Architektur

Komponenten

Das board

Die Grundkomponente ist das *board*. Hier befindet sich das Interface um ein Schachspiel zu spielen (siehe UML), allerdings noch keine genaue Implementierung des Schachfeldes, sondern unter anderem ein Interface *IBoard*, welches definiert welche Aktionen eine Schachfeld zur Verfügung stellt (siehe UML). In *board.simple* befindet sich dann eine Implementierung dieses Interfaces: *Board*.

Ein Nutzer von *board* kann jetzt also ein Board erstellen, was alle Methoden von *IBoard* besitzt und agiert ab dann nur noch über die Methoden vom *IBoard*.

Die Unterscheidung zwischen *IBoard* und *Board* wurde gewählt, um

1. Das Interface einfach, aufgeräumt und verständlich zu halten
2. Eventuell eine weitere Implementierung des *IBoard*s hinzufügen zu können, die z.B. auf Performance ausgelegt ist

Eine weitere Design-Entscheidung war es, dass ein Board unveränderbar sein soll. Damit ist es z.B. leichter für eine KI eine Tiefensuche zu machen, da sie keine Züge rückgängig machen muss, sondern einfach immer ein neues Board bekommt. Wenn man jetzt ein Zug ausführt verändert man nicht das aktuelle Board, sondern erstellt ein neues Board, worauf der Zug ausgeführt wird. (Siehe Abbildung 1)

Die Figuren

Die Figuren werden innerhalb eines Boards in der Klasse *BoardData* gespeichert, indem für jede Figur/Farbe Kombination ein *long* benutzt wird, wobei jeder Bit genau einem Feld auf dem Schachbrett entspricht. Hier haben wir uns dafür entschieden, die Figuren eher Daten orientiert als Objekt orientiert abzuspeichern, da es das Kopieren eines Boards erleichtert und es so auch einfach möglich ist Methoden wie *pieceAt* bereitzustellen. Mit diesen Methoden kann man dann einfach abfragen, welche Figur an einer bestimmten Stelle steht. (Siehe Abbildung 2)

Die ai

Die ai hat eine ähnliche Struktur wie das *board*: Es gibt ein Interface *IAI,* welches definiert was eine AI können soll. In unserem Fall zurzeit nur für ein gegebenes *IBoard* den (vermeintlich) besten Zug zu finden. Wie im *board* gibt es unter *ai.simple* eine Implementierung des Interfaces. (Siehe Abbildung 3)

Das Konsolen-Interface

Die Grundkomponente ist die Klasse *ConsoleMain*. Diese ist für das Verwalten von In-/Output verantwortlich und beinhaltet eine sogenannte “game loop”.

Eingaben werden an *command.CommandManager* weitergeleitet, der diese als einen Command klassifiziert und die spezielle “*Execute*” Methode von dem Command aufruft.

Der *CommandManager* ist als einziges von außerhalb des Paketes *command* sichtbar und dient als Schnittstelle zwischen der *ConsoleMain* Klasse und den Commands. Damit ist eine Abstraktionsschicht geschaffen, die das Interface von den Commands trennt.

Alle Commands erweitern die abstrakte Klasse *Command*. Dies dient dazu, dass der *CommandManager* ohne spezielle Kenntnis über die Implementierung des Commands, diesen ausführen kann. Außerdem ist so die Implementierung von weiteren Commands strukturell klarer gegeben. (Siehe Abbildung 4)

Die 2D-GUI

Für die 2D-GUI, haben wir uns für das MVC Design-Pattern entschieden. Dadurch ist die Trennung zwischen den Daten und der Darstellung gegeben. Dies ermöglicht uns ohne großen Aufwand für dieselben Daten verschiedene Darstellungen anzubieten. Außerdem ist durch die klare Trennung das parallele Arbeiten am Code deutlich einfacher.

