Magnus Kjeldsen
Sebastian Mervog
Anders Kølln 20/05-2020

Digitalt Design eksamensprojekt - Spiludvikling

Forfattere:

Magnus Kjeldsen

Sebastian Mervog

Anders Kølln

Vejleder: John Gerhard Jensen (jgj)

Institution: H.C. Ørsted Gymnasiet - Lyngby

Magnus Kjeldsen
Sebastian Mervog
Anders Kølln 20/05-2020

Resume

I denne rapport forklarer vi hvordan vi har skabt et spil med programmet processing. Spillet er skabt med formålet at underholde en målgruppe af “casual players”. Rapporten gennemgår hvordan vi analyserede problemet og hvordan vi fandt frem til en løsning. Samt en detaljeret teknisk beskrivelse af programmets funktioner. Programmets opbygning og kode vil blive forklaret og blive testet på udvalgte spillere i den rette målgruppe.

Magnus Kjeldsen
Sebastian Mervog
Anders Kølln 20/05-2020

Indholdsfortegnelse

Magnus Kjeldsen
Sebastian Mervog
Anders Kølln 20/05-2020

Problemidentifikation

I dette projekt kunne vil vælge at arbejde med 6 forskellige cases.
Case 1: Spil i undervisningen.
Case 2: Realtime strategispil.
Case 3: Simulering af kroppen.
Case 4: Dataopsamling og visualisering.
Case 5: Neuroevolution.
Case 6: Vælg selv.
Af disse cases valgte vi nummer 6. Til denne case skulle vi selv formulere nogle krav til projektet. Som projekt ville vi gerne arbejde med spiludvikling med fokus på strategi. Det behøvede ikke at være turn-based eller i realtime, men der skulle være et element af strategi. Efter vi valgte denne case opstillede vi disse krav.

- Spillet skal have strategiske elementer.
- Spillet skal kunne tabes hvis spilleren ikke er god nok.
- Spillet skal være underholdende.
- Spilleren skal kunne bruge forskellige “brikker”, units eller objekter med forskellige egenskaber.
- Spillet skal have grafik der passer til spillets historie

Problemanalyse

Til at opfylde de krav vi havde sat til casen lavede vi en liste over spil genre der kunne opfylde kravene. Disse genre blev så analyseret for styrker og svagheder. De genre vi har analyseret er: Grand Strategy, AoE style/Old school, Total War Style / Battle Simulator og Tower Defense.

Idé 1: “Grand Strategy

En mere rigid struktur hvor spilleren må opgive lidt autonomi over sine brikker for at prioritere “realisme”. Der vil være mere fokus på hvorledes spillerens ressourcer benyttes til at

opbygge en stærkere “base”. Brikkerne vil i dette tilfælde bestå af områder som spilleren eller computeren kontrollere hvor målet er at ende med at kontrollere alle “områder” gennem krig med modstandere. Vil fungere bedst med mange forskellige AIs der interagerer med hinanden hvilket vil give mulighed for fredelig/diplomatiske interaktionerne mellem computer/spiller samt computer/computer. I forhold til krig vil brikkerne være unikke i forhold til effektivitet i kamp mod andre brikker, deres bevægelseshastighed samt evt. unikke evner forbeholdt bestemte brikker. De vil være begrænsede i bevægelse med hensyn til de såkaldte “områder” som er opbygget rigidt således at det er muligt at kontrollere hele kortet/banen ved at kontrollere alle “områder”.

Pros:

- Vil give mulighed for mange komplicerede interaktioner mellem computere og spiller samt elementer af selve spillet som vil påvirke hinanden
- Vil give mulighed for at benytte databaser til at holde styr på de mange informationer om hver spiller/computers ressource niveauer og de eller iboende karakteristika af “områderne”
- Kan nemt leve op til kravene Case 2 beskrivelsen

Cons:

- Kan muligvis blive en for stor udfordring at arbejde med mange computere som skal tage stilling til deres unikke situation og beslutte sig for en handling i Real Time hvor den samtidig skal forholde til aktioner foretaget af andre spillere

Idé 2: AoE style/Old school

Mere åben og fri bevægelse end idé 1. Her kunne man begrænse det til 1 spiller mod 1 computer, hvor de vil findes på to sider af et evt. ukendt kort eller en kendt bane. Målet vil være at opbygge en stærk hær bestående af forskellige brikker således at de kombinere deres evner effektivt. Dette vil gøres ved at bygge bygninger i ens base som kan have forskellige funktioner i forhold til a) produktion af “brikker” til ens hær, b) forsvar af ens base mod ens modstander, c) mere effektiv “høstning” af ressourcer der skal bruges til opbygning af disse bygninger samt hæren. Spilleren vil kontrollere sine brikker meget frit og give dem instruktioner som de skal udføre. En instruktion kunne bestå af “Fæld det træ” eller “Angrib

den bygning/modstander brik" hvorefter brikken vil være kodet til at udføre instruktionen. Det ultimative mål vil være at destruere modstanderens base

Pros:

- Kan begrænses til en modstander computer hvilket gør det betydeligt mindre kompliceret
- Kan stadig blive tilstrækkeligt komplekst i forhold til at kode hvordan "brikkerne" skal agere på baggrund af spillerens instruktioner samt interaktioner mellem brikker

Cons:

- Kan blive en udfordring at opbygge et kort/bane der kan vises delvist således at man ser noget nyt når man bevæger sig uden for "skærmen"

Idé 3: Total War Style / Battle Simulator

Næsten eksklusivt fokus på kamp/krig og hvordan man som spiller taktisk bruger ens brikker effektivt i en kamp. Mere autonomi givet til spilleren over ens brikker og kampstrategi vil være centralt. I spiller er der forskellige typer af enheder, som alle besidder forskellige egenskaber. De ser også alle forskellige ud grafisk. De skal kunne modtage instrukser fra spilleren og der skal være en computerstyret spiller som man spiller imod. Man skal have mulighed for at kunne se slaget i et fugleperspektiv som har forskellige højder. Spilleren skal kunne kommandere sine enheder ved brug af musen, spilleren skal kunne bestemme deres bevægelse og deres handlinger til en hvis grad. Ens enheder har forskellige egenskaber som "helbred" "angreb" og "forsvar", alle disse værdier skal kunne ændres dynamisk baseret på omstændighederne som enhederne befinder sig i.

Pros:

- Spillet er meget begrænset, så det er let at realisere.
- Koden kan stadig være meget kompleks, da der ikke er nogen grænse for hvor kompliceret vi kan lave vores kunstige intelligens.
- Der er mulighed for at anvende polymorfisme, når vi udvikler vores brikker.

Cons:

- Siden at spillet er så begrænset, så kan det være vi løber ind i uforudsete problemer, i form af begrænsning af vores ambitioner grundet snæverhed.

- Grafikken er ikke særligt svært at lave, men det er meget tidskrævende. Derfor kan det være at der går meget tid spildt på grafik udvikling.

Idé 4 Tower Defense

Spilleren er ansvarlig for at beskytte en base mens computeren er ansvarlig for at angribe. Her skal spilleren vælge hvilke tårne der skal bygges for at beskytte basen. Computeren prøver at ødelægge basen ved at sende bølger af fjender. Hver bølge bliver svære, men giver spilleren flere penge eller ressourcer til at bygge flere tårne. Spilleren ser banen oppe fra og grafikken er ofte 2D. Strategien kommer fra hvilke tårne spilleren kan placere og hvilke typer fjender computeren sender. Fx. kunne computeren sende flyvende fjender som kun nogle bestemte tårne kan ramme.

Pros:

- Spillet er ikke særligt avanceret, så det er lettere at realisere.
- Nemt at tilføje nye baner, tårne eller fjender
- Kræver ikke meget cpu kraft som grand strategy eller gpu kraft som battle simulator eller old school strategy spil.

Cons

- Da spillet ikke er særligt avanceret kan projektet blive for småt
- Ikke et særligt stort råderum for innovation

Produktprincip

Iterative proces

Den iterative proces går ud på at gentage en samling af handlinger, som bruges til at teste og validere om ens digitale produkt fungere optimalt.
Vores proces startede med, at vi vurderede feedbacken vi fik fra det tidligere stadie. Feedbacken blev samlet vha test på folk som ikke var med i gruppen. Derefter så diskutere vi resultaterne af testene og konkludere på hvad resultatet var. Vi brainstorm derefter

features i vores spil, som kunne blive implementeret som ville fikse de problemer som blev fundet. Featuresne som skulle blive implementeret blev også bestemt ud fra ekspert meninger som kommer udefra som er relevante. Derefter går vi igang med at kode vores program.

De milepæle som skulle nås var tilrettelagt ud fra vores tidsplan, men til tider var tidsplanen ikke præcis pga uforudsete problemer opstod. Derfor så aftaler vi, hvornår det næste skulle være færdigt baseret på, hvor tæt på vi var på deadlinen.

Løsningsforslag

Siden at vi valgte case 6, så skal vi udvikle et spil med underholdningsværdi, så skulle vi først bestemme genren vi ønskede at arbejde i. Vi startede med at brainstorme, nogle forskellige muligheder og så os varme på et par af den.
Den første var, et 4X strategi spil. 4X står for "eXplore, eXpand, eXploit og eXterminate". Vi mente det her var interessant, fordi vi spiller selv de her spil meget og er yderst engageret i kompleksiteten af spillet.
Det andet var, et tower defense spil. Vi har selv spillet en del tower defense spil, hovedsageligt bloons tower defense serien og mente det var sjovt og realistisk at udvikle på egen hånd.
Det tredje var et “puzzle game”, et spil hvori man skal løse forskellige puslespil. Ikke bogstaveligt talt lægge brikker sammen i et traditionelt puslespil, men i stedet for så arbejder man med forskellige abstrakte koncepter for at løse en opgave.

For at vælge mellem de her tre genre, så opstillede vi nogle forskellige kriterier, som vi mente var vigtige på baggrund af tidligere projekter og opgavens formål. Udover dette, så valgte vi også et spil som repræsentere hver genre. Vi valgte et spil, som vi var bekendte med på forhånd og som havde stor succes. Vi vurdere om et spil er succesfuldt eller ej, baseret på hvor positivt feedback det fik. Grunden til, vi gør dette er fordi, så kan vi se hvad genres potentiale er, da det bliver illustreret vha spillets succes.

Underholdningsværdi (10)

Underholdningsværdi er et vagt koncept, men det er en beskrivelse af hvor underholdende/sjovt et program er overordnet. Det er svært at kvantificere, men der er nogle

forskellige ting som man typisk kan måle hvordan et spil er sjovt. Det er fx i form af hvor meget folk spiller ens spil og hvor langt tid ens spillere er engageret. Siden målet for projektet, er at lave et sjovt spil, så er underholdningsværdien alfa og omega. Siden at dette her er ifølge opgavebeskrivelsen det vigtigste mål, så får den her 10.

Realiserbarhed (8)

For at vores program kan være en succes så skal vi kunne realisere det, for at realisere vores program skal vi kunne udvikle det. Det betyder der skal få mandetimer til det og et meget lille budget eller intet budget. Det betyder også at det skal være koncepter/teknikker som vi er blevet undervist i, vi skal kunne anvende i kodningen af vores program. Hvis vi ikke kunne realisere vores projekt, så ville det være umuligt at få succes, da vi aldrig ville blive færdige. Vi giver derfor denne her kategori 8 i vægtning, da hvis vi aldrig bliver færdige, så kan det aldrig blive succesfuldt.

