Technisch verslag

**Git Gud**

Jeroen van Hattem, 1675180

René de Kluis, 1661627

Rik Ruttenberg, 1677636

Documentdatum: 02/02/2017

Versie: 0.0.1

Inhoudsopgave

[Inleiding 4](#_Toc473891026)

[High Concept Document 5](#_Toc473891027)

[Technisch ontwerp 6](#_Toc473891028)

[Beschrijving van de architectuur 6](#_Toc473891029)

[Spel spelen 6](#_Toc473891030)

[Level maken 6](#_Toc473891031)

[Klassendiagram 6](#_Toc473891032)

[Leveleditor 6](#_Toc473891033)

[Dialogbox 6](#_Toc473891034)

[Figure 6](#_Toc473891035)

[Music 7](#_Toc473891036)

[Inventory 7](#_Toc473891037)

[Item 7](#_Toc473891038)

[Main\_menu 7](#_Toc473891039)

[Game 7](#_Toc473891040)

[SQL 7](#_Toc473891041)

[Player 7](#_Toc473891042)

[NPC 7](#_Toc473891043)

[Ability 7](#_Toc473891044)

[Realisatie 8](#_Toc473891045)

[Softwareontwikkeling 8](#_Toc473891046)

[Platformkeuze 8](#_Toc473891047)

[Installatie van de software 8](#_Toc473891048)

[Edit-, compile- en runcyclus 8](#_Toc473891049)

[Ontwikkelmethodiek 8](#_Toc473891050)

[Problemen tijdens realisatie 8](#_Toc473891051)

[Verhelpen van de problemen 8](#_Toc473891052)

[Algoritmen en datastructuren 8](#_Toc473891053)

[Experimenten en resultaten hiervan 9](#_Toc473891054)

[Evaluatie 10](#_Toc473891055)

[Wat beter kon 10](#_Toc473891056)

[Hoe dit beter kon 10](#_Toc473891057)

[Conclusies en aanbevelingen 11](#_Toc473891058)

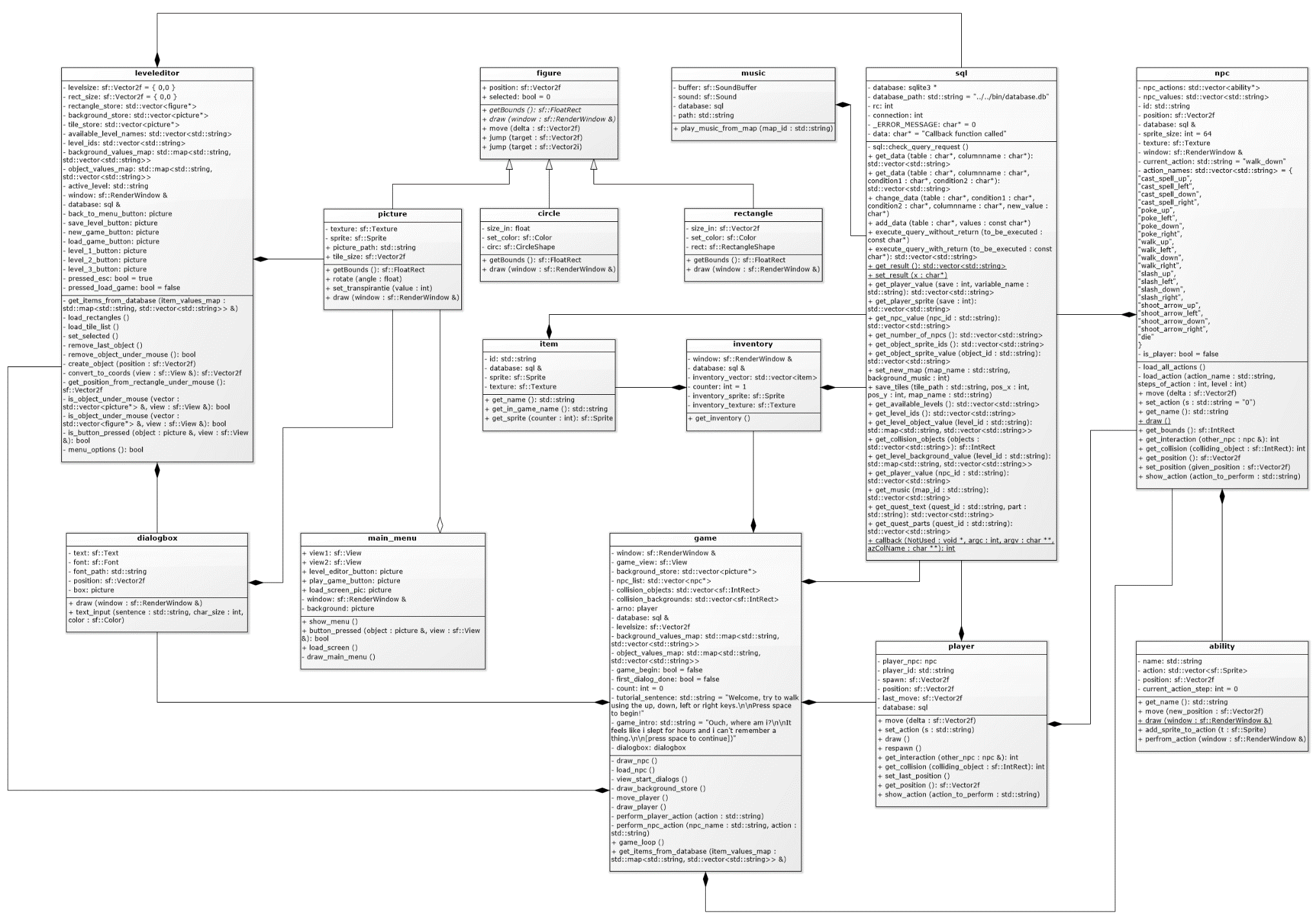
[Conclusies en aanbevelingen op functioneel vlak 11](#_Toc473891059)

