

CS2: The Kiss of death

Tom Kaal, Sander Koenders en Frank Blom



2 november 2015

Han Ica – OOSE OOAD

Harrie van Seters

# Inhoudsopgave

Inhoud

[Inhoudsopgave 1](#_Toc437899381)

[1. Inleiding 3](#_Toc437899382)

[2. Actoren en NPI 4](#_Toc437899383)

[2.1. Actoren 4](#_Toc437899384)

[2.2. NPI 4](#_Toc437899385)

[3. Furps 6](#_Toc437899386)

[4. Use Case Diagram 7](#_Toc437899387)

[5. Use Cases 8](#_Toc437899388)

[5.1. UC1: Verdedigen 8](#_Toc437899389)

[5.2. UC2: Kussen 9](#_Toc437899390)

[5.3. UC3: Attribuut laten vallen 10](#_Toc437899391)

[5.4. UC4: Aanvallen 11](#_Toc437899392)

[5.5. UC5: Chatten 12](#_Toc437899393)

[5.6. UC6: Bewegen 13](#_Toc437899394)

[5.7. UC7: Aanmelden 14](#_Toc437899395)

[6. Domain model 15](#_Toc437899396)

[7. Activity Diagram 16](#_Toc437899397)

[8. System Diagrams & System Sequence Diagrams 17](#_Toc437899398)

[8.1. SD Aanmelden 17](#_Toc437899399)

[8.2. SD Aanvallen 17](#_Toc437899400)

[8.3. SD Attribuut laten vallen 17](#_Toc437899401)

[8.4. SD Bewegen 17](#_Toc437899402)

[8.5. SD Chatten 17](#_Toc437899403)

[8.6. SD Kussen 17](#_Toc437899404)

[8.7. SSD Aanmelden 17](#_Toc437899405)

[8.8. SSD Aanvallen 17](#_Toc437899406)

[8.9. SSD Attribuut laten vallen 17](#_Toc437899407)

[8.10. SSD Bewegen 17](#_Toc437899408)

[8.11. SSD Chatten 17](#_Toc437899409)

[8.12. SSD Kussen 17](#_Toc437899410)

[8.13 SSD Verdedigen 17](#_Toc437899411)

[9. State Machine Diagram 18](#_Toc437899412)

[10. Design Class Diagram 19](#_Toc437899413)

[11. OO-ontwerpprincipes 20](#_Toc437899414)

[12. Architectuurmodel 21](#_Toc437899415)

# Inleiding

# Actoren en NPI

## Actoren

Als actoren hebben wij:

* Speler
* Systeem

## NPI

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Spel | Spel |  | Spel |  |
| Detail |  |  |  |  |
| Overzicht |  |  |  |  |
| Startwaarden |  |  |  |  |
| Variabelen |  |  |  |  |
| Waarden |  |  |  |  |
| File |  |  |  |  |
| Database | Database |  |  |  |
| Kamers | Kamers |  | Kamers |  |
| Grot | Grot |  | Grot |  |
| Deuren | Deuren |  | Deuren |  |
| Elf | Elf |  | Elf |  |
| Dwerg | Dwerg |  | Dwerg |  |
| Monster | Monster |  | Monster | Gedaante |
| Strijder | Strijder |  | Strijder |  |
| Wezens | Wezens |  | Wezens | Levenspunten |
|  |  |  |  | Energie |
|  |  |  |  | Kracht |
|  |  |  |  | Geslacht |
| Boodschappen | Boodschappen |  | Boodschappen |  |
| Vorm |  |  |  |  |
| Schematisch |  |  |  |  |
| tegel | tegel |  | tegel |  |
| speelveld | speelveld |  | speelveld |  |
| Muurdelen | Muurdelen |  | Muurdelen |  |
| Vakje | Vakje |  | Vakje |  |
| Gedaante | Gedaante |  |  |  |
| Kussen |  |  |  |  |
| Speler | Speler |  | Speler |  |
| Medespeler |  |  |  |  |
| Type |  |  |  |  |
| Stategie |  |  |  |  |
| Kikker | Kikker |  | Kikker |  |
| Levenspunten | Levenspunten |  |  |  |
| Energie | Energie |  |  |  |
| Kracht | Kracht |  |  |  |
| Geslacht | Geslacht |  |  |  |
| Attribuut | Attribuut |  | Attribuut |  |
| Zwaard | Zwaard |  | Zwaard |  |
| Harnas | Harnas |  | Harnas |  |
| Knuppel | Knuppel |  | Knuppel |  |
| Gevecht |  |  |  |  |
| Gaten | Gaten |  | Gaten |  |
| Vloer |  |  |  |  |
| Bezemsteel | Bezemsteel |  | Bezemsteel |  |
| Vitamines | Vitamines |  | Vitamines |  |
| Winnaar |  |  |  |  |
| Intranet |  |  |  |  |
| Server |  |  |  |  |
| Client |  |  |  |  |
| Clientvenster |  |  |  |  |
| Spelerskarakter |  |  |  |  |
| Object | Object |  | Object |  |