Kompleksitet (7)

For at et spil skal holde en bruger engageret i længere tid, så skal det havde en hvis grad af kompleksitet. Hvis et spil er meget simpelt og kan monteres på kort tid, så løber spilleren hurtigt tør for ting og lave. Det her hænger sammen med progression som beskrevet tidligere, da progression introducere nye ting, som kan øge kompleksiteten og holde spillere engageret. Spillet World of Warcraft var et godt eksempel på dette. Vi vælger at give kompleksitet 7 i vægtning. Vi giver den 7 fordi, hvis et spil er meget simpelt så holder det ikke spillerne engageret særligt længe. Spil uden super meget kompleksitet har dog været succesfulde, Bloons Tower defense serien er et eksempel på dette.

Brugervenlighed (6)

Nogle genre af spil er typisk mere brugervenlige end andre. Det er vigtigt at have spillet være brugervenligt, da brugerne kan let blive skræmt væk, hvis spillet ikke er let at anvende. Et spil er mere brugervenligt, hvis det har en flad indlæringskurve end en stejl en. Vi mener at det her er vigtigt, da hvis vi ønsker at få brugere i vores spil, så skal det være indbydende. Derfor giver vi det 6 i karakter. Grunden til at vi ikke giver det en højere vægtning, er fordi, at spil som Europa Universalis 4 er ikke brugervenligt overhovedet, men det er stadig succesfuldt, så man kan stadig lave et godt spil som på samme tid ikke er brugervenligt overhovedet.

Bekendthed (7)

Det at lave et spil, er ikke noget vi er blevet undervist særligt meget i. Derfor er det vigtigt for os, at vi er bekendte med genren før vi begynder på et spil inden for en given genre. Der er dog ressourcer man kan trække på online, til spiludvikling, men de er svære at få hænderne på. Derfor vælger vi at give Bekendthed en vægtning på 7, da det ikke er umuligt at lave et spil uden erfaring med genren, men vi mener det ville hjælpe meget.

PV Skema

	Underholdningsværdi (10)	Realiserbarhed (8)	Kompleksitet (7)	Brugervenlighed (6)	Bekendthed (7)	Samlet score
4x	8	1	10	3	9	239
Tower defense	8	7	4	10	8	280
Puzzle spil	9	3	9	8	6	267

Forklaring af karakterer

4x (Europa Universalis 4 som eksempel)
Underholdningsværdien basere vi ud fra bruger anbefalinger og ekspert anbefalinger på metacritic og steam. De er meget positive, derfor giver de det en karakter på 8.

Figur 1. Billedet er et billede af anbefalingerne udarbejdet af brugerne af spillet på platformen steam. Spillet er Europa Universalis IV

Figur 2. Billedet er en opsummering af anbefalinger givet af kritikere og brugere af spillet Europa Universalis IV på hjemmesiden metacritic.com

Realiserbarheden er dog minimal mener vi. Spillet er utroligt kompliceret og har mange forskellige komplekse algoritmer som bestemmer hvad den kunstige intelligens foretager sig. Der er simpelthen for mange features i denne her type af spil, især i Europa Universalis 4, til at vi tror vi kan lave det. Af samme årsag så for kompleksitet en score på 10, vi kan give et

personligt eksempel på hvorfor som også er reflekteret i anmeldelser[1]. Man kan havde spillet i flere hundrede timer og stadigt lære nye ting. Brugervenligheden for den her type spil er som regelt lav, da de er meget komplekse, derfor er det svært for nye spillere at begynde med at spille dem. Bekendtheden score højt, fordi vi har flere tusind timers erfaring med at spille de her spil selv. Den samlede score er 239, hvilket giver den her genre den laveste score af de tre kandidater. Udover dette, så får den kun 1 i realiserbarhed, så selv hvis den havde fået højest score havde vi været nødt til at frasortere den. Den vil blive frasorteret, fordi det ville havde været urealistisk at gøre projektet færdigt.

Tower defense (Bloons Tower Defense 6 som eksempel)

Underholdningsværdien for her også 8, det bliver igen baseret ud fra bruger anbefalinger på steam. Denne gang inkludere vi ikke metacritic, bloons tower defense 6 ikke er på deres hjemmeside.

Figur 3. På billedet kan man se de nyeste og de samlede anbefalinger udarbejdet af brugere til Bloons Tower Defense 6.

Vi giver det 7 i realiserbarhed, da vi meget let kan forstå, hvordan spillet fungere. Vi tror også vi let kan kode det, da det ikke er spækket med ting vi ikke har lært meget om, såsom kunstig intelligens. Det eneste kunstige intelligens der er, er bevægelse og sigte. Pga hvor

[1] https://www.pcgamer.com/europa-universalis-iv-review/

simpelt spillet er, så får den kun 4 i kompleksitet. Siden det er så simpelt. så får den også 10 i brugervenlighed, spillet har meget få inputs fra spilleren og et koncept som er let og forstå. Vi giver den her 8 i bekendthed, da vi selv har spillet de her spil meget, men det er for længere tid siden, så vi husker dem ikke skarpt. Samlet score er 280, hvilket giver tower defense genren den højeste score. Det er også derfor vi ender med at vælge denne her genre.

Puzzle spil (Portal 2 som eksempel)

Siden portal 2, fik utroligt gode bruger anmeldelser på steam, så giver vi denne her genre 9 i underholdningsværdi. Realiserbarheden er dog lav, hvis man skal lave et spil med svære puzzles, så bliver de meget svære at lave. Kompleksiteten er dog meget høj, da man kan indsætte flere måder at løse et problem på, eller indføre nye funktionaliteter i spillet, som kan ændre på ens tilgang. Brugervenligheden i sådan et her spil er også højt, da man kan starte let ud og fra hvert niveau man går videre indføre en ny kompleksitet. Dette giver mulighed for en flad indlæringskurve. Denne her genre er den som vi har spillet mindst, derfor giver vi den kun 6 i bekendthed. Den samlede score for denne her genre er 267 point, dette gør det til den næst højest scorende. Den bliver også frasorteret da tower defense genren scorede højere pointmæssigt.

Vision for vores projekt

Vi har valgt at tage udgangspunkt i forskellige spil inden for “tower defense” genren, det mest kendte spil og det mest succesfulde spil inden for genren er bloons tower defense serien. Så vi vil drage meget inspiration derfra.

Hvorfor ønsker vi at efterligne Bloons Tower Defense 6?

Bloons tower defense 6, er den nyeste udgivelse i Bloons Tower Defense serien. Bloons Tower Defense serien, er nok den mest succesfulde spil inden for tower defense genren, og en serie vi personligt synes meget godt om. Vi ønsker derfor, at efterleve mange af spillets funktionaliteter i vores eget spil, da de forstår, hvordan man udvikler et godt tower defense spil. Vi vil dog ikke være en tro kopi, pga så er der ingen grund til at spille vores spil, da man lige så godt bare kunne spille Bloons Tower Defense 6.

Hvor succesfuldt er bloons tower defense?

Den nyeste udgave af bloons tower defense er den sjette i serien ved navn bloons tower defense 6. Den er udgivet på spil platformen Steam, hvor den har over 20000 anmeldelser

fra brugere af dette spil. De 2000 nyeste er 96% positive og samlet set er de 20749 anmeldelser 94% positive. Dette viser at dette spil er meget vellidt blandt brugerne på Steam, det er overvældende positivt, hvilket gør dette en ekseptionel succes ud fra et underholdnings perspektiv. Vi har ikke adgang til, hvor mange gange spillet er blevet solgt og om det har været under et udsalg, så derfor kan vi ikke vurdere præcist hvor stor en finansiel succes spillet har været. Denne her artikel estimere at den gennemsnitlige rate af anbefalinger i forhold til salg for spil udgivet i 2017 er omkring 63 salg for 1 anbefaling. Spillet var udgivet i 2018, men 2017 er det tal som er tættest på så det er det tal vi bruger til at estimere hvor mange enheder af bloons tower defense 6 er blevet solgt. Lige nu er der 20479 anbefalinger, så det estimeret antal er 1290177 enheder solgt.
Ifølge denne her artikel så tager steam 30% af pengene generet fra salg, hvor indtægten er under 10 millioner dollars og Steam tager 25% af pengene generet fra salg fra over 10 millioner dollars til 50 millioner dollars og fra summer over 50 millioner dollars tager de 20% af pengene. Den samlede mængde af indtægt var 10566549.63 dollars, hvilket betyder at Ninja Kiwi fik $(10000000\ US\$ * 0.7) + (566549.63 US\$ * 0.75) = 7\ 424\ 912.22\ US\$$. Det her tal tæller kun enhederne solgt på Steam og ikke de som er blevet solgt på mobile enheder og andre platforme på PC, Linux eller Mac. Vi kender ikke til budgettet af Bloons tower defense 6, så det er svært at sige hvor stor en profit Ninja Kiwi fik og hvad Ninja Kiwis økonomiske ambitioner og forventninger var. Alt dette beviser der er et marked for denne type af spil, hvis der kan sælges så store kvantiteter og blive udviklet et halvt dusin spil fra denne serie. Hvis der ikke var, så tror vi ikke at Ninja Kiwi havde valgt at udvikle så mange iterationer af spillet, da det har omkostninger og et firmas mål er, som regelt at tjene penge.

Hvad gjorde Bloons tower defense 6 sjovt?

For at finde ud af, hvad der gjorde spillet sjovt, så har vi læst 10 af de mest “hjælpsomme” anbefalinger fra brugere på steam og ser på, hvad der gjorde spillet godt eller dårligt og kategorisere dem i forskellige kategorier. De mest hjælpsomme anbefalinger fandt vi ved, at sortere for mest hjælpsomme på steam.

Opsummering af anbefalingerne

Dem som giver disse anbefalinger er brugere som ikke spiller mere end 100 timer typisk, det er som regelt mellem 20-60 timer som de når at spille og aldrig mere end 10 timer spillet i de seneste to uger. Dette fortæller os, at folk som ikke spiller spillet meget i gennemsnit, bedst kan lide det, hvilket får os til at tro at de er “Casual” spillere, hvilket betyder de ikke spiller det

seriøst men bare som noget til at dræbe tiden. Mere seriøse spillere havde fx spillet meget mere og nok også givet en anden slags anmeldelse. Dette kan fortælle os noget, om den målgruppe vi skal appellere til og kan forklare noget om spillernes forventninger til spillet.

Udover dette så er der også som regel en reference til de forrige spil i serien, oftest bloons tower defense 5 som var versionen af spillet før den nyeste. De snakker hovedsageligt om, at dette her spil er en positiv udvidelse på det forrige spil, features såsom nye tårne er ofte citeret som en positiv ting og er den hovedsagelige årsag til anbefalingen.
Anbefalingerne er oftest kortfattet og nævner ikke andre substantielle årsager til deres positive oplevelse med spillet, men der er et negativt tema som går igen. “Microtransactions” er små betalinger man giver, hvor man som spiller for adgang til forskellige features og eller ressourcer, de er givet med rigtige penge og er generelt set ned på. Flere anbefalinger citere dette som en negativ del af spillet.

Konklusion
Vi kan konkludere at havde forskellige typer af tårne er meget vigtigt for spillerne af denne her type af spil. Udover dette så lader det til mange af dem allerede er fans af genren og denne serie specifikt, så vi appellere til brugere som på forhånd er interesseret i genren i stedet for brugere, hvis interesse skal startes. Dette betyder, at vi kommer til at skulle udvikle et spil som minder om bloons tower defense 6, men som bringer en ny form for nuance ind i spillet. Dette kommer nok til at være i form af nye features som fx nye slags tårne, fælder eller måske noget helt tredje så der er noget nyt for spilleren.

