Tampere University of Applied Sciences

Designing and Creating a Deck-building Game

Marko Saari

BACHELOR'S THESIS
June 2024

Degree Programme in Business Information Systems
Games Production

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tietojenkäsittelyn tutkinto-ohjelma
Games Production

SAARI, MARKO:
Pakanrakennuspelin suunnittelu ja toteutus

Opinnäytetyö 30 sivua
Kesäkuu 2024

Opinnäytetyöprosessissa suunniteltiin ja toteutettiin prototyyppi pakanrakennuspelistä. Työ koostui sekä teoreettisesta osasta, jossa käsiteltiin pelien tärkeitä ominaisuuksia, että käytännöllisestä osasta, jossa esitellään prototyypin käytännön toteutusta. Vaikka työtä lähestyttiin korttipelien näkökulmasta, osaa sen sisällöstä voidaan soveltaa myös muiden genrejen peleihin.

Aiheen tutkimisessa hyödynnettiin olemassa olevia korttipelejä sekä aihetta käsitteleviä julkaisuja. Tutkimuksesta saatuja havaintoja pyrittiin hyödyntämään pakanrakennuspelin prototyypin kehityksessä. Prototyyppi toteutettiin Unity-pelimoottorilla ja sen verkkotoiminnallisuuteen käytettiin Photonia. Prototyyppiä kehitettiin useamman vuoden ajan, jonka seurauksena sen suunnitelmat ehtivät muuttumaan prosessin aikana.

Pelin prototyypin tekeminen oli arvokas kokemus, joka opetti tekijälleen paljon sekä pelisuunnittelusta että peliohjelmoinnista. Työn tuloksena syntyi pakanrakennuspelin prototyyppi, joka antaa jatkokehitykselle hyvän perustan.

Asiasanat: Unity, Photon, pelisuunnittelu, pakanrakennus peli, prototyyppi, peliohjelmointi

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in Business Information Systems
Games Production

SAARI, MARKO:
Designing and Creating a Deckbuilding Game

Bachelor's thesis 30 pages
June 2024

The main purpose of the thesis was to become better at designing and programming games. The written part of the thesis covers what elements are needed when designing games as well as design choices of the game project being developed. Although the thesis is written from the perspective of card games, some of its contents can be applied to games of other genres as well.

The topic was researched through studying existing card games and publications from professional game designers. The findings from this process were then utilized in the development of a prototype for a deckbuilding game. The prototype was implemented using the Unity game engine. The project's online functionality was done through Photon. The production of the prototype progressed slowly over several years, which allowed its design to change during the process.

Working on the project was a valuable learning experience that taught a lot about both game design and game programming. Although the resulting prototype still needs some improvements, it provides a good platform for further development.

Key words: Unity, Photon, game design, deckbuilding game, prototype, game programming

CONTENTS

GLOSSARY

CCG	collectible card game
PvP	player versus player
RPC	remote procedure call
TCG	trading card game

1 INTRODUCTION

Card games originate from ninth-century China where they were first played by using paper money as cards. Over the following centuries card games started appearing in other parts of the world while being heavily influenced by local cultures and symbolism. The first records of card games in Europe are set during the thirteenth and fourteenth centuries where they were brought through various trade expeditions. Card games quickly became a common past time activity and were often used alongside gambling and divination. The digitalization of card games began around the twentieth century with already popular games like solitaire. The new medium opened a world of possibilities when it came to unique mechanics and entertaining audio-visual effects. (Global Game Network, 2023)

Deckbuilding games or deckbuilders are a subgenre of card games that revolve around players modifying their deck during the game. This is in stark contrast to card games in other genres which focus more on playing with deck that was prepared before game began. Deckbuilders were first popularized in 2008 by the release of the board game Dominion and have since become a popular subgenre of card games. Deckbuilders have also spawned their own subgenres like deckbuilder roguelikes. (PlayBetterGames. 2022)

This thesis aims to find out what things are required for card games to be good and ways of implementing them into own projects. The methodology used was to investigate existing card games and what they do well as well as other available content from professional game designers. These findings were then used when developing a games prototype with focus on deckbuilding elements. The thesis documents various design aspects of the prototype with their reasonings as well as some of its more integral implementation related topics.

2 DECKBUILDING GAMES

2.1 History and current state of the genre

Deckbuilding games or deckbuilders are relatively new subgenre withing card games that were started by the release of Dominion in 2008. Before that deckbuilding elements were used as smaller parts of the game instead of being the main mechanic. One such game was ‘StarCraft: The Board Game’ which was released in 2007. In that game players could improve their combat decks by adding new cards to it through researching new technologies (BoardGameGeek, n.d.).

Deckbuilders became popular as tabletop games soon after the release of Dominion, but it took a while longer before they became popular in digital space as well (Nakamura, D, 2014). However as of the writing of thesis the genre has become popular in digital gaming as well and for example digital game distribution platform Steam has around 1700 games that use deckbuilding as their tag (Steam n.d). While the assigned tags can be quite loose, and the list seems to have some traditional CCGs as well it does give estimate on the genre’s current popularity.

