Department of Computer Science

Generating Dungeons & Dragons dungeons that make visible sense in the context of a narrative

Using Procedural Content Generation to generate a D&D dungeon that tells a story, and makes contextual sense, including environmental storytelling.

Author: Joseph Shufflebotham

Supervisor: Dr Rob Alexander

April 2019

This report is submitted as partial fulfillment for the degree of

BEng Computer Science (with a year in industry)

in the department of Computer Science at the University of York

ACKNOWLEDGEMENTS

This project would not have been as successful if not for the support of the Dungeons and Dragons community on reddit.

Thanks to my supervisor Dr Rob Alexander for his support and feedback throughout this project.

STATEMENT OF ETHICS

To study the effectiveness of my solution, I will be sending out a questionnaire to potential users (Appendix 1). This questionnaire was approved using the Physical Sciences Ethics Committee fast-track ethical approval procedure.

No social, legal or professional issues are raised by this project.

TABLE OF CONTENTS

TABLE OF FIGURES

Executive Summary

Dungeons and Dragons (D&D) is a role-playing game where players climb into the shoes of a character they created, to explore a fictional world with a story to tell [1]. In this project, our aim is to build a system that can generate these dungeons that players explore. The system will take a user-centred design (UCD) approach to identify the requirements.

Research was necessary to understand the needs of the average player, which we built the requirements from. First, we created assumption personas that represent the general target audience then the requirements list was built from a list of goals that these users had for such a system and the issues they had with existing generators. Once the requirements were identified, research was conducted to identify the best way to implement a solution that met all the requirements. From these personas we determined that the process of creating an adventure on your own from scratch is a very time consuming task and requires a very creative mindset, which can turn people away from being a DM. With this project I aim to reduce that time it takes to create an adventure and give people a creative boost in their adventures.

Procedural content generation (PCG) is the method of generating game content using random or pseudo-random algorithms that result in an unpredictable range of possible game states. It is used throughout the video game industry and is an ideal solution to generating dungeons. We researched many potential algorithms that would allow us to meet the requirements and for future expansion. The decision was to settle on an algorithm designed for the game TinyKeep[2] and posted on Reddit by user phidinh6[3]. The dungeon requirements for TinyKeep are similar to that of my generator, but the algorithm will be modified slightly to fully meet the needs of this project.

Story is a big part of a D&D campaign, players usually expect a campaign to have a background story and a reason for their exploration. In the video game industry, there are lots of examples of games built around their story being a huge success and receiving outstanding ratings. A requirement of this generator will be that the dungeon generated must make sense, and it must tell a story. The

monsters and features in the dungeon will need to exist for a reason, allowing the Dungeon Master (DM) to get an adventure up and going quickly.

A questionnaire was used to evaluate the effectiveness of my solution and whether it met the requirements. The questionnaire was the only ethical concern for this project, but it was approved using the Physical Sciences Ethics Committee fast-track ethical approval procedure. No other social, legal or professional issues were raised by this project.

This questionnaire was sent to people who either have experience being a DM or want to be a DM and they were asked to consider creating an adventure or campaign from the generated dungeon and discuss how my solution may have improved or not improved on the usual process. The results of this questionnaire showed that my solution was successful in meeting the requirements set out from the user-centered design. Overall feedback for the generator was positive and there were many suggestions on how to improve the generator to increase the quality of the generated scenarios.

1 Introduction

This report will use procedural content generation to generate a Dungeons and Dragons (D&D) dungeon that meet the needs of users who play and run D&D campaigns, with an emphasis on environmental storytelling in the generated dungeon. The following D&D terminology will be used throughout the report:

Session: A session is one sitting of D&D, usually spread over a few hours and is pre planned so all players can come together.

Adventure: Adventures are played across multiple sessions, usually three to six sessions each.

Campaign: A campaign is a string of linked adventures usually played by the same people in the same universe. These campaigns can take a long time to complete and are played across many sessions.

1.1 Dungeons and Dragons

Dungeons and Dragons (D&D)[1] is a popular example of a fantasy role-playing game, where players create a fictional character to explore worlds created by the Dungeon Master (DM). Games usually consist of 3 to 5 players and 1 other person who is the Dungeon Master. The DM's task is to guide the players through the world they created to tell a story or complete a task.

There are many premade D&D campaigns, some shipped with the game, and many more posted by players online. These campaigns allow players to get up and going quickly, and are usually good for first-time players and/or DMs. Premade campaigns generally consist of a dungeon layout (Fig 1), backstory, monsters and everything else needed to run a game. Therefore, the difficult task of generating a world and a story for it that makes sense is avoided. However, whilst this is an easy option, most experienced players prefer to create their own story and a world in which to tell it. This allows for each game to be unique and more enjoyable. DMs tend to have a desire to tell a story, and this is a great way to tell one. However, since the task of creating a world and a narrative to go with it is time-consuming and difficult, most people tend to use dungeon generators to at least get inspiration. Most generators only generate the layout of your dungeon for you, so it's up to you to build a story from the dungeon that is provided for you.

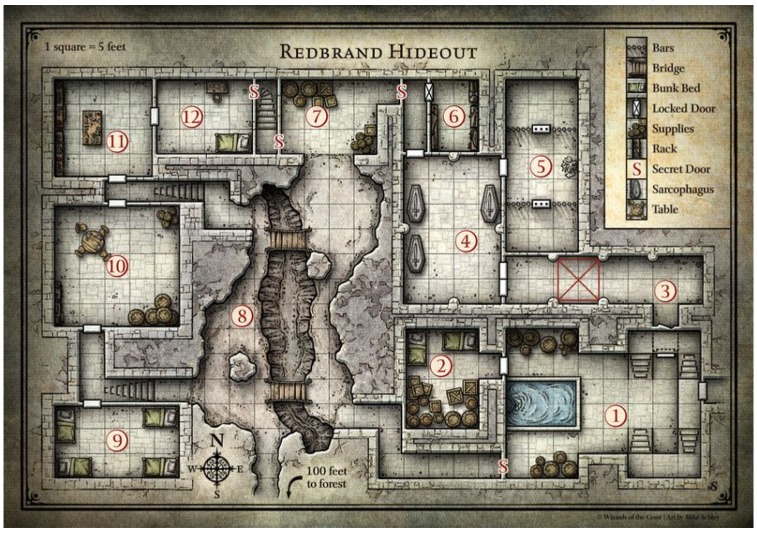


Fig 1. Example of a pre-made dungeons and dragons map. Redbrand Hideout from the Lost Mine of Phandelver campaign.

1.2 Narrative in Games

Along with narrative, both plot and story are equally important in giving players an enjoyable experience within a make-believe world. Whilst they sound similar, these three elements are all different and work together to create the overall adventure. P. Cobley describes them as follows; the story is the collection of all the events that occur; the plot is the reason behind the events occurring, the "chain of causation" showing how events are linked. The narrative is the method that is used to tell the story, how to present it to the user[4]. For example, the narrative could be directly telling the user the events through narration or allowing the user to explore the environment to discover the events for themselves, which is called environmental storytelling. These two narrative methods are both at play in the Fullbright game "Gone Home"[5], where there are sections of the story told to you through journal entries read aloud on a cassette tape, and other aspects of the story are conveyed through objects in the environment, such as books or letters you come across.

Video games are more popular than all other forms of entertainment, seeing more industry revenue than music, movies and television[6]. With this growth, there has been a surge in developers that has been enabled by the introduction of new distribution platforms such as the Discord Store[7] and the Epic Games Store[8]. Platforms like these make it easier than ever for indie developers to get their games published, either through a publishing company or self-publishing, and start making money. This increase in indie developers has contributed to the growing popularity of story-driven games. Some popular examples include Life is Strange by DONTNOD[9], and more indie games such as What Remains of Edith Finch[10] by Giant Sparrow and Gone Home[5] by Fullbright.

These games have been enjoyed by over 3.5 million people and are among the top rated adventure games of all time[11][12][13][14]. This shows that story can and is a hugely important aspect to games, both video games and tabletop games. Chelsea Cariota states that games like these take storytelling to the next level, making the player feel as though it is THEIR narrative by giving them control over their own fate. This feeling of ownership and control keeps the player playing right until the end, maybe even causing them to return and try making different decisions[15]. The ability to interact with a world falls perfectly under the description of tabletop games such as Dungeons and Dragons.

1.3 Environmental Storytelling

Stewart[16] states that environmental storytelling is “the art of arranging a careful selection of the objects available in a game world so that they suggest a story to the player who sees them”. Environmental storytelling is becoming more and more popular in AAA games as computing power increases enabling a higher level of detail in the worlds we explore.

The recent release of Red Dead Redemption 2 in 2018 has seen critical acclaim with it being the top critic rated PS4 game of 2018[17]. One critic writes

"No other game has probably offered so many options to express yourself in the open world, and at the same time did not have such an interesting, rough and moving story."

LEVEL - Czech Republic Gaming Magazine [17]

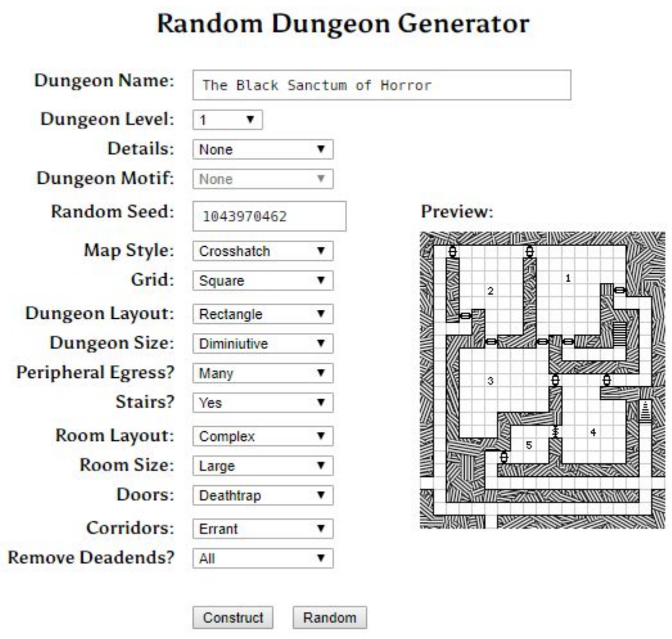
Reviews like this, and many from normal players, show the importance and appreciation of environmental storytelling, and the setting of a dungeon is the perfect place to use the scenery to tell a story.

1.4 Generators

As previously stated, it is very common for Dungeon Masters to use dungeon generators to aid them in the creation of their world. There are many generators that are all used for different reasons. In a questionnaire performed by Brown[18] on people who have DM experience, he found that 73% of them had previously used dungeon generators to some extent when designing a campaign.

There are many popular dungeon generators out there, most notably is DonJon, a series of generators which has a lot of customisation options, allowing the user to create a dungeon for their exact needs. DonJon released its first generator in 1999 and has since released many generators for all needs, from fantasy name generators to the random dungeon generator, which we assume had a similar target audience to this project.

However, the donjon random dungeon generator doesn't generate any content along the lines of storytelling and instead assumes that the user has a narrative in mind already. This is assumed based on the fact that in the customisation options before generating your dungeon, users have to input their chosen environment and other key attributes to allow the generation to happen (Fig 2). Given that the key challenge of this project is generating a dungeon with a story that makes sense, comparing my solution to the donjon random dungeon generator, which doesn't generate anything to do with dungeon context, might not be fair.



Random Dungeon Generator

Dungeon Name:	The Black Sanctum of Horror
Dungeon Level:	1
Details:	None
Dungeon Motif:	None
Random Seed:	1043970462
Map Style:	Crosshatch
Grid:	Square
Dungeon Layout:	Rectangle
Dungeon Size:	Diminiutive
Peripheral Egress?	Many
Stairs?	Yes
Room Layout:	Complex
Room Size:	Large
Doors:	Deathtrap
Corridors:	Errant
Remove Deadends?	All

Fig 2. donjon generation configuration.

