|  |
| --- |
| Haagse Hogeschool |
| Informatica 2 - Elaboration rapport |
| Docent: Tim Cocx |

|  |
| --- |
|  |

# Inhoudsopgave

Inhoud

[Inhoudsopgave 1](#_Toc346658010)

[1. Inleiding 1](#_Toc346658011)

[2. Refined requirements 1](#_Toc346658012)

[1. Functional requirements 2](#_Toc346658013)

[2. Non-functional requirements: 3](#_Toc346658014)

[3. Use Case Model 4](#_Toc346658015)

[1. Use cases 4](#_Toc346658016)

[2. Use case descriptions 5](#_Toc346658017)

[3. Analysis class diagram 9](#_Toc346658018)

[4. Sequence diagram 9](#_Toc346658019)

[4. Architecture Proof of Concept 11](#_Toc346658020)

[5. Logical and fysical testcases 12](#_Toc346658021)

[6. Pacman manual 13](#_Toc346658022)

[7. Attachments 13](#_Toc346658023)

# Inleiding

Het Elaboration rapport bevat het use case model van het project. In de elaboratie fase was het de bedoeling om het probleem van het project te analyseren en een basis voor de applicatie te maken. Om het probleem te analyseren is er een use case model opgesteld, dat een ‘verhaal’ verteld van hoe het systeem zal gaan werken. Het use case model is een belangrijk onderdeel van het project, omdat het verduidelijkt hoe de applicatie moet worden gebouwd. Het architectuur proof of concept is bedoelt om een idee te geven, van hoe de applicatie gaat werken en eruit gaat zien.

# Refined requirements

In de elaboratiefase kunnen de requirements nog worden aangepast, omdat bij het uitbreiden van het use case model en nog meer requirements kunnen worden gevonden.

## Functional requirements

Hieronder is een lijst weergegeven met de requirements waaraan de applicatie ‘Happer’ moet voldoen.

1. Er moet een score aanwezig zijn die de speler ten alle tijden kan zien.
2. De score wordt bijgehouden op basis van het aantal stappen dat het karakter heeft gedaan(de score wordt verhoogt per stap).
3. Door middel van de pijltjestoetsen moet de speler kunnen bewegen.
4. Er moet een speelveld zijn, groot genoeg om het speelbaar te houden en dat is afhankelijk van de andere spelverhoudingen.
5. Het speelveld moet Positions hebben, waarop de spelobjecten kunnen worden geplaatst.
6. Het speelveld word gemaakt op basis van een String die wordt ingeladen. Afhankelijk van de String worden er op de Positions spelobjecten geplaatst.
7. Het spel wordt gestart wanner de speler de applicatie opstart, een level opnieuw begint, het spel herstart.
8. Zodra het spel wordt gestart, moet er een speelveld worden aangemaakt, dat direct word gevuld met spelobjecten.
9. Zodra het spel wordt gestart, word de score op nul gezet.
10. Zodra het spel wordt gestart, begint het monster de bewegen.
11. Er moet een object “Wall” zijn, waar de speler niet doorheen kan bewegen .
12. Er moet een object “Player” zijn, die door de speler word bestuurt.
13. Alle objecten moeten dezelfde grootte hebben, ze nemen dus allemaal een gelijke hoeveelheid Positions in van het speelveld.
14. Er moet een object “Monster” zijn.
15. Het moet altijd mogelijk zijn het Monster in te sluiten, zodat de speler het spel kan winnen.
16. Het spel moet een melding “Game Won” , wanneer het Monster is ingesloten en geen beweging meer kan maken. Het level dat wordt gegenereerd is het volgende level of hetzelfde, afhankelijk van de speler keuze.
17. Het Monster moet om de seconde een beweging maken, dichter naar de speler toe.
18. Wanneer de Monster de speler niet kan bereiken, beweegt hij in willekeurige directies.
19. Het Monster kan niet door kisten of muren bewegen.
20. Er moet een melding worden gegeven “Game Over” wanneer de speler is gevangen(aangeraakt) door de Monster; daarna wordt het level opnieuw gegenereerd of gestopt, afhankelijk van de keuze van de speler.
21. Er moet een object “Box” zijn.
22. Kisten kunnen verschuiven door er tegen aan te lopen.
23. Kisten kunnen niet worden verschoven als er een muur achter staat.
24. Een Childmode, waarbij het mogelijk word om meerdere kisten tegelijk te verplaatsen(het word dus mogelijk een rij kisten achter elkaar te verplaatsen).
25. Kisten kunnen niet worden verschoven wanneer er een of meerdere kisten achter staan, tenzij Childmode aanstaat.
26. Moeilijkheidsgraden per level. Iedere keer als de speler een level heeft behaald, moet er een nieuw(moeilijker) level worden gegenereerd.
27. Het spel moet een GUI hebben, zodat de speler makkelijk kan navigeren naar de functionaliteiten, zoals beschreven in de use cases.
28. Er moet een Menu in de GUI zijn, zodat de speler het spel kan beheren.

