Introduction to Games Design DM6127: Game Design Document

Due on 14 November 2014

Group: 5, Dracolabs
Game: Freakolution

Bladin, Karl - N1408590D - N1408590D@e.ntu.edu.sg
Nilsson, Michael - N1408591A - N1408591A@e.ntu.edu.sg
Béraud, Sacha - G1402647C - sacha001@e.ntu.edu.sg
Mangs, Ludvig - N1408590D - N1408589A@e.ntu.edu.sg

Contents

1 Game Overview

Humans have conducted biological experiments on mystical creatures for decades and lost all respect for ethics and life that is different from their own. One group of scientists took this too far and during torturous like circumstances, four mutant creatures escaped their cages to get vengeance on every human being that has stood by and let this happen. Freakolution is a multiplayer wave survival game. The main focus of the game is the cooperation between the players. Each player gets to choose its own class and chemical which all have strengths and weaknesses. This means that a good mix is necessary to get far in the game.

Overall, the game can be considered a mix of a Hack'n'Slash (i.e. Diablo) and a Tower Defense of the maze type (where the path of monster changes over time). The game is a 2D and 3D hybrid and have a tilted top-down perspective and all players will share the same screen.

The target audience are the people who like to play together with other people, preferably in the same room. It is the perfect Friday night entertainment when friends get together after a long week of school or work. The game should not necessarily have to require much time, which makes it easier for a quick play. It is a game that only takes a couple of minutes to learn but a lifetime to master.

1.1 Interface

1.1.1 HUD

A small partition of the game screen will be reserved in each corner of the screen for displaying information about the players' health and number of enemies killed. The health will be displayed as a circle that will lose fill color when the player is hurt. The fill color will also gradually go from green to red when it becomes empty. This is to make the player aware of the situation before it is too late. The kill counter will also be a circle but with a number inside it. The fill color will be the same as the players chemical color. Other than this there will be a number in the center top of the screen which indicates the difficulty of the wave. The number will increment with the step of one each wave. The number of enemies left for the current wave is also shown here.

Each character and enemy also have a small health bar above their head so that the attention of the player does not have to leave the character. Above the health bar there will be a small shape that will vary depending on the players class.

- Circle: Healer
- Square: Tank
- Triangle: Marksman
- Upside down triangle: Engineer

The shape is black, and inside the shape the player number is shown so that it will be easier to distinguish between the players. See figure 1.

1.1.2 Menus

1.1.2.1 Main Menu

The main menu will consist of these buttons:

- Play
- Settings
- Exit

When navigating through the menu, the selected options will be highlighted. To move to another option the up or down key will have to be pressed and if the move is successful there will be a short sound effect for clarification. See figure 2.

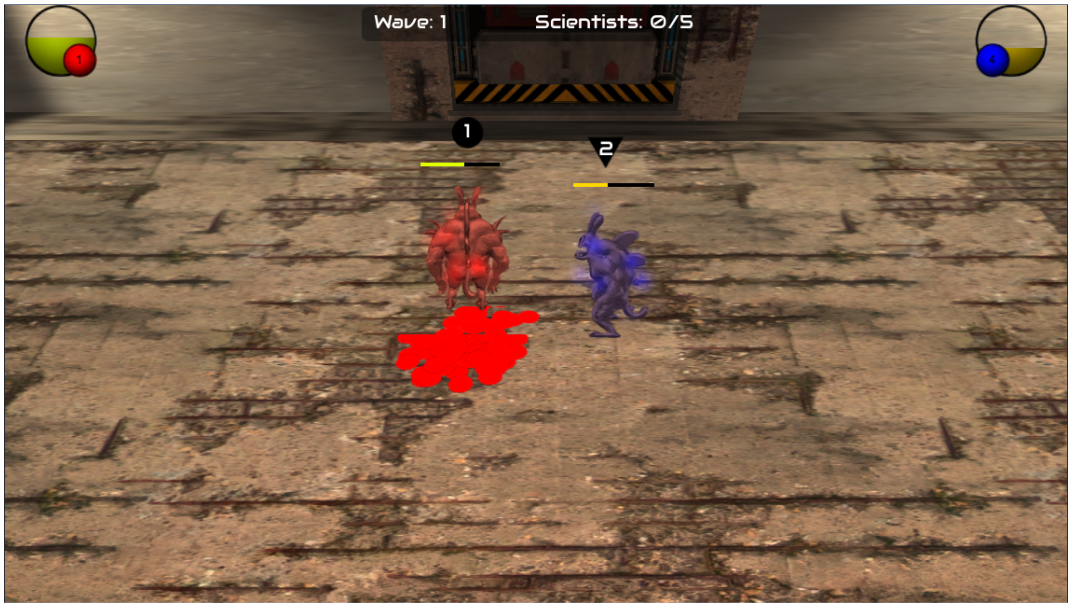

Figure 1: Character shapes and HUD.