[Conclusies en aanbevelingen op technisch vlak 11](#_Toc473891060)

[Bronvermelding 12](#_Toc473891061)

[Bijlage 1: High Concept Document 13](#_Toc473891062)

[Bijlage 2: Klassendiagram 15](#_Toc473891063)

[ 15](#_Toc473891064)

# Inleiding

Dit technische verslag betreft het project “Themaopdracht Devices”. In dit project wordt van ons als leerlingen verwacht in groepsverband een game te maken. Dit verslag is bestemd voor mensen die meer willen over de ontwikkeling van de game en hoe deze in elkaar zit.

Om dit verslag te openen beginnen we met het bespreken van het High Concept Document. We zullen hier door de verhaallijn van het verhaal gaan en de personages. Ook zullen we andere aspecten van de game bespreken.

Hierna gaan we kijken naar het technisch ontwerp van het spel, hiermee bekijken we de architectuur van de game en het klassendiagram.

Vervolgens gaan we kijken naar de realisatie van het spel. Hieronder verstaat men hoe de softwareontwikkeling in zijn werk gaat, de keuze van het platform, de installatie van de software, het beschrijven van de edit-, compile- en runcyclus en het beschrijven van de ontwikkelmethodiek. Ook praten wij hier over de grote problemen die zijn opgetreden tijdens re realisatie, hoe we deze hebben opgelost en een beschrijving van complexe algoritmen en datastructuren. Tot slot zullen we onder dit kopje de tests uitleggen en hiervan de resultaten.

Wanneer we klaar zijn met het verdiepen in de realisatie gaan we kijken naar de evaluatie, wat voor problemen zijn wij tegengekomen tijdens het project en hadden wij deze problemen beter kunnen oplossen? Dat gaan we hier bespreken.

Daarnaast gaan we ook conclusies trekken en aanbevelingen geven met betrekking op het project, op zowel functioneel vlak als technisch vlak. Hier zullen we zeggen wat men beter wel en niet kan doen in een project.

Tot slot hebben we ook een bronvermelding waar we naslagwerken zullen lijsten die wij geraadpleegd hebben in loop van het project.

