MDA of TF2 Board Game

Group 1: Lincoln Freedman, Yuzhe Xie, Lanqing Gao, Serena Yang

Team Log:
https://docs.google.com/document/d/1hk5714j_vZjtmSp7jNeuCHimlrwmCwZufEf_UCIlHy8/edit?tab=t.0

Trello Board: https://trello.com/b/y3nJIvLn/gsnd-5110-project-2-group-1

MDA Analysis

TF2 Overview

Keywords: Tactical, Fellowship, and Individualistic

Team Fortress 2 (TF2) embodies a unique blend of visual and gameplay elements that support the following qualities:

Despite being a fast-paced shooter, TF2 emphasizes coordination between different character classes to achieve objectives. Players must think strategically about positioning, timing, and roles, often making use of specific class strengths to control the battlefield. The game's class system creates a strong tactical layer, where each character has a specific role to play—be it the Engineer building sentries, the Medic healing, or the Spy infiltrating enemy lines. This requires players to adapt their tactics based on the class makeup of both their team and the opponents.

Each class in TF2 is richly designed with unique personalities and visual styles. This individuality extends beyond appearance, as each class has a distinct playstyle—such as the Medic's healing strength or the Scout's agility—allowing players to express themselves through their preferred class. The game rewards individual skills, from mastering trick shots with the Sniper to perfecting the timing of headshots. This mastery over individual mechanics highlights the player's unique approach and skill set.

The individualistic aspect of TF2 gives the player a sensation of heated combat in a multiplayer battlefield. In our prototype, such a sensation is transformed into the excitement created by the randomness of dice.

In summary, TF2 blends tactical depth with a strong sense of individuality while maintaining fellowship and collaboration, encouraging players to use their characters' unique traits while strategically contributing to their team's success.

Our Adaptation

To actualize the game essence, we first analyzed each class's tactical value in a multiplayer setting from our gaming experience and professional plays. We identified key parameters that influence each class's performance, which are speed and range. The design question that can be answered by the prototype is how these two factors should be balanced in the game mode of capture points. While vertical space in 3D is also a primary factor that affects the balancing of each class in the original TF2, we omitted this factor in our prototype for the consideration of both complexity and not relating to our purpose.

We selected classes in TF2 that featured their mobility or shooting range. Though the balancing of all classes in TF2 is affected by their damage, range, and speed, a few classes rely heavily on them as they become their typical traits.

The classical TF2 game happens in a continuous 3D space with a linear map for capturing points in order. Our prototype divides the entire continuous space into multiple discrete spaces to represent possible encounters in different routes. This allows us to replicate the macro view of dynamics happening in a TF2 game and evaluate each class's performance under various strategic placements.

The classes in our prototype build on the abstraction of original classes in TF2. Due to our primary focus lies on the tactical aspect of TF2, attacks and weapons are valued through class abilities and modifiers. The modifier enables us to quantify the performance of class weapons. For example, Soldier has +2 modifier when facing more than 1 enemy. With such adaptation, the player would move each of their characters as chess pieces and precalculate their chance of winning while either taking risks in aggression or strategically countering the enemy formation.

The primary parameter that we are focusing on in this prototype is the class design. Thus, our prototype will be most useful for analyzing how the classes interact with each other (synergies, counters, etc.).

Prototype Design

Our Game

The game includes different classes each with different characteristics that serve as units with different functions in the game. While two opposing teams contend against each other, the team with better coordination and formation has a better chance of winning. The formation of the team also ensures the team is versatile in every situation in the battle. The element of shooter mechanics from TF2 is replaced with randomness, representing the possibility for one character to kill another in as simple a manner as possible.

Team Distribution: 1 vs 1 format in which each player controls one 'team'. there are 5 classes to choose from in our game. Each player chooses which classes they want each of their characters to be at the start of each game.

The player with the offensive token of a **gun** is the **offensive player**.

Offensive and Defensive:

The player with the gun token is **offensive** while the other player without the token is **defensive**. The **offensive** player takes their turn according to the following phases.

Game Start

- Both players must first choose the composition of units that they want to start the game with. This can be any configuration of five units, such as one of each class, five of one unit, or anything in-between.
- Players should write down their starting compositions and then reveal them to each other at the same time.
- The **red player** must then place their units freely anywhere on the map, either together or split in any way.
- The **blue player** takes the first turn.

Turn Phase Order

1. **Capture/Revert:**
    - For **blue player**: If any number of blue characters you control stand in the capture zone, **add** that number of **blue capture tokens** to the capture zone.

- For **red player**: If any number of red characters you control stand in the capture zone, **remove** that number of **blue capture tokens** on the capture zone.
- If any capture zone has a **blue capture token** more than or equal to **5**, remove all **capture tokens** on it. Place a **blue flag** token on it, meaning it has been captured. Fully captured zones can never be uncaptured.

2. **Move Character:**
    - The player may move each character on the board (including those within the respawn area at this current moment) to zones they can reach according to the character's **movement speed**. You cannot move them into your **opponent's spawn area**. You also cannot move them through zones occupied by enemy units (you can move them into these zones, but not back out).

3. **Respawn Character:**
    - If the number of characters **alive** controlled by the player is less than the **character limit** (5) of the game, the player may **revive** a character of **any class** and put them into the player's respawn area. The player may **repeat** the previous step until the number of characters they control reaches the maximum limit.

