**Design:** client (framework)

Datum: 25 - 3 - 2014  
Auteur: Stephan Veenstra

# 1. Versiebeheer

|  |  |
| --- | --- |
| **Versie** | **Aanpassing** |
| 0.1 | Opzet document |
| 0.2 | Toevoeging van eisen |

# 

# 

# 2. Inhoudsopgave

[1. Versiebeheer](#h.25ekfa8el28g)

[2. Inhoudsopgave](#h.hsie9z3hglku)

[3. Inleiding](#h.5bph7dtmtdma)

[4. Eisen](#h.590ugkbisw8y)

[4.1. Uitbreidbaarheid](#h.h6shg9orfq9y)

[4.2. Dynamic AI](#h.ievo2ttr9b5b)

[4.3. Networking](#h.ns8hi12014rf)

[5. Interactie model](#h.rsa8y12u63kw)

[5.1. Model](#h.ls4840o7cqnz)

[5.2. View](#h.mqi2hvb6wdn)

[5.3. Controller](#h.45gsr9a5vbqj)

[5.4. AI (Artificial Intelligence)](#h.puzp59l2txzj)

[5.5. Server](#h.o7ewfe18ihpr)

[6. Structuur](#h.mz0k37st7s35)

[7. Klassendiagram](#h.6sew7ha5prwf)

# 3. Inleiding

Dit document bevat het globale ontwerp van de client. De client is het framework dat moet kunnen worden uitbreid met nieuwe modules zonder dat het framework hoeft worden aangepast. Er wordt verder niet ingegaan op de diepere ontwerpen van bijvoorbeeld de interface. Daarvoor zijn aparte ontwerpdocumenten opgesteld.

Eerst worden de eisen van de client behandeld om een duidelijk beeld te krijgen van wat er van de applicatie verwacht wordt.

Dan worden de interacties tussen de verschillende componenten toegelicht. Daarnaast worden de verantwoordelijkheden van die componenten benoemd en omschreven.

Vervolgens wordt de mappen/ namespace structuur behandeld die wordt gehanteerd bij het programmeren van de client.

# 4. Eisen

Hierin worden de eisen besproken waaraan het framework moet voldoen.

## Product eisen

**Uitwisselbaarheid (P1.1) (M)**

Het framework moet nieuwe spellen kunnen laden en spelen(AI) zonder dat het framework daar verder voor moet worden aangepast.

**Koppelbaarheid(P1.2) (M)**

Het framework moet via de StrategicGameServer kunnen spelen tegen andere frameworks. Dit betekent dat ze via zich moeten houden aan het zelfde protocol.

**Functioneel geschiktheid (P2) (M)**

Het framework moet de gebruiker in staat stellen de verwachte acties die betrekking hebben tot de spelregels van de ingeladen spellen te kunnen uitvoeren.

**Snelheid (P3.1) (M)**

Het AI van het framework moet de volgende zet kunnen berekenen/ uitvoeren binnen de bedenktijd van het spel (10 sec).

**Capaciteit(P3.2) (M)**

Het framework moet alle vergelijkbare spelen( turn-based, two-player) kunnen spelen aan de hand van de ingeladen spelregels.

**Betrouwbaarheid (P4) (M)**

De gebruiker moet ervan kunnen uitgaan dat de AI van het framework de beste zet berekent.

**Modulariteit(P5.1) (M)**

Het framework moet zo worden opgebouwd dat nieuwe modules(games) kunnen worden ingeladen. Daarnaast moet het mogelijk zijn nieuwe views (weergaven) te kunnen maken zonder dat deze de rest van het programma (logica) beïnvloeden.

**Herbruikbaarheid (P5.2) (M)**

Het framework moet gebruikmaken van de zelfde vorm van modules als de StrategicGameServer zodat deze niet tweemaal gemaakt hoeven worden.

**Wijzigbaarheid (P5.3) (M)**

Het framework moet zo worden opgebouwd dat het aanpassen van onderdelen geen invloed heeft op andere onderdelen van het framework.

## Gebruikeisen

**Bruikbaarheid (G1.1) (M)**

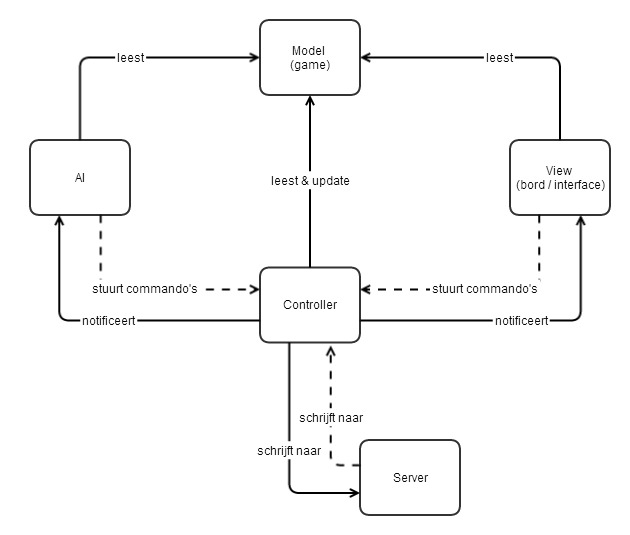
De gebruikers moeten intuïtief gebruik kunnen maken van het framework.

**Vertrouwen (G1.2) (M)**

De gebruikers moeten vertrouwen hebben in het AI van het systeem dat deze de juiste keuzes zal maken.

# 5. Interactie model

In dit hoofdstuk wordt omschreven hoe de componenten met elkaar in verbinding staan en wat hun relatie is.



## 5.1. Model

De model is het spel dat gespeeld kan worden. De model houdt informatie bij als bijvoorbeeld welke speler aan zet is, wat het speelveld is en wat de spelregels zijn.

Het model kan worden gelezen door de Views en de AI. De controller kan de model zowel lezen als updaten(bewerken).

## 5.2. View

De views tonen informatie aan de gebruiker. Ze maken het voor de gebruiker makkelijker om de acties van de controller uit te voeren.

De views lezen het model. De controller kan aangeven dat het model is veranderd waarna de view kan updaten.

## 5.3. Controller

De controller handelt acties af van de user-input(views), AI en de server. Hij stuurt ook de commando’s naar de server.

De controller weet welke views er zijn en kan deze een notificatie sturen als het nodig is dat ze worden geupdate. De controller weet ook van het model zodat hij deze kan aanpassen en lezen.

## 5.4. AI (Artificial Intelligence)

De AI maakt gebruik van de game(model) om de win-kans en beste zet te berekenen. Daarnaast kan de AI ook zelfstandig spellen gaan spelen.

De AI leest het model om zijn acties te bepalen en berekeningen te doen. De AI stuurt opdrachten naar de controller om het spel te spelen.

## 5.5. Server

De server ontvangt en verstuurt opdrachten van/ naar de clients. Deze is al operationeel en heeft een protocol waarop onze client gebaseerd wordt.

# 6. Structuur

De klassen worden netjes onderverdeeld in packages. Hieronder volgt in grote lijnen hoe dit wordt gedaan.

* framework.ai <package>
* framework.model <package>
* framework.view <package>
* framework.controller <package>
* framework.gui <package>
* gamesmodules <folder>

De gamesmodules folder bevat .jar bestanden die de informatie van een game bevatten. Deze worden dynamisch ingeladen.

# 7. Klassendiagram