(Deze punten voor in het menu)

1. Het spel moet direct beginnen, nadat de speler de applicatie opstart.
2. Er moet een herstartSpel knop zijn, waarbij de moeilijkheidsgraad terug gaat naar het laagste niveau en er op dat niveau weer doolhoven worden gegenereerd, wanneer de speler erop drukt en bevestiging geeft, dat hij/zij het spel wilt resetten.
3. Er moet een RestartLevel knop zijn, waarbij de betreffende doolhof worden gereset, wanneer de speler erop drukt en bevestiging geeft, dat hij/zij de doolhof wilt resetten.
4. Er moet een enableChild knop zijn, waarbij de kindmodus wordt aangezet en het mogelijk word om een rij kisten tegelijk te verplaatsen, wanneer de speler erop drukt en bevestiging geeft, dat hij/zij de Childmode wilt aanzetten.
5. Er moet een QuitGame knop zijn, wat de applicatie afsluit wanneer de speler erop drukt en bevestiging geeft, dat hij/zij het spel wilt afsluiten.
6. Zodra de speler op een van de vorige menuknoppen klikt, moet het spel gepauseerd worden terwijl er om een bevestiging word gevraagd. Dit houd in dat de monsters op dat moment niet bewegen.
7. Een handleiding moet met het spel meekomen, zodat de gebruiken deze kan doorlezen en weet hoe het spel werkt.
8. In de GUI moet een GameWindow zijn, waarin de spelactiviteiten zichtbaar zijn voor de speler.

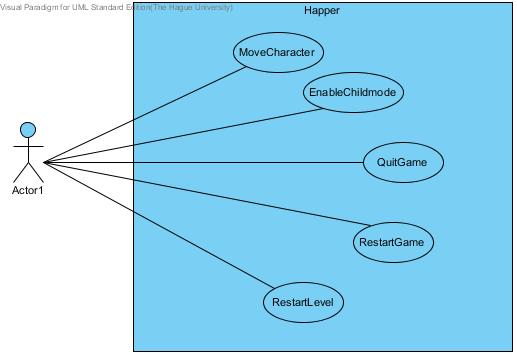
## Non-functional requirements:

1. Het spelinterface(GUI) moet er mooi en strak uitzien.
2. Het spel moet snel reageren op een actie van de speler (gewenst < 75 ms).
3. Er moet een teller zijn voor het level waarin de speler zich bevind, die zichtbaar moet zijn voor de speler.
4. Monsters schieten LaserBeams af. Als de speler wordt geraakt door een beam begint het level opnieuw na de melding “Game Over”.
5. Eventuele uitbreidingen op het spel
6. De structuur van de levels moet kunnen worden opgehaald uit XML files die worden verwerkt in de applicatie.
7. De .XML files moeten los staan van het project.
8. Het spel bevat geavanceerde graphics en ook animaties.
9. Het spel bevat audio.
10. De speler moet de levels uit kunnen kiezen die hij heeft gehaald, om te spelen.

# Use Case Model

Het use case model bevat de analyse voor het project. Het bevat de (aangepaste) use cases, de use case beschrijvingen, analysis klassendiagram en sequence diagram, die hieronder zijn weergeven.

