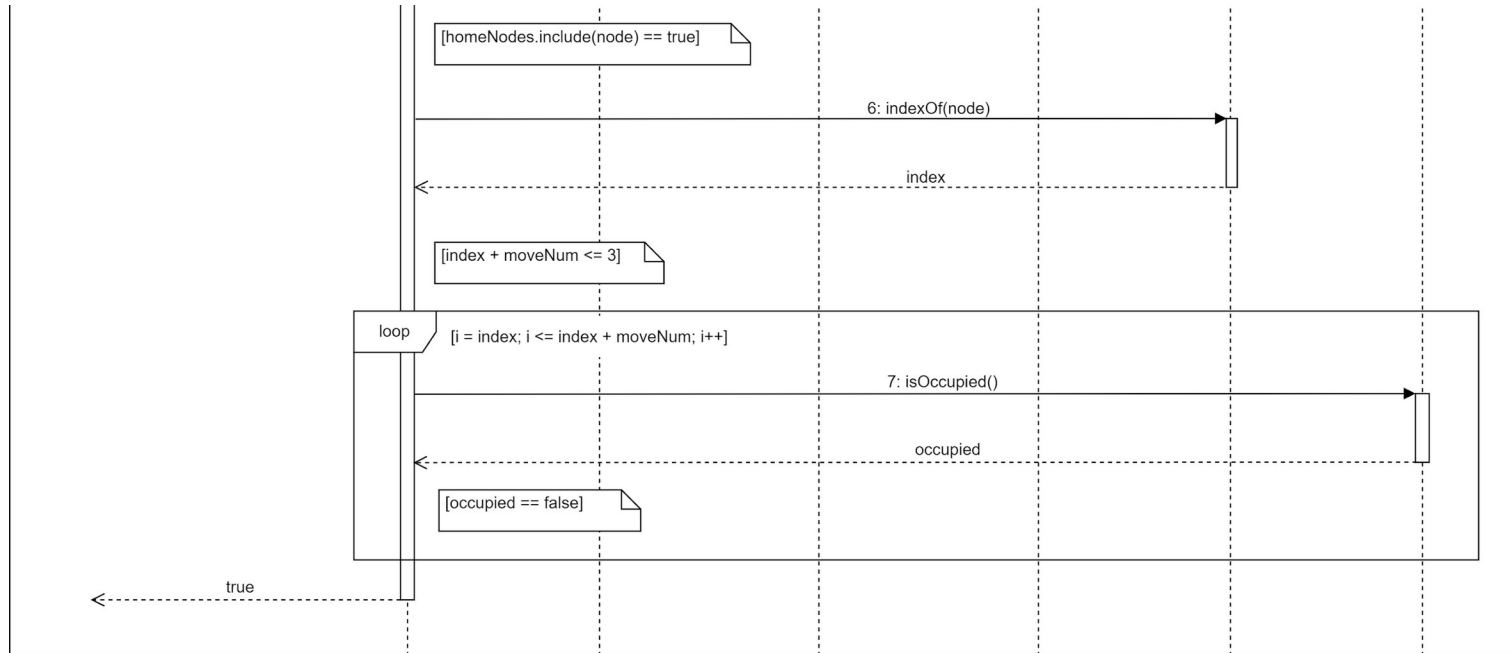












Implementierungsphase

Die Umsetzung des zuvor erstellten Designmodells.

Neubiberg, den 17. Dezember 2024



Erstellte Artefakte:

- **Code**
- Use Cases (Status)
- Testhandbuch (Status)
- Benutzerhandbuch
- Paketdiagramm
- Abweichungen zur Designphase
 - Klassendiagramme
 - Zustandsdigramme
 - BPMN-Diagramme
 - Flussdiagramme
 - Sequenzdiagramme

ID	Name	Priorität	Impl.-Status
-----Use-Cases GAME-----			
UC-Game-01	Spiel beginnt mit Startaufstellung	1	implementiert
UC-Game-02	Spiel endet	1	implementiert
UC-Game-03	Spieler hat alle Figuren im Haus	1	implementiert
UC-Game-04	Spieler hat alle Figuren im Wartebereich	1	implementiert
UC-Game-05	Spiel wird beendet	1	implementiert
UC-Game-06	Reihenfolge wird ausgewürfelt	1	implementiert

ID	Use-Case	Test-Name	Beschreibung	Beteiligte Klassen	Art	Status
T001	UC-Game-01	testStartLineUp	Nachdem das Spiel durch den Host gestartet wurde, hat jeder Spieler drei Figuren im Wartebereich, eine Figur auf dem Startfeld und eine zufällige Sonderkarte auf der Hand.	Game, Board, Player, BonusCard, BoardView, GUIView	JUnit	erfolgreich
T002	UC-Game-01	testCreatePowerCardDeck	Ein gemischtes Deck mit einer bestimmten Anzahl jeder Bonuskarte wird erstellt.	Game, BonusCard, BoardView, GUIView	JUnit	erfolgreich
T003	UC-Game-02	testGameFinishes	Alle außer ein Spieler haben ihre vier Figuren in das Haus gebracht. Das Spiel wird beendet und die Siegerehrung beginnt.	Game, Board, , Piece, Node, BoardView, GUIView	JUnit	erfolgreich
T004	UC-Game-03	testPlayerFinishes	Ein Spieler hat alle Figuren ins Haus gebracht und ist somit fertig. Mit den Figuren kann nun nicht mehr interagiert (ziehen, schlagen, tauschen, etc.) werden und der Spieler wird in den folgenden Runden übersprungen.	Game, Board, , PlayerState, Piece, Node, BoardView, GUIView	JUnit	erfolgreich