Hvad er bloons tower defense?
Bloons tower defense er et spil, hvori der er en bane som balloner bevæger sig hen ad og de skal nå et endemål for at de har udført deres opgave. Deres opgave er at modarbejde spilleren ved at nå deres endemål, dette fjerner et liv fra spilleren. Spilleren skal kunne komme gennem så mange bølger som muligt. Spilleren beskytte endemålet fra disse balloner ved at opstille forskellige typer af aber som skyder ballonerne med pile og andre projektiler. Spillet bliver progressivt sværere, ved at der kommer forskellige typer af balloner og flere balloner end runden før. Spilleren kan så opgradere deres aber, så de kan stoppe ballonerne mere effektivt.
På Wikipedia kan man også læse et resume af spillet “***Bloons Tower Defense*** (also ***Bloons TD*** or ***BTD***) are a series of tower-defense games that were created by Ninja Kiwi. The focus of

the games is to use towers to defeat an enemy called Bloons to progress through the game. As you play the game, the Bloons become more difficult to defeat. Players are able to upgrade their towers to help defeat stronger enemies. Bloons can come with power ups, such as camouflage and regeneration. Most of the Tower Defense games can be played on computers and mobile/tablets." [2]

Hvad er vores vision

Vi ønsker at vores spil i stor grad kommer til at minde om bloons tower defense serien. Der kommer dog til at være nogle betydelige forskelle. Vi ønsker et mere seriøst tema, så vi vælger noget som minder om Terminators mod mennesker. Udover dette så er bloons tower defense også et spil som har adgang til langt flere ressourcer end vi har, både i form af penge og mandetimer. Dette gør at vi bliver nødt til at nedskalere vores projekt i forhold til bloons tower defense. Det betyder fx, at vores grafik og lyd nok ikke bliver lige så skarp eller man har mindre indhold i spillet i form af diverse tårne og "fjender".

Produktudformning

Spil Udviklingsteknikker

Progression

For at holde ens brugere engageret, så skal man give dem noget at arbejde mod. Dette koncept hedder progression. Progression kan bliver ofte set i form af et "level up" system, hvor spilleren går niveauer op og målet er at nå det sidste niveau. Et godt eksempel på dette kunne være World of Warcraft, hvori man starter fra niveau 1 og målet er at nå niveau 120. Det stopper dog ikke der, man kan ende med at bruge mange hundredvis af timer på at finde det rette udstyr når man når niveau 120, på den måde så holder Blizzard deres spillerbase engageret i meget langt tid og brugeren har konstant noget at arbejde henimod.

For Bloons Tower Defense, så er der nogle forskellige måder de indbygger progression i deres spil. Jo flere runder man spiller, jo sværere bliver spillet, så spiller bliver progressivt sværere. Man kan også få adgang til nye slags tårne ved at spille mere. Dette holder brugeren engageret, da som vi kunne se på anbefalingerne, så ville de gerne havde

[2] https://simple.wikipedia.org/wiki/Bloons_Tower_Defense

forskellige tårne. Udover dette, så er der også forskellige typer af sværhedsgrader som man kan gennemføre, så man kan spille den samme bane på, let, mellem og høj sværhedsgrad. På den måde, så kan man forlænge den tid brugeren er engageret på, da de nu spiller et stykke indhold tre gange i stedet for en. I alt er der over 100 opgraderinger man kan opnå i spillet, så der er meget indhold man skal lukke op for som bruger og man kan derfor investere dusinvis ekstra timer i spillet, for at give mulighed for at spille med det låste indhold.

Man skal dog også passe på, at mængden af progression ikke er for høj, det skete nemlig for World of Warcraft, hvor de nu prøver at modvirke det. Det kan nemlig være overvældende for en ny spiller, at se hvor meget arbejde der er foran dem, før de er færdige med spillet. I World of Warcraft går man fra lvl 1 til 120, når man når lvl 120 er man ikke engang færdig, selvom det måske har taget dusinvis af timer at nå dertil. Derfor så har Blizzard udviklerne af World of Warcraft, valgt at i specielle zoner, så starter man i level 50 og kan komme op på level 60. Ifølge udviklerne selv, så tror de det kommer til at få hvert level up, til at være specielt da man nu har 10 i stedet for 119, hvilket gør de 10 mere specielle.

Udover at en karakter kan stige i niveau, så er det også smart, at spillet bliver gradvist sværere, så spillerne kan nå at nyde spillet og se hvad spillet har at tilbyde, inden de potentielt bliver skræmt væk af en stor udfordring.
https://www.polygon.com/2019/11/2/20944794/blizzard-level-squish-50-60-120-shadowlands-cap-leveling-experience-new-player
https://www.makinggames.biz/feature/essential-retention-mechanics-to-keep-players-engaged8091.html

Grafik

Et andet eksempel på dette, er listen af tårne i højre side af skærmen. Den bruger et andet farveskema end resten af banen.

Udover dette, så skal der også være specielle indikatorer som fortæller spilleren at ting er anderledes fra hinanden[3]. På billedet foroven kan man fx se, at der er en blå pøl af vand, den drastiske farveforskel hjælper det spilleren at differentiere, at dette er et forskelligt type af terræn.

[3] https://gameanalytics.com/blog/3-steps-to-improve-your-games-graphics.html

Gestaltlovene

Gestaltlovene er principper som hvis fulgt kan hjælpe med at producere god grafik. Der er mange af dem, vi forklarer her hvordan de bliver anvendt og hvad de gør.

Figur 5. På billedet kan man se et screenshot fra spillet Bloons Tower Defense 6, hvori man er midt i en runde.

Loven om nærhed

I billedet kan man også se nogle forskellige gestaltlove blive anvendt. Man kan se princippet om nærhed, som fortæller os at ting som er tæt på hinanden hænger sammen. Man kan se det, i alle firkanterne med tårne i til højre. De kan alle sammen købes og det kan man se, da de alle hænger sammen.[4]

Loven om lighed

Loven om lighed bliver også anvendt, da man kan differentiere mellem de "allierede" og "fjenden" i spillet. Det kan man, da det er aber mod balloner og man kan se at ballonerne hænger sammen og aberne hænger sammen. [5]

Loven om lukkethed

Loven om lukkethed bliver også anvendt i højre side af billedet. De tårne som man kan købe er i en separat boks, dette fortæller os at de ikke hænger sammen med det som sker uden

[4] https://informatik.systime.dk/?id=1132&L=0
[5] https://informatik.systime.dk/?id=1132&L=0

for boksen. Dette giver også mening i forhold til spillet, da tårnene til højre ikke deltager i hvad der sker på banen, da de ikke er blevet købt endnu.[6]

Loven om forbundethed

Loven om forbundethed bliver også anvendt i billedet foroven. Man kan se at der er en abe med en beholder på ryggen. Beholderen indeholder en gul væske og man kan se at lige foran den, så er der en gul væske som bliver skudt ud. [7]

Lyd

Når det kommer til lyd, så er det vigtigt at man anvender de rette lydeffekter. De skal give brugeren en klar fornemmelse om det de gør er rigtigt eller forkert. Dette bliver gjort i mange skydespil fx, der er der en karakteristisk lyd som bliver afspillet når ens karakter tager skade eller giver skade til modstanderen. Det kan fx være lyden af hjertebanken når man bliver ramt, eller en "hitmarker" når man rammer noget, dette ses ofte i Call of Duty spil. Man skal også sørge for at lydeffekterne passer sammen med temaet af spillet, hvis man fx laver et arcade spil giver det ikke så meget mening at optimere ens lyd for realisme, det omvendte er også sandt.

I Bloons Tower Defense 6 så bruger de forskellige lydeffekter til at give feedback til spilleren. Der er en pop lyd, når man nedlægger en ballon. Det her er et godt valg af lyd, pga spilleren forventer en pop lyd, når en ballon eksplodere, fordi det er det som sker i den virkelige verden når en ballon eksplodere. Det her fungere som en hitmarker som forklaret tidligere. Temaet passer også godt sammen med spillet, da man skal helst ikke havde nogle voldelige temaer i Bloons Tower Defense 6 lydmæssigt, pga det ville være usammenhængende med resten af temaet. Derfor, så er et uskyldigt pop bedre, end hvis fx et skrig havde kommet i stedet for.

Målgruppe

Når man designer og udvikler et produkt, så er det altid vigtigt at forstå, hvem det er man appellere til. Det er vigtigt, for ellers så inkludere man måske ting i sit produkt som støder en bestemt gruppe af mennesker væk, eller tiltrækker de forkerte.

[6] https://informatik.systime.dk/?id=1132&L=0
[7] https://informatik.systime.dk/?id=1132&L=0

For at forstå hvilke type af brugere vi har med at gøre, så besøgte vi steam reviews for bloons tower defense 6, da det er det mest velkendte spil inden for tower defense genren. Vi kunne tydeligt se at det var “casuals” som spillede det her spil. Der var nogle forskellige karakteristika ved spillet og dets brugere, som fortalte os, at det var casuals.

Hvordan fandt vi ud af det var casuals?

Ifølge denne her artikel er en casual er en som nyder sine spil uden at investere store mængder af tid ind i spillet, en som nyder spil uden en stejl indlæringskurve og som typisk spiller spontant[8]. Man finder ofte de her spillere på computere og telefoner. Hvis man kigger på de typer af spillere som ligger anbefalinger på steam, så spiller de typisk set ikke mere end 40 minutter om dagen og deres samlede spilletid lå mellem 20-60 timer. Hvis man havde appelleret til folk som ønsker et mere konkurrence drevet spil, så ville man kunne se spillerne var mere dedikeret til spillet. Det ville så reflektere sig i deres spilletid, som kan komme op på flere timer om dagen. Eller i Counter Strike Global Offensive, hvor mange spillere også spiller flere timer om dagen for at blive bedre[9].

Udover dette så er der nogle karistika ved spillet selv, som indikere at spillet er for casuals. Spillet er meget simpelt og har derfor ikke en stejl indlæringskurve. Det er meget simpelt fordi, at konceptet er let og forstå. Man skal bare beskytte et endepunkt mod balloner, som man popper ved at kaste en dart på ballonerne, det er et koncept som er “down to earth” og ikke særligt abstrakt. Det er ikke konkurrence drevet, det er også vigtigt for casuals da de bare nyder et spil for sjov og ikke for at vinde nødvendigvis.

Alt det her leder os til at tro, at vi skal appellere til casuals.

Udover dette, så er der også et system på steam som lader brugerne af et spil give spillet et “tag”. Et tag er en måde at kategorisere et spil på. En af disse tags er casual[10], man kan se

[8] https://www.computerhope.com/jargon/c/casual-gaming.htm
[9] https://www.criticalhit.net/gaming/casual-vs-competitive-gaming-differences/
[10] https://store.steampowered.com/app/960090/Bloons_TD_6/

det på denne her hjemmeside. Det er en af de mere populære tags af spillet.

Figur 4. På billedet kan man se en liste af de tags som er tilknyttet til spillet Bloons Tower Defense 6 på Steam.

Så udover vores egen analyse, af hvilken bruger som anvender dette her spil. Så har brugerne af spillet selv klassificeret dette spil, som at være for casuals.

Hvordan appellerer man til vores målgruppe

Hvis vi vil appellerer til vores målgruppe, så skal vi forstå hvad de godt kan lide. Vi har allerede kigget på nogle anmeldelser af spillet, men man kan også kigge på tagsne fra før. Tagsne fortæller os noget, om hvordan brugerne af bloons tower defense 6 forstår spillet. Kombineret med, at spillet har overvældende positive anmeldelser, så må disse her tags være positive.

Inden for grafik delen af spillet, så kan vi kigge på nogle tags som appellere til brugerne.