- If the first option is selected and the up key is pressed, nothing will happen
- If the last option is selected and the down key is pressed, nothing will happen

When an option is selected and the X / Enter button is pressed, a short sound effect will be played and the screen will be replaced accordingly.

Figure 2: Freakolution main menu.

When Play is pressed, the players will be given a chance to choose what character to play, which class, and which chemical. See figure 3 After choosing the settings, and the Start button is pushed by all of the players, the game will start immediately. The Play menu is located to the right of the Main Menu, and will therefore slide in from that direction.

In the Settings menu, there is a couple of settings available

- Sound on/off (toggle when X/Enter is pressed)

Figure 3: Freakolution character selection menu.

- Music on/off (toggle when X/Enter is pressed)

The Settings menu is located to the left of the Main Menu, and will therefore slide in from that direction, see figure 4. If the Back button is pressed, the settings menu will slide to the left, out of the screen and the Main menu will slide in from the right.

Figure 4: Freakolution settings menu.

1.1.2.2 Pause Menu

If the ESC/Start button is pressed in game, the game freezes and the center partition of the screen will be overlapped with the Pause menu. The Pause menu will have two buttons:

- Resume
- Exit to main menu

In the background, the game will be darkened to clarify that the game is paused. Any player will be able to use the Pause Menu regardless of who pressed the ESC/Start button. The Selected option will be displayed in white text, and the other button will be darker to indicate that it is not chosen, and green to indicate that it is selectable.

1.1.2.3 Highscore board

When the game is over, the scoreboard will show together with a "Game Over" title. See figure 5. The scoreboard has a sqaure with statistics for each player that participated in the game. The square has the same color as the players chosen chemicals. Statistics that are available for the player are the following:

- Number of kills
- Damage done
- Damage taken
- Number of barrels built
- Number of barrels moved

Figure 5: Freakolution game over screen.

2 Gameplay and Mechanics

Freakolution is a cooperative multiplayer survival game. It allows up to 4 players to work together in order to survive waves of enemies in form of humans of increasing numbers.

2.1 Physics

2.1.1 Dimensions

The physics of the game are in 3D as well as most of the objects in the scene. Although most of the player's interactions with the environment are restricted to 2D, since the characters only walk on the ground, there are events happening in all 3 dimensions (particle effects, and range attacks).

2.1.2 Camera

The camera view is very similar to an RTS (real-time strategy) game: it shows a plane on which players can move from a high angle. Player's height should be not more than a 5th of the size of the screen. The in-game camera always shows all the players on the screen, it moves to always be as close to the players as possible but still manages to keep all players on screen.

2.1.3 Collisions

Players, static objects (i.e. barrels, walls), and monsters all have proper 'colliders' and always collide with each other. They never go through each other.

2.1.4 Gravity

The game has typical gravity: a player going off an edge will fall, objects and characters are attracted to the ground.

2.2 User input

Under gameplay, the input from the users comes from game controllers. See figure 6 and table 1.

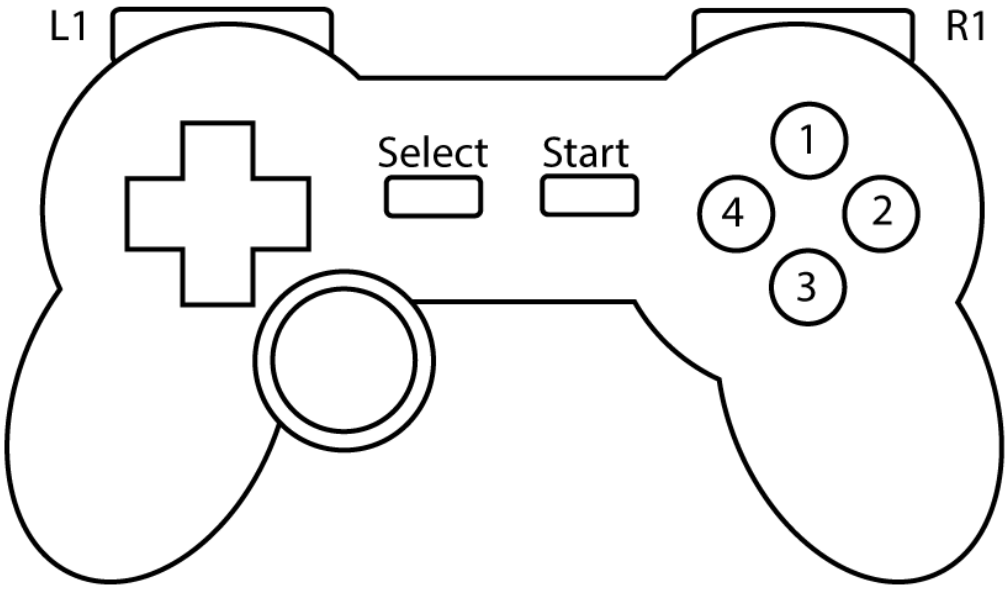

Figure 6: Typical game controller.