# High Concept Document

Het High Concept Document is te vinden op te laatste pagina, of door een klik **hier**. In het High Concept Document is ieder detail over onze game te vinden. We zullen er hier stapsgewijs doorheen gaan.

Ten eerste hebben we de features. Wij wilden een spel hebben waarin de speler vrij is te bepalen wat deze wilt doen, al is het vrij rondlopen, ergens een minigame spelen of strijden met monsters. Wij hebben dus vanzelfsprekend alles gedaan wat wij konden om dit te realiseren.

Waarom zou iemand juist dit spel willen spelen? Veel personen in onze doelgroep hebben zich weleens in een vergelijkbare situatie kunnen vinden, kortom helemaal van de kaart zijn na een mooie avond, net als ons hoofdpersoon. Aan deze situatie hebben wij een komische draai gegeven. Om het af te maken wilden wij ook retro minigames in het spel gebruiken, zodat spelers zich kunnen binden aan de nostalgie terug in hun jeugd door het spelen van Pacman en Snake.

Als genre hebben wij gekozen voor een kruising tussen avontuur en retro. Dit omdat het een waar avontuur is om als iemand die van de kaart is je weg terug naar huis te vinden. Het retro deel komt door het gevoel waarmee je het spel ervaart. Het trekt je terug naar je jeugd door het 8bit uiterlijk van de game en de klassieke spellen die je erin kan spelen.

Onze doelgroep is studenten. Dit omdat studenten zich het meest kunnen inleven in de rol van de hoofdpersoon, veel studenten zijn de hoofdpersoon zelf ook weleens geweest.

Hierna komen de mechanics aan bod. De speler kan NPC’s aanspreken om tips of items te krijgen die nodig zijn het spel verder te kunnen spelen, vaak zal voor deze tips of items een handeling moeten worden verricht, zoals het zoeken van een ander item.

Vervolgens komen we bij competitie/coöperatie. Het spel bevat competitie, omdat er veel uitdaging is, zo moet de speler op sommige stukken hard nadenken om bijvoorbeeld minigames te winnen of een puzzel op te lossen. Er is sprake van coöperatie omdat NPC’s de speler kunnen helpen.

Ons doelplatform is de PC. Dit vanwege de mankracht en tijd die wij hebben, hadden we de tijd was er wellicht een mogelijkheid uit de breiden naar andere markten, maar deze is er simpelweg niet.

We hebben een baklijst met ontwerpdoelen, ik zal de voornaamste noemen, de rest is te vinden in de bijlage. Wij willen dat het spel niet crasht bij kleine foutjes, zoals het fout lopen van de speler. Het spel moet eenvoudig te bedienen zijn, we willen dat de speler een plezierige en geen verwarrende ervaring op doet tijdens het spelen van dit spel. Ook moet de speler rustig kunnen rondlopen in de spelwereld, dit moet ontspannen kunnen.

Dusver hebben wij 3 karakters verzonnen. “De eerste is Commander El Theodoro”, hij is een pratende teddybeer die de speler bij zich heeft, hij is de “sidekick” van de speler. Vervolgens hebben we “Barkley the Great” Hij is de koning van het rijk. Ten slot hebben we Mache Pwason, zij is de vrouw die de speler op weg helpt.

# Technisch ontwerp

## Beschrijving van de architectuur

De architectuur van ons spel bestaat uit meerdere aspecten. Wanneer het spel wordt opgestart krijgt men een laadscherm te zien, in de achtergrond worden hier alle gegevens ingeladen zodat dit niet tijdens het spelen hoeft te gebeuren. Wanneer het spel geladen is komt er een keuzemenu tevoorschijn, hier wordt de gebruiker de keuze voorgelegd om het spel te gaan spelen of om een nieuw level te maken.

### Spel spelen

Wanneer de gebruiker het spel gaat spelen krijgt deze een wereld te zien van 64 tegels hoog en 64 tegels breed. Eén tegel heeft een grootte van 32 bij 32 pixels, dit maakt de wereld in z’n geheel 2048 pixels hoog bij 2048 pixels breed. De speler kan door deze wereld heen lopen, hier kan deze met NPC’s handelingen verrichten, zoals items ontvangen of praten.