4. **Sniping Phase (only occurs during the blue player's turn):**
    - Both players can choose targets for their snipers to snipe, if any are in range. Note that this means the **red player's** snipers shoot during the **blue player's** turn, instead of their own. After sniping rolls, remove any character that died from the board.

5. **Combat:**
    - For each zone with characters from both teams, a **battle** happens between them.
    - Roll a d6 for each character in the zone. Then add the **modifier** to each player's rolls in accordance with their classes. Compare the sum of dice from both teams, the team with the higher total roll wins the battle. If players get the same roll result, everyone dies.
    - All **characters** on the losing team in the zone are **dead.** Remove them from the board.

6. **Exchange:**
    - Pass the **gun** token to your opponent. You now become **defensive** while your opponent is **offensive** and takes their turn.
    - **Increase the turn counter by 1**.

Winning/Losing:

Blue player wins if they successfully capture all capture zones before the turn counter reaches 40.

Red player wins if they successfully prevent blue players from capturing their zones before the turn counter reaches 40.

Classes
The number in the top right of the **class cards** shows the **movement speed** of units of that class. The number in the bottom left shows the **modifier** that it adds in battles. These values are also listed for each class below.

Classes (current finalized version after playtests and Adjustments)**:**

Design Justifications for Each Class:

- Scout: The most agile class with fast moving pace and captures zones faster than other classes. Known for running all round the map at a quick pace to harass enemies and capture zones.
- Soldier: Moderate health and speed, can attack quite effectively for both the offensive and defensive team, making him one of the most well-balanced in the game. Soldiers are known for their direct combat adaptability, making them helpful in group combats.
- Sniper: The best long ranged class, specializing in neutralizing targets from a distance. Sniper is very effective at targeting specialized classes such as Medic that add positive group modifiers.
- Medic: The main healer in the game. Since our prototype does not feature health, we decided to make him add modifiers to attacking units to simulate the effect of “instant

healing" during combat. Medic moves at a decent pace allowing him to stay close to other teammates and protect them when needed.

- Heavy: Moves slowly but very strong defensively. Known for being used as an "area of denial."

Classes:

Scout:

Speed = 3
Modifier = 0
Scout counts as 2 characters when capturing a zone.

Soldier:

Speed = 2.
Modifier = +2 when facing two or more enemies, +1 otherwise

Sniper:

Speed = 2;
Modifier = -3
(Snipe) Can snipe a target enemy character in zones within range of 2 during Sniping phase. Roll a dice. If the dice is 4 or higher, target is killed and removed from the game. Snipers cannot shoot enemy units inside their spawn point. Furthermore, the sniper cannot shoot if there is any enemy unit within the same zone as it.

Medic:

Speed = 2;
Modifier = -3 + (2 * number of non-medic allies in the same zone)
(Pocket Healing) Can move three zones if it moves with a scout.

Heavy:

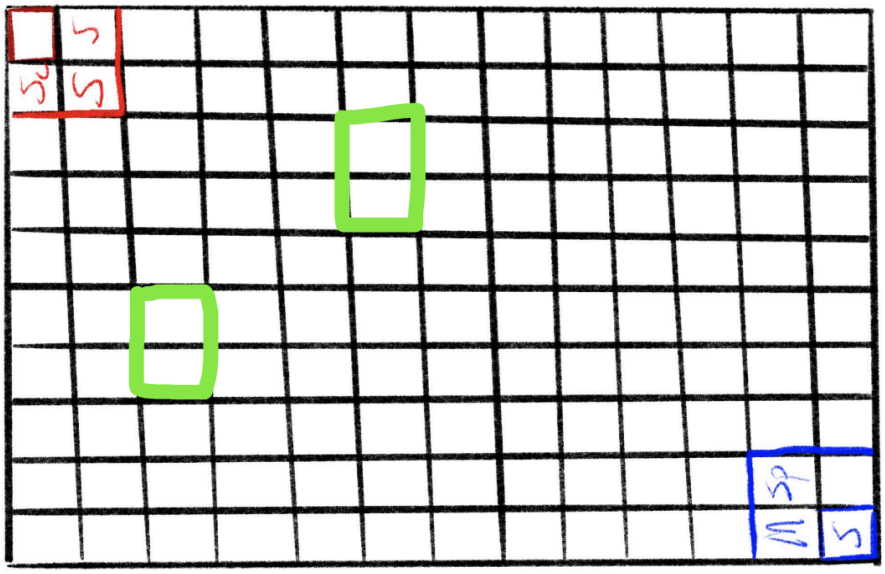
Speed = 1
Modifier = +4 on Defensive turn, 0 on Offensive turn

1. Scout can move three zones and capture zones double the speed of other classes.
2. Soldier gains dice roll advantage when fighting multiple enemy characters. (+2 modifier for 2 or more enemies)
3. Sniper has -3 modifier in combat. Sniper can only snipe on defensive turn. Sniper can only snipe enemies in adjacent zones. When sniping, roll a d6. Enemy is killed when the roll is 4 or higher. Blue team can snipe on offensive turn and red team snipe on defensive turn.
4. Medic (Heal) has disadvantage on all combat rolls (-3 modifier), but grants an increased dice modifier for each ally in the same zone (+2 modifier to allies). Cannot heal other medics.