1.5 Procedural Content Generation

Procedural content generation (PCG) is the method of generating game content using random or pseudo-random algorithms that result in an unpredictable range of possible game states.

There are many different methods of PCG as the definition only states that it uses algorithms to generate content, but doesn’t mention specific ones. There are some popular algorithms that are used often, but most of the time, a custom algorithm needs to be designed and created for the individual needs of the project.

Thrall’s[19] solution uses a common PCG algorithm called spatial partitioning trees to generate his dungeon content. Spatial partitioning trees starts with a full map and splits into 2 smaller areas, and then recursively does this until a minimum constraint is met. Then when the splitting is complete, a room is placed in each section. One benefit of this is that you can guarantee that rooms won’t intersect with each other which is a difficult task to complete otherwise.

Minecraft[20] is a famous and very successful example of a game that implements PCG. It is also an incredibly complex example of procedural content generation, with the inclusion of biomes, caves, oceans and much more. In the early days, developer Markus states he used a 2D Perlin noise heightmap to generate the shape of the world, but as the game got more complex they needed to change their methods[21]. Whilst they mostly implemented their own algorithms to meet their exact needs, they also took inspiration from some existing methods, namely the Whittaker Diagram[22]. The Whittaker Diagram (Fig. 3) is a biome diagram designed by RH Whittaker that sets a biome based on precipitation and temperature.

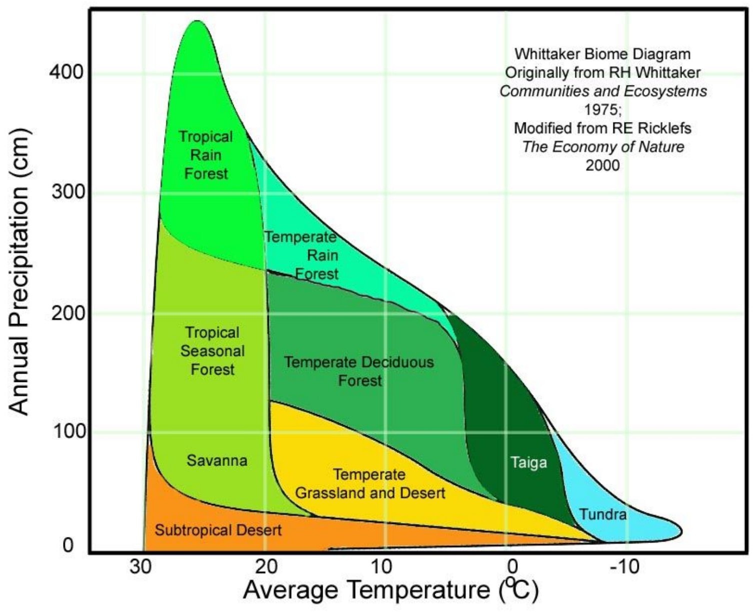


Fig 3. Whittaker Diagram is a geographical theory that maps the combination of average temperature and annual precipitation to a specific biome type[22].

For this project, my options were to expand upon Thrall's solution, using Spatial partitioning trees, or to explore other options and implement my own solution. I explored Thrall's solution in the context of implementing my own goals, and whether achieving them would be easy, if possible at all. After some exploration, I decided to implement my own solution as the amount of work required if I were to expand on Thrall's solution would be enough to warrant a rewrite. This will also allow me to choose a better algorithm that will more accurately suit my needs.

2 Problem and Proposed Solution

2.1 Problems with Existing Solutions

Whilst there are many existing solutions out there, there is a lack of a “perfect” generator. As my solution is inspired by Thrall's[19] solution I will focus on issues with that solution.

As previously mentioned, Thrall’s solution utilizes a Spatial partitioning tree, which, whilst appropriate for his needs, won’t be suitable for my goals. This is because Thrall’s system guarantees a whole floor of connected rooms, but I plan on implementing a multileveled generator, that would ideally allow for multiple sections on the same floor that aren’t connected. Imagine for example a haunted castle that has 2 towers, one on either side of the castle. The floors in these towers will be at the same level, but they won’t be directly connected to each other on that floor. This scenario wouldn’t be possible with Thrall's solution, hence the decision to reimplement a new algorithm.

Overall, Thrall's solution doesn’t allow for a lot of common D&D scenarios and as a result, can make running a campaign on a dungeon generated by his solution very difficult. Other generators such as Donjon have a lot more to offer in terms of potential results, but most of them don’t provide narrative or story generation.

2.2 Proposed Solution

My proposed solution will reimplement the generic room generation using a different algorithm, engineered by a game developer from Reddit. This algorithm was used in his game TinyKeep[2], a dungeon crawling game whose needs are similar to mine.

The algorithm utilizes multiple mathematical functions to ensure some key features such as no rooms overlap and it guarantees that there are no unreachable rooms[3].

In addition to this, I will be adding some new features to the system. In Thrall's solution, rooms are a singular type, either treasure, trap, monster or none. Whilst this provides a wide variety of room types, it’s not well representative of an actual D&D situation. For example, with Thrall's solution, it would be impossible for a large treasure chest to be guarded by 5 hungry werewolves, but treasure being guarded is fairly common in D&D. Because of this, I will be adding a system where rooms have a list of features, rather than one type. These features can be treasure, monsters, traps, or stairs, which allow players to travel between floors. Adding more features to the system in the future would be easy because of the way I have designed it. With the addition of this system, my generator can have rooms that are filled with the world's most valuable treasure but surrounded by the deadliest traps or guarded by the strongest of monsters.

As briefly discussed, rooms can have a stairs feature which allows them to go to a different floor. This is another one of my additions to the system, multi-level dungeons. From experience, I know that a more detailed dungeon can lead to a more immersive and believable experience, and after all, not all dungeons are strictly one level deep. My system allows rooms to have small damaged ladders in the corner leading to a new level or trap doors that drop you down to the floor below, which opens up a whole new level of exploration.

As well as physical additions, I will also add the ability for dungeons to have a background, a reason for its existence. The Dungeon Master Guide states a process to follow for creating an adventure environment[23]. This process involved rolling several 20 sided dice then using lookup tables to give the dungeon a location, creator, purpose and a history. I will implement these rolls to give my dungeon all of the above attributes. The selected attributes will also affect the rest of the dungeon generation by applying modifiers, this is explained in more detail in section 3.6.2.

3 Design and Implementation

3.1 Personas and Scenarios

I will use these personas as the ideal target audience for my solution. The design will be made with the requirements of these personas in mind. M. Barlett states that personas are used as descriptions of typical users and are produced using gathered data[24]. He gives example fields that personas should record in general, and I adapted these for the exact needs of this project. Barlett also states that it is important for personas to have goals and not tasks, these goals can then be assessed as to whether or not the solution helped personas achieve these goals.

David Mackleby

Age: 37

Gender: Male

General Information: Married for 11 years and has 2 children, a 9-year-old and a 2-year-old. Works for a bank as a software developer working on updating old systems.

Background: David has always been a self-defined geek, he loved playing video games growing up and he especially loved playing D&D with his friends. Since he got married and had children, he has had much less time to plan and play D&D sessions with his friends, so is open to any ideas to make that process quicker and easier.

Their D&D Story: David started playing D&D in 1995 after he got a D&D guide for his 13th birthday. He immediately gathered his friends around and took them through a pre-made adventure from the book. Since then, David has created his own adventures from scratch, using nothing but his imagination.

Issues with Existing Generators: David has tried many different dungeon generators but he is fed up of one not offering everything he needs. Donjon doesn't populate the dungeon with monsters and other encounters. He doesn't like searching around to find all the parts he needs.

<u>Technical Ability:</u> As David is a software developer, he is highly skilled in technology and has access to all high-level tech. He uses a smartphone daily and has a house full of technology.

<u>Clara Standall</u>

<u>Age:</u> 26

<u>Gender:</u> Female

<u>General Information:</u> Works as a waitress in the city where she studied Maths at university.

<u>Background:</u> Clara has recently taken an interest in RPG games, she was introduced to gaming when she was a child but was never exposed to role-playing games. She went to study Maths at university where she spent 5 years, doing a masters degree.

<u>Their D&D Story:</u> During university, Clara was introduced to role-playing games by her roommates who were always into the format. She played a few campaigns with her friends and still considers herself a new player. Recently, she has missed playing, so has proposed that she host a game, so now she is looking into how to do that.

<u>Issues with Existing Generators:</u> As this is Clara's first campaign, she wants to make it as easy as possible. Clara has tried many generators but hasn't been able to find a perfect, easily usable one. Clara browses the internet on her smartphone.

<u>Technical Ability:</u> Clara doesn't use computers, she doesn't own a laptop either. All of Clara's interaction with technology is done using her Samsung smartphone, which she can use, but isn't an expert.

<u>Amber Brennan</u>

<u>Age:</u> 17

<u>Gender:</u> Female

<u>General Information:</u> Finishing college and hoping to go to university next year to start on her pathway to becoming a vet.

<u>Background:</u> Amber is a very enthusiastic student who loves trying new things. She is currently finishing her A Levels and is working really hard to get a place at her preferred university next year.

Their D&D Story: Amber has tried lots of different activities, most of which she doesn't really catch onto, but one that she really enjoyed, was role-playing games. Her first RPG was a simple board game but this sparked her interest and she later joined some people in her school who play D&D regularly. Whilst Amber really enjoys the game, she doesn't see herself being a DM anytime soon, she much prefers being a player and experiencing these stories first hand.

Issues with Existing Generators: As Amber isn't a DM and currently doesn't have any desire to be, she hasn't used any generators herself, however, the DM of her usual group does use generators and creates the stories that she subsequently plays. Amber sometimes feels that the story isn't quite deep enough and has a lot more potential to tell a better story, she thinks it's a waste of a very good environment.

Technical Ability: Amber uses technology all the time during school and out of school. However, most of her time spent on technology is on her mobile phone or her notebook laptop. She can use them but is by no means an expert in computers.

3.2 Requirements Identification

From these personas, we can derive a list of requirements that the system will aim to meet.

REQ1 - The system must be easily accessible to all of the target audience.

REQ2 - The system must be able to generate a dungeon with rooms and 1 or more floors. Features will be spread across the rooms, some with none, some with multiple. The generated dungeon should make sense with a creator, history purpose and location, and the content inside the dungeon will be reflected by these, naturally telling a story.

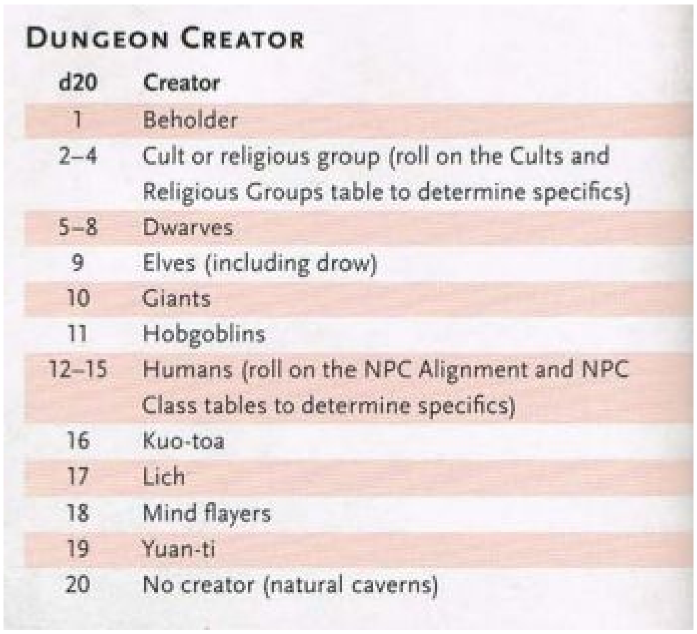
These two requirements have been deemed the most important two for our target audience and the best two to measure the success of the system as a solution to the personas goals.