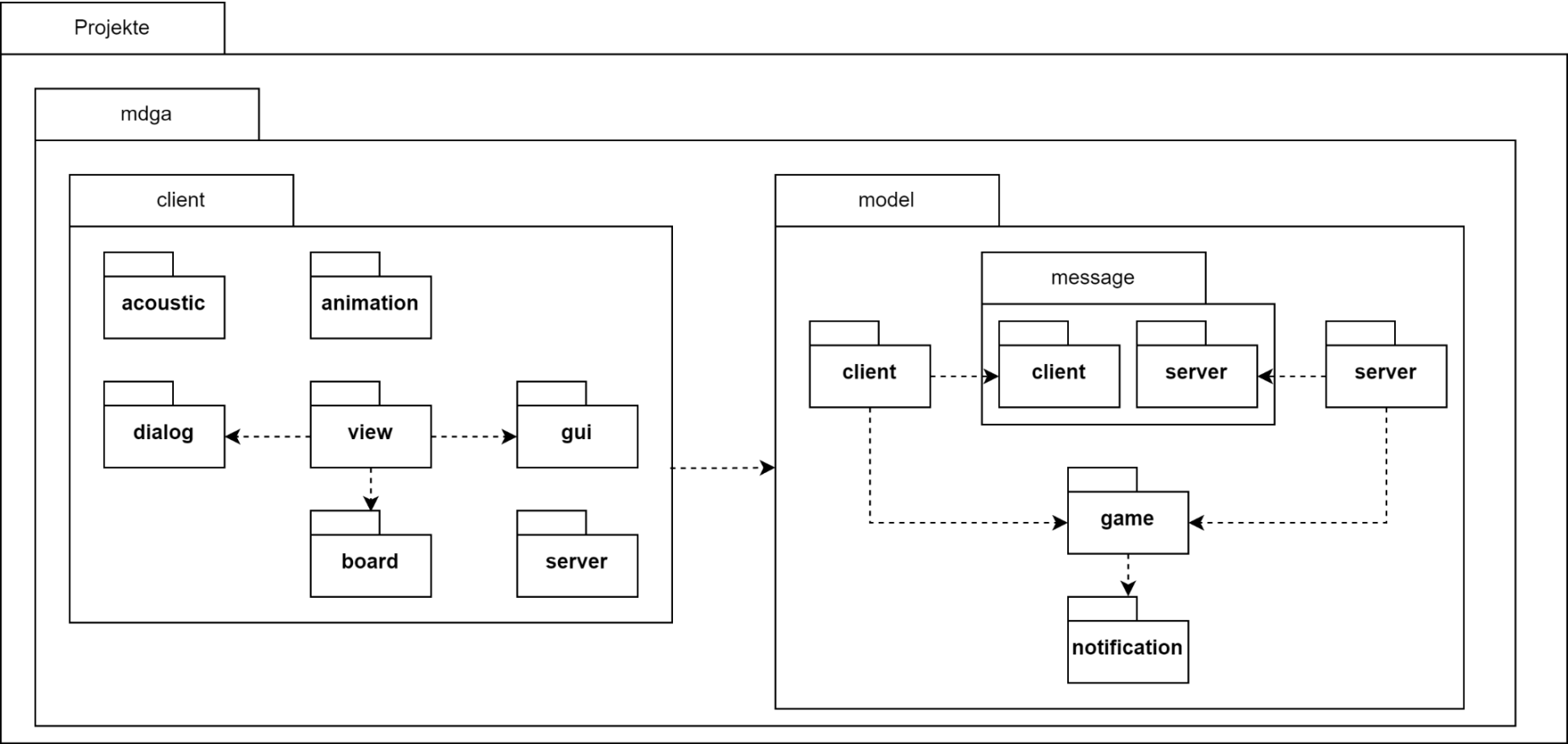
b. Aufbau der GUI

Hier wird der Aufbau der GUI grundlegend erklärt.