2.2 Characteristics

Deckbuilding games are defined through the fact that they are card games in which building a deck of cards is an integral part of the gameplay loop. Deckbuilders generally start the players off with weak and simple decks that players then continue to improve for the whole duration of a game. The decks improvement is usually achieved mainly by adding better new cards to it, but it can also involve getting rid of existing cards or upgrading them into better versions of themselves. (Wrobel, L. 2023)

While most deckbuilders have differences in the ways new cards are added, a large part uses some form of store and in game currency. In some deckbuilders all cards are available from the start, while others limit players choice to smaller amount cards at a time (Picture 1). Some deckbuilding games give new cards to

players outside of stores as well. For example, roguelike deckbuilders tend to use cards as a reward for clearing challenges. In these games players aim to improve their deck to get as far as possible before losing and having to start over.

PICTURE 1. Star Realms has a shared store with five cards that get replaced when bought (PlayBetterGames. 2022).

Other things deckbuilders tend to have in common is the way cards rotate between deck, hand, and discard pile. This is quite different from CCGs and TCGs where cards in discard pile tend to be gone for good outside of specific card effects. Related to this, games centered around improving decks also tend to have players draw more cards. Dominion takes this to the extreme by having players start every turn with a new cards hand which was done for players to see the cards they added multiple times over the duration of a (Vaccarino, D. X. 2013).

3 THINGS ALL GAMES NEED

Every game needs to have a goal. A game's goal tells players what they should aim for, and it is often the thing that gets explained (Rosewater, M. 2011). Like other games, deckbuilders do not have any set goal. However common ones include trying to get the highest score or defeating all enemies. In Magic: The Gathering the players goal is to reduce opposing players life total to zero, which works well due to its clarity (Rosewater, M. 2011).

Game's rules define how the game is played. They limit what players can do in games which enables them to be challenging (Rosewater, M. 2011). Like the game's goal, its rules should guide players towards having a good time and being easily understood. Unclear rules will break players immersion and make planning possible actions difficult (Rosewater, M. 2011). One common problem that games tend to run into is that they get more complicated as they grow. This makes the game harder to learn which can deter potential players from trying it.

Interaction between players makes games more fun and it is the main reason playing them in person is as popular as it is. Just making players compete is enough for some interaction but it should be further supported through game design choices. (Rosewater, M. 2011.) In deckbuilders specifically this can be done for example by making players use a shared store for cards or creating cards for it specifically.

Including a catch-up feature into a game is generally a good idea as it helps to keep players invested in it even while doing badly. This can take on many forms like directly helping players that are behind or games later stages being more meaningful than earlier ones. Card games come with an inherent catch-up feature in the form of drawing randomized cards from a deck. Since the value of drawn cards varies, there will usually be a chance to make a comeback into the game with some luck. (Rosewater, M. 2011.) Developers should however be careful when adding catch-up features, or they risk making lead meaningless.

Games with inertia end naturally before becoming boring. When done right game ends with players wanting for more. This is achieved by making sure that the game is naturally moving towards its end. (Rosewater, M. 2011.) In practice there are multiple ways of creating inertia. One of these is to make the later stages of a game swingier to the point where one player is likely to win (Rosewater, M. 2011). Other options can include setting turn limit for the duration or creating a condition for the game's end which will be reached naturally. Games where the player's main goal is to survive as long as possible tend to either become harder at a pace where survival becomes impossible or include some ending after a certain point is reached.

Games usually contain unpredictable moments that players get surprised by. These moments are useful in making each play time feel unique and adding strategic depth to the game. When players can never know what the most optimal way of playing is they must adapt to given situations, creating interesting moments. When it comes to card games unpredictability can be added with randomness and hidden information. (Rosewater, M. 2011.) While unpredictability is usually beneficial, if it is overdone it can take away from players ability to strategize and get better at the game.

Most games involve some amounts of strategy. These games allow players to improve at them over time, strengthening players bond with them. The feeling of growth is important as it also motivates players to continue playing and learning. (Rosewater, M. 2011.) Strategy and unpredictability are often seen as opposing elements and balancing them can be tricky. Ideally a game has room for both and allows players to be surprised without harming their ability to make meaningful decisions and improve as a player.