# Furps

**Functionality**

* Een spel
* Items oppakken.
* Door deuren heen lopen.
* Over gaten heen springen.
* Boodschappen sturen.
* De speler kan op een vakje staan.
* De grot (oftewel speelveld) kan meerdere vormen aannemen.
* Elkaar aanvallen waardoor de levens punten gewijzigd worden.
* Een speler kan zich verdedigen.
* Een speler kan zich bewegen in het spel maar alleen wanneer de speler genoeg levenspunten heeft.
* Een speler kan zich ook sneller dan normaal lopen, dit kost wel het dubbele aantal punten.
* Een speler kan een andere speler kussen.
* Vliegen met een bezem om naar een willekeurige plaats te vliegen.
* Een speler kan vitamines opeten om levenspunten aan te vullen.
* Elk type wezen kan een bijbehorend attribuut dragen.
* Een speler kan een attribuut oppakken en neerleggen
* Het bijhouden van een attribuut kost extra levenskracht.
* Een speler kan zich aanmelden in systeem om mee te doen.
* Het spel start pas wanneer er 9 personen in zitten.
* Het spel genereert 3 teams met daarin 3 wezens van dezelfde types.
* Eén van de gegenereerde wezens wordt automatisch een monster.
* Het spel wordt afgesloten wanneer het monster verslagen is of wanneer alle spelers dood zijn.
* Het stel wordt gespeeld in 2D.
* Speler kunnen chatten met andere speler die van dezelfde types zijn.
* Bij 0 levenspunten is het wezen dood.
* Het spel kan altijd gestopt worden.

**Usability**

* Het spel moet gemakkelijk speelbaar zijn

**Reliability**

**Performace**

* Realtime communicatie

**Supportability**

* Het spel moet op het intranet werken
* Het draait op een server

# Use Case Diagram

# Use Cases

## UC1: Verdedigen

|  |  |
| --- | --- |
| **Primary Actor:** <Speler> | |
| **Stakeholders:** <Tegenspeler> | |
| **Brief Description:** Een speler verdedigt zichzelf tegen een aanval van een <Tegenspeler>. Het verdedigen kost altijd energie. Of en hoeveel levenspunten er af gaan is afhankelijk van of de aanval lukt en hoe sterk de <Speler> en de <Tegenspeler>is. | |
| **Preconditions:** De <Speler> heeft al zijn levens nog en staat naast een <Tegenspeler>. De <Speler> word succesvol aangevallen door de <Tegenspeler>maar verdedigd op tijd. | |
| **Postconditions:** De <Speler> heeft minder levenspunten wanneer het gevecht is afgelopen. | |
| **Main Success Scenario:** | |
| 1. De <Speler> geeft aan te willen verdedigen. | 2. Het systeem haalt van de <Speler> energie af. |
| . | 3. Het systeem berekent op basis van kans of er schade aan de <Speler> wordt toegebracht. |
|  | 4. Het systeem berekent op basis van de hoeveelheid kracht en attributen van zowel de <Speler> als <Tegenspeler>, hoeveel levenspunten er af gehaald moeten worden. |
| **Extensions (Alternative Flows):** | |
|  | 3a1 [Systeem berekend dat er geen schade aan de <Speler> word toegebracht]  3a2 Er worden levenspunten afgehaald van de <Tegenspeler>. |