i. GUI im Spiel

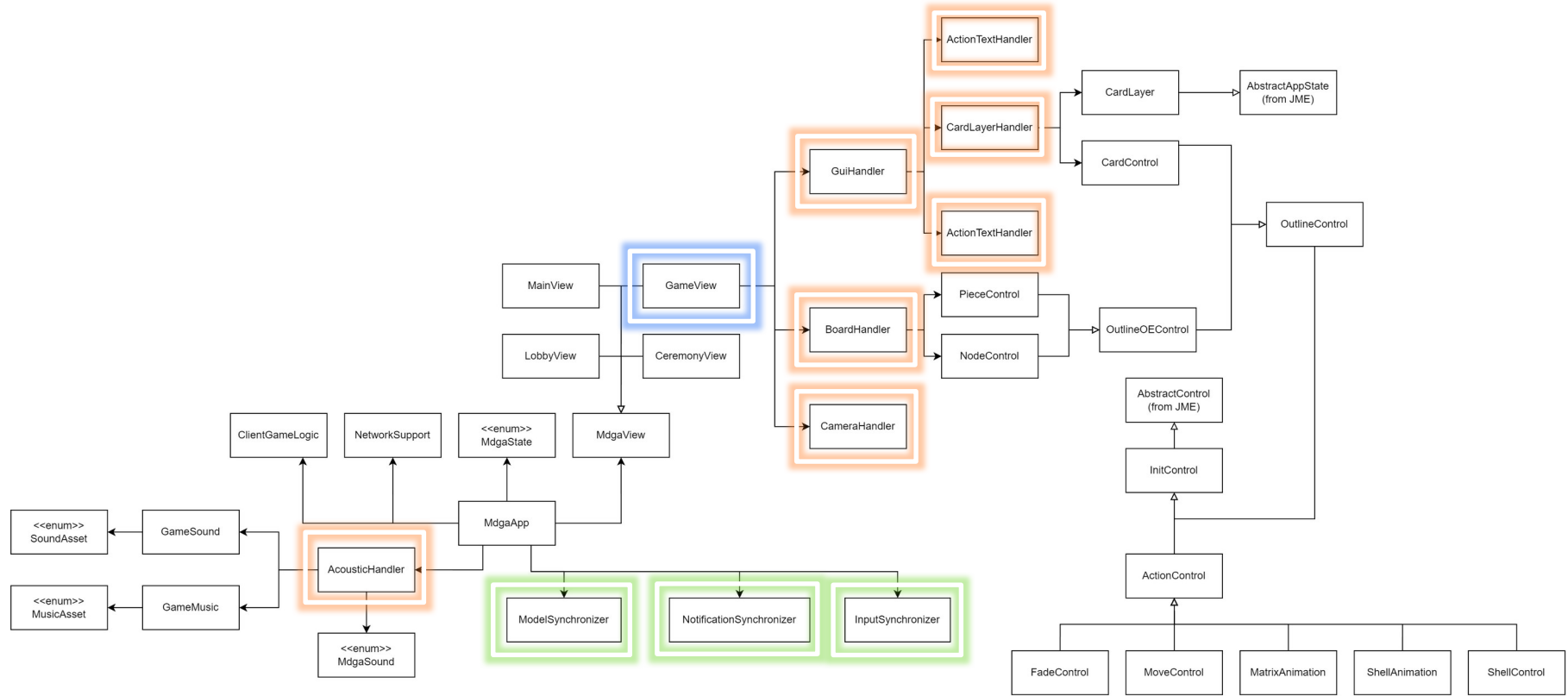


Zu Beginn startet die Kamera jedes Spielers in einer festgelegten Startposition, zusammen mit den Bonuskarten, die „auf der Hand“ im GUI angezeigt werden. Die Kamera kann nach Belieben gemäß der Steuerung bewegt werden, um das Spielfeld aus verschiedenen Blickwinkeln zu betrachten.





View



MapLoader

```
public enum Asset {  
    bigTent,  
    cardStack,  
    cir,  
    heer,  
    jet,  
    jet_noGear("Models/jet/jet_noGear.j3o", "Models/jet/jet_diff.png"),  
    lw,  
    marine,  
    node_home_blue("Models/node_home/node_home.j3o", "Models/node_home/node_home_blue_diff.png"),  
    node_wait_blue("Models/node_home/node_home.j3o", "Models/node_home/node_home_blue_diff.png"),  
    node_home_black("Models/node_home/node_home.j3o", "Models/node_home/node_home_black_diff.png"),  
    node_wait_black("Models/node_home/node_home.j3o", "Models/node_home/node_home_black_diff.png"),  
    node_home_green("Models/node_home/node_home.j3o", "Models/node_home/node_home_green_diff.png"),  
    node_wait_green("Models/node_home/node_home.j3o", "Models/node_home/node_home_green_diff.png"),  
    node_home_yellow("Models/node_home/node_home.j3o", "Models/node_home/node_home_orange_diff.png"),  
    node_wait_yellow("Models/node_home/node_home.j3o", "Models/node_home/node_home_orange_diff.png"),  
    node_normal,  
    node_start("Models/node_normal/node_normal.j3o", "Models/node_normal/node_start_diff.png"),  
    node_bonus("Models/node_normal/node_normal.j3o", "Models/node_normal/node_bonus_diff.png"),  
}
```

map.mdga

```
#CIR Pos  
cir 4,5 0  
cir 4,4 0  
cir 5,4 0  
cir 5,5 0  
  
#Assets  
jet -12,0 45  
jet -17,-2 55  
small_tent -9,7 45  
small_tent -10,5 60  
small_tent -7,8 30  
ship 11,0 169  
small_tent 6,8 340  
small_tent 8,7 320  
big_tent -10,-9 130  
big_tent 9,-10 225  
radar 0,10 -20  
tank -1,-10 135  
#tank 0,-18 180  
tank_shoot 0,-18 180  
tank 3,-18 180  
tank -3,-18 180
```