### Level maken

Wanneer de gebruiker besluit een level te maken, krijgt deze een raster van 64 bij 64 lege tegels voorgeschoteld. Vervolgens kan de gebruiker aan de rechterkant van het scherm tegels en objecten selecteren en deze in de wereld te plaatsen. Als de gebruiker tevreden is met het level dat deze gemaakt heeft kan het level worden opgeslagen. Als de gebruiker later weer verder wilt met het maken van een level kan deze een level inladen.

## Klassendiagram

In de bijlage is ons klassendiagram te vinden. Wij hebben besloten om onze klassen zo ver mogelijk op te delen. Dit zodat het makkelijker is bepaalde functies te vinden tijdens programmeren. Ook is het op deze manier eenvoudiger te bedenken welke klasse wat moet includen. Daarnaast zijn kleine klassen beter voor het geheugen, omdat een gehele library wordt opgenomen in het .obj bestand, en deze zichtbaar kleiner zijn op deze manier.

De klassen zal ik hier uitleggen.

### Leveleditor

De leveleditor klasse bevat alles wat nodig is voor de leveleditor, denk hierbij aan het opslaan van het level, het inladen van het level, tegels plaatsen in het level.

### Dialogbox

De dialogbox klasse is voor het weergeven van tekst op het scherm, hier kunnen conversaties met NPC of tips op worden weergeven.

### Figure

#### Picture

De picture klasse is een afbeelding die kan worden weergeven op het scherm. Zo kan je met iedere sprite eenvoudig werken.

#### Circle

Met de circle klasse kan je eenvoudig een cirkel maken en deze op het scherm weergeven.

#### Rectangle

Met de rectangle klasse kan je eenvoudig een vierkant maken en deze op het scherm weergeven.

### Music

Met de music klasse kunnen we eenvoudig muziek inladen, afspelen en stoppen.

### Inventory

De inventory klasse houd alle items bij, laadt deze uit de database en weergeeft ze op het scherm wanneer de inventory wordt opgevraagd door de gebruiker.

### Item

De item klasse vereenvoudigt het gebruik van de inventory door van alle items in de database gelijkgestemde objecten te maken. Met de item klasse kan eenvoudig de eigenschappen worden opgehaald.

### Main\_menu

De klasse main\_menu is de klasse waarmee het startmenu wordt weergeven. Hier kan het spel laden of de leveleditor worden aanroepen.

### Game

De game klasse bevat alle elementen op het spel te laten draaien. Deze klasse laadt alle NPC’s, objecten, tegels enzovoorts uit de database en slaat deze lokaal op zodat deze sneller ingeladen kunnen worden. Deze klasse zorgt er ook voor de speler wordt samengevoegd in de wereld met de NPC’s en overige objecten.

### SQL

In de SQL klasse wordt een connectie met de database tot stand gebracht. Ook staan hier alle functies die nodig zijn om data te halen uit de database en deze om te zetten in bruikbare variabelen.

### Player

In de player klasse staan alles functies die nodig zijn om te speler te beheersen. Zo staan hier functies in om de speler te verplaatsen, te weergeven op het scherm en om botsingen tussen NPC’s en objecten te detecteren.

### NPC

In de NPC klasse staan alle benodigde functies voor de NPC. Deze moet kunnen bewegen, verschillende acties kunnen doen en een botsing kunnen detecteren.

### Ability

In de ability klasse staan de functies op de speler verschillende handelingen te laten verrichten. Dit heeft betrekking op een andere sprite laden, waardoor het lijkt dat de speler naar boven loopt of met een wapen slaat.

# Realisatie

## Softwareontwikkeling

Van te voren zijn wij gaan denken, wat hebben we nodig en hoe gaan we dit realiseren? Wanneer we dit hadden, hebben we dit op het bord geschreven. Je had dan bijvoorbeeld een klasse “player”, deze klasse zou het personage zijn dat de speler kan besturen. Dit personage moet kunnen lopen, ook zou hij niet door een huis door mogen lopen. Dit hebben wij voor het hele spel gedaan en uiteindelijk hebben wij zo een grote lijst gekregen, hierdoor hebben wij een beter beeld van wat er moet gebeuren.