The single most important thing games need is for them to be fun. If a game is not fun to play nothing else will matter as players do not want to play it again. Making sure that the game is fun is difficult due to its subjective nature. The only way of finding out if a game is fun to play is by having it be played by multiple people. According to Rosewater new designers especially struggle with making their fun as they tend to focus too much on smaller details. (Rosewater, M. 2011)

Games flavor is important as it makes the game more fun and helps players connect with it. On top of that flavor can be a valuable teaching tool when learning the game for the first time. (Rosewater, M. 2011) In his GDC talk Rosewater gives an example of this in the form of flying creatures in Magic: The Gathering. In this example the name of the ability itself is enough to create ideas about its effect on gameplay. (Rosewater, M, 2016)

A good hook also works as a center piece when marketing the game and helps get people interested in it. While a game can be good without a strong hook, it will harm its ability to find players. Any aspect of a game can be a hook if it is immediately easy to understand and can catch attention. (Rosewater, M. 2011)

4 DESIGNING A PROTOTYPE

4.1 Useful games for reference

4.1.1 The Bazaar

The Bazaar is a game project developed by Andrey “Reynad” Yanyuk and his team at Tempo Storm which has its development process documented in depth on the game’s YouTube channel. The game’s original concept (Picture 2) combined the class system of Hearthstone with gameplay elements from deckbuilders. The goal of the game was to improve your deck and ultimately win by dealing enough damage to your opponent. The game also included neutral monsters which could be defeated for additional rewards. This concept was ultimately scrapped as the game changed direction during its development process.

PICTURE 2. Old concept of The Bazaar (The Bazaar. 2020).

4.1.2 Dominion

Dominion was published in 2008 and in it the players goal was to have the largest amount of victory points at the end of the game (Nakamura, D. 2014). While later expansions added additional ways of getting victory points, the main way

of getting them is by buying specific cards with them and keeping them in the players' deck until the game ends. According to Dominion's designer Donald X. Vaccarino (2013) this was mainly done due to it sounding cool, but he also liked how collecting cards needed for winning also made the deck weaker.

Every turn of dominion follows the same pattern. Play action cards, spend resource cards to get new cards from store (Picture 3) and then end the turn by moving all played, bought and unused cards into discard pile and drawing new hand of five cards. If players were to draw a card while their deck is empty, discard pile is shuffled after which it becomes the new deck. Every card available for players to buy or get is shown in the store at the same time. Out of these, the ten cards pile on the grey background change between the games while other cards stay the same. The game ends if either province pile or three of the other piles are empty. (Rio Grande Games, 2021)

PICTURE 3. Cards for a single game of Dominion (Rio Grande Games, 2021).

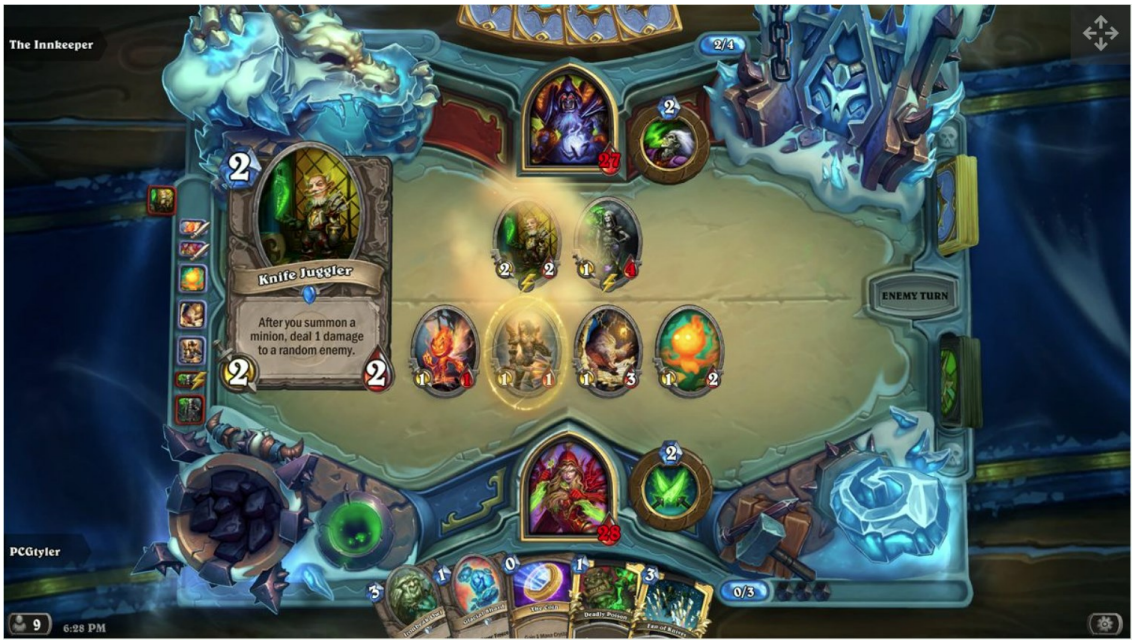
In dominion a starting deck consists of seven Copper cards and three Estates. The reason for Estates is to introduce some variance into the first couple turns. Starting decks size was chosen to allow players to play two turns before running out of cards for the first time. Vaccarino did not want to make the starting deck

larger than that either since it would have made the impact of newly added cards smaller. The reason players always draw a new hand in Dominion unlike in the most TCGs was that Vaccarino wanted players to get through their decks fast and see their cards multiple times during a single game. (Vaccarino, D. X. 2013)

4.1.3 Hearthstone

Hearthstone was released in 2014 as a free to play collectible card game (CCG) that was played exclusively online. While Hearthstone was not the first digital CCG to exist, it was the one to raise the genre into spotlight. This was likely due to it being free to play, easy to learn and highly polished compared to existing games in the genre. (Bindloss, W. 2018)

The place where Hearthstone shined the most was in its simple game design. The game's goal was simple to understand, hit opposing player until their health goes to zero (Picture 4). This combined with other design choices made Hearthstone an easy game to pick up and learn. One example is how players get a new mana crystal automatically at the start of each turn guaranteeing an increase in resources over the game. Another meaningful design choice is how Hearthstone handles players turns. In Hearthstone players take turns one normally but can perform no actions during their opponents turns. This worked to simplify the game while also improving its overall flow.