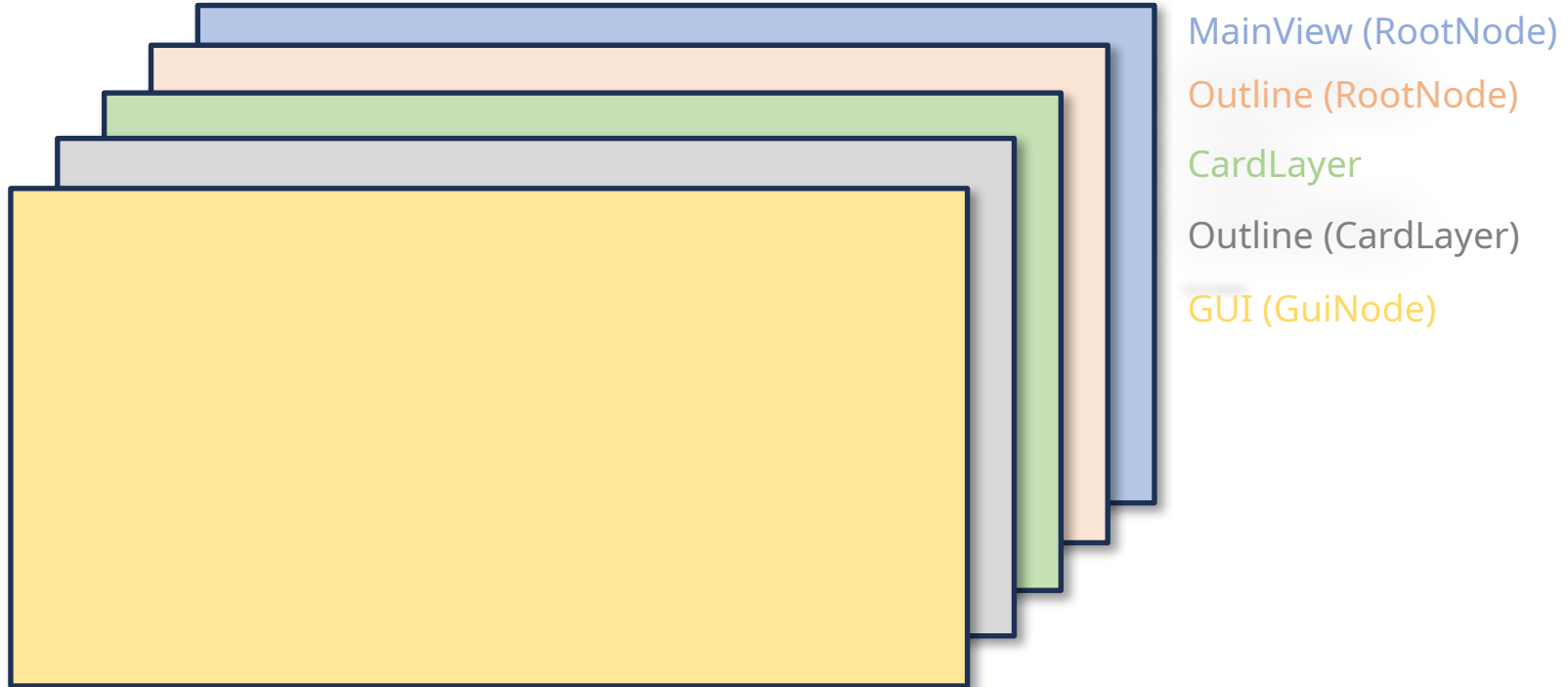
Outline



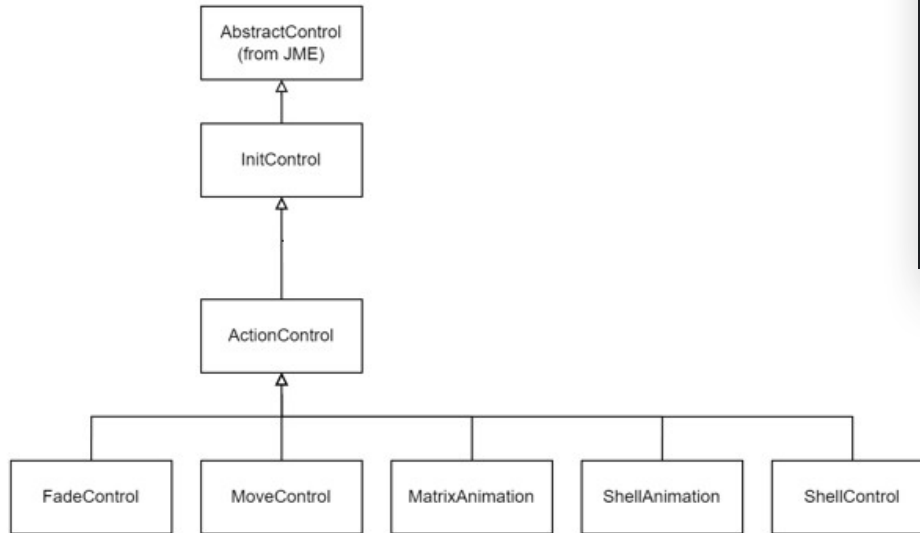
PostProcessing

- DirectionalLightShadowFilter
- SSAOFilter (Ambient Occlusion)
- FXAAFilter (Antialiasing)