Playtest and Adjustments

Initial Board Design (two versions to choose from)

- We had two different designs for the game board.
- Version 1 involves moving in a grid system in addition to using the class abilities. Combat happens whenever two units clash.
- Version 2 excludes movement across grids altogether where units travel from area to area. Combat breaks out simultaneously for every character on the opponent team in the same zone.
- Here is a picture of the initial game board for version 1:



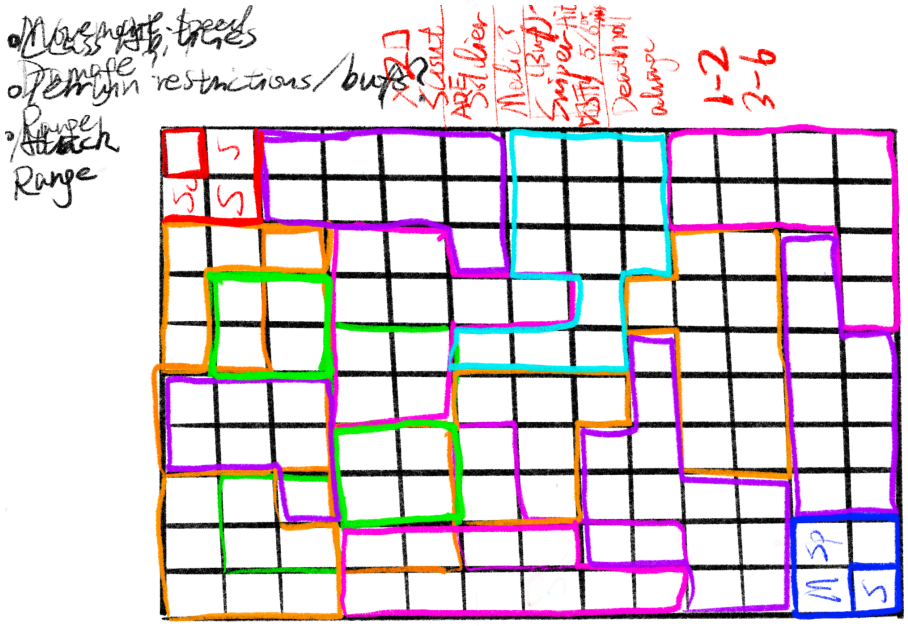
- After a complete failure of trying to use the grid version because it takes forever to progress in things since characters die very easily and will be sent back to spawn point too frequently, we decided to use version 2 instead. At the time of this playtest, the classes had the following designs:
  - ❖ Scout: speed = 2 grid, d6
  - ❖ Soldier: speed = 1 grid, AOE shockwave (shaped like a cross) range = 2, d6
  - ❖ Medic: speed = 1 grid, Healing range = 1, d6-3
  - ❖ Sniper: speed = 1 grid, sniping range = 5, d6-4 in the same combat zone, roll 5-6 to succeed in sniping
- As we can see above, another clear issue with the initial class design is that some classes lack differentiation. For instance in practice, scout and soldier played very similarly because the attack range of soldier limits the position where the shockwave can take hit and scout moving 1 extra unit doesn't make him fast enough, at some point the two became very similar in game play. Another major issue was that soldier's cross shaped

AOE attack made it difficult to plan its movement and position so that it will attack the enemy at the right spot.

- So overall, this made board 1 our better alternative, which at the time still needed to be tested.

Board Design Improved (previous version 2)

- After the failed version 1, version 2 is being tested.
- The quick sketch looked like this:



- Each area is mapped out with different colors to simulate different battle zones. Units in the same battle zone fight each other. When died, characters respwan from the spawn area.
- At the time, we were playing with almost a paper prototype, drawing how each unit moves by simply writing and erasing on a drawing tablet.
- Despite the primitive setting, this game design with the board was proven to be much more effective. Combats happened much faster and progression of the game was made. Because grid was no longer a restraints and every unit in the same zone would attack each other, AOE attack of soldier was also canceled. Medic's ability also had to be changed accordingly since healing range was no long relavent given he has to be in the same zone with the character he is healing.
- At the time, rules of classes were changed to the following:
    - ❖ Scout: speed = 2 zones, d6
    - ❖ Soldier: speed = 1 zone, d6, +1 for every enemy unit in the same zone
    - ❖ Medic: speed = 1 zone, d6-3, +2 modifer on every ally unit in the same zone

    - ❖ Sniper: speed = 1 zone, sniping range = 1 adjacent zone away, d6-4 in the same combat zone, roll 5-6 to succeed in sniping.
- Mechanics in combat resolution also changed accordingly. Both players in the same zone roll dices and whoever has the highest wins the battle, hence killing all the enemy units in the same zone.
- Despite the map improvement, abilities of the classes still had significant problems.
- For instance, soldier and medic become overly powerful when paired up. Since players are allowed to repeat classes as long they control a total of 4 characters (at the time), the game is essentially broken when such combination of classes are placed together.
- These issues needed to be fixed, so we made adjustments and entered the next stage of play testing.
- In addition, we built our game in Tabletop Simulator which made testing easier. Each piece is easier to move around and visualize. Dice rolling and calculation was convenient and us group members can run two playtest sessions at the same time separately before rejoining for discussion.