## Platformkeuze

Wij hebben ons gericht op de PC, draaiend op Windows. Hoewel de code ook te compilen is voor Linux en Macintosh. Wij hebben dit gedaan, omdat hier de grootste doelgroep zich bevindt.

## Installatie van de software

De gebruiker zal in een .exe bestand nodig hebben wat het spel zal opstarten, ook heeft de gebruiker meerdere libraries nodig en een database. Wanneer deze goed zijn gekopieerd, kan de gebruiker het spel spelen.

## Edit-, compile- en runcyclus

Wanneer wij aan het programmeren zijn, her-compilen we vaak na een kleine verandering, dit om te testen wat het verschil is na een kleine verandering. Soms kan een kleine verandering een groot effect hebben. Wanneer je meerdere dingen verandert en daarna pas compiled, is het moeilijk te zien waar een error vandaan komt.

## Ontwikkelmethodiek

Wij hebben gebruik gemaakt van scrum. Scrum valt onder de “agile” ontwikkelingsmethode. Bij scrum maakt men gebruik van sprints, wij hebben 3 sprints gehad van ieder 3-5 dagen. Dit was variabel. Tijdens iedere sprint hebben wij doelen gesteld die wij wilden halen. Deze doelen zijn terug te vinden op ons Trello-bord[[1]](#footnote-1).

## Problemen tijdens realisatie

Tijdens de realisatie van het spel zijn wij vaak op problemen gestuit, vaak was dit omdat wij te makkelijk dachten over een taak. Andere problemen waren dat we soms dubbel werk deden, dit was voornamelijk wanneer iemand vergat het Trello-bord te updaten.

## Verhelpen van de problemen

Vaak werden onze problemen opgelost door met elkaar te gaan zitten en hierover na te denken, want samen sta je sterker. Vaak hebben wij het whiteboard gebruikt in een lokaal en hebben wij hierop onze gedachten uitgebeeld.

## Algoritmen en datastructuren

Wij hebben niet veel complexe algoritmen en datastructuren gebruikt. Wel hebben wij het “factory pattern” meerdere malen gebruikt.

## Experimenten en resultaten hiervan

Dit is terug te vinden in het bestand “testverslagen”

# Evaluatie

## Wat beter kon

1. We hadden een betere voorbereiding moeten treffen. Toen we de eerste sprint begonnen wisten we eigenlijk nog niet precies hoe we alles wilden gaan aanpakken.
2. Van sommige dingen hadden we nog niet genoeg kennis, waardoor we deze kennis moesten op doen tijdens het project.

## Hoe dit beter kon

1. De volgende keer voor de eerste sprint niet alleen goed nadenken over wat we willen, maar ook hoe we dit willen realiseren. Of we wel of niet een database willen gebruiken e.d.
2. Alvorens de eerste sprint te beginnen, zorgen dat je genoeg kennis hebt voor het maken van de taken die jou zijn toegewezen. Als je in een bepaald onderdeel beter bent dan de andere groepsleden, is het beter deze taak op je te nemen.

# Conclusies en aanbevelingen

## Conclusies en aanbevelingen op functioneel vlak

Wij raden het aan om alvorens het project te beginnen een goed beeld te hebben van wat er gedaan moet worden.

Zorg er ook voor dat iedereen een duidelijke taak heeft met wat hij/zij moet uitvoeren. Anders kan het voorkomen dat sommige teamleden dubbel werk doen

## Conclusies en aanbevelingen op technisch vlak

Alvorens te beginnen met het project is het noodzakelijk dat ieder teamlid genoeg kennis heeft van de te gebruiken libraries en programmeertaal, dat hij/zij geen extra lessen nodig zal hebben om op weg te komen met het programmeren. Wanneer dit niet het geval is zal dit veel tijd kosten.