Viewport Reihenfolge



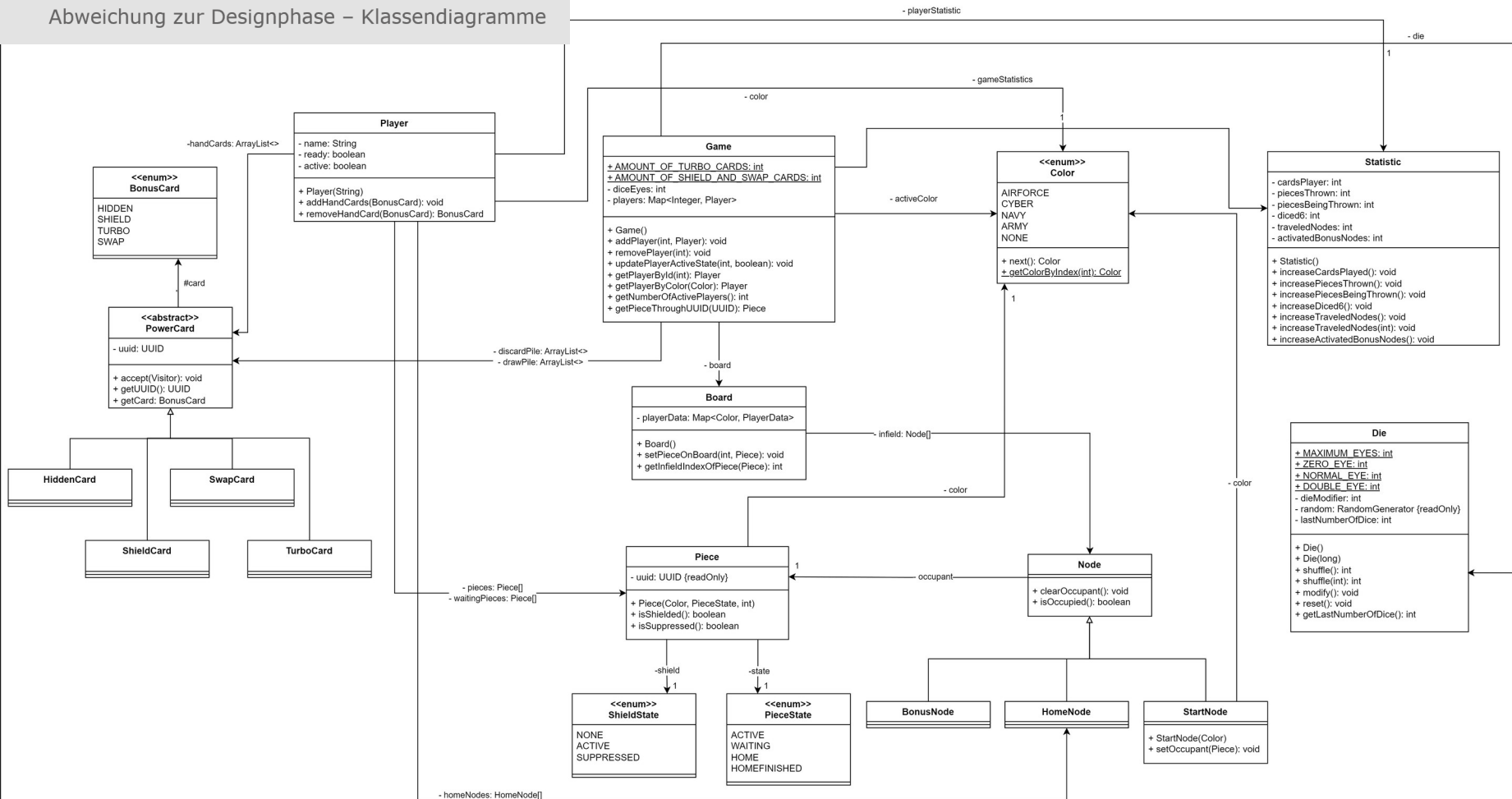
Animation



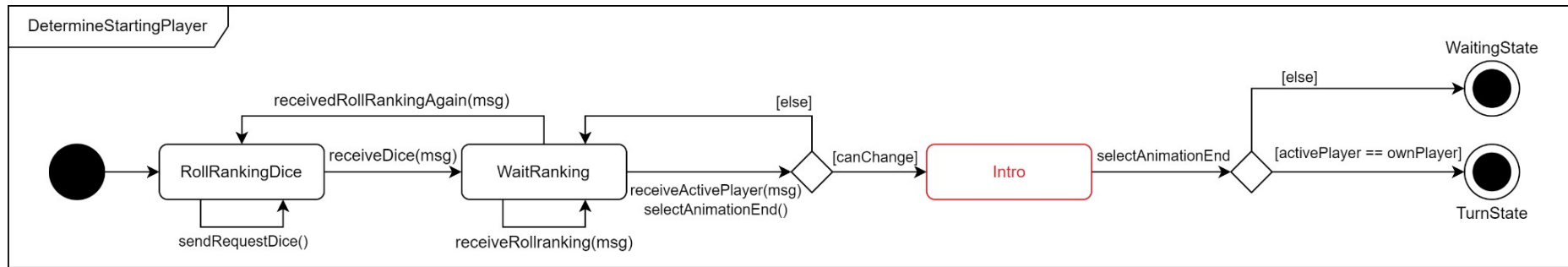
```
private void throwMatrix(UUID uuid) {
    app.getAcousticHandler().playSound(MdgaSound.MATRIX);