Class Balance Modeling

Because the primary element of TF2 that we chose to emphasize most in our prototype is the class system, it could even be said that the primary purpose of the prototype is to test how the classes interact with each other. One of the fundamental precepts that we discovered early on in testing is a sort of rock-paper-scissors relationship, by which scouts counter snipers, soldiers/medics counter scouts, and snipers counter soldiers/medics. We were pleased to see this relationship, as it adds significant compositional & strategic depth to the game, as well as being an accurate, albeit simplified representation of the class counters that exist in TF2.

However, as we tested more, we determined that a skilled player could often defeat a sniper based composition using only soldier + medic. Furthermore, scouts often felt extremely strong, even in circumstances where they should be countered according to the logic of TF2. This led us to feel that there was an issue with the game's balance, so we decided to numerically model each class' power in order to make some informed adjustments.

Initial Class Power Model

Class	Speed	Modifier	Ability	Total
Scout	2	0.5	1	3.5
Soldier	1	1	1	3
Medic	1	-3	5	3
Sniper	1	-3	4	2

For this model, the speed value was determined by the number of zones one unit could move in one turn. The modifier value was simply determined by what base modifier would be applied to the unit in dice battles (the scout's value of 0.5 is averaged between 0 and 1 for defensive and offensive battles respectively). The ability category was the most subjective and difficult to evaluate, with all team members discussing how many points we felt each class's special effects should be worth. From this process, we determined that the scout was too strong, while the sniper was too weak.

This led us to make several adjustments, the first of which was to remove the scout's +1 modifier on offensive rolls. This had initially been added to the game to make scout better at offensively assassinating isolated snipers and medics, however after more clearly understanding the mechanics of our own game, we realized that this wasn't necessary in the first place, as even 2 isolated snipers already had very low odds of beating a single scout *without* the +1 modifier. The next change we made was to increase the speed of all units from 1 to 2, except for scout which was increased from 2 to 3. The intent of this was to help offset the scout's movement advantage (effectively making it 50% faster rather than 100%), as well as to make games faster and increase the odds of a blue victory (more on this later). In order to match this global increase in movement speed, we also increased the range of the snipe ability from 1 to 2. Our new model looked like this:

Second Class Power Model

Class	Speed	Modifier	Ability	Total
Scout	3	0	1	4
Soldier	2	1	1	4
Medic	2	-3	5	4
Sniper	2	-3	4	3

This version of the game felt a lot better overall. However, the sniper still felt weak, as shown by the power model. We experimented with increasing its odds of a successful snipe from 50% to 67% as one potential way to buff the class. However, it still did not seem to solve the fundamental issue of snipers. The snipe ability is, at best, able to kill one enemy unit per turn. However, the fact that its range is the same as the movement of soldiers, and that it only shoots after moving, effectively means that it can never get off a shot without being in range to be attacked. In other words, the best scenario for the sniper is to kill one enemy unit, and then be attacked the next turn. Since the average roll of a d6 is 3.5, the sniper's effective contribution to battle rolls is 0.5. This means that, even after killing an enemy unit, it contributes almost nothing to the following battle, and has traded barely better than evenly.

Consider the following example scenario: 3 soldiers, 2 snipers VS. 5 soldiers. Even if the snipe has a 67% chance of succeeding, the 5 soldier composition will, on average, only lose 1

unit. The following engagement between 4 soldiers and 3 soldiers + 2 snipers will be greatly favored for the 4 soldiers, showing that the sniper would still be too weak even with this change. Of course, we could increase the snipe chance even more, to 83% or even 100%, but this would make the sniper a unit the purpose of which is essentially to trade 1 for 1 - not what we were looking for in modeling TF2. Rather, we felt it was necessary to make some change which could create the potential for a sniper to get off multiple shots before being forced to engage in combat. One possibility was to increase the snipe range even further to within 3 zones. However, we felt this might be easily exploitable, and decided on a different option.

This was to add the heavy - a new unit with a strong bonus modifier in defensive combat. The idea behind this was that snipers need a babysitter of sorts to stay either with or in front of them, and prevent soldiers and scouts from easily pouncing on them after they shoot. Furthermore, we decided that the heavies should have a slow movement speed, as they do in TF2. Tactically, this means that heavies are extremely bad at attacking enemy snipers, as their movement is outranged by the snipe ability. Thus, if heavies are commonly used, snipers as a direct counter will have their stock increased. In light of these changes, our final power model looked like this:

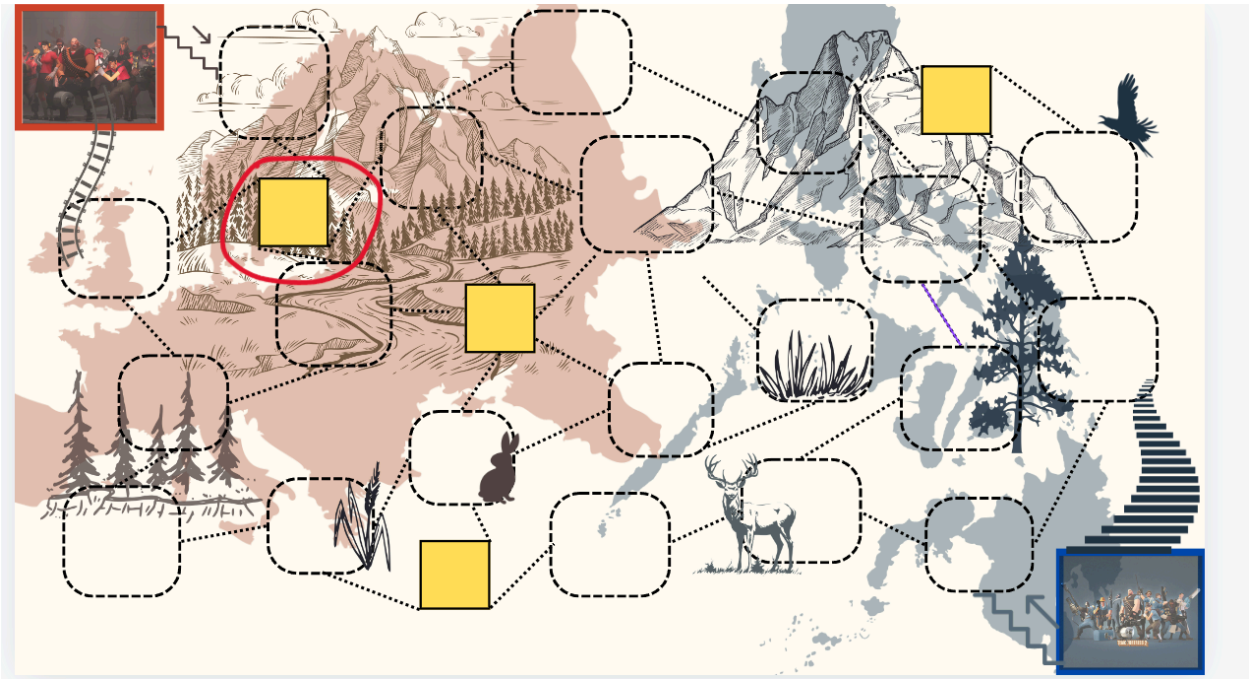
Final Class Power Model

Class	Speed	Modifier	Ability	Total
Scout	3	0	1	4
Soldier	2	1	1	4
Medic	2	-3	5	4
Sniper	2	-3	5	4
Heavy	1	0	3	4

Team Balance

In addition to class imbalances, our game has the potential for team imbalances, due to its asymmetrical design. Almost all of our earliest playtests resulted in a red (defensive) victory, which led us to believe that blue side (offensive) was underpowered. In order to fix this, we had several options of potential parameters that we could adjust. These were the number of capture tokens needed to capture a zone, the number of capturable zones, the positions of and accessible routes to the capturable zones, the movement speed of the units, and finally, the number of turns before a red victory.

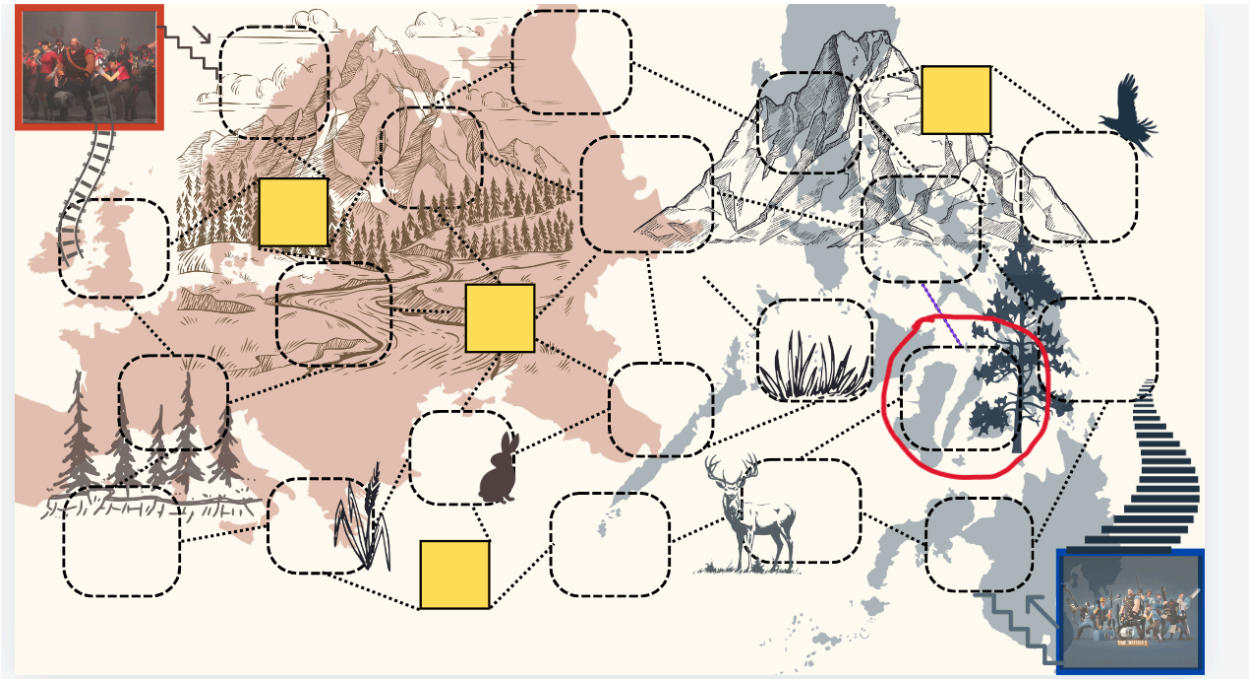
The first change which we chose to implement was to increase the movement speed of all units by 1. This was, of course, an extremely significant buff to blue side, as it reduced the number of turns that blue units would have to waste in transit to capturable zones. Following this, we also decided to reduce the number of capture tokens needed to fully capture a zone from 6 to 5. This change was especially important, because it allowed a full non-scout army to capture a zone in a single turn, rather than two turns. We had come across multiple scenarios while playtesting in which a red player would turtle with a strong battle composition (typically 4 soldiers + 1 medic) in the capturable zone closest to their base (pictured below).