Table 1: In game button inputs

Buttons	Function
Button 1	Puke with cooldown varying per class. Healer has healing puke.
Button 2	Pick up / put down barrel
Button 3	Create barrel
Button 4	Melee attack

2.3 Movement

2.3.1 Players

The game is played using a typical gaming controller The players are able to move up, down, left and right on the screen, as well as all the respective diagonals.

2.3.2 Enemies

Enemies move according to a grid used by the A* algorithm to find the shortest path to the player it is currently chasing.

2.3.3 Environment

Environment will mostly be static, although some objects (barrels) can be moved and created by the players in order to take advantage of the enemies' restricted movements.

2.3.4 Projectiles

The range attacks of the players is puking. Players can hit and damage several enemies by doing this. The damage depends on the class of the player and also the chemical of the enemy and the player attacking.

2.4 Objects

Players are able to carry barrels around using the assigned button. The action is very similar to Half-life's gravity gun: the player use the action on a barrel which then floats in front of their character, and can then move around with it and drop it using the same button. These barrels can be used to block the enemies' path.

2.5 Actions and Combat

Most of the players' actions in the game are combat-based. There are various ways to fight. Combat is the main part of the game. Players and enemies have health, damage and chemical properties that affects the outcome of the combat.

2.5.1 Melee

Several characters in the game are optimal for close-combat, since their melee damage is greater than the ranged damage. Melee combat is basically an attack that can be done at any time with a small cooldown(i.e. swinging arms towards an enemy and damaging it). This basic attack can have different effects based on the character's class.

2.5.2 Range

All characters have a ranged attack in the form of puking. It deals different amount of damage to the enemy depending on chemicals and class. The characters with a lot of ranged damage are more fragile to enemy attacks.

2.5.3 Tactical Combat

Since enemies always come towards the players in order to attack them using a simple pathfinding AI, it is possible for players to use movable objects to redirect the enemies' path in order to increase their safety and possibilities to deal ranged damage.

2.6 Player classes

There are four number of classes in Freakolution; all with different strengths, weaknesses and capabilities. Each class have different cooldown times and different meelee and range damage. In table 2 the combat specifics are shown for each of the four classes.

2.6.1 Engineer mutant

The engineer will have the ability to with a low cooldown build blocks which everyone can use to build defensive structures to protect themselves from the hordes of enemies.

2.6.2 Marksman mutant

The long distance mutant is able to deliver most damage with long distance attack, puking. This character is very vulnerable and therefore not suited for close up battle.

2.6.3 Tank mutant

The tank is the character that has to take most damage from enemies. With a lot of health points it can withstand a lot of strikes from the opponents.

2.6.4 Healer mutant

The healer mutant can heal its allies by puking on them. It can still deal damage both with range attacks and melee attacks but not as much as the other characters.

Table 2: Player classes statistics

	Engineer	Marksman	Tank	Healer
Melee damage per second	75	30	150	75
Range damage per second	65	368	25	65
Health	200	100	300	200
Healing per second	0	0	0	98
Movement speed	4	4	4	4
Melee attack cooldown (sec)	0.4	0.5	0.4	0.4
Range attack cooldown (sec)	3	2	4	3
Build cooldown (sec)	3	20	20	20

2.7 Enemies

Enemies are portrayed as scientists with protection suits. They all have individual combinations of chemicals which makes them stronger or weaker against different players depending on their chemical. The enemies have a glow effect with color based on how much of each chemical they have. For example an enemy with a mix of blue and red chemicals would have a purple glow. The statistics of the enemies are shown in table 3.

2.8 Chemical specialisation

There are 3 kinds of chemicals: Redion, Bluorine and Greenogen. These chemicals are attributes that both the players and enemies have. They each are stronger or weaker against each other depending on the combinaton, i.e. the result of the damage formula.

Table 3: Enemy statistics

	Scientist Enemy
Damage per second	3
Health	100
Movement speed	3
Detection range	50

2.8.1 Damage Formula

The damage per second that a player character can affect an enemy with depends on the respective chemical weight. The calculation is done for all chemicals and then added to the loss of the enemies life.

$$ChemicalDPS_{total} = DPS_{base} \cdot (1 - ChemicalWeight_{player}) \cdot ChemicalWeight_{enemy}$$

2.8.2 Example of damage calculation

Say an Archer with the chemical $redion = 1$ and all other chemicals set to 0, attacks an enemy with chemicals $redion = 0.58$, $greenogen = 0.58$ and $bluorine = 0.58$.