De tags er “Sødt”, “2D” og “Tegnefilmsagtigt”. “3D” er også angivet, men hvis man kigger på spillet så er der ikke noget 3D grafik, så det er svært at finde ud af, hvad brugerne mener her. De tre tags nævnt udover 3D overrasker os ikke. Vi kan se, hvorfor man mener det er sødt, da der ikke er noget vold eller måde nogle ting lider på. 2D og Tegnefilmsagtigt hænger lidt sammen, da langt de fleste tegnefilm er i 2D. Det at det er tegnefilm hænger også sammen med, at det er sødt, da man ser det som noget for børn. Så vi kan se, at det grafiske tema i det her er, et sødt og uskyldigt tema.

Figur 5. På billedet kan man se et screenshot fra spillet Bloons Tower Defense 6, hvori man er midt i en runde.

Som man kan se på billedet ovenfor, så er det meget uskyldigt og børnevenligt. Det er tegnefilms aber som skyder dartpile mod balloner. Spillet i dette screenshot, tager også sted i en romantiseret efterårsskov. Det foregår fx ikke på en slagmark, med telte hvor folk bliver behandlet for sår. Det ville havde gjort spillet grusomt og havde skræmt nogle af de brugere væk, som spiller pga det søde og uskyldige tema.

Vi skal dog prøve at differentiere os selv fra bloons tower defense serien, da hvis vi bare laver en kopi, så kan man lige så godt bare spille bloons tower defense i stedet for, da det er bedre lavet. En måde vi kan differentiere os selv på, er ved at have et andet tema i vores spil. Hvis man laver et spil med et mere brutalt tema, så kan man tiltrække nogle af de spillere, som mener at bloons tower defense serien er for sødt og uskyldigt. Der er selvfølgelig også andre måder, vi ønsker at differentiere os selv fra bloons tower defense

serien på, pga kun at have et andet tema ville nok være for overfladisk. Vi kan derfor fx havde en anden slags tårne, som fungere anderledes, eller havde anderledes typer af fjender man skal bekæmpe.

Spillets funktionalitet er lidt bedre beskrevet her, i tags kategorien.
Tagsne som beskriver spillets funktionaliteter er “Strategi”, “Tårn Forsvar”, “Singleplayer”, “Multiplayer”, “Co-Op”, “Online Co-Op”, “Casual”, “Svært”, “Action” og “Sjovt”.

Det sidste tag “Sjovt” tæller vi ikke med, da vi regner med, at det er en kombination af alle de andre tags, som gør det sjovt. Strategi kommer fra, at man skal planlægge på forhånd, hvordan man vil håndtere forskellige faktorer. Disse faktorer kommer fra, de forskellige typer af balloner og varierende antal. Det er også langsigtet, da man skal spille over hundrede runder for at vinde. Casual kommer fra, den flade indlæringskurve og hvordan det bliver spillet, altså lidt tid ad gangen og spontant. Svært kommer fra, hvor svære de sidste runder af spillet er, hvor man er op imod umulige odds, som kun kan overkommes med planlægning og strategi. Det skal dog nævnes, at selv de sidste runder se svære, så hører det stadig til den flade indlæringskurve, det er bare omkring slutningen af den indlæringskurve. Action har vi svært ved at genkende hvor det kommer fra, pga spillets langsomme tempo. Udfra disse her tags, så kan vi forme noget af vores spils funktionalitet, da vi skal prøve at emulere bloons tower defense 6 til en vis grad. Vi har dog valgt ikke, at inkludere Co-Op og Online Co-Op, da vores vision ikke er at lave et spil, hvori man samarbejder. Udover dette, så mener vi også det ville være for udfordrende, at lave en online version af spillet.

Produktkrav

Høj grafisk kvalitet

Med høj grafisk kvalitet så er der nogle forskellige kriterier som skal mødes. Grafikken skal være i høj fidelitet, det kan fx betyde at man har høj opløsning på ens tårne fx[11]. Et godt eksempel på dette er fx bloons tower defense serien, hvor de har en simpel grafisk stil, men den er eksekveret med høj fidelitet og professionel kvalitet.

[11] https://gameanalytics.com/blog/3-steps-to-improve-your-games-graphics.html

Figur 5. På billedet kan man se et screenshot fra spillet Bloons Tower Defense 6, hvori man er midt i en runde.

Her kan man fx se, at alle de forskellige karaktere på banen er i høj kvalitet, selvom de alle har en simpel kunststil. Med simpel kunststil så mener vi, at de ligner tegnefilm en smule i stedet for en realistisk kunststil som plejer at være de sværeste at eksekvere.Høj lydkvalitet
Med høj lydkvalitet så mener vi, at det skal følge de principper vi præsenterede i vores lyd afsnit. Det betyder at der først og fremmest skal være høj fidelitet i lyden, derefter så skal man også sørge for at lydeffekterne er passende. Det kan betyde, at en kanon fx skal lyde kraftfuld og et mindre stærkt tårn lyder selvfølgelig mindre stærkt.
Det er også standard for et spil at havde et soundtrack, så det soundtrack vi har skal passe til temaet og selvfølgelig også havde lydkvalitet i høj fidelitet.

Diverse spiloplevelser

Noget vi kunne konkludere ud fra vores læsninger af spilleres anbefalinger af Bloons tower defense 6, er at de elskede diverse tårne. Dette betyder at vi skal prøve at havde diverse tårne selv, da vi appellere til den samme slags bruger. Vi skal dog også huske, at vi skal havde andre slags tårne end Bloons Tower Defense 6. Hvis vi kopiere tårnene fra Bloons Tower Defense 6, så er der ikke meget grund til at spille vores spil over Bloons.

Flad indlæringskurve

Siden vi appellere til casuals, så er det ekstremt vigtigt at de ikke bliver skræmt væk, af en stejl indlæringskurve. Vi skal derfor sørge for, at vores spil er let og forstå, det har et simpelt koncept og kræver ikke komplicerede inputs fra spilleren. Med komplicerede inputs, er det ment, at trykke på knapper i komplicerede mønstre fx. Det kan betyde, at man har meget simpelt control layout med få knapper og multifunktionalitet fra knapperne.

Projektforberedelse

Scrum metoden

En normal metode, som anvendes til at producere et digitalt produkt er scrum metoden. Product owner i det her tilfælde var vores to lærer Anders Juul Refslund Petersen og John Gerhard Jensen. Vi anvendte en model for scrum metoden med 8 trin[12]

Produkt vision

Vores produkt vision blev udviklet på baggrund af analyse modeller såsom krav matrixen og ekspert meninger og datasæt. Visionen blev opdateret i samme tempo som vores analyse blev opdateret. Vi aftalte den nye version af visionen for projektet, hver gang vi mente vi havde fundet ny viden, som bør modificere vores vision. Det vil sige, der var ikke fastlagt en diskussion for et møde, hvori vi diskuterede visionen.

Product roadmap

Vi gennemgik vores tidsplan hver gang vi havde scrum møder, for at se om det er realistisk at vi bliver færdige og om vi skal justere ambitionsniveauet op eller ned.

Release plan

Vores release plan var todelt. Den første del var en simpel prototype som kunne teste vores koncept, for at se om det var tilfredsstillende. Den anden del var sporadiske brugertest, hver gang vi havde produceret nye features testede vi dem og samlede vores viden i vores scrum møder.

Product backlog

[12] https://advanz.dk/blog/hvad-er-scrum-metode/

Siden at vores case var at vi skulle vælge et produkt selv, så var de her krav mere fleksible. De blev i stor grad baseret ud fra vores analyse i stedet for vores product owner. Product owneren satte dog nogle krav, som var at vi skulle udvikle et spil med underholdningsværdi.

Sprint backlog

Vi delte opgaverne ud på baggrund af vores kompetence sæt. Hvis en person havde svært ved at udføre en opgave, så kan et andet gruppemedlem tage over eller supplere med deres egne kompetencer.

Sprint planlægning

Vi planlagde vores sprint på baggrund af, hvad det var vi manglede og tiden for vores sprints afhang af hvor meget tid vi havde tilbage. Ambitionsniveauet af vores sprints blev bestemt ud fra, hvor meget vi manglede.

Dagligt stand-up

Vi havde først daglige stand-up til sidst i vores proces, da vi manglede at koordinere på en daglig basis, så vi vidste præcist hvad vi manglede de sidste dage og hvem der lavede hvad, så vi kunne bruge den sidste tid bedst muligt.

Sprint review

Vi vurderede hvor godt vores sprint var, ved at teste vores produkt om det så var internt eller på en bruger som ikke var en del af gruppen. Ud fra resultaterne af testene, vurderede vi hvor succesfuldt sprintet havde været.

Product owner

Product owner i det her tilfælde var vores to lærere Anders Juul Refslund Petersen og John Gerhard Jensen.

Scrum master og scrum team

Vi havde en egalitær måde at organisere vores gruppe på, der var ikke en scrummaster, til gengæld så var vi alle sammen med til at sætte dagsordenen. Alle var også del af scrum teamet, dette var nødvendigt da vi alle skulle arbejde, hvis vi ønskede at nå vores mål.

Prototype

Vores første prototype var en simpel version af programmet, hvorpå der er en bane som "fjenden" skal igennem og steder hvorpå man kan placere sine tårne som skal stoppe "fjenden". Vi havde rimeligt let ved at implementere dette. Der var dog ikke noget avanceret grafik eller lydeffekter, da vi bare ønskede at tjekke om konceptet var solidt og noget man kunne arbejde videre på.

Første test af simpel prototype

- Konceptet er godt
- Kunne ikke rigtigt lave noget
- Graffiken skal op+
- Gutter der løber rundt på banen

Vi kunne bekræfte at konceptet var godt, det betyder at det er let at arbejde med og vi kunne forestille os et godt produkt som bygger videre på vores prototype. Det gode produkt er det som opfylder vores produkt krav.

Vi kunne ikke rigtigt lave noget i programmet, så vi skal tilføje flere features hvis vi ønsker at opnå vores vision med programmet.

Grafikken skal opgraderes betydeligt, en af vores krav var høj grafisk fidelitet og det er på ingen måde til stede i den nuværende tilstand.

Næste build af programmet
I det næste build af vores program, så skal vi havde implementeret forskellige typer af "fjender" og forskellige typer af tårne. Dette vil være en mere præcis repræsentation af vores endelige program, da det endelige program også kommer til at indeholde mulighed for at opgradere ens tårne og forskellige typer af tårne. Det endelige program kommer også til at indeholde forskellige typer af fjender som har forskellige egenskaber.

Der er tre hovedsagelige features der skal implementeres.

Den første er man kan opgradere et tårn. Det betyder at når man har nok af en ressource så kan man opgradere et tårn, så det for nye egenskaber. Det kunne fx være at det skyder hurtigere eller skyder bedre skud.

Den anden feature er et andet type af tårn.
Det kunne fx være et tårn som gør “fjenderne” langsommere, ved måske at skyde lim på det område som de bevæger sig hen over. Denne feature har vi implementeret ved at anvende Extend funktionen i processing. Denne funktion videregiver informationen fra en eksisterende klasse til en ny klasse.