Since this zone was within two movement range of the red respawn point, even if blue players won a critical engagement on the point, they would not be able to fully capture it. Rather they would only be able to place 5 capture tokens on the point before the red player's entire respawned army could engage once again. If each battle was viewed as a coin flip, then the blue player would essentially need to win two in a row (25% chance) to ever capture the final point. After changing the number of capture tokens needed to fully capture to 5, a single battle victory would suffice.

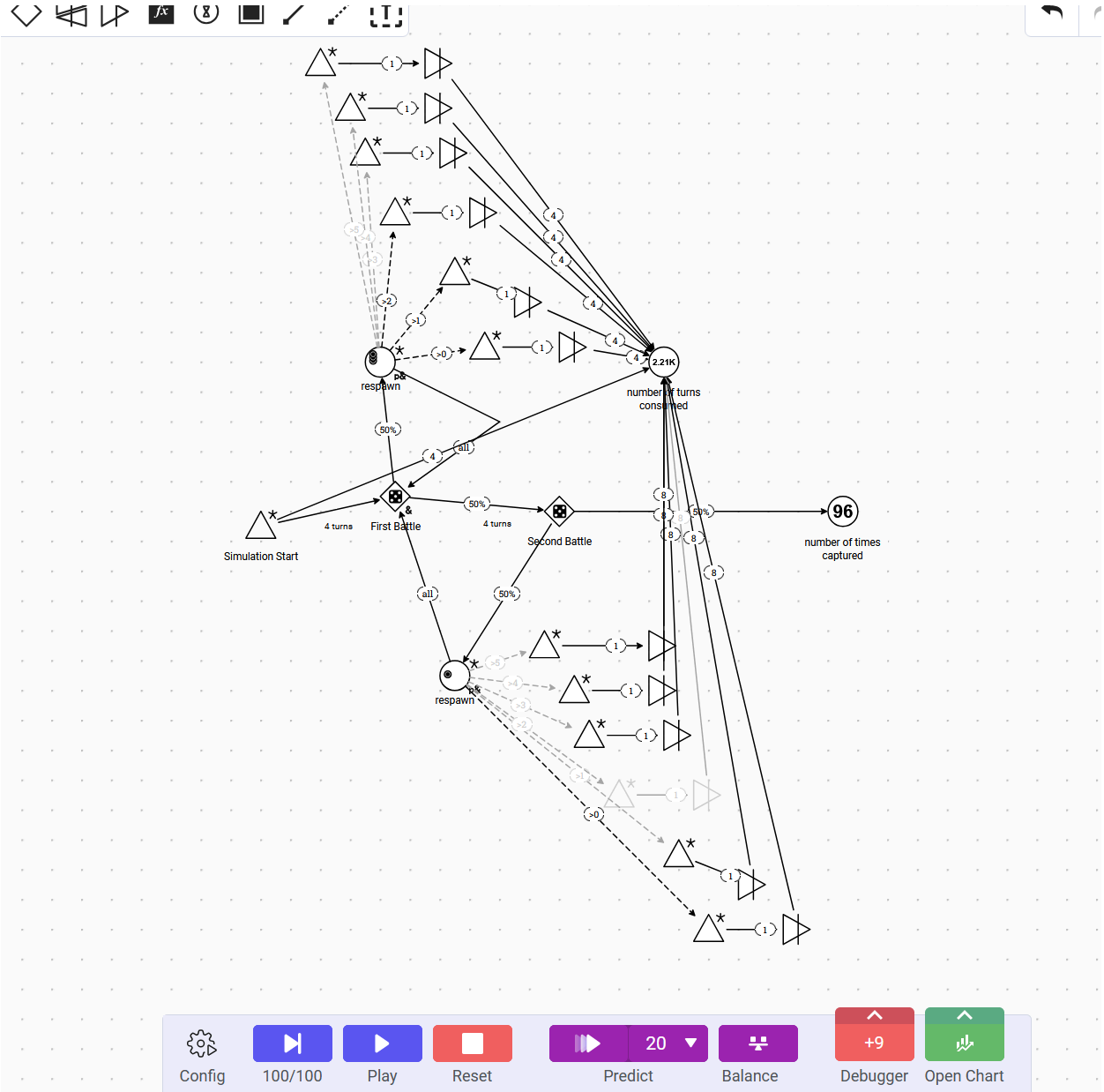
We attempted to use this form of 'coin flip' modeling further to determine the probability of a blue victory within a certain number of turns. To this end, we began with a simplified and generalized version of a common scenario that we had observed during playtesting - one in

which the blue player had already captured all three outer zones. Furthermore, within this scenario the blue player had just lost their entire army, and was respawning everything in base (this commonly occurred because blue players would use scouts to capture the outer zones, before sacrificing them to remake a stronger fighting composition of soldier medic which could actually break the final capture target). The red player would start with their entire army on the final capturable zone, and would take the first turn (because the blue player would have to actively attack with their scouts to sac them, and afterwards pass the turn to red). The red player would then move their entire army two spaces forward in order to block the blue player's advance. Since the blue player would have to spend their turn respawning an army after that (units cannot move on the same turn that they are respawned), the red player would get one more turn to move, allowing them to reach this zone:



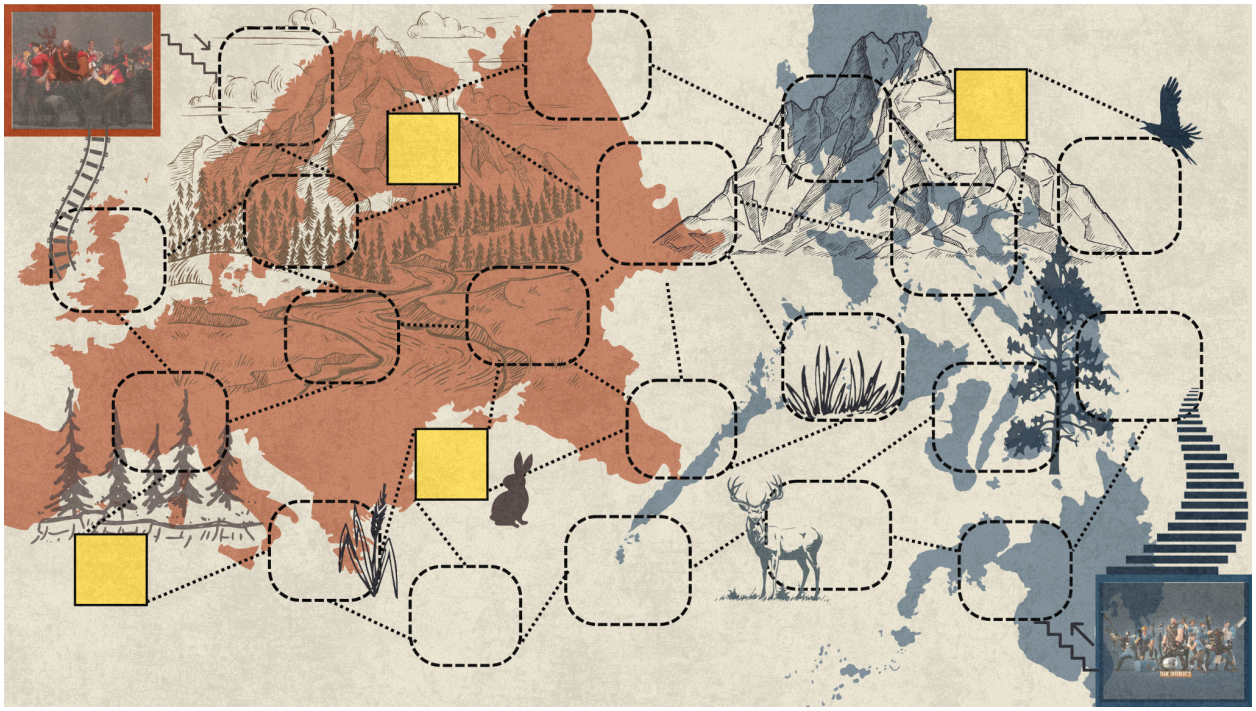
From here, the red army can always prevent a blue advance, even if the blue player attempts to use scouts. This means that an engagement was guaranteed to occur on the red player's following turn. 50% of the time, the blue player would lose this engagement, and would effectively be sent back to square one with four less turns to work with. The other half of the time, the blue player would win the battle. From here, it would take them four more turns (only two of their own movement phases) to reach the final capturable zone (unless they were using scouts, which would make their odds of winning a battle in the first place drastically worse than

a coin flip). This would give enough time for the red player to respawn their entire army, and still fight the red army on the capture target (this is only possible because the target is within two movement of the red base). This battle is another coin flip, with a 50% chance of resulting in a blue game victory, and a 50% chance of sending the blue player back to spawn with 8 less turns to work with. We created a flowchart in Machinations to simulate this scenario. The machinations link is available here.



(NOTE: the design of the flowchart is very inefficient and can be inaccurate under a very rare circumstance, but we couldn't figure out a better way due to our inexperience with the software).

From this, we determined that the blue player in this scenario will capture the final point and win the game in approximately 2210/96 = 23 turns. This felt not terribly imbalanced, but definitely a bit skewed towards red, seeing as it means the blue player must capture all 3 other zones within only 17 turns to have an even or favored chance of winning. For this reason, and to generally reduce the degree to which the game was centered around the capturable zone close to the red base, we chose to rearrange the map a bit.