3.3 Dungeon Design Algorithm

This section will explain step by step how I implemented the algorithm described by Reddit user u/phidinh6 that was used in the video game TinyKeep[2] and how I slightly modified it to allow me to meet my own requirements[3].

3.3.1 Dungeon Information

The first step of the dungeon generation, before I start placing rooms, is taken from the Dungeon Master Guide page 99-101(Fig 4), that describes a method of how to generate a location, creator, purpose and history. This creates a story for the dungeon and can also assist in the future generation. For example, if the creator of the dungeon turns out to be the elves that created the dungeon as a death trap, then the likelihood of traps and elf encounters should be increased for that dungeon. This is all handled by the Modifier system that I implemented, which will be explained in 3.6.2.


DUNGEON CREATOR

d20	Creator
1	Beholder
2–4	Cult or religious group (roll on the Cults and Religious Groups table to determine specifics)
5–8	Dwarves
9	Elves (including drow)
10	Giants
11	Hobgoblins
12–15	Humans (roll on the NPC Alignment and NPC Class tables to determine specifics)
16	Kuo-toa
17	Lich
18	Mind flayers
19	Yuan-ti
20	No creator (natural caverns)

Fig 4. The Dungeon Creator roll table from the Dungeon Master Guide (pg 100)

A small but impactful part of the dungeon generation is giving the dungeon a name. All of the premade campaigns come with a powerful name that sets the players expectations or provides them with a hint of the horror that waits within, all before they've even begun playing. The dungeon name can be a very important part of the campaign and the story it tells. To generate a dungeon name, I define some base layouts of a dungeon name, with slots to fill by choosing from a list of possibilities. For example, "The {adjective} {type}", could yield a dungeon name of "The Gloomy Caverns".

3.3.2 Room Generation, Placement & Selection

There are many layers to the general room generation. The main object is a class *Dungeon* that contains a list of the object *Floor*. A *Floor* represents a level of the dungeon as dungeons can spread across multiple levels. Floors are then split into sections, so that there can be parts of floors that aren’t connected, allowing for a castle with multiple unconnected spires to be possible. These sections are where the rooms are generated.

The first step of room generation is to fill the section with rooms. Rooms are generated up until the total room area has surpassed a percentage of the available space in the section. This threshold is default 65%, but dungeon modifiers can change this amount. Room size is generated using Gaussian normal distribution, this ensures a general room size, but also allows for variety. Excessive sizes will be ignored later on in the generation.

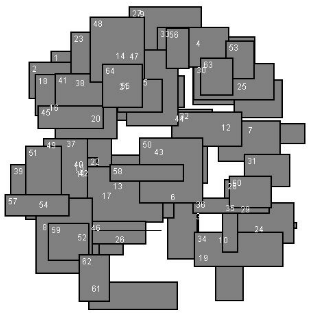


Fig 5. The first step of generation upon completion.

These rooms are generated within a certain radius of the centre of the section. Once the generation is completed, the rooms are all crammed together in the middle, which is not the desired result. So I use a procedure called separation steering behaviour, that makes the rooms push each other apart and forces them to move away from each other (Fig 6). There are some catches in this, it is entirely possible for rooms to get caught in a pushing loop, where they would constantly be moving each other back and forth, so I had to implement some blocking cases that identified repeating movements then just marks the room as rejected to break the loop.

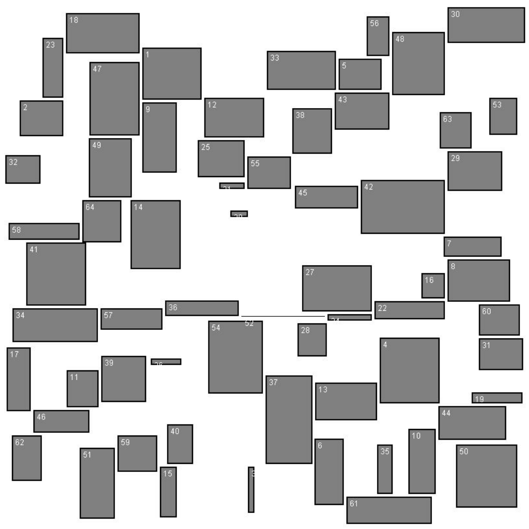


Fig 6. Separation Steering results

Once the rooms have been separated out, we need to choose which rooms are valid and which aren't viable as rooms. This is done by checking the room size against the normal distribution used to generate the sizes. If both the height and width of the room are within 70% of the centre then the room is accepted. All rooms that fall out of that threshold are rejected and don't get to the next stage of calculation (Fig 7).

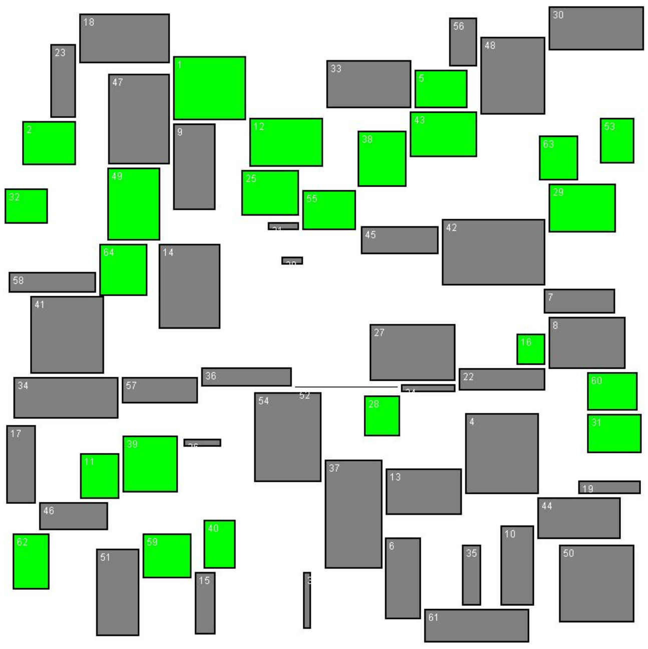


Fig 7. Third step after completion. Green rooms are the accepted ones and all others (grey) haven't been accepted due to their size.

3.3.3 Connecting Rooms

After the approved rooms have been selected, we need to generate the connections between them, this is done in several steps. Firstly we run an algorithm called Delaunay Triangulation. Delaunay triangulation takes in a set of points, P, which in this case, is the centre of each accepted room, and calculates a set of triangles such that no point in P is inside the circumcircle of any triangle in DT(P) (Fig 8). This gives us connections between rooms, but it doesn't always guarantee connections, for example, if there were 2 rooms, there wouldn't be a triangle. Also, the triangulation has a pattern to its result, which is why these aren't the final connections between rooms, we save these edges for after the next step has completed.

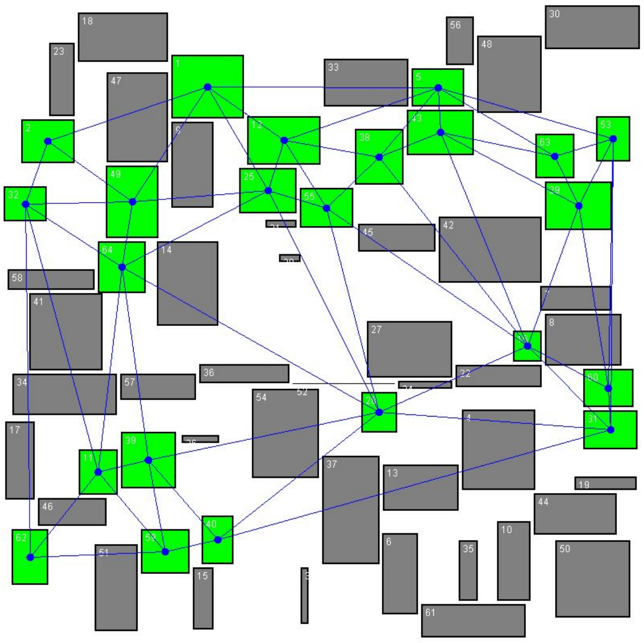


Fig 8. Step 4, the Delaunay triangulation between accepted rooms

The next step of the algorithm is to apply a minimum spanning tree to the same set of points P (Fig 9). This is the part that ensures all rooms are reachable. However, with this, it can lead to very linear progress, which might be okay for some situations, but in most D&D premade dungeons, there are loops in corridors, this is where the Delaunay triangulation comes back in. We add some of the edges from the Delaunay triangulation to the minimum spanning tree to create loops in our dungeon. For every edge in the triangulation, there is a chance that it will be added to the final list of corridors.

The chance is default to 15%, but again, that can be changed using dungeon modifiers.

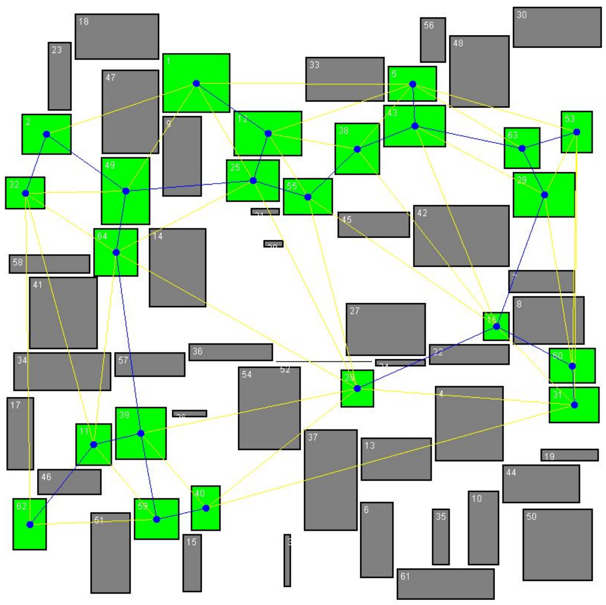


Fig 9. Step 5, Applying a minimum spanning tree to the rooms. Minimum spanning tree can be seen in blue.

Now there is a set of rooms, and a set of connections between them, the next step is to generate the corridors that will connect the rooms. To complete this, I need to implement a pathfinding algorithm that will create straight or ‘L’ shaped corridors. There are many pathfinding algorithms out there that will do the job, but I already have a solution that I wrote myself 2 years ago for my university software engineering project. I implemented the A* pathfinding algorithm[25] in Java for my project which was a 2D top-down exploration game, as the game was tiled I was able to reuse the code for this project, with some minor changes (Fig 10).

However, during the final stages of development, I found a constant issue was that it took on average around 3-7 minutes to complete generation which is too long to generate a dungeon for a good user experience. I looked further into what was slowing down the process, and it turned out to be the A* pathfinding. As a combination of Dijkstra and Greedy pathfinding, it explores a lot of nodes to find the best path available, which in a map of 800x800 can be a lot of nodes. The algorithm is run for every connection on the map, which causes a long time delay. After some research, I decided to slightly modify the algorithm to take a more greedy approach, which caused the algorithm to speed up significantly but may result in non-optimal corridors.