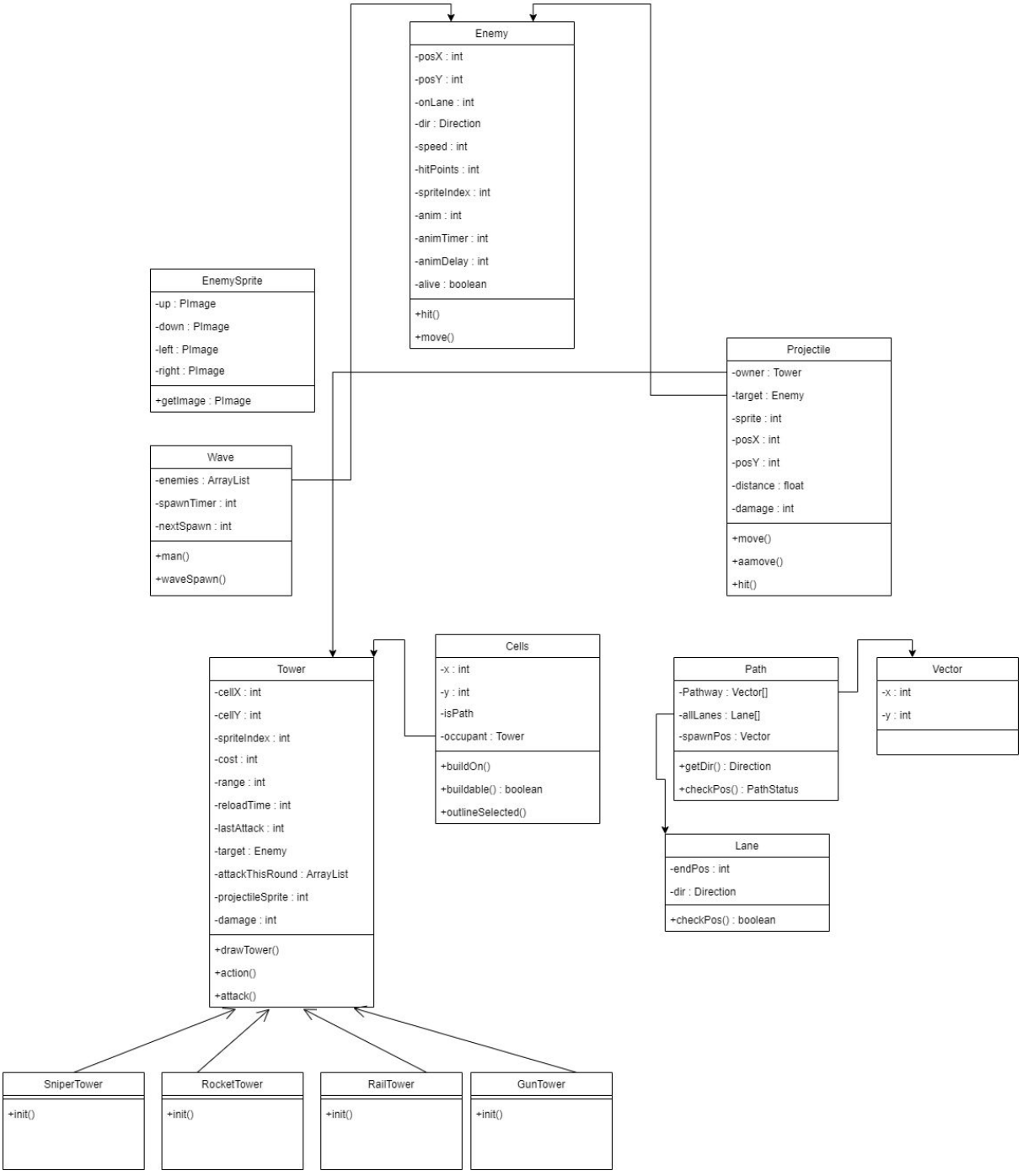
```
class RocketTower extends Tower {

    void init() {
    type = TowerTypes.rocketTower;
    spriteIndex = 0;
    range = 200;
    reloadTime = 1000;
    cost = 25;
    damage = 20;
  }

  RocketTower(int x, int y) {
    cellX = x;
    cellY = y;
    init();
  }

  RocketTower() {
    init();
  }
}

class GunTower extends Tower {

    void init() {
    type = TowerTypes.rocketTower;
    spriteIndex = 1;
    range = 400;
    reloadTime = 300;
    cost = 50;
    damage = 10;
  }

  GunTower(int x, int y) {
    cellX = x;
    cellY = y;
    init();
  }

  GunTower() {
    init();
  }
}
```

Figur 6. På billedet kan ses et stykke af kode fra RocketTower klassen i en tidlig iteration af programmet.

Her kan man se, at vi “Extender” vores Tower klasse ned til andre klasser. Tower er vores superklasse som vores andre klasser arver deres egenskaber fra. Det nye type af tårn er et raketårn ved navn “RocketTower”, det som adskiller dette tårn fra det andet, er at dette tårn skyder langsommere da den har en højere “reloadTime”, den er 1000 i stedet for 300. Den koster det halve, da dens “cost” variabel er 25 i stedet for 50. Damage er dobbelt så højt og rækkeviden er halv. Da “damage” er 20 i stedet for 10 og “range” er 200 i stedet for 400.

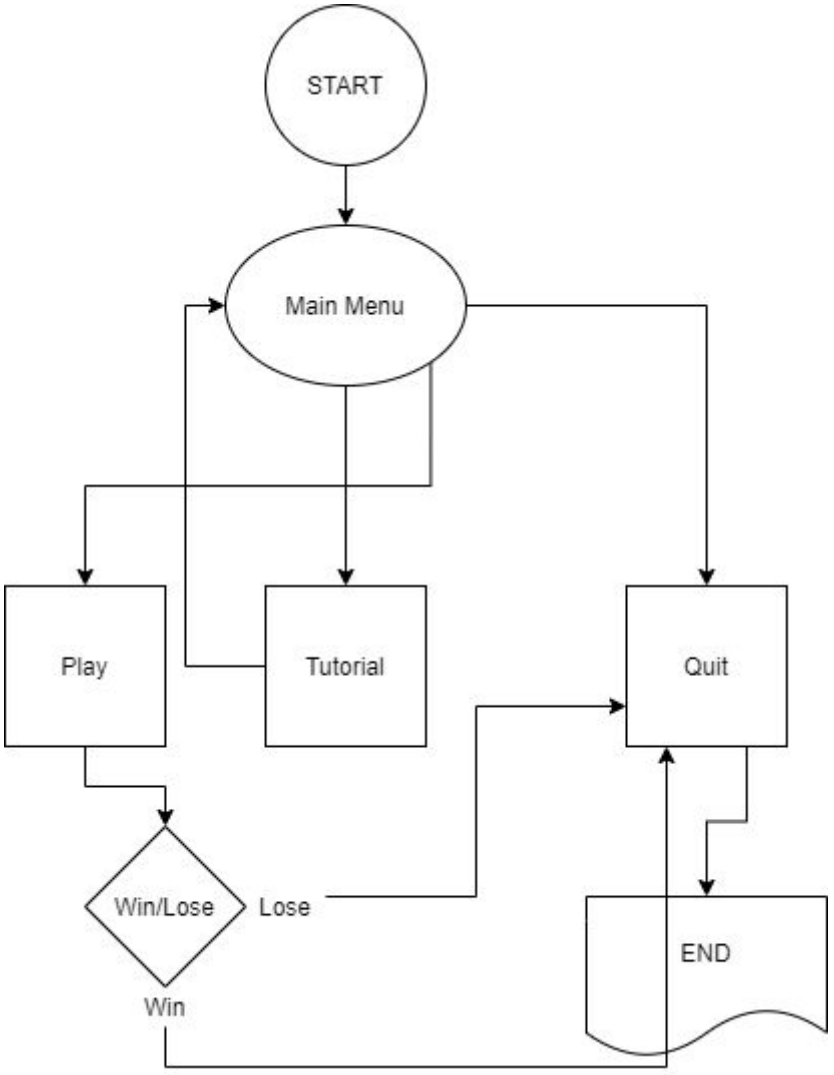
Den tredje feature og den sidste feature vi ønsker i det næste build af vores program er forskellige typer af “fjender”.
I det spil vi basere vores spil på, så er der forskellige typer af “fjender”. Nogle af dem regenerere liv og andre kan kun skydes under bestemte betingelser. Så hvis vores vision skal blive tilfredsstillet, så skal vi udbygge forskellige typer af fjender.

Realisering



Figur 7. Figuren viser et klassediagram over alle programmets klasser

I klassediagrammet ses et overordnet syn på programmets klasser samt hvorledes de hænger sammen.



Figur 8: Flowchart over programmets overordnede struktur

Det overordnede flowchart viser programmets centrale struktur på en helt overordnet plan i forhold til den grundlæggende funktionalitet.

Teknisk Beskrivelse

Spillet er bygget op i forskellige stages. Stage er et integer variabel der holder styr på hvilken stage spillet er i. Der findes en stage tilsvarende til hver del af spillet, f.eks. main menu, selve spillet, tutorial samt når spilleren vinder eller taber spillet.

Stage -1 er vores main menu som er den første stage der bliver loadet når man starter spillet.

Ved at trykke Play kommer man ind i stage 0 som er selve spillet.

Stage -10 er vores tutorial til selve spillet.

Stage 10 er når spilleren vinder spillet.

Stage 20 er når spilleren taber spillet.

Grunden til det ikke hedder stage 1,2,3 er så der er plads til andre baner. Stage 2 kan så være en anden bane.

Grafikken

Spillet er foregår i Terminator universet. Jorden er blevet overtaget af Skynet og de resterende mennesker prøver at udrydde robotterne. Spilleren spiller som robotterne der skal forhindre at menneskene ødelægger deres base. Banen ligner derfor en gold jordklode. Ligesom i filmene er banen mørk og farverne nedtonede.

Visuelt bliver banen genereret ud fra nogle billedfiler i programmappen som bliver loadet i spillet gennem PImage variabler.

Figur 7. Billede et af banen som man spiller på.

Banen som den ser ud i spillet

Vi benytter 3 billedfiler til at generere banen. Den ene er den overordnede baggrund og den anden er stien:

Figur 8. Billedet viser det bagerste lag af vores spillebane

Baggrunden til banen

Figur 9. Billedet viser den tekstur som vi tegner vores sti med

Teksturen til stien (groundpath.jpg)

For at placere stien på banen som vist i spillet benytter vi map.png som er et sort/hvid billede af banen som vi forestiller os den. Ved at benytte funktionen groundpath.mask(pathMask) kan vi reducere groundpath.jpg til den del som vi vil bruge som banen ud fra vores skitse i map.png.

Figur 10. På billedet kan alfa masken til vores bane ses

Det gode ved at gøre det på denne måde er at alfa masken kan byttes og så har man en hel ny bane. Hvis man ikke kan lide hvordan banen ser ud kan de 2 billedfiler skiftes ud.

Grid

Bag den visuelle del af banen ligger et grid af cells som bestemmer hvor spilleren kan placere Towers. Alle cells bliver genereret gennem et dobbelt for loop i setup.

```
for (int x=0; x<Grid.length; x++)
{
  for (int y=0; y< Grid[0].length; y++)
  {
    Grid[x][y] = new Cell(x, y);
  }
}
```

Figur 11. På billedet kan et dobbelt for loop ses, som bliver brugt til at iterere igennem vores to dimensionelle array

Inde i vores for loop, så har vi en variabel ved navn x. Den stiger i værdi, hvis x er mindre end den første værdi i vores array. Så dens maks værdi kommer til at være 20. Det samme sker i det andet for loop, men med 2 ændringer. Variablen hedder y, da vi nu tager udgangspunkt i y aksen og vi tjekker om y er mindre end den anden værdi i vores array. Når vi anvender vores for loop, så tjekker den først en hen ad x aksen og så ned langs y aksen indtil den når bunden, derefter går den videre til hvor den næste celle skal være på x aksen og gør det samme, indtil der ikke er plads til flere celler på banen.

Hver cell objekt indeholder sin position samt information om hvorvidt den ligger på stien (og dermed ikke kan bygges på) og hvilket tower der på nuværende tidspunkt er placeret i den cell. De cells som er en del af stien bliver bestemt når vores Path genereres så bliver hele vores Grid gennemgået for cells der er en del af vores Path og deres isPath variabel bliver sat til true.