$ArcherDPS_{total} = 368\ dmg/sec$

$RedionWeightOnArcher = 0$

$GreenogenWeightOnArcher = 1$

$BluorineWeightOnArcher = 0$

$RedionWeightOnMeleeEnemy = 0.58$

$GreenogenWeightOnMeleeEnemy = 0.58$

$BluorineWeightOnMeleeEnemy = 0.58$

$RedionDPS_{total} = 368 \cdot 0 \cdot (1 - 0.58) dmg/sec = 0\ dmg/sec$

$GreenogenDPS_{total} = 368 \cdot 1 \cdot (1 - 0.58) dmg/sec = 0\ dmg/sec$

$BlourineDPS_{total} = 368 \cdot 0 \cdot (1 - 0.58) dmg/sec = 0\ dmg/sec$

The total DPS is obtained by adding all DPS from all chemicals:

$DPS_{total} = 0 + 155 + 0\ dmg/sec = 155\ dmg/sec$

This will mean that the more different the chemical of the enemy is from the chemical of the player, the more damage the player will do to the enemy. In the opposite way, the more similar the chemicals are between player and enemy, the more damage the enemy does to the player. Also note that the chemicals are always normalized. The magnitude of the chemical should always be 1 to be able to perform these calculations.

2.9 Screen Flow

The screen flow consists of three different stages; the intro stage, the settings stage and the gameplay stage. See figure 7

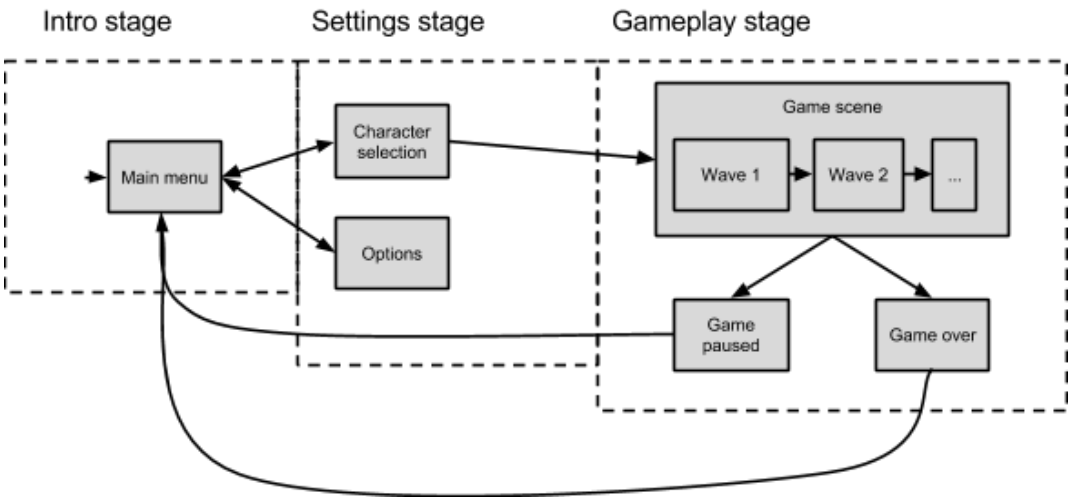

Figure 7: Screen flow diagram.

2.9.1 Intro stage

The intro stage concist of the main menu which leads to the character selection or the options state. The colors are greenish and there is a background music playing in this stage.

2.9.2 Settings stage

From the main menu, the player can go to the settings stage to change parameters that affect the actual game play. All the screens in the settings stage have a similar style as the main menu. A similar background and the same style on buttons.

2.9.3 Gameplay stage

The gameplay stage consists of one game mode, Survival. The Survival game mode takes place in one scene. From the actual game scene, the player can pause the game. When the game is paused, the players will be shown two buttons. One to continue the game and one to go back to the main menu. The stage is displayed in figure 8.

2.10 Game progression

The game consists of only one main survival level. During the game, waves of enemies that comes after the players and attack in larger number as the game progresses.

At the end of the game, when all the players run out of life and are dead, a scoreboard shows information about each player's performance. It includes: damage dealt to enemies and who got hurt the most as well as number of barrels built and moved.

3 Story

An evil organisation, Draco Labs, has been conducting illegal animal testing of their latest biochemical research products focusing on DNA alteration.

The latest generation of bunny subjects were considered more dangerous than the others since they were considerably stronger and smarter than the rest and therefore it was decided that it would be too dangerous to let them live. Smart as these creatures were, they understood what was going on and started a revolt against the humans with the goal of escaping the experiment facility. The humans had the upper hand and almost instantly killed off most of the creatures when they attacked one by one.