    Spatial piece = pieces.get(uuid).getSpatial();

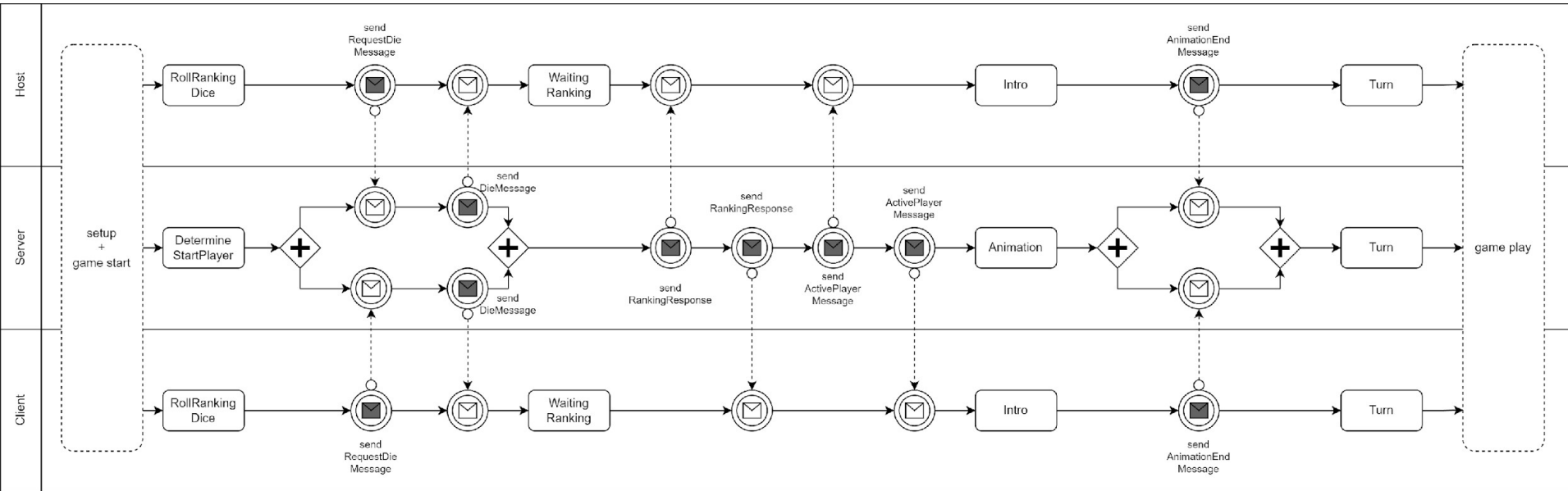
    piece.addControl(new MatrixAnimation(app, radarPos,
        () -> throwPieceAnim(uuid)
    ));
}
```