PICTURE 4. Typical view in a game of Hearthstone (Bindloss, W. 2018).

In Hearthstone players can choose from a pool of different classes, each with unique cards to select from and a different play style coupled with their own strengths and weaknesses. Different classes not only add a ton of flavor into the game but also help new players by guiding them towards certain strategies. (Bindloss, W. 2018)

4.2 Key attributes

The prototype designing process started from narrowing down the core ideas for the prototype. While The Bazaar was the main inspiration for the project it had plenty of things I wanted to change. Overall, the design process ended up being far from organized and plans often changed as new ideas came up over a long period of time. In the current concept players are fighting over control of a city and playable classes would consist of characters with varying occupations.

4.2.1 Multiple classes

From the start the project's core concept was to be a deckbuilder with multiple classes for players to choose from. In deckbuilders, class systems are mainly seen in single player games like Slay the Spire. When it comes to deckbuilders

with multiple players the more standard route is to have all players start identical starting decks and share a common pool of cards. This choice has some big benefits like requiring a smaller number of cards to be made and enabling the use of a shared store. However, it also gives up on benefits derived from multiple classes like increased flavor and more distinct playstyles. This flavor combined with the game concepts unique nature form a strong hook while making the game easier to pick up.

The theme I chose for the game would be some form of city over which players would fight. In my own project I was planning to tie the classes into different occupations in the city. This solution offered clear distinctions between classes as well as giving a good idea of what their cards would look like. In this way the story of the game ends up being about different characters fighting each other for power over the city.

Ideally every class should contain enough cards for the game to feel distinct from each other. This would also force players to adapt available cards to play optimally. Slay the Spire excels in this category by each class having multiple different core themes to build around while having plenty of cards that do not directly belong to any of them. This allows players to easily pivot their strategy based on situations promoting interesting and exiting gameplay.

While going for multiple complete classes is problematic at the start due to the number of required cards, it stops each new addition from becoming harder than the previous one. In games without classes or in ones that get expanded on, every new card needs to be tested and balanced with other cards already in the game. However, by making each class completely self-contained every new class requires the same amount of work to be created. This has a similar effect on games complexity as the number of possible card combinations in a single game does not increase. Of course, all these notions are meaningless when considering the scope of the prototype.

4.2.2 Rotating store

When it came to implementing the store for cards, I ended up going with a solution like the one used in The Bazaars original concept. The main points that carried over are being in the middle of the game and containing offering of cards that get changed every turn. The points of difference came down mainly to the number of cards available at the time and only showing cards for the player whose turn it was.

In the current prototype available cards are divided into different tiers with each tier having a different number of slots in store. The idea behind this was that cards meant for different stages of the game could always be displayed without risking situations where players would not have any viable options to choose from. One idea that I played around with but ended up discarding was to change the number of slots different card tiers have as the game progresses. The idea felt needlessly complicated relative to what it achieved.

4.2.3 Unique win condition

In the prototype the player's goal is to fill progress bar to the full. The games difficulty comes mainly from the fact that players opponent tries to achieve the same thing and progress bar is shared between players. This results in a situation resembling a game of tug of war. Overall, I ended up liking this as an idea since it felt fitting and quite unique.