Før man kan bygge et tower i en cell bliver der kørt funktion buildable() som returner en boolean. Funktionen checker hvorvidt der et tower i den cell eller om den er en del af stien. Når spiller holder musen over en cell vil den blive outlinet i enten grøn eller rød alt efter hvilket boolean buildable() funktionen har returneret for den cell.

Towers

Når man forsøger at placere et tower i en cell hvor det er muligt bliver der checket selectedTower variablet som styrer hvilket tower man forsøger at bygge. Hvis man ikke endnu har klikket på et Tower i spillets UI er variablet sat til 0 og man kan ikke placere noget. Når man klikker på de tilsvarende ikoner for de fire Towers i spillet bliver selectedTower variablet ændret til et tal mellem 1-4. Derefter bliver der checket om spilleren har tilstrækkeligt Scrap til at bygge det tower de forsøger at bygge.

Den overordnede Tower class indeholder 4 subclasses der har forskellige karakteristika indenfor de centrale Tower variabler som cost, range, reloadTime, damage. Hvert Tower indeholder positionen af den cell de er placeret i, samt deres target som er det Enemy de sigter efter. Når funktionen action() bliver kørt bliver der først checket om vores Tower har mulighed for at udføre et attack(). Her tjekkes der den tid der er passeret siden sidste attack

(lastAttack) er over reloadTime og herefter om der er et target og om det target er inde for vores Towers range. Hvis alle if statements passeres bliver der kørt attack().

```
void action() {
  if (millis() < lastAttack + reloadTime) return; //still on cooldown

  if (target != null) { //check our current target
    if (getDistance(this, target) < range) { //attack them if we can
      attack(target);
      return;
    }
  }
```

Figur 12. Viser funktion action som kører hvis en fjende er indenfor rækkevidde og tårnet er klar til at skyde.

Attack() funktionen forudsætter at Toweret har et target som den kan skyde herefter genereres et projectile fra Towerets position med samme target som vores Tower

Projectiles

Projectiles bliver genereret af vores Towers. Når et projectile har forladt vores Tower og bevæger sig mod sit target bliver der først tjekket om det target er i live og derefter bliver der udregnet den vinkel der findes mellem projectilen og det enemy den sigter mod. Herefter bliver den rykket med med en bestemt afstand i den retning. Da vores Enemy bevæger sig samtidig med projektilet skal der konstant udregnes en ny vinkel men afstanden den bevæger sig vil altid være det samme da vi benytte sinus og cosinus relationer i en retvinklet trekant således at projectilen bevæger sig hen ad den hypotetiske hypotenuse som altid har en bestemt længde.

```
void aamove() {
  distance -= 2;
  float tX = target.posX + cellSize/2;
  float tY = target.posY + cellSize/2;
  float dx = tX - posX;
  float dy = tY - posY;
  float angle1 = atan2(dy, dx);
  posX = round(tX - (cos(angle1) * distance));
  posY = round(tY - (sin(angle1) * distance));

  image(ProjectileSprites[sprite], posX, posY, cellSize/2, cellSize/2);
  if (distance < 0) hit();
}
```

Figur 13. Viser funktionen der rykker projektilerne mod fjenden de er skudt efter.

Når afstanden mellem vores projectile og vores target er under 0 bliver funktion hit() kørt som påvirker vores Enemy.

Når hit bliver kørt bliver projectilen fjernet og vores target bliver ramt.

Enemy

Der er 8 forskellige images for vores enemies. 2 images for hver retning som vores enemy kan bevæge sig i(venstre, højre, op, ned). Når et enemy bevæger sig en retning bliver der skiftet mellem de to forskellige sprites for at emulere bevægelse. Transitionen mellem de to forskellige sprites kører på en simpel animTimer

Figur 14. På billedet kan ses de to billeder som bruger til at animere personen som går

```
animTimer++;
if(animTimer > animDelay)
{
  animTimer = 0;
  if (anim == 0) anim = 1;
  else anim = 0;
}
```

Figur 15. Viser timeren der styrer hvilket billede der skal vises på skærmen.

Som jævnligt skifter animation ud fra variablet animDelay.

Enemies bliver genereret i waves som bevæger sig hen ad den definerede Path. Hver enemy har et antal hitpoints som bliver reduceret når funktion hit() bliver kørt hvilket sker under kollision med et projectile.

Waves

onWave variablet bestemmer hvilken wave spillet er i gang med på nuværende tidspunkt. Alle waves bliver initieret i starten af spillet med antallet af enemies og af hvilken type der skal genereres i den specifikke wave.

```
void levelInit() {
  ArrayList<Enemy> wave1 = new ArrayList<Enemy>();
  for (int i = 0; i < 5; i++) wave1.add(new Enemy(0, 5));
  ArrayList<Enemy> wave2 = new ArrayList<Enemy>();
  for (int i = 0; i < 10; i++) wave2.add(new Enemy(0, 5));
  ArrayList<Enemy> wave3 = new ArrayList<Enemy>();
  for (int i = 0; i < 3; i++) wave3.add(new Enemy(1, 15));
  ArrayList<Enemy> wave4 = new ArrayList<Enemy>();
  for (int i = 0; i < 20; i++) wave4.add(new Enemy(0, 5));
  ArrayList<Enemy> wave5 = new ArrayList<Enemy>();
  for (int i = 0; i < 10; i++) wave5.add(new Enemy(1, 15));
  ArrayList<Enemy> wave6 = new ArrayList<Enemy>();
  for (int i = 0; i < 0; i++) wave6.add(new Enemy(0, 5));
```

Figur 16. Viser listen over waves og hvilken fjende der skal sendes og hvor meget liv fjenden har.

De waves bliver placeret i en ArrayList over alle vores waves allWaves.

```
allWaves = new ArrayList<Wave>();
allWaves.add(new Wave(wave1, 1000));
allWaves.add(new Wave(wave2, 500));
allWaves.add(new Wave(wave3, 500));
allWaves.add(new Wave(wave4, 500));
allWaves.add(new Wave(wave5, 500));
allWaves.add(new Wave(wave6, 5));
```

Figur 17. Viser listen over waves og hvor lang tid der er til den wave starter.

De andet variabel i vores constructor for vores wave bestemmer waveDelay som er den tid der går mellem afslutningen af den forrige wave og starten den næste wave. Den første wave tager længere tid at starte så spilleren har tid til at opbygge sit initielle forsvar.

Spillets UI viser hvilken wave man er på under variablet waveInfo. Hvis boolean noWave er true viser den i stedet tiden der er til næste wave starter.

Funktionen startWave() kører for at starte den næste wave og bestemme hvilket waveInfo der skal vises på skærmen

```
void startWave() {
  if (millis() > startWave) {
    if (!endGame) {
      noWave = false;
      waveInfo = "Wave " + (onWave + 1);
    } else {
      if (AllEnemies.size() < 1) println(stage);
      win();
    }
  } else {
    waveInfo = "Wave " + (onWave + 1) + " in:  " + round( (startWave - millis()) / 1000) + " sec";
  }
}
```

Figur 18. Viser funktionen der starter en wave og når der ikke er flere fjender tilbage vinder spilleren.

Den tjekker også om spillet er slut fordi spilleren har vundet. Her tjekker den ArrayListen AllEnemies som er en liste over alle enemies. Når et enemy mister sine hitpoints og ”dør” bliver det fjernet fra arraylisten. Det samme sker hvis et enemy kommer igennem banen og skader basen. Derfor kan vi benytte størrelsen af arraylisten til at bestemme om spilleren har

vundet da den kun når under 1 hvis alle enemies er elimineret og spilleren har overlevet uden at lose() funktionen bliver kørt. Lose() bliver kørt i tilfælde hvor baseLives er 0 eller lavere.

Mouse stuff

For simple ting som knapper i menuer bliver der benyttet musens præcise position og tjekket for om den ligger inde for forskellige områder på y-aksen og x-aksen således at den er placeret over en bestemt knap. Når musen bliver trykket bliver knappens funktion kørt forudsat af at musen er over en knap på det tidspunkt. De områder der tjekkes afhænger af den stage man er på således at der ikke kan trykkes når man er i stages der ikke indeholder de knapper.

Når man er selve spillet skal spillet vide hvilken grid cell man holder musen over på alle tidspunkter således at der kan genereres en outline til den cell. Dette gør den ved at finde musens position og dividere med størrelsen på en celle således at man kan afrunde det til en præcis celle som musen er over. Derfor bliver der brugt int i stedet for float da man ikke er interesseret hvor tæt musen er på en grid cell men nærmere den præcise celle som musen visuelt befinder sig over.

```
void mouseCheck()
{
  int x = (int)(mouseX/ cellSize);
  int y = (int)(mouseY/ cellSize);

  if (x<Grid.length && y<Grid[0].length)
  {
    hoverCell = Grid[x][y];
    hoverCell.outlineSelected();
  }
}
```

Figur 19. Viser funktionen der tjekker hvilken celle musen er i.

Brugertest

Når man laver sin brugertest, så er der en masse teori man skal havde på plads og meget man skal gøre parat på forhånd, Man skal vælge den rette bruger man tester på, man skal tænke på hvilken effekt omgivelserne har på brugerens oplevelse. Man skal også tænke på, hvilke ting brugeren skal teste for en og om de må modtage hjælp under testen eller om de skal udføre testen selvstændigt. Nogle brugere ønsker også at være anonyme, så man skal også tænke på den effekt, det ville havde på resultaterne om de er anonyme eller kendte. Siden vi udføre de her test digitalt, så skal vi tænke på, hvilken effekt det har på dem at vi ikke er fysisk tilstede under prøven.

Udover dette her, så mener vi også, at det er vigtigt brugeren forstår, hvad det er resultaterne af prøven bliver anvendt til. Dette her er vigtigt, for at brugeren føler sig tryg, når de udfører brugertesten, og så de ved hvad de deltager i.

- Man kan afbryde undersøgelsen, hvis man gerne vil.
- Den bliver brugt til, at fejl undersøge, hvordan vi kan forbedre vores program.
- Vi nedskriver alle dine handlinger på skrift
- Du vil være anonym i vores rapport og du behøver ikke angive dit navn.
- Du skal angive din erfaring med processing eller andre java baseret sprog.

Det sidste vi siger til dem er “Du skal angive din erfaring med processing eller andre java baseret sprog”. Dette her er vigtigt, da hvis man har arbejdet med i Java eller Processing, så kan det være lettere at forstå hvordan spillet skal startes og sluttes. Vi har nemlig ikke en dedikeret start og stop knap, da vi bare bruger processings “run” og “stop” knap til det. Fra tidligere brugertest, så har vi kunnet se, at det har været svært for folk at starte og stoppe spillet på denne måde, hvis man ikke var erfaren med kode editoren.

Brugertype

Vi ønsker at appellere til folk som har spillet Tower Defense spil før. Det mest populære spil er jo Bloons Tower Defense 6, så vi vil eftersøge brugere som har erfaring med dette her spil. Udover dette, så differentiere vi os selv fra Bloons Tower Defense 6, med et andet mere seriøst/modent tema, så vi skal også spørge nogle af dem vi håber dette tema appellere til, for at se om temaet er godt.

Grundet Corona Virussen, så kan det være svært at finde den helt rigtige bruger, da vi ikke kan opsøge dem fysisk, som var vores originale plan. Derfor så må vi prøve at finde dem digitalt i stedet for. Dette gør vi ved, at spørge klassekammerater, om hvorvidt de har erfaring med spillet og ud fra deres personlighed, så vil vi vurdere om de ønsker det andet tema.