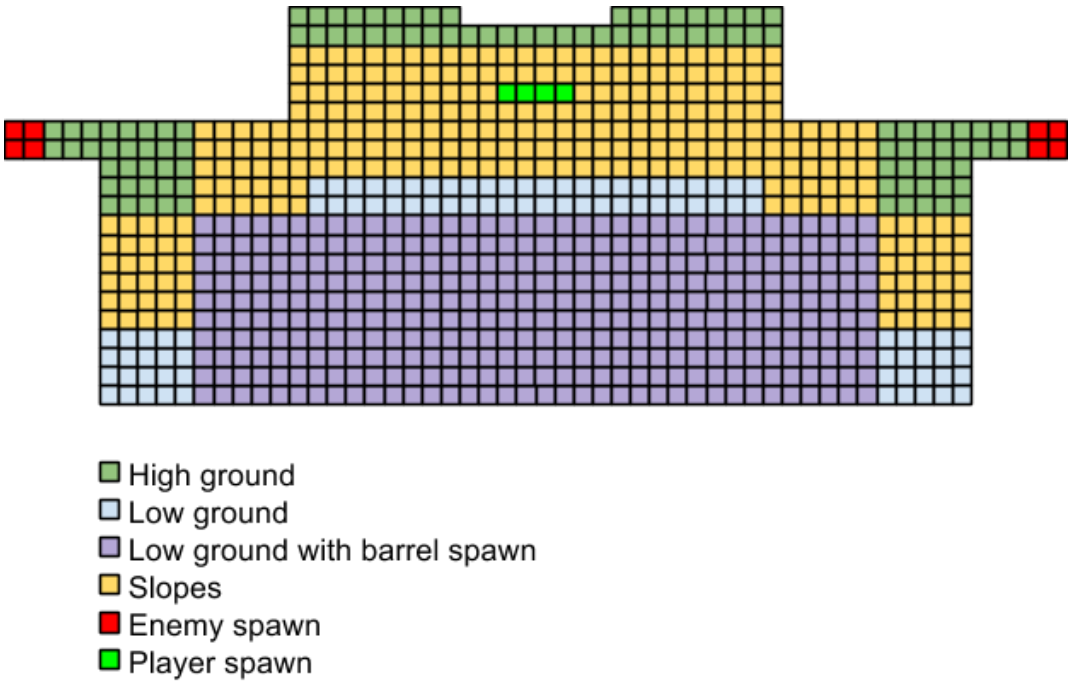

Figure 8: The gameplay stage.

In the end only four of them were left. They realized that they had to work together with their different unique abilities to escape.

Having seen the dark and dangerous side of humans the four experimental animals swore to avenge all the animals by killing every human involved in the organization Draco Labs, which were conducting all these experiments. To be able to do this it is necessary to raid all the facilities located at different parts of the world and to kill off all the presidents of this company.

4 Artificial Intelligence

4.1 Enemy AI

Enemy AI can be split into fighting and pathfinding.

4.1.1 Fighting

Enemies can either attack in melee or from a specific distance (range). The AI is simple; melee enemies reach players using the A* pathfinding algorithm. and hit players every specific period of time once they reached the destination. If the players move again, the enemy will reactivate the pathfinding algorithm and reach them again.

Collisions detection and handling between various characters in the game (players, enemies and environment) is done using the in-built physics of Unity. Specifically, by having ‘colliders’ on these entities, collision and blocking is to be achieved fairly easily.

To be more specific, an invisible “sphere” used only to find an intersection is created in front of an enemy, and it will attack if this sphere intersects with any player colliders (very similarly to the actual player’s attack collisions).

4.1.2 Pathfinding

For pathfinding, a library called Simply A* has been used. This is a free library for Unity. This library’s main algorithm is A* (‘a-star’), which is a very popular pathfinding algorithm that uses manhattan distances to find the best path to a destination.

Hence, the plane on which the game happens must be divided into a grid. This process is included in the before-mentioned asset package.

This library allows the enemies in the game a simple mechanism to reach and attack the players. Moreover this library includes pathfinding algorithm that can detect when no path is available to reach a specific place (the players).

The asset "Simply A*" had to be modified drastically to be adapted to the game due to its current specifications. For instance, in Freakolution, the environment and obstacles varies because barrels can be moved and added to the scene to block enemies' paths, and enemies are not able to stand on each other. There are actually 2 separate pathfinding algorithms in the game, which runs independently depending on the situation:

The first one is computationally expensive and will compute the best path while taking other enemies into account, which allows enemies to properly surround the players.

The second one is simpler since it does not care about other enemies in the scene and will give the optimal path, making enemies queue to reach a player. It was necessary to keep this simple algorithm for 2 reasons: performance, and because the other algorithm breaks and makes the computer lag when queuing is actually necessary for enemies - i.e. in a player-made maze of barrels.

Note that due to the complexity of the implementation of this algorithm, an important part remains to be done: the detection of "no path available" and destroying of barrels. It is actually already possible to detect if no path is available to the players; it is actually done to trigger the use of the simpler algorithm, however, there was not enough time to add a completely new third algorithm to find the optimal path and destroy everything in the path, since, on top of its inherent complexity, it could have broken the previous algorithms and hence would require much more work. It is important to note however that the game can still be played if players do not use this "cheat", and although the game is not perfect yet, it can be considered a working prototype which is already useful as it can be play-tested.

5 Technical

5.1 Target hardware

This game aims to be played on any typical game console (i.e. Playstation, Xbox, etc.) or a computer (Mac, Linux or Windows). The players' input will be done using classic console game controller (Playstation or Xbox controllers, and for computers most likely Logitech controllers).

5.2 Development hardware and software

During the development, classic game controllers will be used (mostly likely made by Logitech) on Mac and Windows, although thanks to Unity it should be exportable and playable on any of the before-mentioned consoles.