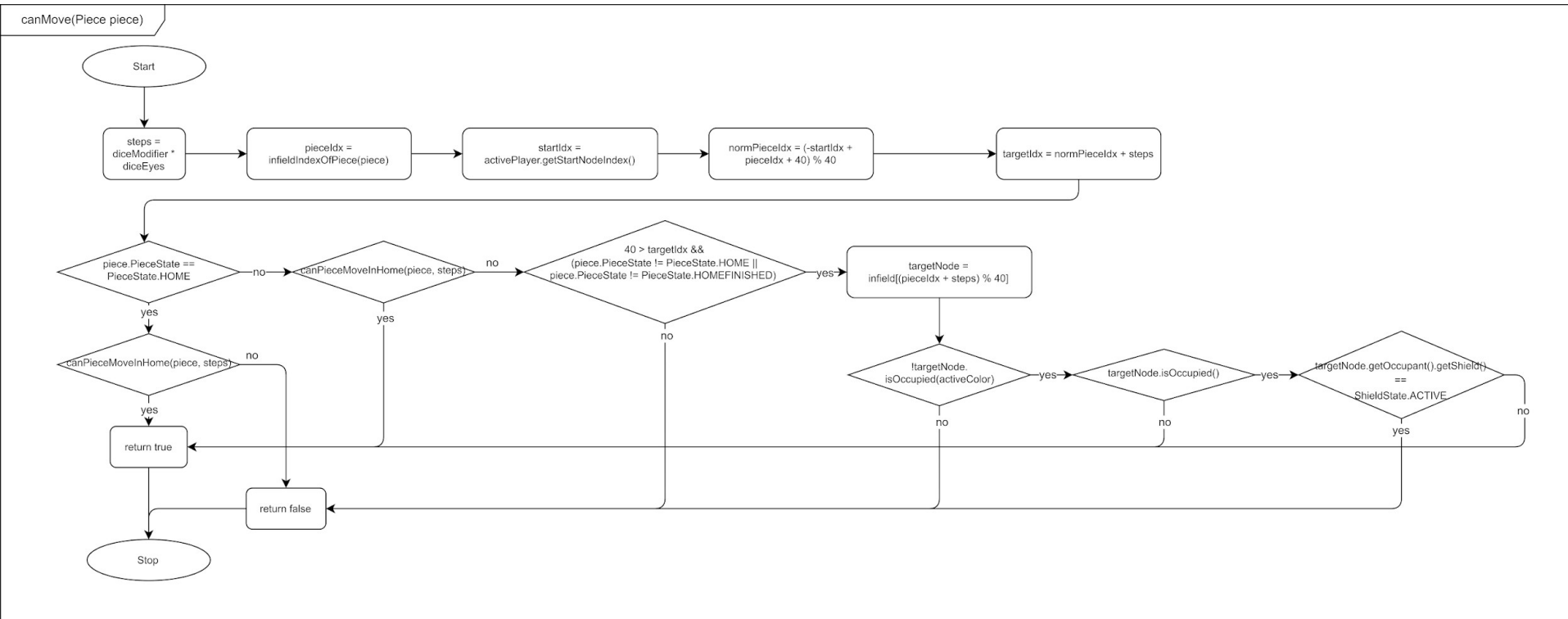


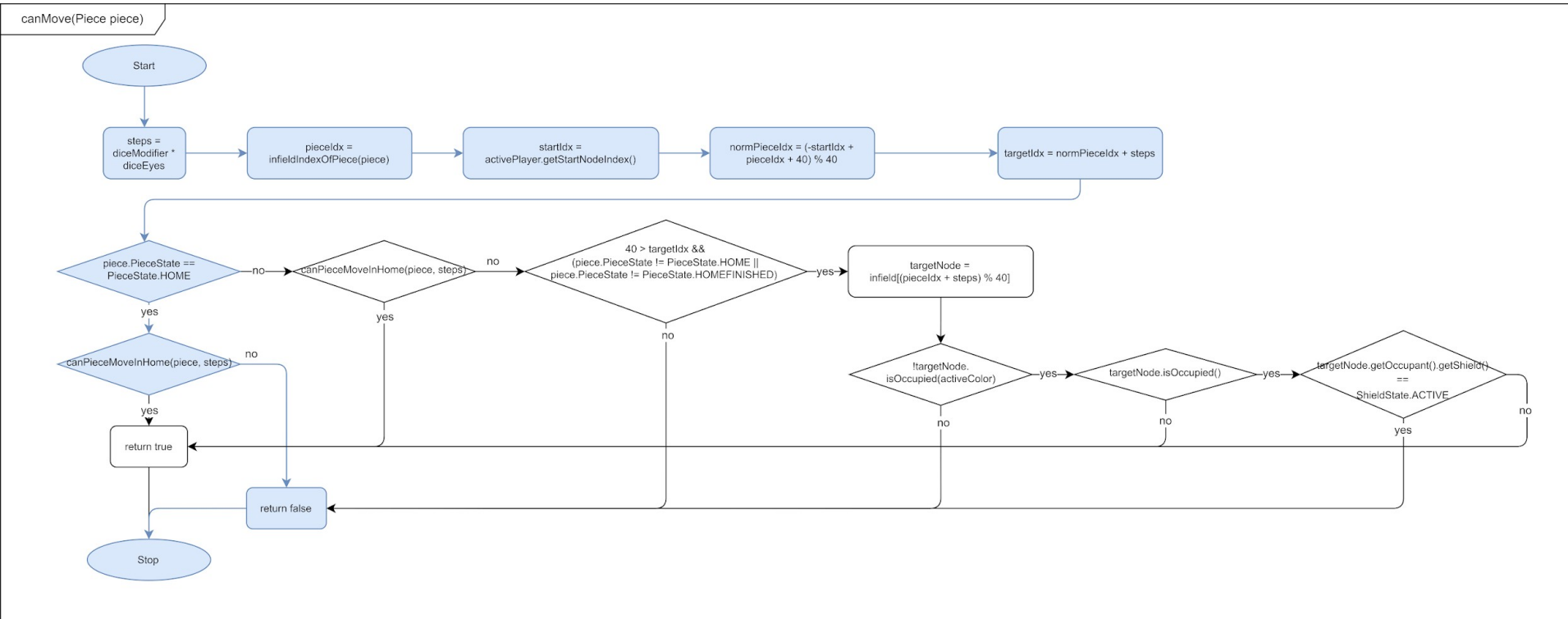
- neuer Zustand: Intro im Automaten DetermineStartPlayer des Clients