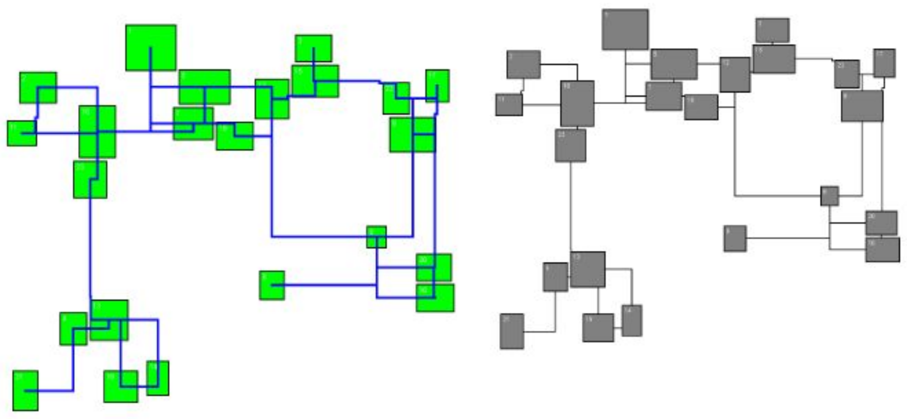


Fig 10. Final Step, Greedy pathfinding algorithm to generate corridors between the rooms (Left) and the final dungeon (Right).

3.4 Room Features System

3.4.1 Thralls System

As previously mentioned, Thrall's[19] solution had the rooms generated be of a specific type, either Monster, Trap, Special or empty. Rooms could only be one of the four, which proposed a huge limitation for D&D campaigns. This limitation was an issue when it came to expanding upon Thrall's system to implement my requirements.

My requirements would have added at least two new room types, whilst easy to do, these types need more information. For example, if a room was type stairs, those stairs would have to have a destination and a description of what type of stairs it is. All of this would not be possible with Thrall's existing system and I would need to completely re-do this system to meet my requirements.

My motivation for the need to improve or reimplement this system is so that the generated dungeons can be more representative of an accurate dungeons and dragons dungeon, similar to those in premade campaigns.

3.4.2 New System

Implementing my own system in the new solution would be best for meeting my requirements. First I will need to design the system so that it can meet all my needs.

The ideal system will allow for a room to have multiple features, that each has its own information that can differ per room. For example, 2 rooms that have a “monster” feature, can represent a different type/count of monster(s).

When I had finished the layout generation, I discovered adding features to a room wasn’t going to interfere with this layout generation, as the creation of features is handled within the room, therefore abstracted away from the Dungeon. To implement the features, I created an interface called *RoomFeature* that implemented the methods that would be required by all features, such as *generate(), getDescription()* that the dungeon could use to easily get the information about all features.

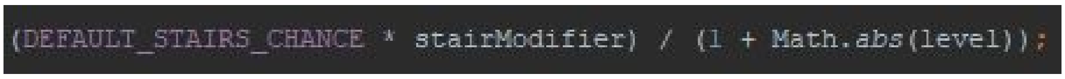
The individual features have a static base chance of being generated in a room, for example, a monster feature has a 15% chance of being placed in a room, for each room. However, this chance of being generated is affected by changes applied to the system by Modifiers. Modifiers are explained in more detail in 3.6.2. Modifiers contain a field that will change the chance of each feature occurring in a room. For example, the modifier from the selected creator of the dungeon might say that monsters are more likely to occur in this dungeon, so it will double the chance from 15% to 30%, resulting in the monster feature being in more rooms. Modifiers have a different field for changing the possibility of appearing for every feature.

This system will result in much more detailed dungeons which more accurately depict the sort of environment that D&D players will expect from a dungeon.

3.5 Multiple Level Generation

Dungeons can be home to some of the world's most valuable treasures or the scariest of monsters, but it’s not always the case that dungeons are contained on a single floor, you often come across stairs, or ladders or even a caved in ceiling. All of these scenarios would be impossible to achieve in Thrall's[19], Brown's[18] and even donjon's dungeon generators. My system will aim to allow the possibility for dungeons to spread across multiple floors, enabling the players to explore the deepest of dungeons or the tallest of castles.

Rooms having stairs is the first step to allowing branching generation, so a new feature was added that can be generated into a room. The *StairsFeature* was added with a default chance of generation being 5%. As there was a potential issue where it could eternally branch, creating floor after floor, I had to modify the chance algorithm so that as you got further from the ground floor (level 0), the chance of generating decreases, the formula(Fig 11) makes it less likely for stairs to occur the further away that floor is from the ground floor.



```
(DEFAULT_STAIRS_CHANCE * stairModifier) / (1 + Math.abs(level));
```

Fig 11. Formula to derive the chance of a StairFeature being placed in a room.

For example, assuming the default chance being 0.05 (5%) and the *stairModifier* being 1, the chance of a room generating stairs on different floors is displayed in the below table.

Level	Chance of StairsFeature
0	5%
1	2.5%
-2	1.666%

Once the stair feature has been placed in a room, all the handling of direction and floor generation is handled by the Feature object. Stairs can either go up or down, and the chance of this decision is default 50/50, but again this can be changed in the modifiers. A castle dungeon location changes the up/down chance to 80/20 and increases the chance of stairs occurring, which on average should result in a tall dungeon.

When a direction has been chosen, the room feature tells the dungeon that they would like to connect to a floor either 1 above or 1 below. Then, if there is already a floor in the given direction, it creates a new section on that already existing floor, else, it creates a new floor, and adds a new section centered around the location of the room that generated stairs. When a floor begins placing rooms, it locks itself from adding new sections, should another floor attempt to do so. This means that if another floor attempts to add to this floor, it won't create a new section, but later on, would choose a room on that floor to connect to. It is completely possible that during room placement, several rooms have stairs leading to the same floor. When this occurs, it's possible for the sections placed by rooms to be overlapping. To solve this, the floor will compare each section to the others, if it finds overlapping ones, it will combine them to a new one, and rerun the checking algorithm, eliminating all overlapping sections and ensuring there are no overlapped rooms on this floor (Fig 12). Originally, this algorithm produced a new section which encased both of the overlapping sections (Left of Fig 12), but I found that this ended up filling a lot of empty space which could then encase another section that wasn't originally overlapping, I found this to be unwanted behaviour. So I changed the algorithm to take half the total width and height of the two sections and centered between them (Right of Fig 12). This achieved the desired behaviour.

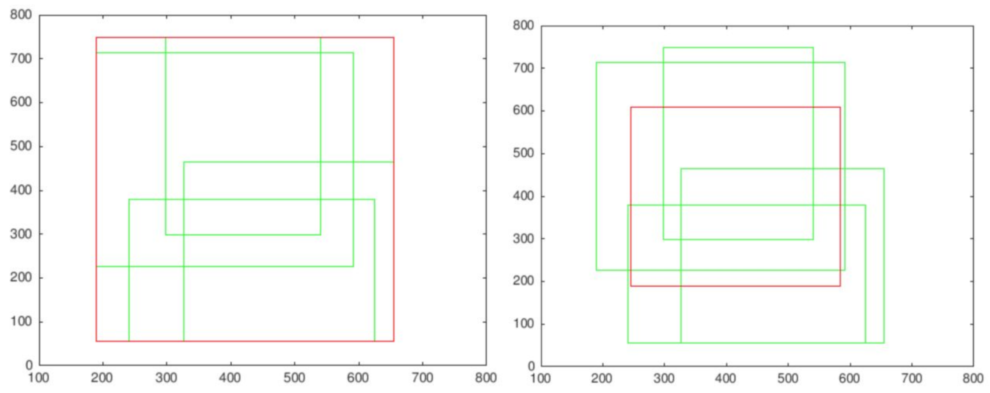


Fig 12. Example section combination with the original algorithm (Left) and the final algorithm (Right). The green squares were the overlapping sections. Red is the resulting combined section.

Due to the order of execution, it is possible that when a room decides that it will have stairs, that the floor it leads to has already been placed rooms, and a new section cannot be added to it. Therefore, it simply chooses the room that is closest to its position on the other floor and that room becomes the target room.

The room that is chosen as the target room (the room that these stairs lead to), is also given the StairsFeature with a reversed description (stairs down become stairs up) and the starting room as a target.

3.6 System Parameters

3.6.1 Input Parameters

Before the user chooses to generate the dungeon, they are presented with a few customisation options to change the result of the dungeon. My generation algorithm allows for a few input parameters that will change what style of dungeon gets generated. For example, the room size is generated on a normal distribution with a standard deviation, if we change the mean and the standard deviation, we can change the size of the rooms generated. As this would result in different dungeon coverage (percentage of the dungeon floor covered), changing the room size will also affect the number of rooms generated in a section.

Another simple modification that can be made is with the minimum spanning tree, if the user wants corridors to be longer because they want to add encounters in them, then we simply change the minimum spanning tree to a maximum spanning tree, which will lead to longer corridors being generated by the pathfinding algorithm.

Finally, java’s Random library allows a Random object to be seeded, which will guarantee the random numbers generated will be the same each time, for the same seed. My system allows users to input a seed on which to generate the dungeon, this means that a user can share their seed with a friend who can then use the same software, input parameters, and seed, generate the exact same dungeon.

3.6.2 Modifiers (Internal Parameters)

Modifiers are used to manipulate the generation parameters internally so that we can generate themes within a dungeon. Modifiers are mostly supplied to the dungeon from the attributes of the dungeon which are explained in section 3.1. Each of the four attributes, location, creator, purpose and history have modifiers that have effects on the outcome of the algorithm.

Each modifier defines how it wants to change the generation parameters, then when it’s time to start generating, the dungeon collates all the modifiers together to generate the final modifier, which it will then apply to the respective fields. A table of all the fields of a modifier and an example usage is Appendix 2.

To collate all the figures of the modifiers together, I take the average of all the modifiers that want to change an attribute. So for example, if two modifiers want to change the likelihood of monsters being place in a room, one by 0.7 (reduce by 30%) and one by 1.5 (increase by 50%), then the final modifier will change the monster likelihood by 1.1 (an increase of 10%). Some of the fields are chance modifiers, but others are static numbers, the size of a monster group for instance. When a monster feature is generated it has 2 fields, one is the type of monster, and the other is the number of monsters in this group. The average size of a monster group is default to 3, but a modifier might say that it wants to increase that number, then the average of all the new numbers will be the final amount, any values left default will be ignored.

The end goal of the modifier system is for the generated dungeon to reflect the dungeon attributes that were generated for it. This will ensure that the dungeon, its inhabitants, its encounters, all fit together and make sense. If the user reads that the dungeon was created by the hobgoblins as a treasure vault, then the players will expect the dungeon to be full of treasure and hobgoblins guarding the riches. The modifier system allows these situations to occur.

3.7 User Interface Design

User interface design is a key part of software design as a user needs to be able to use your system with ease. If not, it could result in them using the software incorrectly, being unable to utilize all of its features, or worst case, just giving up with your software. Achieving a good UI is best achieved when using user-centred design. UCD is the iterative design process which is dictated by the user's needs[26]. The user (or someone who represents the user and their needs) should be involved in every part of the process to ensure the final product or service is tailored to their specific needs. Moxey addressed user-centred design in the banking industry. She claims that traditional banks have shown "resistance to technical change" until recently, ever since banks such as Monzo[27] have surfaced. Moxey states the reason that these apps have gained popularity is their ease of use and their pleasant looking apps, all due to their UCD approach. I will be taking a similar UCD approach in the design of my app, however, as I won't be able to have direct contact with potential users for ethical reasons, I will be representing the end user in the form of the personas defined in section 3.1.

3.7.1 Thrall's UI Feedback

As part of his project, Brown[18] performed a heuristic analysis on his supervisor, Dr Rob Alexander, and recorded all of Rob's feedback on the UI. Most of the feedback of the test was centred around the user experience, with common complaints like "very confusing", "not clear" and even complaining that there are no instructions, which, if the UX was designed well, wouldn't be required anyway. This was, in my opinion, a huge drawback to the software as the user is not going to be able to confidently create a dungeon if they can't use the software.

This feedback was for both Thrall's[19] original generator and the generator that Brown build on top of that, however, I will also discuss the issues raised with Browns UI as they are very similar and the feedback is valuable to designing my UI.