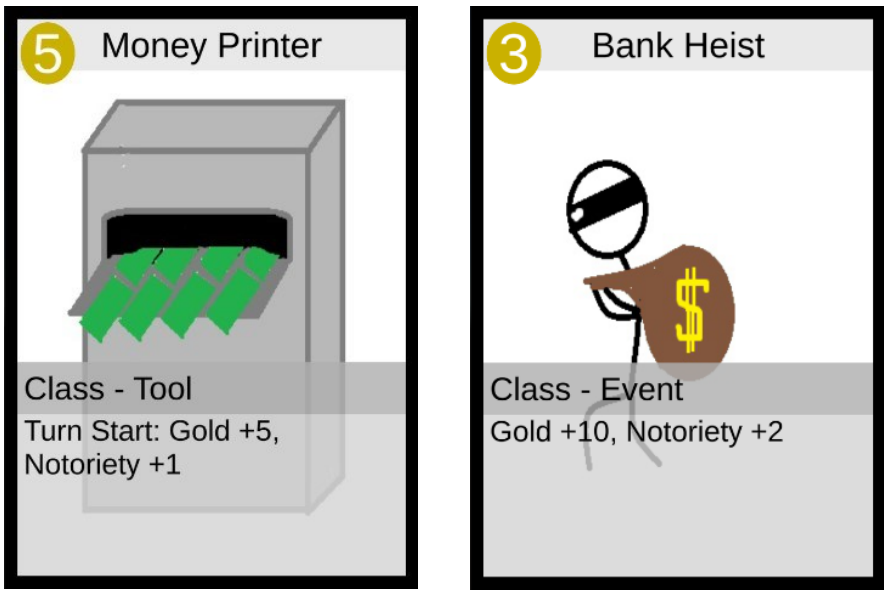
The main things I tried to focus on when deciding on the game's goal were clarity, intractability, and flavor. The idea of progress bar ended up hitting all these goals quite well. When it comes to clarity the game's goal can be explained simply and players current progress can be seen easily at all times. As intractability goes the fact that players are pushing against each other means that ignoring the opponent while going for the win is impossible. While theme can be harder to evaluate, I liked how the tug of war could be tied into players control over city. without being directly tied into physical health the same way life totals can be.

Other possible win conditions that I considered were traditional health-based systems and ones centered on collecting points. While health-based systems tend to play quite well in practice, I was not the biggest fan of them due to how they thematically tie into physical combat. As for score-based systems, I could not figure out how to implement them in a way that made the ending feel satisfying. While this comes down to personal taste, as a player I tend to prefer systems where the game ends due to something players did instead of arbitrary time limits. On top of that just collecting points can easily make it feel like players are playing their own individual games.

The biggest problem that I ran into with this system was game length. If both players get similar amounts of influence per turn the games could easily drag on. This problem can be mitigated to some amount by offering players benefits from controlling the middle point or by making late game cards swingy enough that they will make ending the game easier.

4.2.4 Different card types

The project currently divides its cards into three different types. events, tools, and locations (Picture 5). Out of these three, event cards are the simplest as they get put into discard pile right after getting played and having done their effect. Tools and locations are more similar in the sense that stay on board after playing and provide value over time. The main difference between the two is that each player can have up to four tool cards at the same time, but location cards are limited to one. Some tools also include a durability value which tells how many turns it survives on board before breaking and going to discard pile.



PICTURE 5. Concepts of simple cards with different card types.

Cards coming in different types is useful by itself since it introduces natural variety to card pool. On top of that, having access to card types opens up more possibilities when designing new effects. While it is important that every class can use all kinds of card types, the option to make some classes lean more into some of them is an additional way of adding flavor and variance to the game.

4.2.5 Persisting resources

Every card game comes with different kinds of resources that players need to manage when playing the game. Some of the more common ones include cards in hand, life totals, and one that is used when playing cards. Deckbuilders often include another one that is needed when acquiring new cards.

In deckbuilder recourses can either be included as a part of deck like in dominion or tracked separately. While keeping resources on deck can certainly work and has its own charm it comes with some drawbacks as well. This is mainly due to the way in which resource cards dilute the players deck, making other cards harder to find. This problem can be counteracted by making sure that players draw enough cards that they will get to see other cards as well. This solution does, however, cause another problem in the way it requires players to play a bunch of cards every single turn. This is especially problematic in digital mediums where cards are often played by dragging and dropping them one at a

time. While this problem can be mitigated to some extent by streamlining the way cards are played, I would prefer to circumvent the whole problem completely by keeping resources separate from the deck.

In the projects current concept decks have only playable cards. This led to the decks staying smaller than otherwise would be possible, allowing players to get through them while drawing less cards overall. Through some testing I ended up with each player drawing two cards at the start of a turn. This felt like a nice compromise while a single card per turn made the game a bit too slow. Now that players were drawing a modest number of cards it felt natural to allow them to be saved in hand over multiple turns. Being able to save cards allows players to hold up answers to their opponents' actions as well as play cards together in order to increase their effectiveness. As a side benefit, lowering the baseline in card draw allows for more variety between classes.

4.3 Mechanics that changed over time

4.3.1 Locations

The concept of dividing cards into three categories was present in the project from the early stages. However, the role of locations changed multiple times over the development process. At first locations were meant to function as cornerstones for different playstyles within a class. This concept was a bit troublesome in the sense that players' game would suffer too much due to not finding the correct location card for the situation. Additionally, location cards being tied into a specific playstyle made them close to useless outside of it.

In the second iteration both players started with a location card already on board. This was meant to further solidify the class's strengths by providing unique benefits from the start of the game. For a short while the plan was to make the starting location evolve over the game, but that was scrapped relatively fast. The main weakness of this concept was how in order to justify the amount of space locations took they had to be meaningful enough. This, however, led to cards from the store feeling less impactful which was too big of a problem.

In the current version locations are pretty similar to tools and having one is not mandatory in order to win. The idea of starting locations was used as a smaller class bonus that could still make classes more distinct from each other without taking the focus away from actual cards.