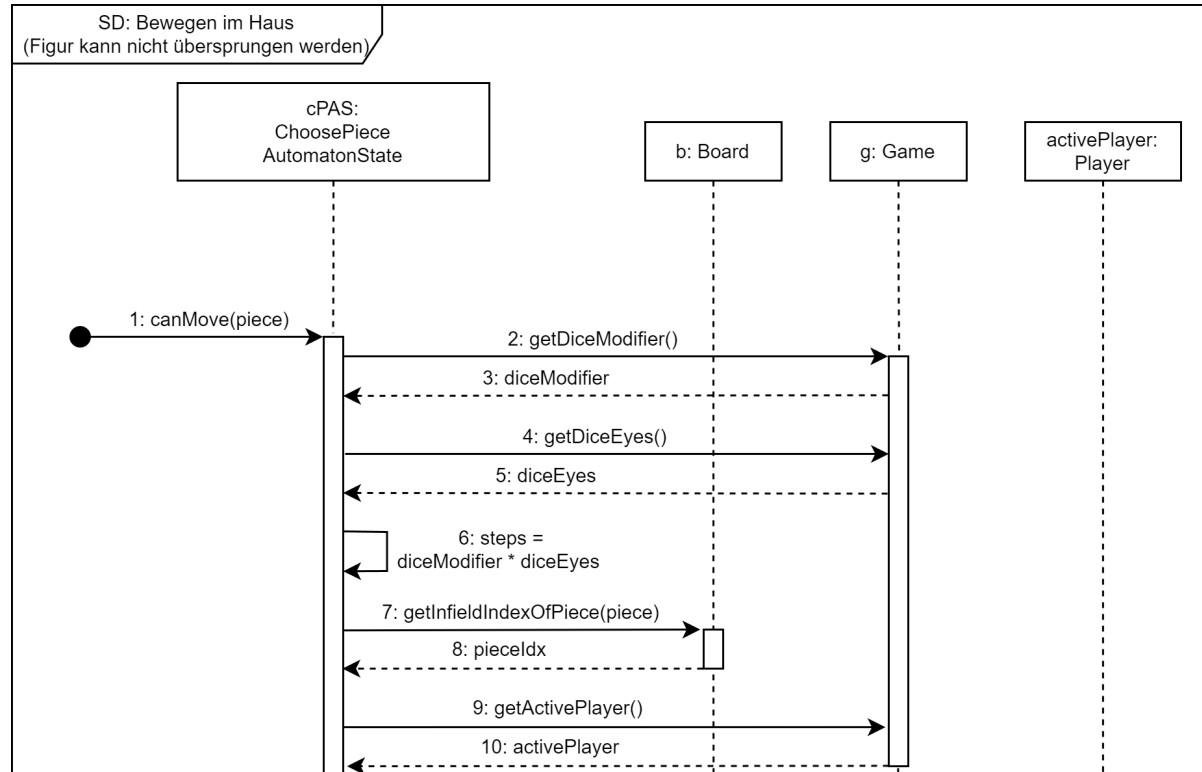


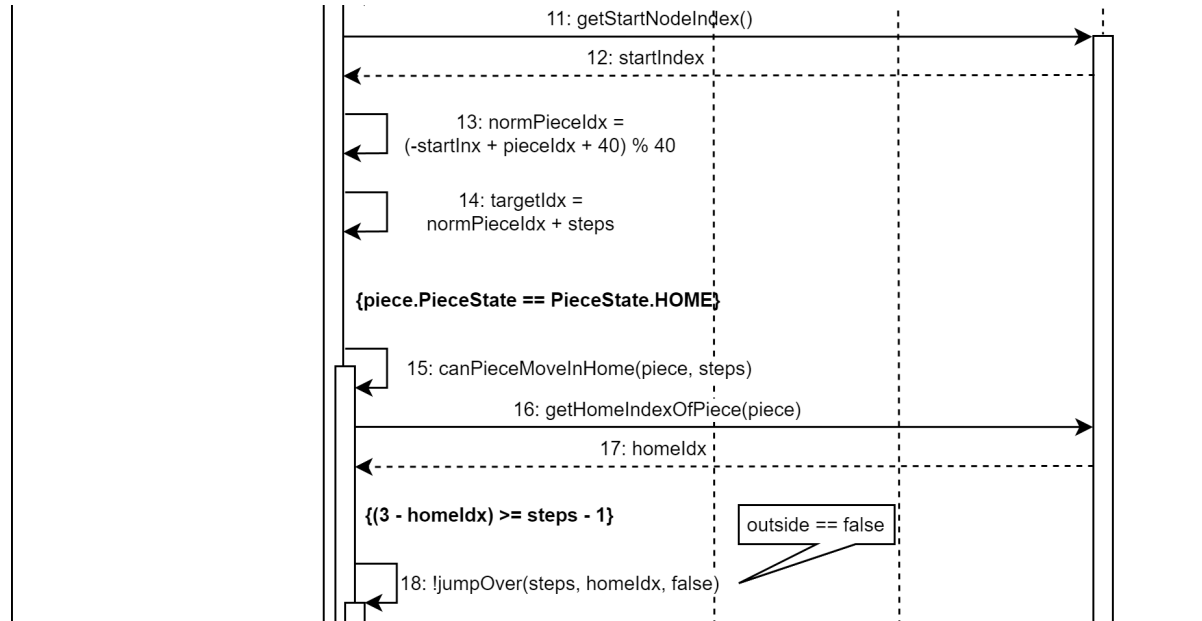
■ BPMN-Diagramm: Bestimmen des Startspielers













Vielen Dank!