## Use cases



## Use case descriptions

|  |  |
| --- | --- |
| Naam | MoveCharacter |
| Id | K.1 |
| Samenvatting | Het bewegen van het karakter. |
| Primairy actors | Speler |
| Secondairy actors | Geen. |
| Precondities | Er is een level gestart. |
| Main Flow | 1. De actor drukt op een pijltoets. 2. Het systeem verplaatst het karakter in de richting van de pijltoets. **[A] [B] [D] [G][I]** |
| Postcondities | Karakter heeft zich verplaatst. |
| Alternative Flows | **[A]**   1. Er bevindt zich een muur op de plaatst waar het karakter commando krijgt naartoe te verplaatsen. 2. Systeem verplaatst het karakter niet.   **[B]**   1. Er bevind zich een kist, op de plaats waar het karakter commando krijgt naartoe te verplaatsen. **[C][D][E]** 2. Systeem verplaatst het karakter en de kist in de richting van de pijltjestoets.   **[C]**   1. Er staan nog een of meerdere kisten achter de kist waar het karakter naartoe moet worden verplaatst.**[D]** 2. Het systeem beweegt het karakter niet.   **[D]**   1. De childmode staat aan. 2. Het systeem verplaatst de kisten en het karakter in de richting van de pijltoets. **[F]**   **[E]**   1. Voor de kist staat een muur of monster. 2. De kist en het karakter worden niet verplaatst.   **[F]**   1. Voor de rij met kisten staat een muur of een monster. 2. Het systeem verplaatst het karakter en de kisten niet.   **[G]**   1. Er bevindt zich een monster op de plaatst waar het karakter commando krijgt naartoe te verplaatsen. 2. Het systeem beweegt het karakter niet.   **[I]**  1. Er is geen positie meer in de richting waar het karakter commando krijgt naartoe te verplaatsen.  2. Het systeem verplaatst het karakter niet. |

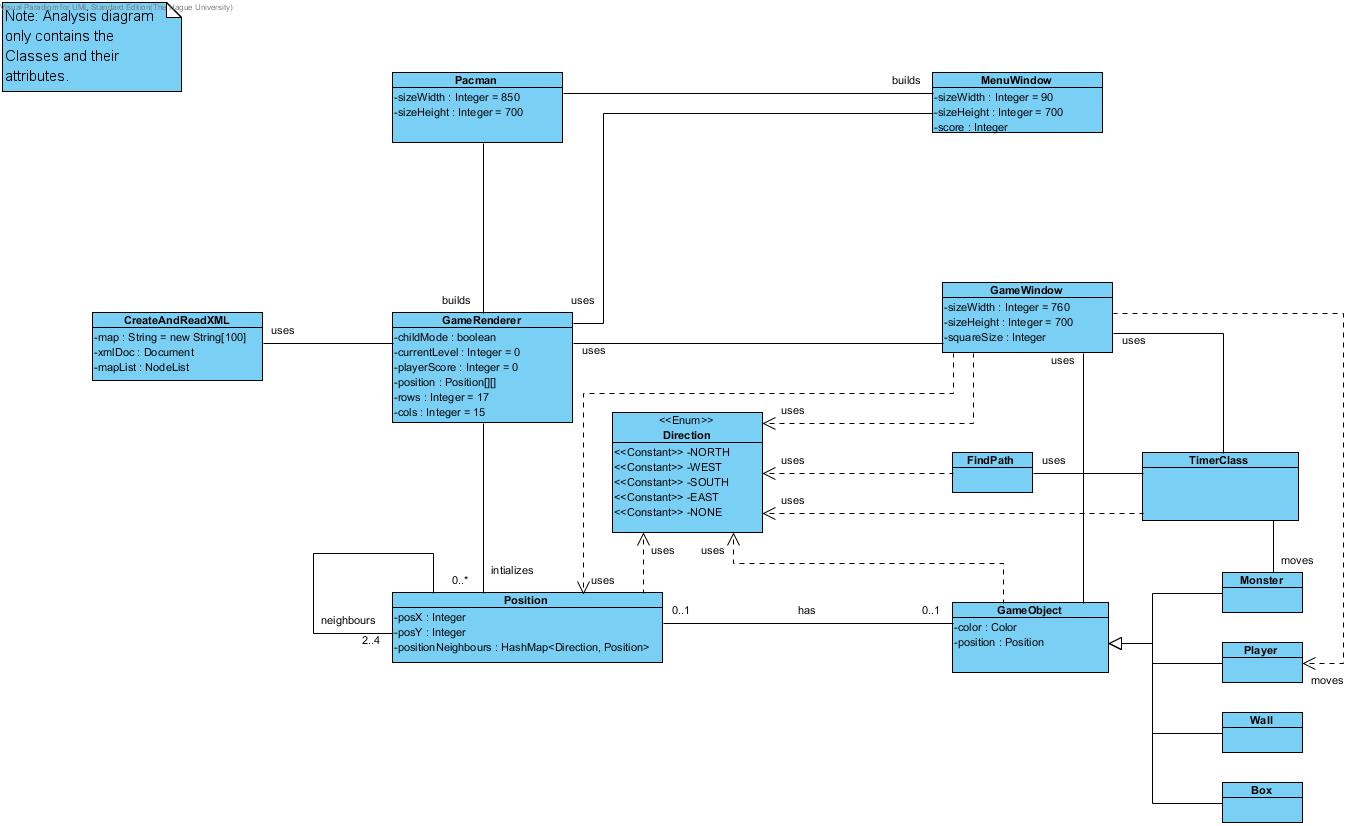
|  |  |
| --- | --- |
| Naam | RestartGame |
| Id | G.1 |
| Samenvatting | Actor herstart het spel. |
| Primairy actors | Speler |
| Secondairy actors | Geen. |
| Precondities | Het spel is gestart. |
| Main Flow | 1. De actor kiest optie HerstartSpel. 2. Het systeem vraagt om een bevestiging van de actor. 3. De actor kiest voor ja. **[A]** 4. Het systeem zet het levelniveau op 0. 5. Het systeem verwijdert alle objecten en initialiseert nieuwe spelobjecten. 6. Het systeem zet de speler score op 0. 7. Het system hertekent het level. 8. Actor kan beginnen met spelen. |
| Postcondities | Spel is herstart. |
| Alternative Flows | **[A]**   1. De actor kiest voor nee. 2. De actor gaat verder met spelen. |