## UC2: Kussen

|  |  |
| --- | --- |
| **Primary Actor:** <Speler> | |
| **Stakeholders:** <Tegenspeler>, <Medespeler> | |
| **Brief Description:** Een <Speler> kust een <Tegenspeler>. | |
| **Preconditions:** De <Speler> staat naast een <Tegenspeler>en heeft voldoende levenspunten om te kussen. | |
| **Postconditions:** De <Tegenspeler> veranderd in een kikker. | |
| **Main Success Scenario:** | |
| 1. De <Speler> geeft aan te willen kussen | 2. Het systeem haalt de helft van de <Speler> zijn levenspunten er af. |
| . | 3. Het systeem veranderd de <Tegenspeler> in een kikker. |
| **Extensions (Alternative Flows):** | |
|  | 3a1 [<Tegenspeler> is het monster]  3a2 Het systeem geeft aan dat het monster dood is en het team waar de speler in zit gewonnen heeft. |
|  | 3b1 [<Tegenspeler> is een kikker van een ander type]  3b2 [Use Case eindigt] |
|  | 3c1 [<Medespeler> is een kikker van een hetzelfde type]  3c2 Het systeem verander de kikker terug naar zijn originele gedaante. |

## UC3: Laten vallen <Attribuut>

|  |  |
| --- | --- |
| **Primary Actor:** <Speler> | |
| **Stakeholders:** - | |
| **Brief Description:** Een <Speler> laat een attribuut vallen. | |
| **Preconditions:** De <Speler> is in bezit van een attribuut. | |
| **Postconditions:** De <Speler> is niet meer in het bezit van een attribuut. | |
| **Main Success Scenario:** | |
| 1. De <Speler> geeft aan zijn attribuut te willen laten vallen | 2. Het systeem haalt het attribuut weg bij de <Speler>. |
| . | 3. Het systeem legt het attribuut neer op het vakje waar de <Speler> zich bevind. |

## UC4: Aanvallen

|  |  |
| --- | --- |
| **Primary Actor:** <Speler> | |
| **Stakeholders:** <Tegenspeler> | |
| **Brief Description:** Een <Speler> valt een andere speler aan | |
| **Preconditions:** Het karakter van de <Speler> is geen kikker en de <Speler> heeft genoeg energie | |
| **Postconditions:** Levenspunten zijn afgeschreven | |
| **Main Success Scenario:** | |
| 1. De <Speler> geeft aan een <Tegenspeler> te willen aanvallen. | 2. Systeem vraagt op wie de aanval uitgevoerd moet worden |
| 3. De <Speler> geeft aan op welke <Tegenspeler> de aanval uitgevoerd moet worden | 4. Het systeem checkt of er een <Tegenspeler> aanwezig is op de aan te vallen locatie. |
|  | 5. Het systeem valt aan op de aan te vallen locatie met <<init>> vertraging. |
|  | 6. Het systeem controleert of er verdedigd word door de <Tegenspeler>. |
|  | 7. Het systeem berekent de slagingskans van de aanval. |
|  | 8. Het systeem bepaald m.b.v. de slagingskans de winnaar en de verliezer van de aanval. |
|  | 9. Het systeem verminderd de levens punten van de verliezer met <<init>> |
|  | 10. Het systeem verminderd de energie van de <Speler>. |
| **Extensions (Alternative Flows):** | |
|  | 9a1 [<Speler> heeft te weinig levenspunten na de aanval van de <Tegenspeler>]  9a2 De <Speler> wordt uit het spel verwijderd. |
|  | 9b1. [<Tegenspeler>heeft te weinig levenspunten na de aanval van de <Speler> en is het monster]  9b2 De <Tegenspeler> verandert in zijn werkelijke ‘gore’ gedaante. |