In the new map, none of the capturable zones are within 2 movement range of the red base. So that the capturable zones are not all right next to each other, we moved two others as well. Based on our playtests, these changes seem to have made the game easier for blue, but not so much so that red is greatly disadvantaged. However, due to fatigue and general dissatisfaction with the Machinations simulation we had made, we decided not to attempt to model the victory probabilities on this new map. The Machinations model was not sophisticated enough to represent aspects of the game like sniper's range, scout's faster movement, or even how long it would take on average for blue players to capture the outer zones. We ultimately ended up feeling that Machinations was just fundamentally not the right tool for our project. We also did not feel that trying to determine the ideal number of turns to set the red victory condition to (or the ideal value for any other team balancing parameter) using pen & paper calculations would be

very fruitful. Instead, we just playtested more, and ended up feeling that the 40 turn win condition, as well as all other elements of red/blue balance, were quite fair.

30% Difficulty Increase

Because this prototype is competitive, increasing its difficulty is not a very straightforward thing to do. That we could think of, there were three completely different routes we could take to make the game harder:

- Make the game more difficult for either blue or red side. Of course, this would have the side effect of making the game easier for the other one.
- Make one particular class 30% stronger. This option was suggested to us by the professor.
- Make the game more complex. People will often say that more complex competitive games are more difficult, IE: chess compared to checkers.

Regarding the first option, we would say that we have already outlined numerous ways in which the game can be made more or less difficult for one side throughout the 'team balancing' section. The version of the prototype that we had before rearranging the map, for example, was certainly more difficult for blue, though probably not by 30%. We could also simply increase or decrease the number of turns necessary to trigger a red victory. Unfortunately, our Machinations simulation outputs the average number of turns needed to capture a zone, not the probability that a zone will be captured within a certain number of turns (let alone the probability that the game will be won within a number of turns). Even without being experts in statistics, we are fairly confident that decreasing the number of turns by 30% (ie: red player wins after 28 turns) would result in a much greater than 30% decrease in the probability of a blue victory. We could also argue that adding the heavy class resulted in an increase in complexity, making the game more difficult. However, numerically quantifying this increase would be even more difficult to do.

For these reasons, we chose to pursue option #2 - to increase the power level of individual classes. To this end, we used an online dice calculator to determine that the chance of a d6 rolling higher than a d6 +1 is 27.78%. Furthermore, the chance of the d6 rolling exactly equal to the d6 +1 is 13.89%. 13.89%. Under the rules of the game, equal rolls cause all units on both sides to die, so we can essentially count this as a half good outcome for both sides. The chance of a good outcome for the d6 is therefore 27.78% + (13.89% / 2) = 34.725%. Naturally, the chance for a good outcome (plus half the chance for an even outcome) between two d6 +1 is 50%. Comparing these two values, we found that 50/34.725 = 1.44, meaning that the d6 +1 is

44% better than the d6. This value wasn't the ideal 30% we were looking for, so we tried comparing d6 to d6 + 0.5 as well. Performing the same calculations (minus the part about even outcomes, since the 0.5 prevents those), we found that d6 + 0.5 has a 41.67% chance of winning. Since 50/41.67 = 1.2, the d6 + 0.5 is effectively 20% better.

The coveted 30% lies roughly in between the values of 20% and 44%. We thus concluded that a unit with +0.5 half the time and + 1 the other half would be approximately 30% better than other units. Fortunately our game already includes a conditional method for giving units modifier bonuses half of the time - this being the distinction of offensive and defensive turns. We could simply give each class + 1 on offensive combat rolls and +0.5 on defensive ones, thereby making each of them about 30% overpowered. However, this would not be a very interesting way to fulfill the requirement. Instead, we chose to give this bonus to each class in slightly different ways, as follows:

- Scout: We chose to simply give scout back its + 1 on offensive combat rolls, alongside the +0.5 on defensive ones. This fits the class identity of an aggressive unit that is stronger at attacking than defending well.
- Heavy: For this big guy, we chose to simply give him the opposite; +0.5 on offensive rolls, and +1 on defensive rolls. This accentuates the heavy's class identity of defensive play and acting as an 'area of denial' even further.
- Soldier: The soldier's rule was slightly more difficult to come up with. We decided on giving it +0.5 all the time, and an additional +0.5 when it is fighting 3 or more enemy units. The reason we chose 3 or more instead of 2 (which is the threshold which activates its existing +1 modifier) is because our playtests have shown that large fights are common, and the soldier tends to face 2 or more enemy units in combat significantly more than half the time.
- Medic: For the medic, we decided to do something somewhat similar; giving it +0.5 all of the time, and an additional +0.5 when it has all 4 allies in combat with it. We felt that the medic would be able to activate an ability conditional on being with 3 or more allies in combat too often, as there is little reason for a player to split a medic away from their main army. With this condition, the medic can at least be prevented from gaining its additional +0.5 (and thereby being effectively 44% overpowered) when one unit has been

split off of its army, or else killed by a sniper, which we felt would be the case approximately half the time.

- Sniper: In this case, the +0.5/+1 conditional modifier gain was not needed in the first place. The sniper's active ability has a 50% chance of succeeding in its balanced state. By simply changing its success condition from 'rolls of 4 or higher' to 'rolls of 3 or higher', we could easily adjust this success chance to 66.7%. Because 66.7/50 = 1.33, this change is enough to make the snipe ability (which is the unit's main form of value) 33% stronger. We decided this was close enough to our goal.