Der Code der 2D-GUI befindet sich in dem *gui* package und dessen subpackages. Jedes subpackage von dem GUI package korrespondiert dabei mit einem Interface (*game* = Spiel Ansicht, *main* = Hauptmenü, …). Diese subpackages bestehen aus einem View, einem Controller, einem Modell und gegebenenfalls Hilfsklassen. Der View ist für die Darstellung der Daten, die in dem Modell gespeichert werden, verantwortlich. Der Controller verwaltet die Interaktionen zwischen der 2D-GUI und dem Benutzer.

Erweiterbarkeit und Wartbarkeit

Durch die klare Trennung zwischen Board <-> Game <-> UI/Console, können alle Komponenten leicht erweitert werden, ohne dass die anderen Komponenten angepasst werden müssen.

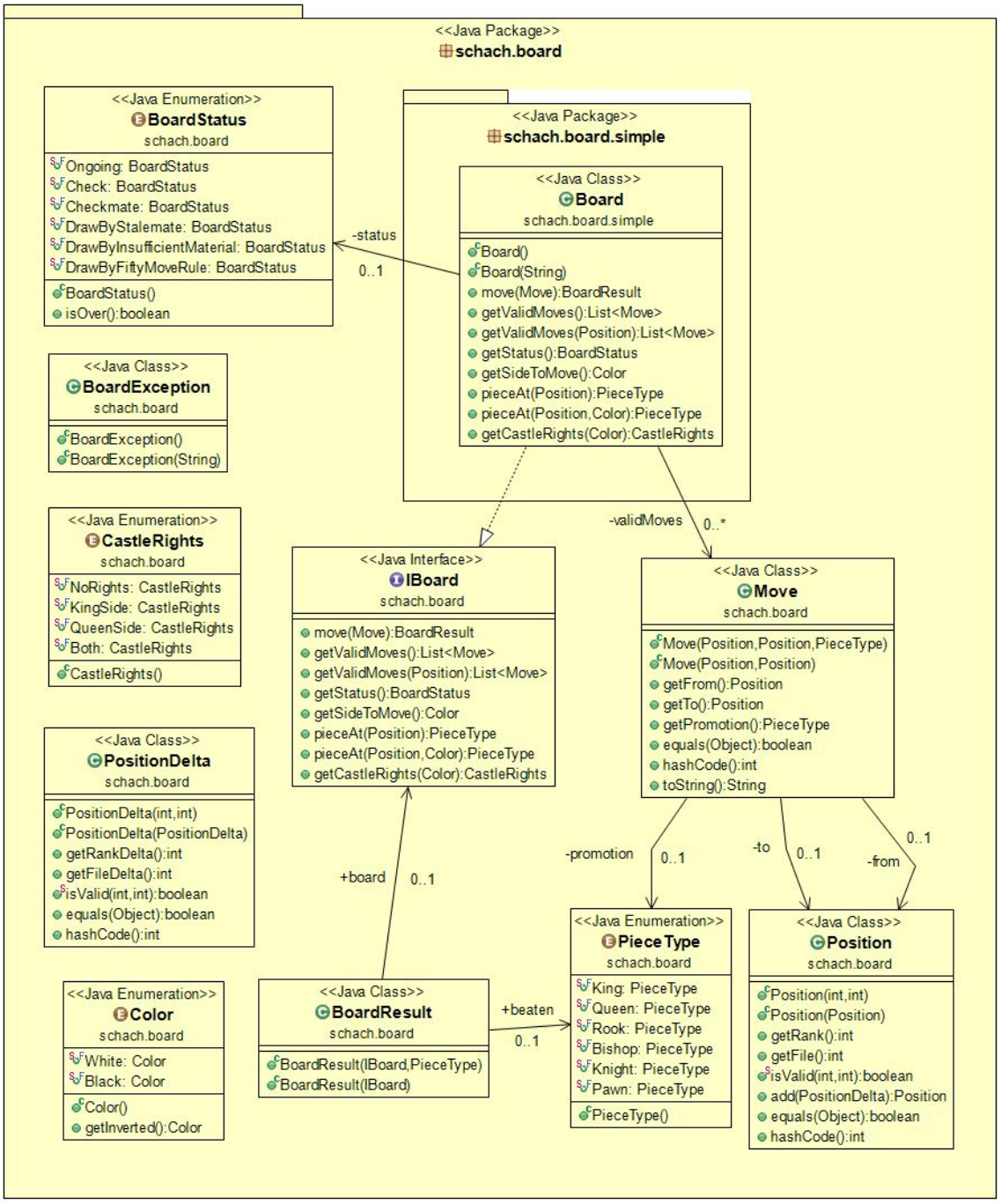

Abbildung 1: board Architektur

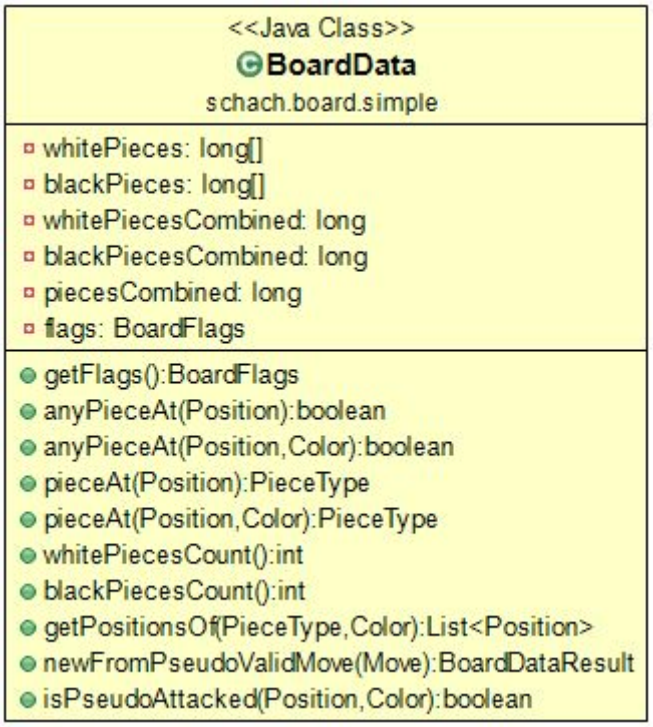

Abbildung 2: BoardData

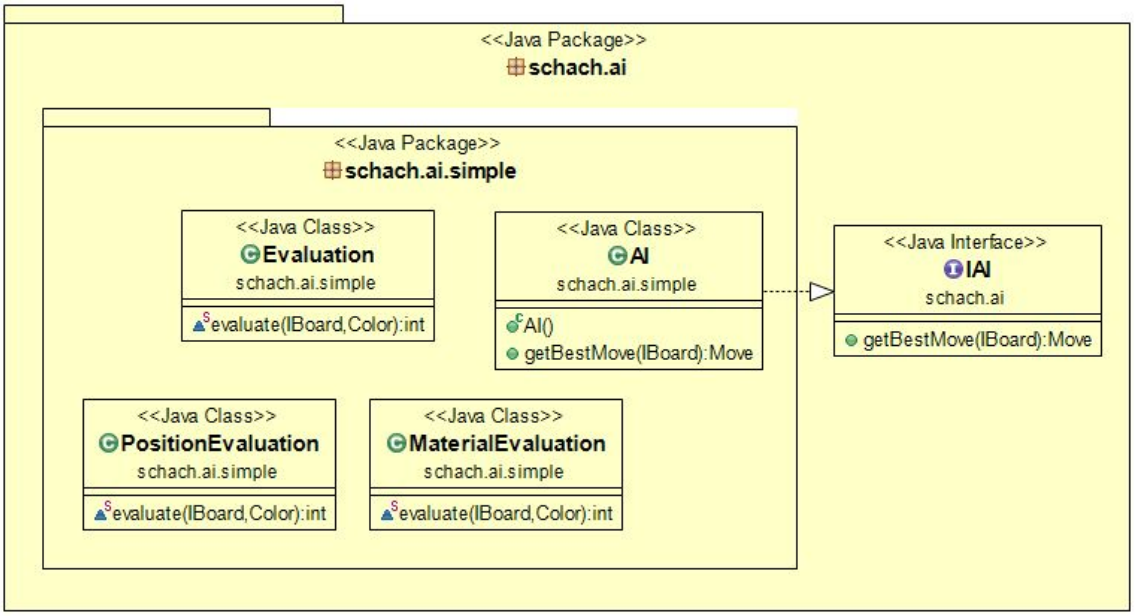

Abbildung 3: AI Architektur

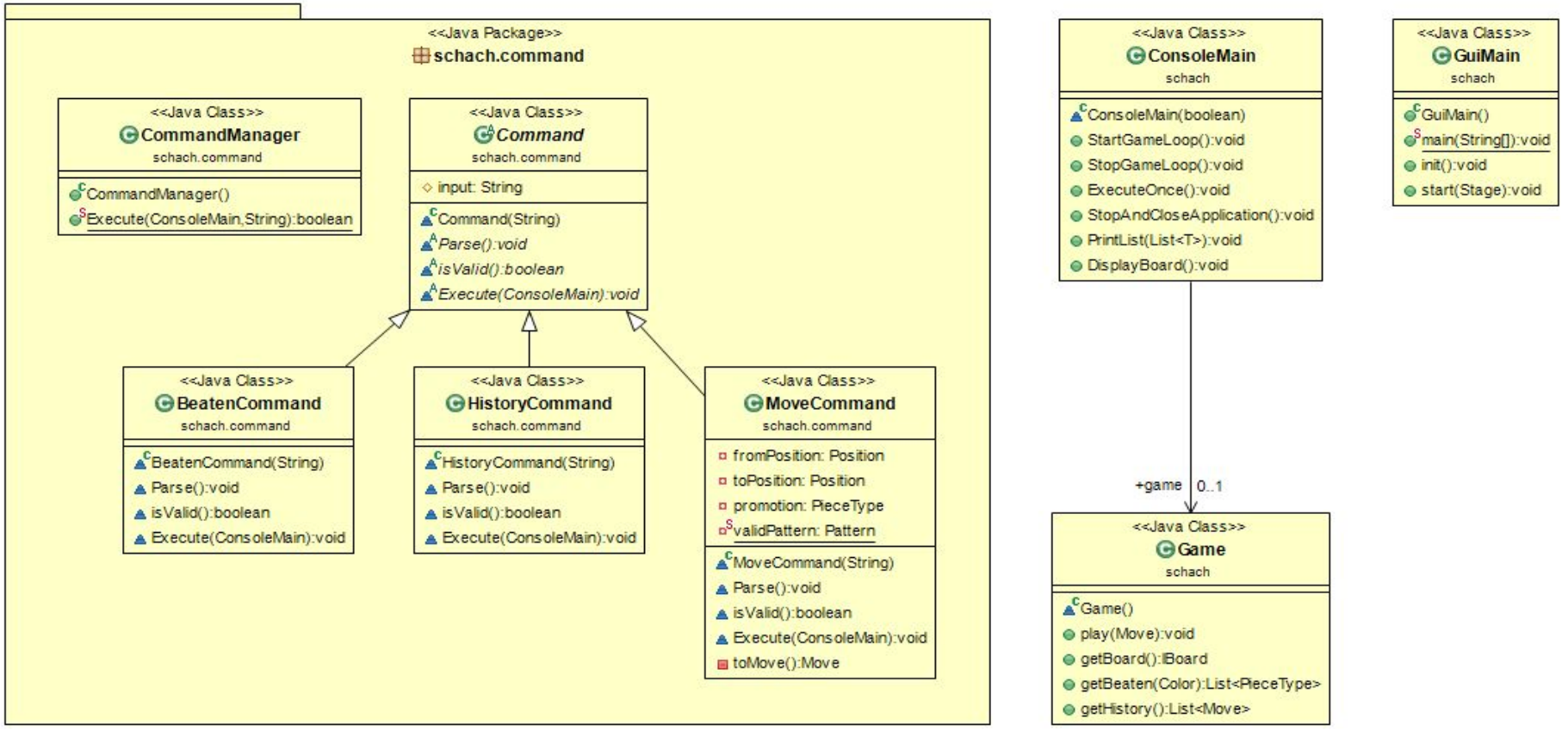

Abbildung 4: Konsolen-Interface