5.3 Development procedures and standards

Although the goal is to get the basic functionality of the game done as soon as possible (much like an Extreme programming approach), we all made it very clear that our design should be flawless and our code irreproachable. In order to achieve that goal, we included peer reviewing of any task done using our Agile project management (see Section X Management). Moreover, we are following Unity's code guideline in order to have a common basis for proper coding.

Since Unity works with a lot of separate entity-based scripts, there won't be too much work to do in the 'Technical Design' of the whole system.

5.4 Game engine

The Game Engine used for this project is Unity, as specified in the project requirements. It allows export to many different targeted platforms and has in-built support for many aspects of the game development such as game controllers support, easy camera views and scripts, character controllers, physics, collision detection, etc.

5.5 Network

Although the game is played locally (on only one machine), it is multiplayer and the original idea was actually to build it as an online game. However, to be able to create a working prototype faster, the game currently have no network support. In future versions, as an insight, the game might be ported to an online game using the Photon Unity library.

5.6 Scripting language

Unity allows for several different scripting languages. We believe that in our case it would be better to stick to only one, in order to make debugging easier and to increase the readability of the code. The coding is done in C#, but some parts are in Javascript. This is due to the fact that many outside resources are used (such as the pathfinding algorithm) in order to increase the productivity and to focus on the final result.

6 Game art

6.1 Concept art

The concept art consists of:

- Some basic sketches of the gameplay
- Ideas for menus
- Level design ideas

See figure 9, 10 and 11 below.

Figure 9: Basic character mutant and enemy sketches.

Figure 10: Bunny character concept.

Figure 11: Gameplay sketch.

6.2 2D art

The 2D art of the game consists of menu backgrounds, title screen intro image, and sprites for the characters. The 2D elements and the style of the game are semi realistic and focuses a lot on colors and light.

6.3 3D art

A big part of the game art is in 3D. The scene where the actual game play takes place is in 3D. All the objects except the characters are 3D objects.

7 Management

The design of Freakolution is progressed using the scrum methodology with some modifications to suit this specific project and everything that comes with it.

7.1 Scrum

Each sprint should be finished in two weeks resulting in three sprints plus one extra sprint, sprint 0, in the beginning to start up the project and get everything going. In sprint 0, the time for each task is not estimated since the stories mostly are about setting up project files and deciding upon the general direction of the game design process. For the remaining sprints, the time of each task should be estimated and measured to make sure that the correct amount of time is spent on each sprint.

All group members acts as the product owner and can therefore always add stories to the backlog. This makes everyone a part of the decision making but can lead to more time spent on planning and prioritising stories and tasks.

Meetings are held each week and consist of both standup meetings and sprint meetings. The standup meetings are held twice a week, and should be ten minutes at maximum.

Every second week before each sprint, a sprint meeting is conducted. Here the group discuss the previous sprint with a retrospective. Also, stories for coming sprint are discussed and prioritised. Every story should also consist of tasks which have to be time-estimated. The sprint meetings are documented by one member of the group.

For handling events such as story and task management, the web application trello was used. This gave the opportunity to handle each sprint as a “board” with stories and tasks as different cards with different color tags to make sure every group member knew what the others were doing, see figure 12.

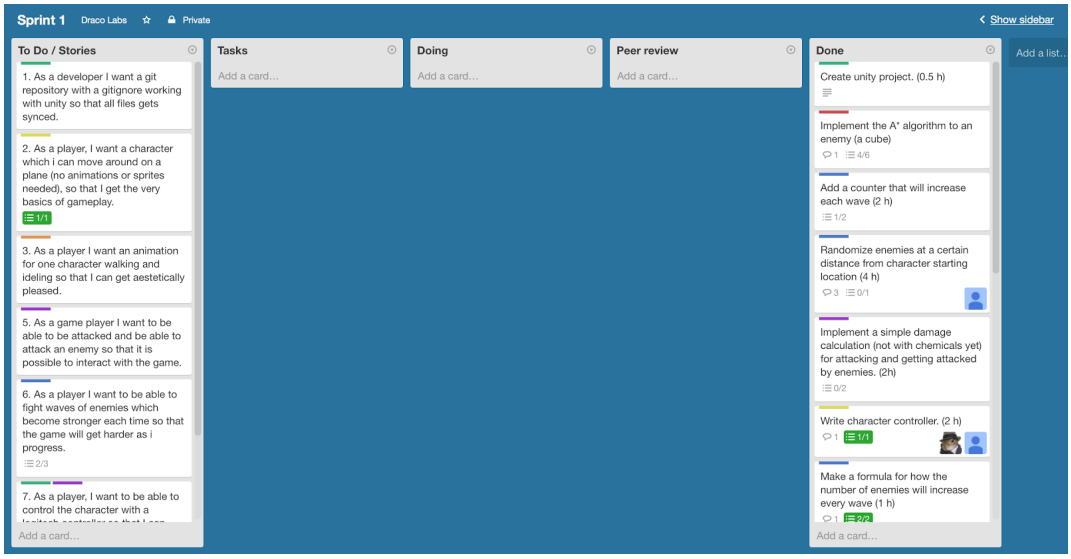

Figure 12: A typical sprint in trello.