4.3.2 Discarding cards for resources

In the original concept one of the core ideas was to make discarding cards the main way of getting recourses. To reflect that every card had their discard value shown on the card which tended to go up with the card's cost. After some play-testing it became clear that discarding cards was just not as fun as getting to play them. This led to the creation of cards that were meant specifically for getting resources. At this point discarding worked only as a backup in case player could not get money through any other means.

Since discarding had now become a far smaller part of the game it made less sense for every card to have their own individual values. Due to this I ended up getting rid of them in favor of having one shared value that could still be increased through some cards. This was a clear improvement as it reduced the amount of unnecessary information on cards while making everything overall easier to understand.

After some more testing and pondering the system felt a bit unnecessary. While there was not anything directly wrong with it, the actual economy cards just felt better to use and were overall more interesting.

While the discarding mechanic did not work as a core system, it was far too interesting to get rid of completely. Instead, I ended up repurposing it as a class specific mechanic that allows player to discard tools. The class in question is centered around having a bunch of different tools and can further augment the benefits received from discarding them.

4.3.3 Secondary resources

While the game worked well with only one resource, I started playing around with adding additional ones. The main reason for this at the time was to get more things to balance cards around as well as enable specific concepts. The main idea I had for these was to include some form of negative resource that would allow cards to be more powerful than normal while being balanced due to interesting drawbacks.

The concept that came from this was to have a number and increase the card's prices equal to its value. This fitted the idea of immediate gains for a short-term benefit well enough without being too punishing. While originally the plan was to use this as an integral part of the game, after a while it was changed to be a class specific system. The main reason for this was to keep the basics as simple as possible in order to make the game easier to learn.

5 TECHNICAL IMPLEMENTATION

The project's implementation has been slowly worked on over the duration of multiple years with longer breaks in the middle. Implementation started immediately after getting the game's basic concept down. It did not take too long before players could get into a game and cards were moving between deck, hand, and discard pile. The prototype stayed in this state for a long while before it started progressing again. Soon after, cards could be bought from the store and played on board. This was followed by multiple other additions and improvements until the prototype got to its current state.

5.1 Choosing the game engine

Game engines are a type of software which is used as a framework when developing games. Game engines aim to make the development process as easy as possible by taking care of its technical aspects like rendering things on screen and running physics simulations. (Sydney, B. 2023) Many game engines are specialized and best suited for certain types of games, but some of them are more generally usable and can support all kinds of games.

In most situations it is good to consider the type of game that is being developed when choosing what game engine to use. Another thing that can affect the choice is the amount of help available for the game engine as well as the game engine's pricing model. Since card games do not limit the available game engines too much and I was already comfortable with using Unity, it was an easy choice.

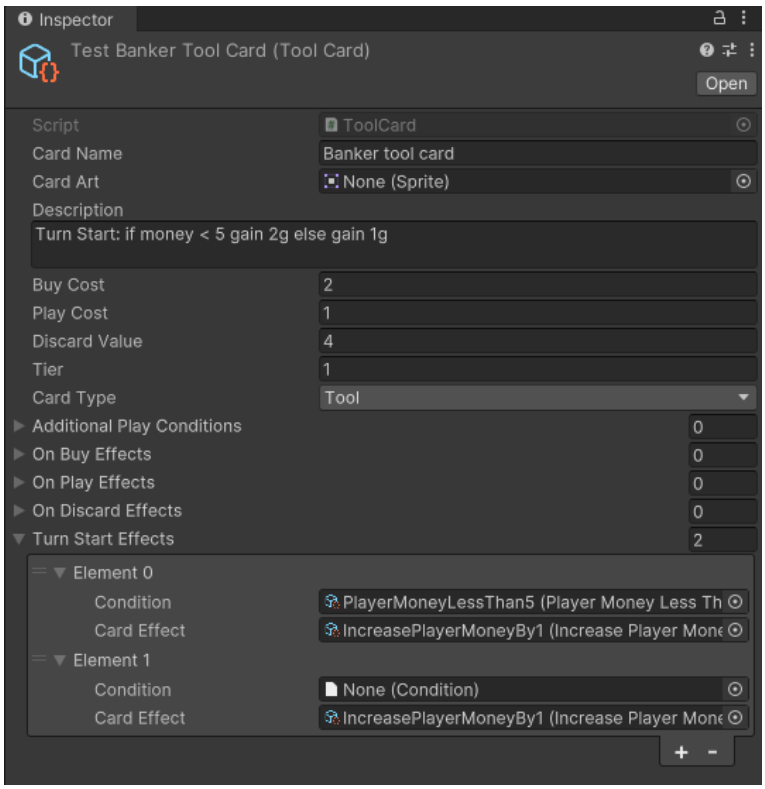
5.2 Scriptable Objects

Scriptable Objects are made based on a class and can be created from menus or dynamically while the game is being played. Scriptable Objects work as data containers and can be useful in reducing the games memory usage as they can be used to get rid of duplication of data. Scriptable Objects can also be useful

when needing to change values during testing since changes made to them do not get reverted when leaving the play mode. (Unity Technologies. 2022)

Scriptable Objects are extremely important in the prototype as all of its cards are done using them. This allows cards to exist without all of the baggage that comes with normal Game Objects. Most of the time, cards are kept in some list and only a small number of them need to be visible at a time. Since every card in the game is a Scriptable Object the amount times, they are referenced does not increase the amount of used memory meaningfully. Due to this a list can have tens of copies of a card without any drawbacks.