Rob Alexander states that after the “Generate Dungeon” button has been clicked, the program halts for around 10 seconds, which is whilst the algorithm is running. There was no UI feedback to let the user know if something was happening so to the end user, it appeared as though the software had just frozen. Once the generation had completed, the resulting dungeon was displayed to the user by opening a tab in their browser which Rob claims was “mildly surprising” which again, a user shouldn’t be experiencing in a well-designed system.

To conquer these issues, I will use this heuristic report to design my UI so that most of the issues raised by Rob won’t be a problem in my solution.

3.7.2 Ideal User Interface

After the feedback on Brown[18] and Thrall[19]’s UI, I will be reimplementing the UI to attempt to remove as many issues as I can with their system. The built-in UI for java, the swing library, isn’t really good for an intuitive UI as it is quite old and doesn’t offer most of the features that we see in our day to day use of software. Due to this lack of features and for ease of use, I will port over my solution to an android app which will increase the usability and accessibility whilst also allowing me to build a better UI based off of Google’s Material Design guidelines[28].

An Android app will be the best option for the solution as it drastically increases accessibility. Based on the user personas, a mobile app is the best way to reach all of our potential users, as not everyone is skilled in using a desktop PC if they even have access to one. In 2017, Google announced that its Android operating system had over 2 billion monthly active Android devices, making it a viable target operating system, ensuring it will reach a large proportion of the target audience[29]. It is for these reasons that I came to the decision to implement the solution in an Android app.

4 User Evaluation and Results

4.1 UE Methodology

To evaluate the effectiveness of my software, I will be sending out a questionnaire to people who have experience being a DM to use my solution and give feedback on their experience. From this questionnaire, I want to obtain an idea of what DMs with different amounts of experience think about the level of detail included in this dungeon generator and overall how many DMs believe this to be a good tool for making campaign creation a simpler task. J. Rowley gives features that are needed to successfully conduct a survey [30]. She states that the first thing to make sure is that the questions use a language that your target audience understand, so I will be writing my questionnaire assuming the user has no advanced knowledge about D&D but are able to contextualize words such as “campaign” and “adventure”. Rowley also recommends using a mixture of open and closed questions, closed questions for the data that you want to be able to quantify and open questions for conveying precise thoughts and opinions. I will want to quantify the amount of experience a DM has had, so I will be using a closed question for that and will also like to ask what the user feels can be done to improve the software, so I will use an open answer to allow them to put whatever they want.

This questionnaire will be used to examine whether this system has met the requirements defined in 3.2. The requirements were set based on the goals of my personas which represented the general user base. I will also use this questionnaire to study the effectiveness of the new UX approach taken.

My questionnaire will not require the subject to actually run a campaign using the dungeon, but to consider the steps needed in turning this dungeon into a campaign. This will be easier on the subject as planning a D&D session can take quite a lot of time and effort. I will be looking for feedback on what made creating the campaign easier for the user, and where the user feels it could make creation even easier. Final questionnaire is in Appendix 1.

4.2 UE Analysis and Results

I posted the questionnaire with a link to download the app to a subreddit dedicated to DMs[31]. I also sent the form to some close friends who I know have led D&D campaigns before and they also shared it with their friends. All results from the questionnaire are in Appendix 3. The risk with sharing with an anonymous community is the chance of the form being filled with fake information or someone could fill the form in again and again. Fortunately, this wasn't an issue as the submissions were detailed enough to be credible. I got 25 responses to the questionnaire with the app receiving 98 downloads as of 24th April 2019. Overall the feedback on the app was positive, and people had lots of suggestions to improve the functionality. Most of these had to do with the dungeon generation itself. I specifically asked people who are or who want to be a DM to answer the questionnaire, to get a wide variety of experience levels.

Figure 13 shows the wide variety in self-judged experience levels. Of the 25 responses, 24 of them had at least "some" experience being a DM, leaving one who was wanting to be a DM but yet to run a session. The experience of the DMs is widely spread, quite a few people said they have only run between 1 and 5 campaigns, each consisting of multiple sessions. There were also some highly experienced people who had been DMing for years, one person said they run weekly sessions and have been for the last 2 years, and someone said they have been DMing for over 15 years.

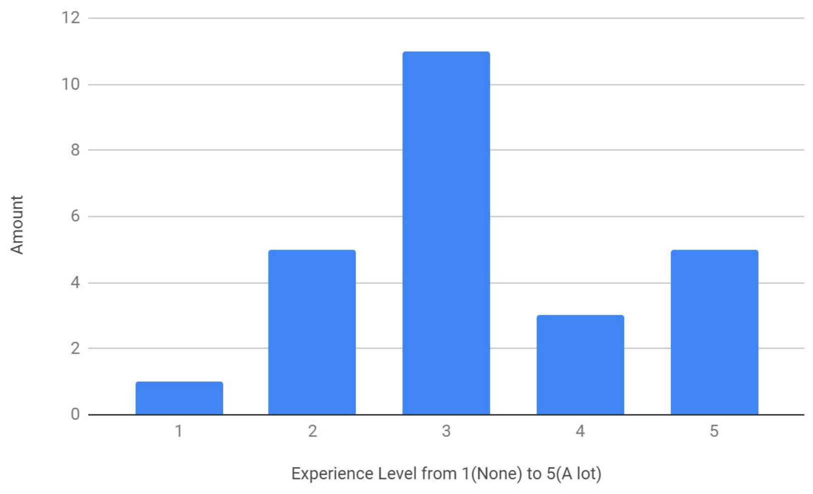


Fig 13. Responses to question "How much experience do you have as a DM?" (Appendix 1)

Of all the people who took the questionnaire, 84% of people state that they have written their own adventure or campaign before so are familiar with the steps that need to be taken in order to produce a campaign which will give further validity to their answer to the question “Imagine you are to create a campaign using the generated dungeon. How difficult do you think it would be to go from the generated dungeon to a campaign with a story”. Whilst this is a lot of people who have written their own adventures, only 47% of these people have used a dungeon generator before, so the other 53% might not know what to expect from a dungeon generator and might either be expecting too much or be too easy to please and therefore slightly affecting their opinion on my generator.

The next question was to find out how intuitive to use the UI was and asked the user how “easy” it was to create and view their dungeon. On a scale of 1 (Very hard) to 10 (Very easy), the average was 8.6 with a standard deviation of 1.3. There was also some feedback on the UI in the open section of the questionnaire. Some people commented on the ambiguity of the slider titles (e.g Reduced, Lots) in the “Create Dungeon” section which allowed users to tweak the generation parameters. Example of this screen can be seen in Appendix 4, Figure 1. While this feedback is very valid, it would be difficult to make these labels more concrete, as the generated dungeon depends on many different parameters, making it impossible to guarantee for example an exact amount of monsters if the monster slider is set to maximum.

Overall, the feedback on the app was overwhelmingly positive and a few people were saying they would like development on it to continue, and most people giving some feedback on what they thought would make the generator better. Specific feedback will be discussed more in sections 5.2 and 5.3.

5 Analysis and Critique

5.1 Project Evaluation & Requirements Analysis

Looking back at the requirement set in part 3.2 we can evaluate how effective the project was in meeting these requirements for our end user.

REQ1 - The system must be easily accessible to all of the target audience.

This requirement was to ensure that all of our user base (the personas) had access to our system that could help them meet their goals. The success of this will be measured both by the fact that all our users had access to a mobile phone, so will have access to the app, but also by the results of question 5 of the questionnaire (Appendix 1) which was “How easy to use was the software to generate and explore your dungeon”. As previously mentioned, the average response of this question was 8.6, which indicates that the users found the UI easily usable, therefore the requirement has been met.

REQ2 - The system must be able to generate a dungeon with rooms and 1 or more floors. Features will be spread across the rooms, some with none, some with multiple. The generated dungeon should make sense with a creator, history purpose and location, and the content inside the dungeon will be reflected by these, naturally telling a story.

This requirement was intended to measure the success of the algorithm itself in meeting the aims of the project, to generate a dungeon that tells a story. This requirement has largely been met, however sometimes the stories the environment tells doesn’t quite have a common theme.

Overall, the project has been a success and met all the requirements. The feedback on the solution was very positive and I am happy with the result.

5.2 Known Limitations

Known limitations of the system have been raised by both myself during development and by subjects of my questionnaire. There were many areas of the solution where I felt needed improvement or needed to be built in a completely different way. The solution was only built for the Android platform, therefore people who use Apple's iOS won't be able to use the system, which might not be a huge issue given the operating system market share (Appendix 5).

There was a lot of feedback from the questionnaire that raised some issues with the current implementation. The most frequently raised issue was to do with the treasure and trap features within rooms and compared them to the monster feature. The monster feature contained a detailed description of exactly what monster was in the room and how many there were, even showing the user the specific stats of the monsters so they didn't have to look it up elsewhere. However, for the trap and treasure features, it simply displays a string that is chosen from a static list. To implement this solution would be very simple and contained within one class. Due to the structure of the features system, modifying and adding new features is easily achievable.

There was also some feedback for the user to be able to slightly modify sections of the dungeon after generation, which in fact is what Brown's[18] project did as an extension of Thralls[19]. However, with the way the algorithm was designed, it would be incredibly difficult to modify the dungeons individually after generation, it would require some modification to the algorithm.

Other popular feedback addressed what monsters are chosen for a dungeon. In D&D monsters have a challenge rating which dictates how difficult that monster is to fight; the lower, the easier it is to kill. This value is not used in my system at all, the monsters are chosen from a list of monsters with the preferred types defined by the modifiers. 5 of the feedback forms and other feedback given via reddit comments mentioned how you couldn't input the players levels or the maximum monster challenge rating, to affect what type of monsters are placed in the dungeons. Sometimes the list of prefered monsters is empty and then it just chooses a random monster from the entire list of monsters, which is a major contributor to the issues with dungeons not having a common theme.

5.3 Potential Future Additions

For this project, I aimed to build a solution that met the requirements identified from the personas, however, the dungeons generated by this solution still require work to turn them into a playable campaign. In the dungeon master guide, there are still some unimplemented tables that generate information about the story, for example, why the adventurers are exploring can be generated from a lookup table. This piece of information can be very important to enthusiastic players as it affects their motivations behind exploring.