7.2 Version control

For version control, git is used. That way it is easy for the developers to work in parallel and control each version of the game that is finished after each sprint.

When a task is finished, it is moved to the Peer Review list. Then another person has to review what has been done before the code can be merged to the branch for the specific sprint. When the sprint is finished the code written in the sprint branch will be merged with the master branch. This is done on the sprint meetings.

7.3 Communication

For communication, facebook and google apps are used. Facebook works as a quick way to get in contact with all the group members for urgent issues or for questions regarding the development. With google calendar, all the group members can create events or schedule meetings which will be seen by the rest of the group.

8 Appendix - Future development

This Section includes all the game features that were included in the original GDD and that could not be implemented to this day. As part of Agile development, the group skimmed through the GDD every beginning of a new sprint and selected the most important features to be done.

Although these features were not implemented, the group believe that they could still add a lot the game as it is currently, hence Future Development is included as an Appendice to give an overview of what the game will be like once it is finished.

8.1 Attacks

8.1.1 Range Attacks

A broad range of projectiles is available through the available weapons. They all have different speeds and effects. This include but is not restricted to: fast single target, slow 'splash'/AOE(area of effect) target, guided missile, hit-through (laser), bouncing, etc. The implementation of such projectiles will most likely vary depending on the progress of the rest of the project.

8.1.2 Friendly fire

If the team does not play well together, they might accidentally (or purposely) hit each other which brings an extra dimension of chaos to the game.

8.2 Scene objects

8.2.1 Buildable objects

Some of the players will be able to build useful objects according to their class or items that they acquired during the game. These objects have special capabilities (traps, automatic fire turrets) and give a more strategic aspect to the game. Once built, these objects are part of the environment, have rigid bodies and can be interacted with by any player.

8.2.2 Reward items

When enemies die, they sometimes leave items behind them. These items can be picked up by simply walking on them. Items that can be picked up include: health potion, weapons and armor items, usable items such as mines, one-use spells, turrets, etc. Possibly if a 'crafting system' is made, they could also include various materials used to craft said items.

8.2.3 Picking up objects

Players are able to pick up reward items by walking on them. They can then use these items using their 'inventory menu', or even make some of these items have a shortcut to a specific button on their controller.

Some characters with special class and powers (telekinesis) can move objects around from a distance by entering 'telekinesis mode', having the character stuck to a location while a pointer appears, which the player can use to select objects and move them around, or throw them at enemies.

8.2.4 Destroying objects

Only enemies can destroy objects, and that happens only if no path is available to reach the players: if the players completely barricaded themselves behind objects, leaving no space for enemies to go through.

8.3 Pause Menu

When the pause is pressed the music will stop to help indicate that the the gameplay has stopped.

If the Exit to Main Menu button is pressed, a warning prompt will pop up in front of the Pause Menu letting the players know that they are about to exit the game and lose their progress. The players will have to press Yes if they want to continue to Main Menu or No if they change their mind.

8.4 Spells

Every class can cast a few special spells that vary according to their class. See Table 4 which also includes other button functionalities.

Table 4: Player classes specialization

Buttons / Character	Tank	Engineer	Marksman	Healer
Button 1	Stomp the ground(stun). Cooldown 10sec.	Whirlwind 7 sec (spinning damaging around). Cooldown 30sec.	AOE damage spray	Healing spray
Button 2	Invisibility for 10 sec. Cooldown 60sec.	Build. 5 sec to build. No cooldown, chanelling (can't attack)	Telekinesis (move objects from a far) No cooldown, channeling.	Bubble built as spray is used which reduce incoming damage in area, can be released granting full life at once and damaging enemies by 70%.
Button 3	Jump	Jump	Jump	Jump
Button 4	Melee attack	Melee attack	Long range attack	Long range attack
Button R1	Move crate	Move crate	Move crate	Move crate
Start	Pause/Unpause	Pause/Unpause	Pause/Unpause	Pause/Unpause

8.5 Player types

Originally each player character was supposed to have its unique special abilty, besides its normal attack. These features would give the players more ways to cooperate and more dynamics to the game.

8.5.1 Engineer

Depending on the environment the things that are possible to build are different which makes it important to think about where to stand when creating these blocks since the materials may change the result. The builder character is very blocky to its appearance and short in height. It has the ability to create blocks which it, and other players, can move to protects themselves.

8.5.2 Marksman

This character has telekinesis powers which makes it possible to easily move blocks around and also to use other materials in the area. This could be used to seal off openings and drop heavy items on enemies and even solving puzzles of different kinds.