# Bronvermelding

* Scrum. (2017). Wikipedia. Geraadpleegd op 2 januari 2017, van  
  <https://nl.wikipedia.org/wiki/Scrum_(softwareontwikkelmethode)>
* Stack Overflow. Meerdere pagina’s geraadpleegd, niet bijgehouden welke, van

<http://stackoverflow.com>

# Bijlage 1: High Concept Document

Inleiding

Een top-down RPG waar een gast met een harde kater en geheugenverlies wakker wordt in een andere wereld. Hij moet uitzoeken waar hij is en proberen terug te komen naar zijn eigen dementie.

Features

* Fetch quest RPG met minigames
* Random interacties met NPC’s
* Verzamelen van Items en gear Items
* Mappen gebonden aan speler level
* Stad 2 mag niet betreden worden voordat de speler level 4 is.
* Vrij rond kunnen lopen in de spelwereld

Speler motivatie

* Het spel bevat een komisch verhaal met onverwachte wendingen, wat maakt dat de speler door wilt spelen.
* Ook zijn aspecten van de verhaallijn gebaseerd op gebeurtenissen waarmee de meeste studenten weleens in aanraking zijn gekomen en daardoor zich ook kan inleven in het verhaal.
* Daarnaast heeft het spel retro aspecten, zoals retro mini games en oude stijl RPG, wat de spelers het gevoel van oudere games teruggeeft en daardoor geïnteresseerd zullen zijn.

Genre

* Avontuur
* Retro-gaming

Doelgroep

* Studenten

Mechanics

* Speler kan NPC’s aanspreken voor tips of resources om het spel verder te spelen. Om deze tips of resources te krijgen moet een opdracht voltooid worden. Bij het niet voltooien van de opdracht krijgt de speler geen tip of resource.

Competitie/coöperatie

* Competitie: oplossen van retro mini-games
* Coöperatie: karakters die vrienden worden met de speler.

Unique Selling Points

* Avontuur game in samenwerking met retro games zoals Pong, Snake & Space Invaders.
* In plaats van een normaal of spannend verhaal is dit bizar en zwaar ongeloofwaardig.

Doelplatform

* PC

Ontwerpdoelen

* Het verhaal heeft een duidelijk begin en eind.
* Het eind van het verhaal moet behaald kunnen worden.
* De speler kan rondlopen in de spelwereld.
* De speler kan praten met NPC’s
* NPC’s kunnen minigames in het spel starten.
* Het spel mag niet crashen bij kleine fouten.
* Het opstarten van het spel moet makkelijk zijn.
* Aan het begin van het spel kan de speler zijn karakter kiezen.

Karakters, verhaal en setting

* Commander El Theodoro:
  + Een pratende teddybeer die de speler bij zich heeft aan het begin van het spel. Hij ziet eruit als een gewone bruine knuffel teddybeer, maar bezit over magische krachten die de speler helpen om zijn weg naar huis terug te kunnen vinden.
* Barkley the Great:
  + Hij is een grote gespierde man, en de koning van het rijk. Hij heeft een erg kort lontje en denkt dat iedereen tegen hem is. Omdat niemand in het dorp de speler kent ziet hij de speler als een spion en wilt hem daarom opsluiten.
* Mache Pwason:[[2]](#footnote-2)
  + Zij is een aardige vrouw die graag mensen helpt. Zij vindt de speler bewusteloos in de bosjes liggen en neemt hem daarom mee naar haar huis. Zij kent de meeste mensen en geeft de speler tips met wie hij moet praten om verder te kunnen met zijn zoektocht. Daarnaast kan de speler in haar huis uitrusten.

Mini games

In het spel zijn verschillende retro mini games te spelen. De lijst met mini games is nog niet helemaal zeker maar de spellen die nu zijn gekozen is de volgende lijst:

* Pong
* Snake
* Space Invaders
* PacMan

# Bijlage 2: Klassendiagram

# C:\Users\jeroe\AppData\Local\Microsoft\Windows\INetCacheContent.Word\Klassendiagram.png

1. Trello-bord (<https://trello.com/b/ZO2BO9mq/project-gaming-git-gud>) [↑](#footnote-ref-1)
2. Deze naam komt uit Haïti en betekent in het Haïtiaans Creools: vismarkt. [↑](#footnote-ref-2)