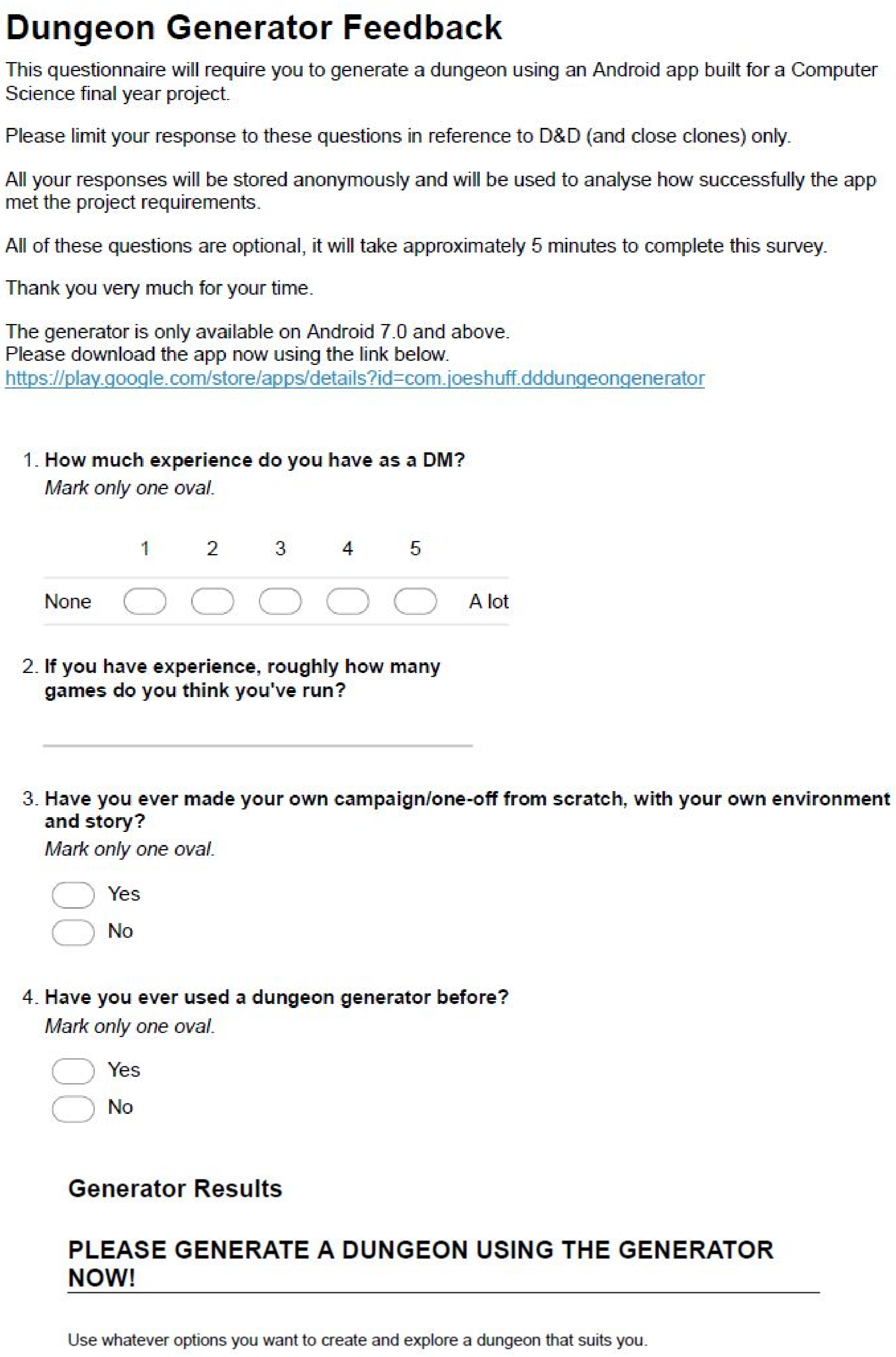
The major feedback points from the user questionnaire were the issues mentioned in 5.2, starting with the monster challenge rating. Adding a field for the users to input a maximum challenge rating will make the dungeon more suitable for a campaign, as with the current solution there could be unnecessarily easy or difficult monsters in the dungeon. Overall the whole monster selection would benefit from a refactoring as there were some issues raised that the monsters didn't seem to have a theme and some monsters didn't make sense in the given environment.

The other frequently raised issue was with the trap and treasure room features. The detail in these is severely lacking in comparison to a monster feature, and yet traps and treasure could be more detailed than a monster. Some people mentioned that the descriptions for these features was repetitive, this could be solved if there was a more detailed system for defining traps and treasure, this could result in unique encounters discovered in each dungeon.

In conclusion, the current system does it's job very well but there is still room for improvement to allow the system to further achieve its goals in creating playable environments with a detailed story.

Appendices

Appendix 1 - Questionnaire



Dungeon Generator Feedback

This questionnaire will require you to generate a dungeon using an Android app built for a Computer Science final year project.

Please limit your response to these questions in reference to D&D (and close clones) only.

All your responses will be stored anonymously and will be used to analyse how successfully the app met the project requirements.

All of these questions are optional, it will take approximately 5 minutes to complete this survey.

Thank you very much for your time.

The generator is only available on Android 7.0 and above.
Please download the app now using the link below.
https://play.google.com/store/apps/details?id=com.joeshuff.dddungeongenerator

1. **How much experience do you have as a DM?**
   Mark only one oval.

	1	2	3	4	5	
None	○	○	○	○	○	A lot

2. **If you have experience, roughly how many games do you think you've run?**

3. **Have you ever made your own campaign/one-off from scratch, with your own environment and story?**
   Mark only one oval.
   - ○ Yes
   - ○ No

4. **Have you ever used a dungeon generator before?**
   Mark only one oval.
   - ○ Yes
   - ○ No

Generator Results

PLEASE GENERATE A DUNGEON USING THE GENERATOR NOW!

Use whatever options you want to create and explore a dungeon that suits you.

5. **How easy to use was the software to generate and explore your dungeon?**
Mark only one oval.

	1	2	3	4	5	6	7	8	9	10	
Very Difficult	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	Very Easy

6. **How detailed do you think the dungeon layout and content is?**
Mark only one oval.

	1	2	3	4	5	
Not Detailed at all	⬭	⬭	⬭	⬭	⬭	Very Detailed

7. **Imagine you are to create a campaign using the generated dungeon. How difficult do you think it would be to go from the generated dungeon to a campaign with a story.**
Mark only one oval.

	1	2	3	4	5	6	7	8	9	10	
Very Difficult	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	Very Easy

8. **What features do you like or dislike about this generator?**
Give as much detail as you want. E.g "I like that the generator tells me exactly what monsters are where so I don't have to choose monsters"

9. **How likely would you be to use this generator again?**
Mark only one oval.

	1	2	3	4	5	6	7	8	9	10	
Very Unlikey	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	Very Likey

10. **Would you recommend this generator to a friend?**
Mark only one oval.

	1	2	3	4	5	6	7	8	9	10	
Very Unlikely	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	⬭	Very Likely

11. **Anything else you'd like to say about the generator?**

Appendix 2 - Modifier Table

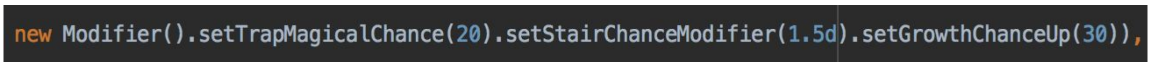
This table shows all the fields that are in a modifier.

Field Name	Field Description	Default
monsterChanceModifier	The chance of a monster feature being placed in a room is multiplied by this modifier to increase/decrease the chance.	1
monsterAverageGroupSize	This number is used to generate the number of monsters in a monster feature. It is generated using normal distribution, this number represents the average and has a standard deviation of 2.	3
trapChanceModifier	The chance of a trap feature being placed in a room is multiplied by this modifier to increase/decrease the chance.	1
trapMagicalChance	When a trap is placed, it can either be magical or mechanical. This number represents the chance of it being magical. If it's not magical, then it will be mechanical. 1 minus this value is trapMechanicalChance.	0.5
treasureChanceModifier	The chance of a treasure feature being placed in a room is multiplied by this modifier to	1

	increase/decrease the chance.	
mapCoveragePercentage	This value represents the coverage of rooms. So the bigger the number, the more tightly packed the rooms will be on the floor.	65
triangulationAdditionChance	When generating connections between rooms, the Delaunay triangulation edges get added to the minimum spanning tree. This is the chance for a single edge to be added. Increasing this will lead to more corridors.	15
bossChanceModifier	When a monster feature is generated with 1 monster, there is a chance this can be a boss. This changes the default chance.	1
growthChanceUp	When a stairs feature is generated, it can either go up or down. This dictates the chance of it going up. 1 minus this value is the chance of the stairs going down.	50
stairChanceModifier	The chance of a stairs feature being placed in a room is multiplied by this modifier to increase/decrease the chance.	1
preferredMonsters	This is a list of monster types, the list defines what the preferred monster types of the	Empty List

	dungeon is, to keep with a theme.	
blockedMonsters	This again is a list of monster types, but this one blocks specific monster types that won’t make sense in the selected dungeon environment.	Empty List

This is an example of a modifier that was created for the dungeon environment, Sewer.



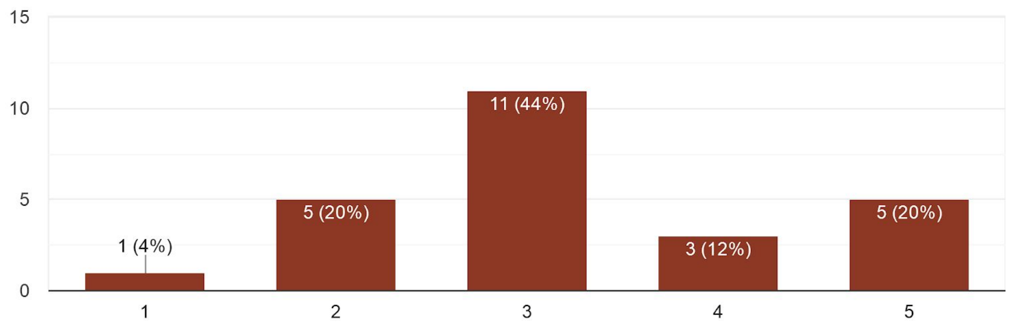
```
new Modifier().setTrapMagicalChance(20).setStairChanceModifier(1.5d).setGrowthChanceUp(30)),
```

This modifier sets the chance of a trap being magical to 20%, therefore making mechanical traps more common. It also changes how stairs are generated. With sewers being below ground level and more likely to grow down, the modifier increases the chance of stairs occurring by 1.5x and the chance of the stairs going up to 30%, meaning that stairs down are more common.

Appendix 3 - Questionnaire Results

How much experience do you have as a DM?

25 responses



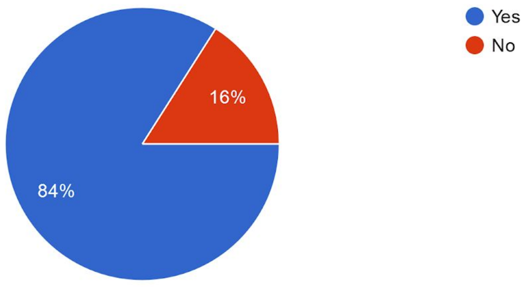
If you have experience, roughly how many games do you think you've run?

24 responses

1	8	4	50	500	20 or 30
3	1	10	7 or so	200+	3 full game
3-4	15	300+	20	60	2
Only 3	Weekly for the last 2 years, so about 104	Sessions? Maybe 20. 3 campaigns	Dmed over 15 years	2 different campaigns. Atleast 20 sessions	2 campaigns, though 1st ended after 2 sessions, second after 4

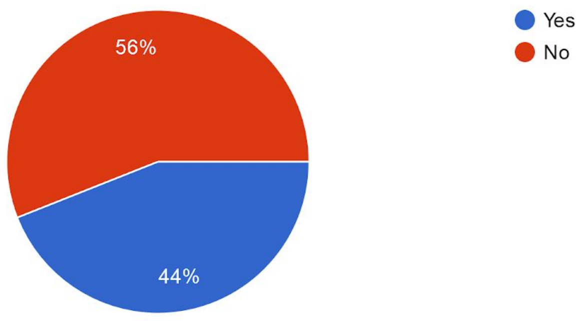
Have you ever made your own campaign/one-off from scratch, with your own environment and story?

25 responses



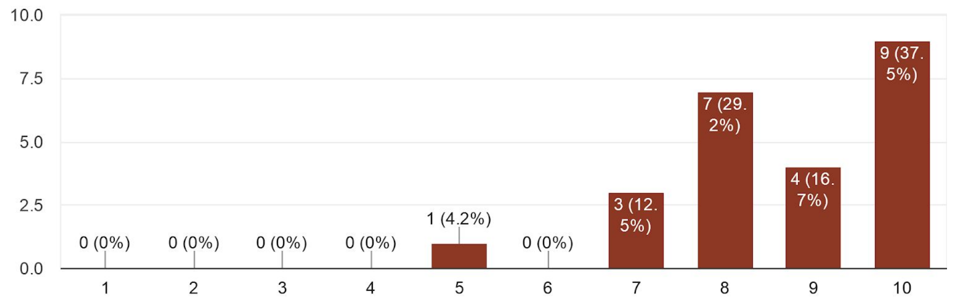
Have you ever used a dungeon generator before?