## UC5: Chatten

|  |  |
| --- | --- |
| **Primary Actor:** <Speler> | |
| **Stakeholders:** <Medespeler> | |
| **Brief Description:** <Speler> van hetzelfde type kunnen elkaar boodschappen sturen. | |
| **Preconditions:** De zendende <Speler> en de ontvangende <Medespeler> zijn van hetzelfde type, de <Speler> is geen kikker. | |
| **Postconditions:** Er is een boodschap verzonden naar de <Medespeler>. | |
| **Main Success Scenario:** | |
| 1. De <Speler> geeft aan een boodschap te willen sturen. | 2. Het systeem vraagt om naar welke <Medespeler> het bericht gestuurd moet worden |
| 3. De <speler> geeft aan naar welke <Meldespeler> het bericht moet | 2. Het systeem vraagt naar het bericht wat verstuurd moet worden. |
| 3. De <Speler> geeft het bericht wat verstuurd moet worden. | 4. Systeem geeft de op optie om het bericht te sturen. |
| 4. De <Speler> geeft aan het bericht definitief te willen versturen. | 5. Systeem verstuurd bericht. |

## UC6: Bewegen

|  |  |
| --- | --- |
| **Primary actor:** <Speler> | |
| **Stakeholders:** | |
| **Brief description:**  De <Speler> kiest een vakje uit om zich naar toe te bewegen. Op het uitgekozen vakje kan een object liggen die de <Speler> automatisch op pakt wanneer hij op dat vakje terecht komt. Het kost 2 tot 6 energie(afhankelijk van de situatie) om zich te verplaatsen. Voor een aangrenzende vakje 2 energie, een gat over springen kost 4 energieën een transitie van kamers kost 6 energie. | |
| **Preconditions:**  <Speler> heeft meer dan 10 energie | |
| **Postconditions (Success Guarantee):**  <Speler> staat op een ander vakje | |
| **Main Success Scenario (Basic Flow):** | |
| **Actor Action** | **System Responsibility** |
| 1 <Speler> kiest om te bewegen. | 2 Systeem vraagt naar welk vakje de <Speler> wilt. |
| 3 <Speler> kiest een vakje. | 4 Systeem vraag <Speler> welke snelheid hij zich wil verplaatsen |
| 5 <speler> geeft snelheid aan | 6 Systeem verlaagt <Speler> zijn energie op basis van de snelheid. |
|  | 7. Systeem verplaatst <Speler> naar gekozen vakje. |
|  | 8 Systeem geeft object aan <Speler>. |
| **Extensions (Alternative Flow):** | |
|  | 2a1 [<Speler> geeft aan met de bezem te willen bewegen]  2a2 Systeem hervat bij stap 7. |
|  | 4a [<Speler> kiest vakje buiten bereik] >> [Use Case eindigt] |
|  | 4b [Tussen gekozen vakje en <Speler> is een gat] >> [Systeem verlaagt energie van <Speler> met 4] |
|  | 4c [Gekozen vakje is een deur] >> [Systeem verlaagt energie van <Speler> met 6] |

## UC7: Aanmelden

|  |  |
| --- | --- |
| **Primary actor:** <Speler> | |
| **Stakeholders:** <Medespelers> | |
| **Brief description:**  De <Speler> meldt zich aan bij een spel. Waarna het spel kan starten. | |
| **Preconditions:** Server is opgestart en er is nog minimaal één plaats beschikbaar. | |
| **Postconditions (Success Guarantee):**  <Speler> heeft zich aangemeld voor een spel en de teams zijn gevormd | |
| **Main Success Scenario (Basic Flow):** | |
| **Actor Action** | **System Responsibility** |
| 1. <Speler> kiest voor aanmelden | 2.Systeem meldt <Speler> aan bij het spel. |
|  | 3.Systeem geeft aan genoeg <Medespelers> te hebben in de teams. |
|  | 4.Systeem formeert teams. |
|  | 5.Systeem start het spel |

# Domain model

# Activity Diagram

# System Diagrams & System Sequence Diagrams

## SD Aanmelden

## SD Aanvallen

## SD Laten vallen <Attribuut>

## SD Bewegen

## SD Chatten

## SD Kussen

## SSD Aanmelden

## SSD Aanvallen

## SSD Laten vallen <Attribuut>

## SSD Bewegen

## SSD Chatten

## SSD Kussen

## 8.13 SSD Verdedigen

# State Machine Diagram

# Design Class Diagram

# OO-ontwerpprincipes

# Architectuurmodel