|  |  |
| --- | --- |
| Naam | QuitGame |
| Id | G.2 |
| Samenvatting | Actor stopt het spel. |
| Primairy actors | Speler |
| Secondairy actors | Geen. |
| Precondities | Er is een level gestart. |
| Main Flow | 1. De actor kiest optie QuitGame. 2. Het systeem vraagt om bevestiging. 3. De actor kiest voor ja. **[A]** 4. Het systeem sluit zichzelf af. |
| Postcondities | Systeem bevindt zich in het hoofdmenu. |
| Alternative Flows | **[A]**   1. De actor gaat verder met spelen. |

|  |  |
| --- | --- |
| Naam: | RestartLevel |
| ID: | G.3 |
| Samenvatting: | Het opnieuw starten van een level. |
| Primary actors: | Speler |
| Secondary actors: | Geen |
| Precondities: | Er is een spel actief. |
| Main Flow: | 1. De actor drukt op knop Menu. 2. De actor kiest optie RestartLevel. 3. Het systeem vraagt om bevestiging. 4. De actor kiest voor ja **[A]** 5. Het systeem verwijdert alle spelobjecten en initialiseren nieuwe spelobjecten. 6. Het systeem zet de spelerscore op 0. 7. De actor kan beginnen met spelen in het actuele level. |
| Post condities: | Level is herstart. |
| Alternative Flows: | **[A]**   1. De actor kiest voor nee 2. Systeem hervat het actuele level. |

|  |  |
| --- | --- |
| Naam: | EnableChildmode |
| ID: | K.2 |
| Samenvatting: | Het aanzetten van de Childmode |
| Primary actors: | Speler |
| Secondary actors: | Geen |
| Precondities: | Geen. |
| Main Flow: | 1. De actor kiest optie EnableChildmode. 2. Het systeem zet de Childmode aan.**[B]** 3. Het systeem geeft een melding dat de Childmode is aangezet. |
| Post condities: | Childmode is aangezet. |
| Alternative Flows: | **[B]**   1. De Childmode staat al aan. 2. Het systeem zet de Childmode uit. 3. Het systeem geeft een melding dat de Childmode is aangezet. |

## Analysis class diagram

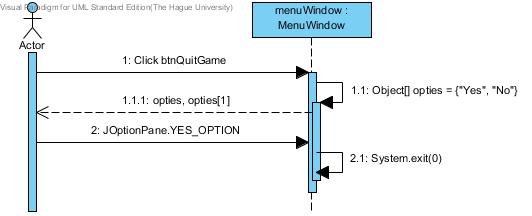


## Sequence diagram

Het sequence diagram is gebaseerd op een aantal sequenties van acties die hieronder zijn beschreven. Ik heb de operaties uit het analysis klassendiagram overgenomen en gebruikt om de sequentie van acties op te stellen. Het moet uiteindelijk een verhaal vertellen over hoe het systeem werkt.