25 responses



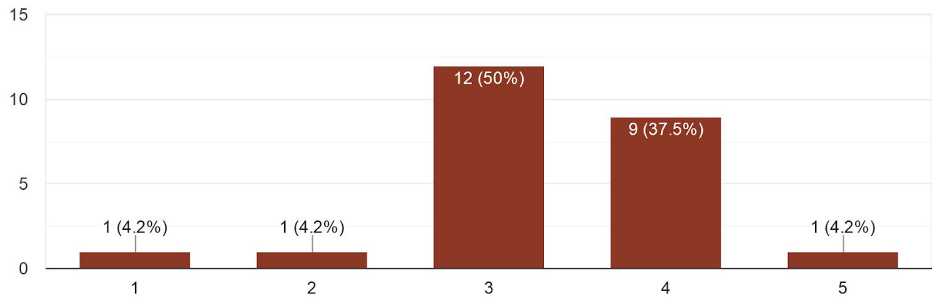
How easy to use was the software to generate and explore your dungeon?

24 responses

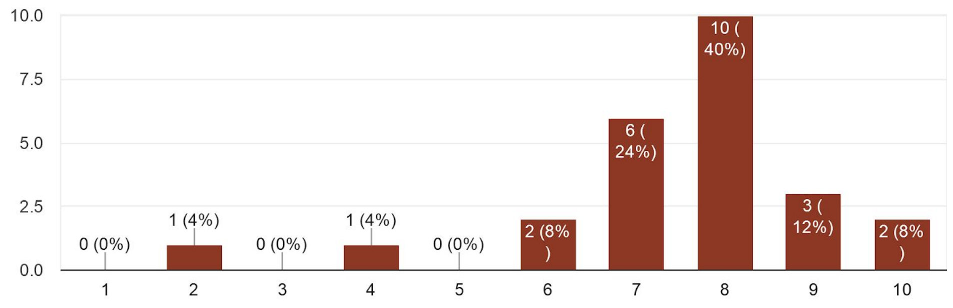


How detailed do you think the dungeon layout and content is?

24 responses



Imagine you are to create a campaign using the generated dungeon. How difficult do you think it would be to g...ted dungeon to a campaign with a story.
25 responses

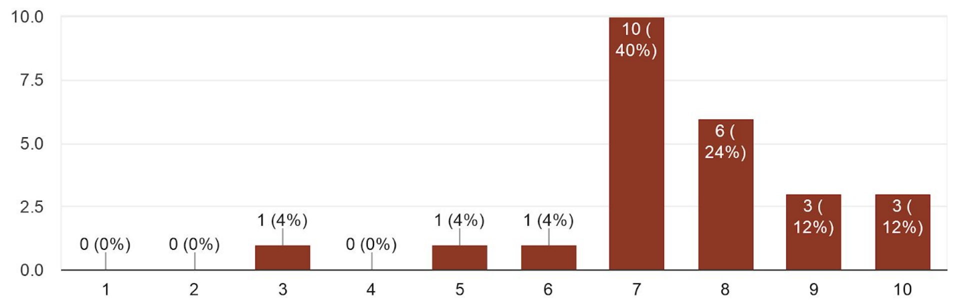


What features do you like or dislike about this generator?
21 responses

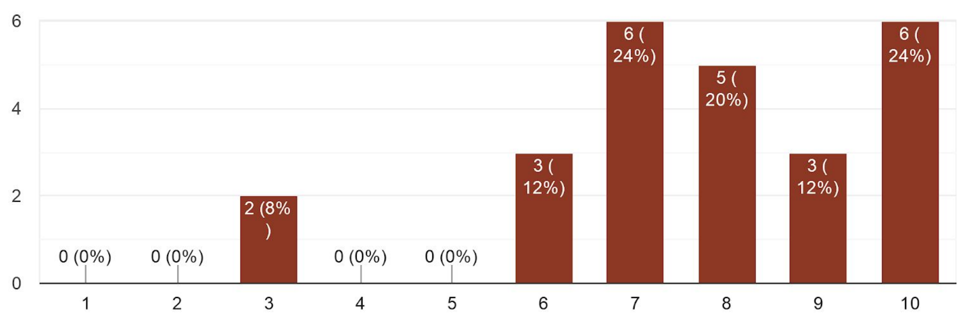
I like that it generates the encounters for you, but some of them feel quite repetetive and some seem to lack purpose or contradict other information
Love most things. Two pieces of feedback: able to set the difficulty (maybe max XP per encounter or something) and label the rooms without having to click on it.
I dislike how there's no setting for difficulty/CR of the dungeon. I dislike how the options for setting up the dungeon generation do not tell you what they all do (for instance, I can't tell what the tall/deep slider does after making a few). I like how I can jump between floors by hitting the goto button. I like how you give an explanation for how the dungeon came to be and what I/the party should expect the theme to be.
I dont like that traps aren't specific. It just says "Refer to traps in the Dungeon Masters guide"
It's a good starting point for something that could get a lot better. I like having some general filler in most of the room but would like to see something done with the corridors that connect them. It also can be a little difficult to find where the stair for each level is located.
Easy to use. Doesn't specify where traps are, rooms seen very random, perhaps an outline of what the building looks like would be good
It is great having the monsters' stats and treasure listings right there in the room. The room descriptions are also very handy.
I like that there are some descriptions of the rooms, even if its not very detailed and repeats itself often (statues with gold coins) I would like to see more detailed maps as well, this just kinda looks like someone had fun in ms paint haha
I think there should be difficulty settings, such as put in players and levels, then

determine monsters based off of CR. Also treasures and traps should be specified
I don't like that the connections between rooms have no properties. I like the details of the descriptions, and that the monster statblocks are included. (ps I couldn't find the obligatory license text?)
Like the on the fly ability to create dungeons. Like the monster stats. Would like filters to make a dungeon more thematic (vampire vs kobold lair, environment, etc.). Needs more details regarding treasure value.
I like the links to the PHB and the monster stats. The map is simple but I like that because it allows me to draw based on the initial layout. I don't like how it gets repetitive quickly. Also, I didn't like an implication of a puzzle but no idea of what the puzzle is.
On my phone, the last option during creation was almost hidden by the generate dungeon button and I couldn't scroll. Would also like the treasure to be generated like the monsters rather than just a description
I like the little story hooks in the rooms, but I dislike that the map isn't really usable in a game.
I dislike that it's not on a grid. It doesnt make traps for you which I also dont like. Dont like how hard it is to find room 1. No clear entrance. Not enough customization options
I like the little details it gives for each room.
Since it saves the seed andeverything. it needs the ability to adjust the creatures/traps and make notes digitally. Otherwise you end up drawing the map and rewriting everythinf yourself. But its super helpful, so if you can add that, 10/10
Humm maybe im just dumb but to put my own setting and make it easier to find the entrance maybe
Rooms are to the foot instead of rounding to 5ft. No level adjustment for monsters. No overview of what's in the dungeon
I don’t dislike anything yet
Monsters are unrelated to eachother. The dungeon seems to lack an overall theme.

How likely would you be to use this generator again?
25 responses



Would you recommend this generator to a friend?
25 responses



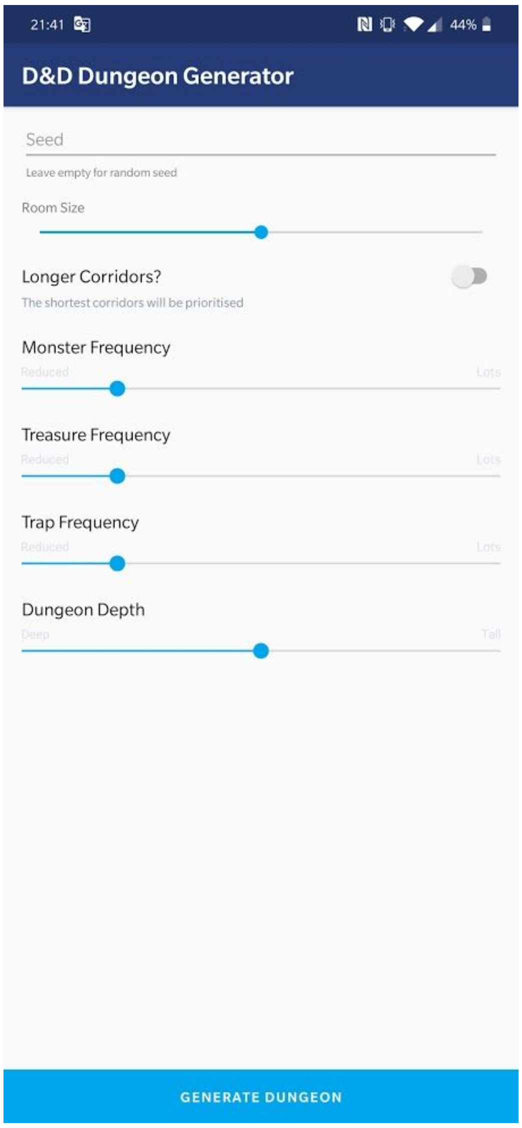
Anything else you'd like to say about the generator?

13 responses

Good work!
Overall better than any online generators I've used before, a lot cleaner too. Only real hangup is being able to tune monsters to the CR I'm looking for and making sure those monsters aren't out of place based on the setting provided. As far as QoL, I'd like to see more information with the sliders. For instance what each does specifically, and how extreme are the ends of the sliders (is it no traps at "reduced" and one per room at "lots"? Those are very abstract labels).
This is awesome!!!
Great work!
Great work. With further development, this could turn into my favorite Dnd app
Keep it up. Add a feature to select level range to increase or decrease monster levels to match a party
Next step would be to be able to re-generate a single room
Amazing work!
There seem to be some stability issues. Seed: 3514881559870 crashed the app. Also isn't including the rules for the monsters a breach of copyright?
It's a good idea. I may use it from time go time and see how it goes. I look forward to updates
Great so far. I'm really digging it. It's heading in the best directions! Excellent job.
Great job overall
I like the concept many dungeon generators are limited for multi floors. I'll be interested to see what it's like when it's finished.

Appendix 4 - Screenshots of App

Figure 1 - Create dungeon screen. This is the screen where the user chooses their input parameters for the dungeon.



Seed

Leave empty for random seed

Room Size

Longer Corridors?