When a card needs to exist as a Game Object an instance of a card template is created after which the template is updated with information from the referenced Scriptable Object (Picture 6.). After the card does not need to exist anymore the object can be deleted and reference to the Scriptable Object added to the necessary list. Since all of the information is copied from scriptable objects this process could be e further optimized by reusing existing card templates instead of deleting them and creating new ones.



PICTURE 6. Inspector view of a test card.

5.3 Creating the cards functionality

Creating the framework for cards functionality ended up being surprisingly challenging and interesting task. In the first version cards functionality was hard coded into every card. This solution worked without, but it meant having to copy paste large amounts of code for each separate card. While this was a factor, the main reason it got scrapped was that I wanted to try out a different solution. I wanted to create a solution in which the card's functionality could be edited by dragging and dropping components in inspector view. This ended up being more complicated than I thought, but it taught me a lot in the process.

In the resulting implementation cards effects are built by combining by adding a varying number of simple Scriptable Objects into different lists inside the card (Picture 6). All of these are extremely simple and only do a single action. These actions either check if a condition is true (Picture 7) or implement an effect (Picture 8). The cards themselves only know how to go through their action lists executing functions inside them as needed. This architecture allows cards to be modified without changes to code just by changing Scriptable Objects inside its lists. This both speeds up the process and removes the need for multiple card base classes.

```
1   using System.Collections;
2   using System.Collections.Generic;
3   using UnityEngine;
4
5    [CreateAssetMenu(fileName = "PlayerMoneyLessThan", menuName = "EffectConditions/PlayerMoneyLessThan")]
6   public class PlayerMoneyLessThan : Condition {
7       [SerializeField] int value = 5;
8
9       public override bool CheckCondition(ManagerReferences managerReferences, Card card) {
10          if(managerReferences.GetGameManager().getMoneyPlayer() < value) return true;
11          else return false;
12      }
13  }
14
```

PICTURE 7. Implementation for a singular condition check.

```
using System.Collections;
using System.Collections.Generic;
using UnityEngine;

[CreateAssetMenu(fileName = "IncreasePlayerMoney", menuName = "Effects/IncreasePlayerMoney")]
public class IncreasePlayerMoney : CardEffect {

    [Range(0, 20)] [SerializeField] int increaseAmount = 0;

    public override void DoEffect(ManagerReferences managerReferences, Card card) {
        managerReferences.GetGameManager().IncreasePlayerMoney(increaseAmount);
    }
}
```

PICTURE 8. Implementation for increasing a player's money by a set amount.

5.4 Online functionality

When it came to creating a game with online functionality there were multiple things to consider and learn. After spending some time reading, watching tutorials, and trying to get stuff to work, I ended up with a workable solution. However, the way online functionality is currently done in the prototype is far from ideal and would need complete overhaul before the project could be released.

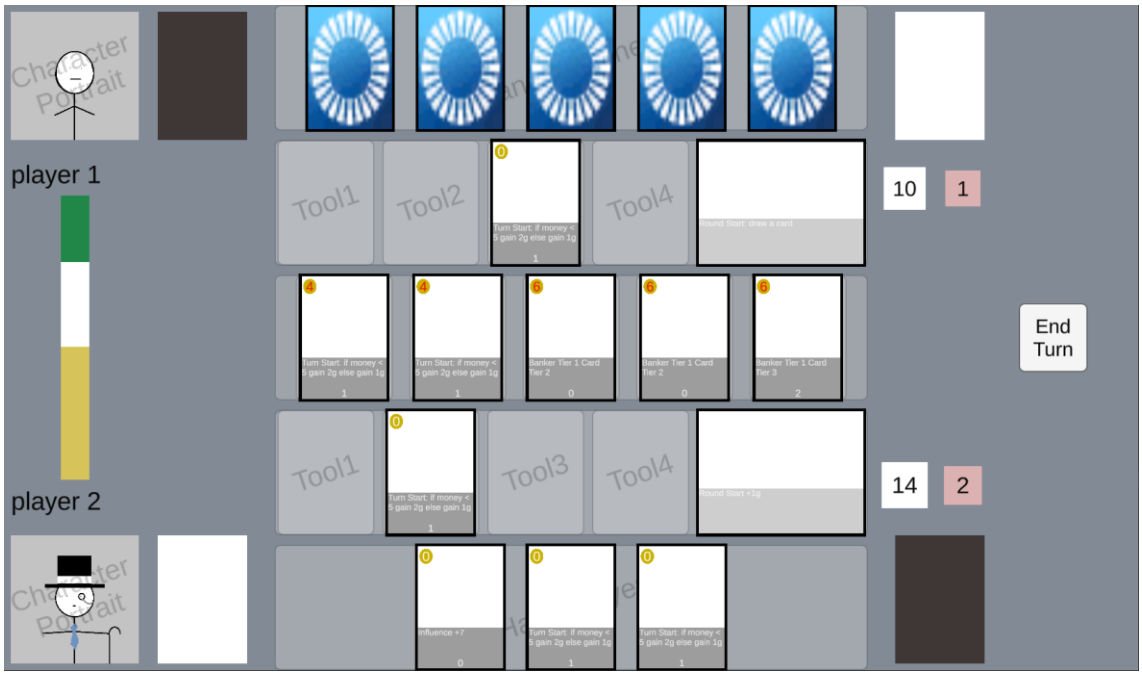
The biggest choice when working on the networking system was whether to use a client-based model or a server-based one. The difference between the two is in where the functionality of the game is executed. In client-based model the game runs on both players own systems at the same time and the information is kept synchronized directly or by using a server as a relay. In server-based model clients only send their input to server where it is used to perform actions in the game and the server keeps players updated on the game's current state. Of these two the server-based model is generally preferred as the other one is extremely vulnerable to all sorts of tampering. However, as the client-based model was easier to implement, and safety was not the main focus in this project I ended up going with it.