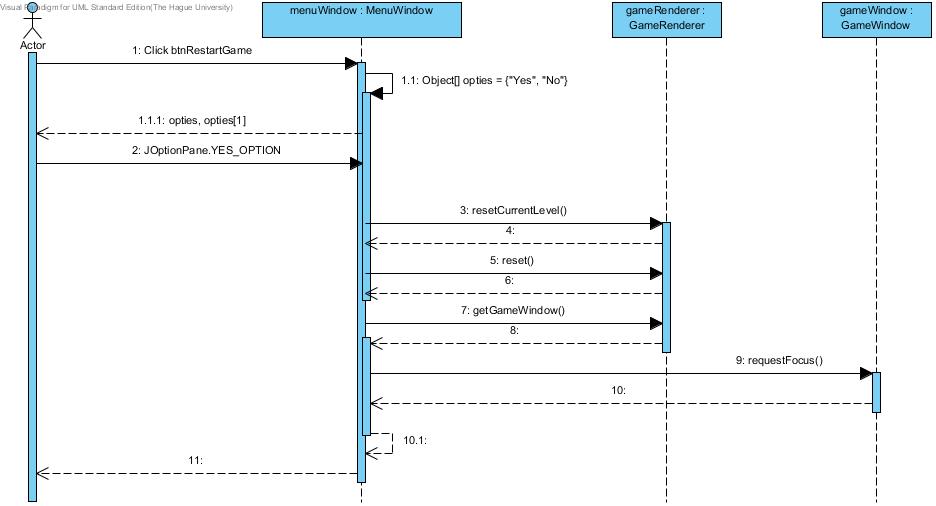
1. **Use case QuitGame**

De speler drukt op de knop btnQuitGame. Het systeem geeft een optiepanel weer, waar de speler ‘ja’ of ‘nee’ kan kiezen. De speler kiest optie ‘ja’. Het systeem sluit zichzelf af.



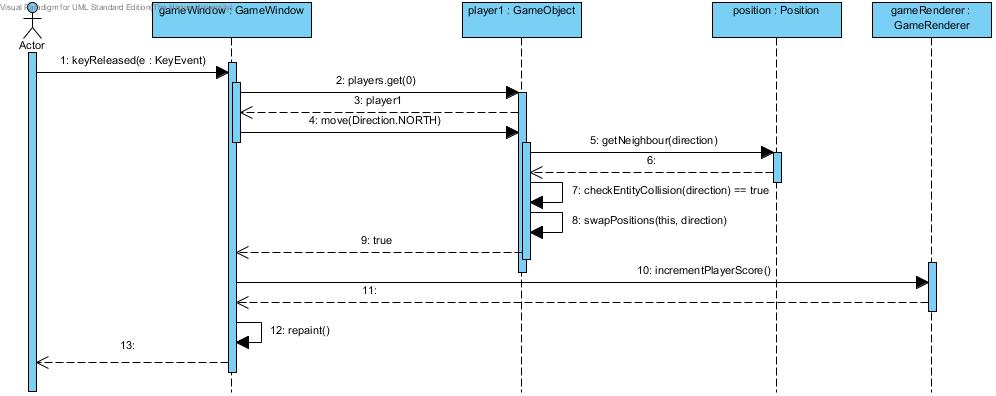
1. **Use case RestartGame**

De speler drukt op de knop btnRestartGame. Het systeem geeft een optiepanel weer, waar de speler ‘ja’ of ‘nee’ kan kiezen. De speler kiest optie ‘ja’. Het systeem zet het levelniveau attribuut op 0, vervolgens worden alle aanwezige objecten verwijdert. Het systeem initialiseert daarna nieuwe objecten. De speler kan verder gaan met het spel.



3. **Use case moveCharacter**

Scenario 1: In dit scenario drukt de speler op de pijltjestoets naar boven. Boven het karakter staan geen spelobjecten, maar is wel een positie. Het systeem checkt of er een object op de positie staat, waar de speler wenst naartoe te bewegen. Er staat geen objecten en het systeem beweegt aldus het karakter naar boven. Als laatste telt het systeem een punt op bij de spelerscore en weergeeft dit in het score veld.



# Architecture Proof of Concept

Het architectuur proof of concept is een basis van de komende gehele applicatie, wat een beeld moet geven van hoe het ongeveer moet gaan werken. Door te kijken naar het architectuur proof of concept kan er worden besloten of het project doorgaat of niet.

# Logical and fysical testcases

De volgende testgevallen zijn ontworpen voor het prototype, en voor de use case moveCharacter. Het is een beslissingstabellentest met condition coverage, wat betekent dat alle mogelijke condities worden getest.

# Pacman manual

De handleiding is de spelbeschrijving, die het voor de gebruiken duidelijk maakt hoe het spel moet worden gehanteerd. De handleiding zit in de bijlagen onderaan het rapport.

# Attachments

Met het elaborationrapport komen ook nog verschillende artefacten die behoren tot de elaboratiefase.  
De volgende bijlagen zijn te vinden in de folder:  
  
- .PNG van de Sequence diagrammen van de use cases QuitGame en RestartGame.  
- .PNG van het Analysis klassendiagram  
- Prototype van het spel Pacman.   
- Handleiding van het spel.