The shortest corridors will be prioritised

Monster Frequency

Reduced Lots

Treasure Frequency

Reduced Lots

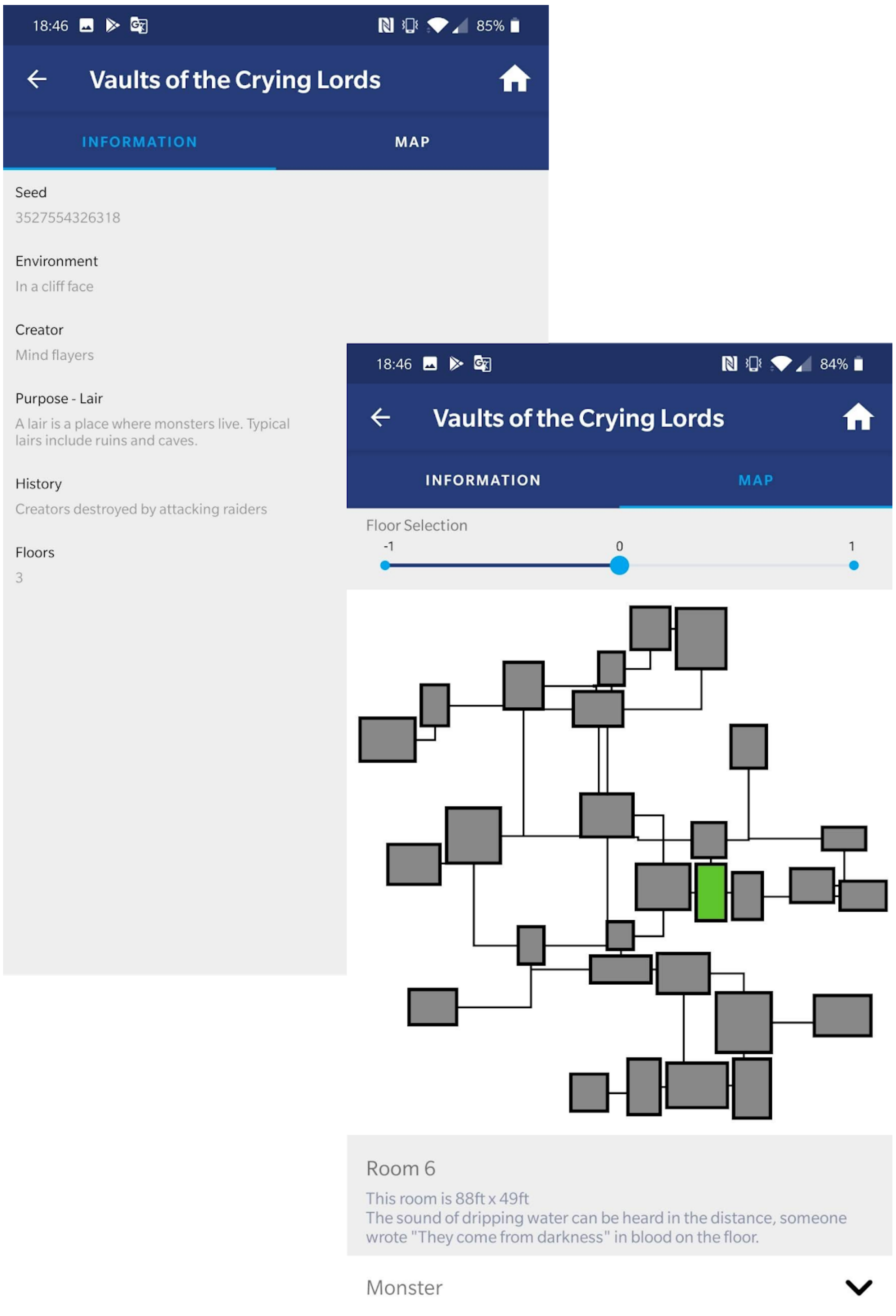
Trap Frequency

Reduced Lots

Dungeon Depth

Deep Tall

Figure 2 - How the app displays a generated dungeon to the user.



18:46 85%

Vaults of the Crying Lords

INFORMATION MAP

Seed
3527554326318

Environment
In a cliff face

Creator
Mind flayers

Purpose - Lair
A lair is a place where monsters live. Typical lairs include ruins and caves.

History
Creators destroyed by attacking raiders

Floors
3

18:46 84%

Vaults of the Crying Lords

INFORMATION MAP

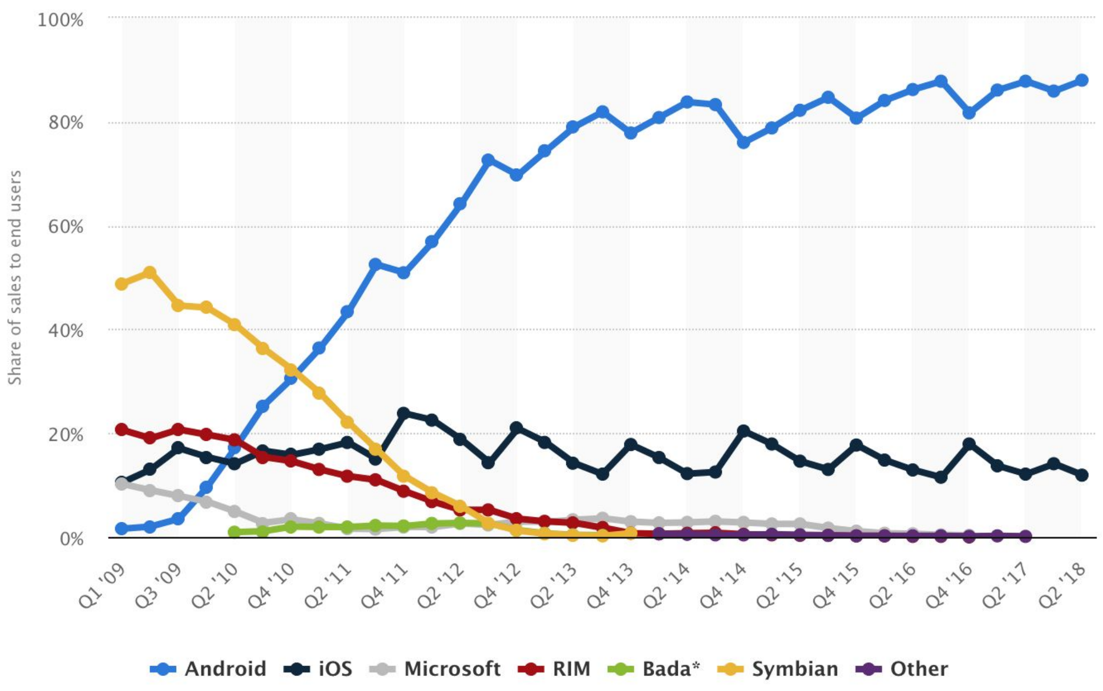
Floor Selection
-1 0 1

Room 6
This room is 88ft x 49ft
The sound of dripping water can be heard in the distance, someone wrote "They come from darkness" in blood on the floor.

Monster

Appendix 5 - Mobile OS Market Share

Mobile Operating system market share [32]



Bibliography

[1] Dnd.wizards.com. (n.d.). D&D Official Homepage | Dungeons & Dragons. [online] Available at: http://dnd.wizards.com/ [Accessed 12 Mar. 2019].

[2] TinyKeep. (2014). Phigames. [online] Available at: https://store.steampowered.com/app/278620/TinyKeep/ [Accessed 12 Mar. 2019].

[3] Reddit.com. (2019). Procedural Dungeon Generation Algorithm Explained [online] Available at: https://www.reddit.com/r/gamedev/comments/1dlwc4/procedural_dungeon_generation_algorithm_explained/ [Accessed 12 Mar. 2019].

[4] Cobley, P. (2001). Narrative. 1st ed. London: Routledge, pp.4-6.

[5] Gonehome.game. (2019). Gone Home: A Story Exploration Video Game. [online] Available at: https://gonehome.game/ [Accessed 12 Mar. 2019].

[6] Investing in the Soaring Popularity of Gaming. [online] Available at: https://www.reuters.com/sponsored/article/popularity-of-gaming [Accessed 4 Apr. 2019].

[7] Discord. (2019). Discord - Game Store. [online] Available at: https://discordapp.com/store [Accessed 12 Mar. 2019].

[8] Epic Games. (2019). Epic Games Store. [online] Available at https://www.epicgames.com/store/en-US/ [Accessed 12 Mar. 19].

[9] Life is Strange, (2019). Life is Strange. [online] Available at: https://lifeisstrange.square-enix-games.com/en-gb [Accessed 12 Mar. 2019].

[10] Edith Finch. (2017). Edith Finch. [online] Available at: http://edithfinch.com/ [Accessed 12 Mar. 2019].

[11] Porreca, R. (2017). A new Life is Strange game is in the works [online] Available at: https://www.destructoid.com/a-new-life-is-strange-game-is-in-the-works-436830.phtml [Accessed 4 Apr. 2019].

[12] Conditt, J. (2014). Gone Home finds 250k sales, most on Steam [online] Available at:

https://www.engadget.com/2014/02/06/gone-home-finds-250k-sales-most-on-steam/ [Accessed 4 Apr. 2019].

[13] What Remains of Edith Finch - SteamSpy [online] Available at: https://steamspy.com/app/501300 [Accessed 4 Apr. 2019].

[14] PC Adventure Games by Metascore - Metacritic [online] Available at: https://www.metacritic.com/browse/games/genre/metascore/adventure/pc?view=condensed&page=0 [Accessed 4 Apr. 2019].

[15] Cariota, C. (2018). Interactive Story Games That Will Change How You Think About Storytelling. [online] Available at: https://medium.com/@chelseacariota/3-interactive-story-games-that-will-change-how-you-think-about-storytelling-fb4530feaec7 [Accessed 4 Apr. 2019].

[16] Stewart, B. (2015). Environmental Storytelling. [online] Available at: https://www.gamasutra.com/blogs/BartStewart/20151112/259159/Environmental_Storytelling.php [Accessed 4 Apr. 2019].

[17] Red Dead Redemption 2 - Metacritic. [online] Available at: https://www.metacritic.com/game/playstation-4/red-dead-redemption-2 [Accessed 4 Apr. 2019].

[18] Brown, S. (2019). *Prototyping software for the complex human task of creating Dungeons and Dragons dungeons with Narratives, using user centred design and procedural generation.* MSc Software Engineering. University of York.

[19] Thrall, H. (2017). *Procedural Generation of a Dungeon for Dungeons and Dragons*. MSc Information Technology. University of York.

[20] Minecraft - Official Site. [online] Available at: https://www.minecraft.net/en-us/ [Accessed 12 Mar. 2019].

[21] Persson, M. (2011). Terrain Generation, Part 1. [Blog] *The Word of Notch*. Available at: https://notch.tumblr.com/post/3746989361/terrain-generation-part-1 [Accessed 6 Mar. 2019].

[22] Whittaker Diagram - Procedural Content Generation Wiki [online] Available at:

http://pcg.wikidot.com/pcg-algorithm:whittaker-diagram [Accessed 4 Apr. 2019]

[23] Mearls, M., Crawford, J., Schwalb, R., Thompson, R. and Lee, P. (2014). Dungeons & Dragons: Dungeon Master Guide. Renton, WA: Wizards of the Coast LLC, pp.99-101.

[24] Mark Barlett, “Users and Personas”, HACS, University of York 2015.

[25] Lester, P. (2005). A* Pathfinding for Beginners. [online] Available at: http://csis.pace.edu/~benjamin/teaching/cs627/webfiles/Astar.pdf [Accessed 4 Apr. 2019].

[26] Moxey, R. (2019). What is user-centred design and how are fintech companies using it? Here are some tips on how to implement a user-centred design when creating a new financial product. [online] Available at: https://eden.agency/lab/blog/fintech-design [Accessed 7 Apr. 2019].

[27] Monzo - The bank of the future [online] Available at: https://monzo.com/ [Accessed 21 Apr. 2019].

[28] Google Material Design [online] Available at: https://material.io/design/ [Accessed 4 Apr. 2019]

[29] Popper, B. (2017). Google announces over 2 billion monthly active devices on Android. [online] Available at: https://www.theverge.com/2017/5/17/15654454/android-reaches-2-billion-monthly-active-users [Accessed 4 Apr. 2019].

[30] Rowley, J. (2014). Designing and using research questionnaires [online] Available at: https://doi.org/10.1108/MRR-02-2013-0027 [Accessed 24 Apr. 2019].

[31] Subreddit: Dungeon Masters Tips and Tricks [online] Available at: https://www.reddit.com/r/dungeonmasters [Accessed 24 Apr. 2019]

[32] Global market share held by the leading smartphone operating systems in sales to end users from 1st quarter 2009 to 2nd quarter 2018 [online] Available at:

https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/ [Accessed 8 Apr. 2019]