Hjælp

Vi ønsker at være til rådighed, hvis testpersonen ønsker hjælp. Vi vil dog ønske at brugeren udføre, det meste af testen på egen hånd. Vi siger dog derfor før testen starter, “Vi er til rådighed, hvis du har brug for hjælp med testen. Vi ønsker dog, at du prøver at gøre det du har svært ved før, du henvender dig til os”. Vi ønsker de ikke modtager hjælp fra os, da en bruger som anvender vores spil, vil ikke kunne spørge os personligt om hjælp, men kun ville kunne anvende de hjælpesystemer som er til stede i spillet på forhånd. På den måde, så bliver testen mere autentisk til en normal brugssituation. Vi ønsker den mere autentiske situation, da det er den situation som vores projekt kommer til at blive bedømt efter.

Omgivelserne

Vi har svært ved at kontrollere de fysiske omgivelser, da vi ikke kan møde fysisk og udføre bruger testen grundet Corona Virussen, men vi kan kontrollere de digitale omgivelser. Vi kan vælge kommunikationsplatformen, om det er tekst eller stemmebaseret. Hvis det er tekstbaseret, så tager det testpersonen længere tid at sende en besked, hvilket giver dem mulighed for at have et mere velovervejet svar. Det bliver dog ikke en normal samtale man har om spillet, da en verbal samtale foregår i et andet tempo og har derfor andre resultater. Vi skal også huske på, at der kan være besvær med at anvende de digitale kommunikationsplatforme for brugeren, så vi skal også vægte ind i vores vurdering af testen eventuelle frustrationer, med anvendelse af kommunikationsplatformen. Det kan derfor være vigtigt, at vælge en bruger som er bekendt, med hvordan man anvender denne form for kommunikationsplatform.

Vi ønsker at anvende discord til at kommunikere online med vores brugere. Vi valgte det her pga, vi mener det har høj lydkvalitet efter personlig brug af discord. Det har også muligheden for at dele filer, som er nødvendigt for at vi kan teste programmet. Da vi skal sende dem filerne. Der er også mulighed for at dele sin skærm i discord, så hvis brugeren har et

spørgsmål om spillet, som de lettere kan illustrere hvis vi kan se hvad de laver, så kan de gør det. De kan også dele deres skærm under hele testen, så vi kan se hvad de laver på deres computer hele tiden. Der er også mulighed, for at bruge et facecam hvis man har det, så vi kan se om de har svært ved at bruge controls i spillet på deres tastatur.

Anonymitet

Som tidligere sagt, så ønsker nogle brugere at være anonyme af personlige årsager. Dette tillader brugeren at give forskellige typer af kritik, som måske ville havde været ignoreret pga identitet. Det samme kan betyde, at de kan give positiv kritik som bliver accepteret på trods af identitet, som måske ville havde gjort at man ville se bort fra det. Et eksempel på, hvordan man fravælger kritik pga identitet kan man fx se i mange konkurrencedrevne spil.

I konkurrencedrevne spil justere man ofte, på forskellige værdier for at sørge for at den konkurrencedrevne oplevelse er så fair som muligt. Spillet bliver dog spillet forskelligt på forskellige niveauer, så mange højt rangerende brugere fravælger kritik fra lavt rangerende brugere, pga de mener de lavt rangerende brugere ikke forstår spillet. Hvis alle feedbacken er anonyme, så forarbejder man dette fænomen, så flere feedback punkter bliver taget seriøst. På samme måde kan stykker af feedback vi ikke kender endnu, risikere at blive frasorteret pga en overbevisning man har om en identitet, man måske ikke forstår helt.

Triangulering

Man kan prøve at ændre omgivelserne eller andre parametre fra test til test, for at se om man får andre resultater. Man kan fx, vælge ikke at hjælpe brugeren overhovedet eller være overbærnde med hjælpen. På den måde så kan man teste forskellige ting og se hvad der fungere bedst.

Testen

Vi testede 3 personer hvor vi gav linket til vores github repository så de kunne hente spillet. Her bad vi dem om at skrive ned hvad de tænker kunne ændres eller forbedres. Vi ville gerne teste spillets værdier som for eksempel hvor meget tingene kostede og hvor svære bølgerne var.

Testperson 1

Den første person vi testede ville gerne have at vi tilføjede flere waves/bølger. De mente spillet var for kort. De ville også gerne have at vi tilføjede flere tårne/towers. Musikken gentog hellere ikke sig selv, så når man var halvvejs stoppede musikken. Der kom ikke forskellige fjender på samme wave.

Hvad vi ændrede på.
Vi tilføjede 5 nye bølger så der nu var 15 i det hele. Vi tilføjede ikke flere tårne da vi mente 4 var godt nok i starten. For at få musikken til at gentage sig selv ændrede vi koden for musikken fra music.play() til music.loop(). Vi ændrede wave strukturen så nu kommer der forskellige fjender på samme wave.

```
ArrayList<Enemy> wave8 = new ArrayList<Enemy>();
for (int i = 0; i < 15; i++) wave8.add(new Enemy(1, 20));
for (int i = 0; i < 5; i++) wave8.add(new Enemy(0, 10));
```

Figur 20. Viser wave 8 der har 2 forskellige fjender i.

Her ses wave 8 hvor der kommer 15 type 1(tanks) fjender og derefter kommer 5 type 0(Soldater).

Testperson 2
Den anden person vi testede på så at life og scrap UI elementerne ramte ind i hinanden nå spilleren havde optjent meget scrap. Vi havde lavet en stavefejl “Humans soldiers” skulle ændres til “Human soldiers”. Testpersonen syntes også at det var svært at læse teksten på tutorial siden og at spillet blev for nemt til sidst. Det sidste tårn kostede 150 når der stod at den skulle koste 200.

Hvad vi ændrede på.
Vi rykkede life elementet mere til højre så det ikke ramte ind i scrap. Vi ændrede stave fejlen og ændrede på baggrundsfarven fra 150 til 100 så den var mørkere på tutorial skærmen. Vi fiksede prisen så nu kostede tårnet 200 scrap.

Testperson 3

Den tredje person blev bedt om at kigge spillet igennem for stavefejl. Her ændrede vi på forskellige ting for eksempel ændre vi too til to og five til four. Personen mente også at man fik for meget scrap når man dræbte fjenderne.

Hvad vi ændrede på.
For at balancere spillet mere fik man kun 2.5 scrap når man dræbte fjenderne i stedet for 5.
For ikke at gøre starten for svær starter man nu med 125 scrap i stedet for 75.

Konklusion

Vores mål var, at vi skulle producere et spil med underholdningsværdi. Vi mener vi har løst casen, da vi opfylder mange af vores produktkrav. Vi har en flad indlærings kurve, idet spillet starter let ud og bliver progressivt sværere. Vi havde forskellige typer af tårne, som diversificerede spille oplevelsen. Vi formåede også at differentiere os selv stilistisk fra Bloons Tower Defense 6, så vi kunne skabe en mere unik oplevelse for brugerne af dette spil. Vi valgte at appellere til “casual gamers”, da vi kunne læse os til fra anbefalingerne, at det typisk set var folk som spillet Bloons Tower Defense 6 sporadisk og med en lille tidsinvestering. Vi kunne også se på tagsne tilknyttet spillet på steam, at det var en “casual” oplevelse at spille, derfor måtte spillet også være for “casual gamers”. Udover at det teoretisk set appellere til “casuals”, så kunne vi også bekræfte ud fra vores brugertest, at det havde opfyldt de produktkrav vi havde stillet til spillet. Vi mener derfor, at projektet har været en succes.
Siden at mængden af forskellige tårne i Bloons Tower Defense 6, er blevet krediteret i anbefalingerne af spillet, som at være en positiv kvalitet. Så bør vi øge mængden af tårne og, gøre så tårnene kan opgraderes, så spiloplevelsen bliver endnu mere divers. Man kan også opgradere den grafiske fidelitet og adaptere soundtracket, så det looper bedre.
Konceptet bag spillet lader dog til at være stærkt, derfor behøver vi ikke nogle konceptuelle ændringer, vi skal bare udvide på det nuværende produkt. Man kan også tilføje flere baner, som måske indeholder nye typer af terræn, så der er nye udfordringer og mere reel progression, på den måde holder vi vores brugere interesseret i længere tid. Til sidst ville vi nok tilføje, nye former af fjender som har nye egenskaber som de andre ikke har. Det kunne fx være camouflage eller regeneration af livspoint.
Grundet corona pandemien, så løb vi ind i nogle organisatoriske problemer, vores gruppe kommunikation ændrede sig radikalt, til at være hovedsageligt fysisk til at være digitalt. Dette var en stor ændring, da vi ikke længere kunne være i samme lokale, hvilket ændre på den

sociale dynamik. Udover dette, så skulle vi også finde en ny måde at udføre brugertest på, dette indeholdt nye udfordringer, da vi aldrig har udført en over digitale platforme og det var nye forhold vi skulle indregne i vores evaluering af testene.

https://drive.google.com/file/d/1XpzwMG9ntJsQjfpj_4oXalth8ilUf7we/view?usp=sharing

Bilag

Kildeliste

Kunstner og titel: Det er titel løst og uden en enkel kunstner. Det er en opsummering af alle anbefalingerne på metacritic af Europa Universalis IV.
Link: https://www.metacritic.com/game/pc/europa-universalis-iv
Vi stoler på denne her kilde, da metacritic citere mange anbefalinger fra officielle kritikere, såsom PC Gamer og IGN. Det eneste man kan stille spørgsmål til, er bruger anbefalingerne, men disse er kohærente med de professionelle anbefalinger, så vi stoler også på disse.

Kilden er uden kunstner og titel og er butikssiden for Europa Universalis IV på steam. Den indeholder både tags for spillet og en opsummering af de forskellige anbefalinger givet af brugerne af spillet.
Link: https://store.steampowered.com/app/236850/Europa_Universalis_IV/
Vi stoler på denne her kilde, da steam er en af de mest respekterede spilplatforme på nettet.

Kilden er uden kunstner og titel og er butikssiden for Bloons Tower Defense 6 på steam. Den indeholder både tags for spillet og en opsummering af de forskellige anbefalinger givet af brugerne af spillet.
Link: https://store.steampowered.com/app/960090/Bloons_TD_6/#app_reviews_hash
Vi stoler på denne her kilde, da steam er en af de mest respekterede spilplatforme på nettet.

Kunstner og titel: Det er titel løst og uden en enkel kunstner. Det er en opsummering af alle anbefalingerne på metacritic af Portal 2.
Link: https://www.metacritic.com/game/pc/portal-2
Vi stoler på denne her kilde, da metacritic citere mange anbefalinger fra officielle kritikere, såsom Euro Gamer og IGN. Det eneste man kan stille spørgsmål til, er bruger anbefalingerne, men disse er kohærente med de professionelle anbefalinger, så vi stoler også på disse.

Kilden er uden kunstner og titel og er butikssiden for Bloons Tower Defense 6 på steam. Den indeholder både tags for spillet og en opsummering af de forskellige anbefalinger givet af brugerne af spillet.
Link: https://store.steampowered.com/app/620/Portal_2/
Vi stoler på denne her kilde, da steam er en af de mest respekterede spilplatforme på nettet.

Kilden er udarbejdet af Nathan Lovato, som er en spil designs ekspert og grundlægger af GDquest. Artiklen er udgivet på Gameanlytics.com. Kilden hedder “3 Simple Steps To Improve Your Game’s Graphics”
Link: https://gameanalytics.com/blog/3-steps-to-improve-your-games-graphics.html

Vi mener denne her kilde er troværdig, da den er udviklet af en som har erfaring med at udvikle spil og grafik.