8.5.3 Tank

The tank is very wide and grounded with a low center of mass. Its characteristic appearance are spikes and armor. this character will have the power to become invisible which is an important game mechanic since it allows you to move behind the enemies and scout areas. This ability will also allow it to lure enemies away from other players and then lose them by going invisible.

8.5.4 Healer

The healer can create a bubble which is another defensive mechanism that can be used. Inside the enemies move slower which makes it possible to move around and hitting them with long range attacks since they cannot catch up to you.

8.6 Enemy Types

8.6.1 Flying enemy

A human using jetpack to fly over obstacles, which makes it harder for the players to protect themselves. These humans are fast and agile but also very vulnerable. This enemy can only be killed by ranged attacks.

8.6.2 Range enemy

A human using a long range weapon (harpoon type) to wound the player characters. When being close enough to a player character, the range enemy will fire its weapon. Otherwise it will walk towards them to get into range. The range enemy can deliver high damage but is also vulnerable to attacks.

8.6.3 Melee enemy (slow)

A human in a big robot armor. This enemy is strong, but have to move close to reach with its punches. The attack cooldown of this enemy is long and it does not move very fast. This enemy will start destroying blocks if there is no way to reach the players.

8.6.4 Melee enemy (fast)

This human is smaller and does not wear the same amount of armor, which makes it more vulnerable to attacks. This also means that it can run faster. When in close enough distance, it picks up its taser to stun the player. The player does not lose a lot of health, but he or she is stunned (can not move) for 0.3 seconds.

8.6.5 Boss enemy

This enemy will be alot stronger than the other enemies, and will require some form of cooperative tactic to defeat.

8.7 Chemical level up

At the end of each wave, players can each choose a specific chemical attribute to increase. Increasing it will basically add 20% to its current value (i.e. Increasing a Redion attribute of 1 will make it 1.2, increasing it another time will make it 1.44).

By adding a percentage rather than a linear value, the strength of the specialisation is exponential and pushes the player to focus on only 1 or 2, since playing a specialisation that is too balanced will result in not doing enough damage at high levels of waves. With this implemented, the damage formula will have to be reworked since it requires the chemicals to be normalized.

8.8 Screen Flow - Intro Stage

The intro stage consists of two screens to give the player an overview of what the game is about. When starting the game the players will first reach the title screen. There will be no buttons to press and there will be no cursor to move. The games theme music will loop whilst still showing this screen. If any button is pressed, the screen will be replaced with the main menu.

8.9 Game Progression

For the wave there will be a rank value which is calculated by the sum of the attack and defense of all monsters. This means that if there are few enemies they will be stronger and if there are many enemies they will be weaker. It also gives the possibility of having enemies that are easy to kill but deals a lot of damage and others that are more difficult to finish off.

Since the rank value will be increasing exponentially the game will become very difficult after a while. To make up for this there will be a level system where the characters get experience from beating waves so that their attributes can be customized to be able to beat more difficult waves.Depending on each individual's performance additional experience will be divided amongst the players; the more points you receive the more experience you get.

8.10 Campain mode with several levels

There will be a number of levels which the players will have to overcome to be able to finish the game. They will consist objectives in the form of bosses that you have to beat and waves of enemies that will try to stop you from reaching them.

The enemy encounters will be different for each level. This means that some of the powers/weapons that can be obtained may be more or less useful in this particular map since some enemies are weak against specific powers. The surroundings for each level will also vary since the facilities are at different parts of the world and therefore all have a unique style.

8.11 Friendly AI

Players are able to build and summon a number of autonomous entities to help them in their struggle. These include:

8.11.1 Turrets

Turrets have a fairly simple AI. Their script checks if any enemies are within their range. If some are, they then pick the closest one and aim at it (using Unity's lookAt method) and fire a specific projectile at regular interval.

8.11.2 Healing Well

Healing well are entities that can be static or follow a specific player. They regularly deliver a specific amount of health to that player or those around it.

9 References

Coding Guidelines: http://wiki.unity3d.com/index.php/Csharp_Coding_Guidelines

9.1 Assets

9.1.1 Scripts

Simply A*: https://www.assetstore.unity3d.com/en/#!/content/6385

Photon: https://www.assetstore.unity3d.com/en/#!/content/1786

9.1.2 3D Models

Barbed wire: https://www.assetstore.unity3d.com/en/#!/content/45

Barrel: https://www.assetstore.unity3d.com/en/#!/content/840

Sci-fi door: https://www.assetstore.unity3d.com/en/#!/content/21813

Streets: https://www.assetstore.unity3d.com/en/#!/content/13811

Tall fence: https://www.assetstore.unity3d.com/en/#!/content/70

9.1.3 Textures

Concrete textures: https://www.assetstore.unity3d.com/en/#!/content/12951

Nature textures: https://www.assetstore.unity3d.com/en/#!/content/13237

9.2 Music

Gameplay music: DST-DarkFuture, http://www.nosoapradio.us/

Menu music: belabar Dark8-320, http://sampleswap.org/mp3/artist/bela/belabar_Dark8-320.mp3