Kilden kommer fra Systime og har titlen “Gestaltlovene”. Den er kernestof for kommunikations/it på b niveau.
Link: https://informatik.systime.dk/?id=1132&L=0
Vi stoler på denne her kilde da den er kernestof, hvilket gør den ekstrem troværdig.

Kilden kommer fra criticalhit.net og er skrevet af Jayson van Kerckhoven.
Link: https://www.criticalhit.net/gaming/casual-vs-competitive-gaming-differences/
Kilden kommer ikke fra en højt respekteret udgiver, men den giver et nuanceret indblik i casual og competitive spillere mening og definition. Udover det, så citere den også spillere de har adspurgt, hvilket giver dem belæg, derfor mener vi den er troværdig.

Kilden kommer fra Computerhope.com og er udarbejdet af en ukendt kunstner. Den er senest opdateret den 13 maj 2019
Link: https://www.computerhope.com/jargon/c/casual-gaming.htm
Kilden mener vi er troværdig fordi, at den gentager hovedsageligt det som vi kan læse criticalhit.net kilden.

Kilden kommer fra Polygon.com og er udarbejdet af Ryan Gilliam. Den er senest opdateret 2. November 2019.
Link: https://www.polygon.com/2019/11/2/20944794/blizzard-level-squish-50-60-120-shadowlands-cap-leveling-experience-new-player

Kilden mener vi er troværdig, da den er udgivet af Polygon som har udgivet nyheder i meget langt tid og de citere direkte udviklerne af World of Warcraft, hvilket gør dem mere troværdige.

Kilden kommer fra Makinggames.biz og er udarbejdet af Mark Robnison. Den er senest opdateret på et ukendt tidspunkt, men inden for de seneste par år da den referere til nye spil som Hearthstone
Link: https://www.makinggames.biz/feature/essential-retention-mechanics-to-keep-players-engaged8091.html
Udgiveren er partnere af Game Developement conference som er respekteret, derfor tror vi at denne her kilde er troværdig. Udover dette, så laver de også argumenter understøttet med gode eksempler, hvilket gør kilden mere troværdig.

Kilden er udgivet på Wikipedia.com og er udarbejdet af en ukendt kunstner.
Link: https://simple.wikipedia.org/wiki/Bloons_Tower_Defense
Normalt ville vi ikke mene wikipedia er troværdigt, da der er en stor risiko for at nogle af detaljerne er ukorrekte. De giver dog et resume af spillet, hvilket ikke indeholder nogle kontroverse meninger og er meget overordnet.

Kilden er udgivet på theverge.com og er skrevet af Nick Statt. Den blev senest opdateret 30. November 2018
Link: https://www.theverge.com/2018/11/30/18120577/valve-steam-game-marketplace-revenue-split-new-rules-competition
The Verge er en udgiver som har meget erfaring med at skrive artikler om teknologi generelt. De har også skrevet meget om computerspil og er respekteret som en god kilde. Derfor mener vi at denne her kilde er troværdig.

Kilden er en artikel som er udgivet på gamasutra.com og er skrevet af Jake Birkett, den blev sidst opdateret den 5. April 2018
Link: https://www.gamasutra.com/blogs/JakeBirkett/20180504/317366/Using_Steam_reviews_to_estimate_sales.php
Denne her artikel er ikke udgivet af en velkendt udgiver, den er dog skrevet af en spiludvikler som laver indie spil og som ejer Grey Alien Games. Han anvender data drevne artikler, hvilket gør artiklen troværdig.

Kilden er en artikel som er udgivet på advanz.dk, den har ikke en specifik forfatter. Den har heller ikke en dato for seneste opdatering.
Link: https://advanz.dk/blog/hvad-er-scrum-metode/
Vi mener denne her kilde er pålidelig, da den stemmer i stor grad overens med det vi allerede har lært om scrum. Derfor tror vi, at de udvidelser de har lavet på, må være velovervejede.

Figurliste

Figur 2

Figur 3


BLOONS TD 6
Fedt tårn-forsvars spil
SENESTE ANMELDELSER: Overvældende positive (1,779)
ALLE ANMELDELSER: Overvældende positive (22,033)
UDGIVELSESDATO: 18. dec. 2018
UDVIKLER: Ninja Kiwi
UDGIVER: Ninja Kiwi
Populære brugerdefinerede tags for dette produkt:
Strategi
Tårnforsvar
Singleplayer
Co-op
3D
+

Figur 4


Populære brugerdefinerede tags for dette produkt: (?)
Strategi
Tårnforsvar
Singleplayer
Co-op
Multiplayer
Online co-op
Casual
Familievenligt
3D
Sandkasse
Tegnefilmsagtigt
Sjovt
Svært
Action
Overlevelse
2D
Godt soundtrack
Indie
Seksuelt indhold
Sødt

Figur 5

Figur 6

```
class RocketTower extends Tower {

    void init() {
    type = TowerTypes.rocketTower;
    spriteIndex = 0;
    range = 200;
    reloadTime = 1000;
    cost = 25;
    damage = 20;
  }

  RocketTower(int x, int y) {
    cellX = x;
    cellY = y;
    init();
  }

  RocketTower() {
    init();
  }
}

class GunTower extends Tower {

    void init() {
    type = TowerTypes.rocketTower;
    spriteIndex = 1;
    range = 400;
    reloadTime = 300;
    cost = 50;
    damage = 10;
  }

  GunTower(int x, int y) {
    cellX = x;
    cellY = y;
    init();
  }

  GunTower() {
    init();
  }
}
```

Figur 7

Figur 8

Figur 9

Figur 10

Figur 11

```
for (int x=0; x<Grid.length; x++)
{
  for (int y=0; y< Grid[0].length; y++)
  {
    Grid[x][y] = new Cell(x, y);
  }
}
```

Figur 12

```
void action() {
  if (millis() < lastAttack + reloadTime) return; //still on cooldown

  if (target != null) { //check our current target
    if (getDistance(this, target) < range) { //attack them if we can
      attack(target);
      return;
    }
  }
```

Figur 13

```
void aamove() {
  distance -= 2;
  float tX = target.posX + cellSize/2;
  float tY = target.posY + cellSize/2;
  float dx = tX - posX;
  float dy = tY - posY;
  float angle1 = atan2(dy, dx);
  posX = round(tX - (cos(angle1) * distance));
  posY = round(tY - (sin(angle1) * distance));

  image(ProjectileSprites[sprite], posX, posY, cellSize/2, cellSize/2);
  if (distance < 0) hit();
}
```

Figur 14

Figur 15

```
animTimer++;
if(animTimer > animDelay)
{
  animTimer = 0;
  if (anim == 0) anim = 1;
  else anim = 0;
}
```

Figur 16

```
void levelInit() {
  ArrayList<Enemy> wave1 = new ArrayList<Enemy>();
  for (int i = 0; i < 5; i++) wave1.add(new Enemy(0, 5));
  ArrayList<Enemy> wave2 = new ArrayList<Enemy>();
  for (int i = 0; i < 10; i++) wave2.add(new Enemy(0, 5));
  ArrayList<Enemy> wave3 = new ArrayList<Enemy>();
  for (int i = 0; i < 3; i++) wave3.add(new Enemy(1, 15));
  ArrayList<Enemy> wave4 = new ArrayList<Enemy>();
  for (int i = 0; i < 20; i++) wave4.add(new Enemy(0, 5));
  ArrayList<Enemy> wave5 = new ArrayList<Enemy>();
  for (int i = 0; i < 10; i++) wave5.add(new Enemy(1, 15));
  ArrayList<Enemy> wave6 = new ArrayList<Enemy>();
  for (int i = 0; i < 0; i++) wave6.add(new Enemy(0, 5));
```

Figur 17

```
allWaves = new ArrayList<Wave>();
allWaves.add(new Wave(wave1, 1000));
allWaves.add(new Wave(wave2, 500));
allWaves.add(new Wave(wave3, 500));
allWaves.add(new Wave(wave4, 500));
allWaves.add(new Wave(wave5, 500));
allWaves.add(new Wave(wave6, 5));
```

Figur 18

```
void startWave() {
  if (millis() > startWave) {
    if (!endGame) {
      noWave = false;
      waveInfo = "Wave " + (onWave + 1);
    } else {
      if (AllEnemies.size() < 1) println(stage);
      win();
    }
  } else {
    waveInfo = "Wave " + (onWave + 1) + " in:  " + round( (startWave - millis()) / 1000) + " sec";
  }
}
```

Figur 19

```
void mouseCheck()
{
  int x = (int)(mouseX/ cellSize);
  int y = (int)(mouseY/ cellSize);

  if (x<Grid.length && y<Grid[0].length)
  {
    hoverCell = Grid[x][y];
    hoverCell.outlineSelected();
  }
}
```

Figur 20

```
ArrayList<Enemy> wave8 = new ArrayList<Enemy>();
for (int i = 0; i < 15; i++) wave8.add(new Enemy(1, 20));
for (int i = 0; i < 5; i++) wave8.add(new Enemy(0, 10));
```

Figur 21

Logbog

16 - 25 April
Denne her periode, blev brugt på idegenerering og valg af emne hovedsageligt. Vi valgte på baggrund af de forskellige cases og mulighederne inden for de forskellige cases.

Dato 26 April - 2 Marts
I den her periode, så fandt vi ud af, hvad ambitionsniveauet skulle være for vores projekt

Dato 3 - 10 Marts
I den her periode, så havde vi fokus på, at specificere vores produkt krav, så vi kunne udvikle en simpel prototype af vores program, som kunne testes på brugere, for at se om konceptet var godt nok.

Dato 11 - 18 Marts
I den her periode så undersøger vi hvilken målgruppe det var vi skulle appellere til, udover dette så udviklede vi en mere avanceret prototype af vores program, som testede konceptet bedre.

Dato 19 - 26 Marts.
Vi undersøgte måder hvordan vi kunne differentiere os selv fra Bloons Tower Defense serien i den her periode. Udover dette, så fik vi en funktionel version af vores program i denne her periode.

Dato. 27 Marts - 3 Maj
I denne her tidsperiode, så tilføjede vi de sidste kerne features i vores program. Det bestod af, addering af den sidste type af tårn og den sidste type af fjende. Vi skrev også på rapporten her og undersøgte primært, hvad det var som gjorde Bloons Tower Defense 6 succesfuldt.

Dato. 4 - 11 Maj
I denne her tidsperiode, så arbejdede vi på grafikken i vores program. Vi skrev også på rapporten. Indholdet i rapporten havde fokus på gestaltlovene og stilistiske valg. Vi udførte brugertest, for at teste om grafikken passede til vores forventninger

Dato. 12-19 Maj
I denne her tidsperiode, så indgik vi vores sidste fase af projektet. Vi fokuserede på, at skrive på vores rapport og finpudse vores program. Vi tilføjede musik til spillet og flere bølger, udover dette så justerede vi også på værdier på vores tårne, så de magtforholdet mellem de 4 tårne var mere balanceret. På vores rapport fokuserede vi på den tekniske beskrivelse og formalier.

Tidsplan

Sprint	**16/3 - 29/3**	**30/3 - 12/4**	**13/4 - 26/4**	**27/4 - 12/5**
	Idegenerering i forhold til specfifikke produktkrav	**Udvikle produktet i processing (Opsætning af klasser og basale funktioner som bevægelse)**	**Begynd at lave mere komplekse funktioner, såsom kunstig intelligens og forskellige typer af tropper.**	**Arbejde ud fra det respons vi fik fra brugertestene og fjerne problemer i programmet aka bug fixing**
	Udarbejdelse af simple prototyper af vores produkt	**Brugertest**	**Brugertest af kunstig intelligens**	**Brugertest igen**
	Test af simpel prototype	**Evt. møde med produkt owner**	**Evt. møde med produkt owner**	**Evt. møde med produkt owner**