After deciding to use a client-based model the next step was to choose what tools to use for it. There were plenty of options to choose from but the main ones for me were Unity's own solutions (lobby, relay, and network for game objects) or Photon. These options were picked due to them being free to use until usage thresholds, which were high enough not to matter while prototyping. After trying both options, I went with Photon since it felt easier to work with.

Connecting players with Photons servers and each other were done using its built-in methods. After the players were connected and in the same game the only thing left was to keep them up to date with each other. Photon has multiple built in methods for this purpose like using components that keep game objects information updated in real time. These, however, were not too useful for this project since most of its gameplay elements needed to have their position mirrored based on the player while keeping them upright. On top of that some of the game's information needs to be visible only for one of the players. Though there might be some built in solutions for the first one, for now the game's prototype runs entirely by synchronizing information with remote procedure calls (RPCs). RPCs were easy to use since they came down to making a function with small adjustments and calling it from the other client. Any of the necessary information could be easily passed along as parameters just like in normal function calls.

6 CONCLUSIONS AND DISCUSSION

At the time of writing, the prototype is in working state with its core systems implemented (Picture 9). The biggest areas for improvements are the small card pool and updating older systems to match the current concept. After these points are addressed, the game would likely be in a state where it can be played with other people. This would likely make further improvements to it easier due to increased testing and new ideas.



PICTURE 9. Current state of the prototype.

The game's concept has potential and contains the elements needed for a game to be good. The one among them that I am slightly worried about is the fun aspect. This is caused by the prototype not being in a complete enough state where it could be properly tested. My plan for the project is to keep working on it on the side as a hobby.

The project fulfilled its main goal as a learning opportunity well. Working on the prototype taught me a lot about making card games as I had no prior experience with it. On top of that, even though the prototype still needs more work, I am pleased with its current state.

REFERENCES

Bindloss, W. 2018. HEARTSTONE REVIEW. PC Gamer. Read on 27.11.2023. https://www.pcgamer.com/hearthstone-review/

BoardGameGeek. n.d. StarCraft: The Board Game. Read on 25.5.2024. https://boardgamegeek.com/boardgame/22827/starcraft-the-board-game

Global Game Network. 2023. The Fascinating History of Card Games: From Medieval Europe to Digital Evolution. Read on 25.2.2024. https://www.global-gamenetwork.com/card-games-history/

Nakamura, D. 2014. So what exactly is a deck-building game anyway? https://www.destructoid.com/so-what-exactly-is-a-deck-building-game-anyway/

PlayBetterGames. 2022. What is a deck building game? Read on 27.11.2023. https://www.playbettergames.com/games-101/what-is-a-deck-building-game/

Rio Grande Games. 2021. DominionRules2021. Read on 23.5.2024. https://www.riograndegames.com/wp-content/uploads/2016/09/Dominion2nd.pdf

Rosewater, M. 2011. Ten Things Every Game Needs. Read on 27.11.2023. https://magic.wizards.com/en/news/making-magic/ten-things-every-game-needs-part-1-part-2-2011-12-19

Steam. n.d. Deckbuilding. Read on 24.5.2024. https://store.steampowered.com/tags/en/Deckbuilding/?facets13268=1%3A11%2C8%3A15%2C9%3A2

Sydney, B. 2023. What Is a Game Engine? Read on 5.2.2024. https://www.howtogeek.com/888619/what-is-a-game-engine/

The Bazaar. 2020. Why Decks are a Problem | The Bazaar Update #1 Watched on 25.5.2024. https://www.youtube.com/watch?v=fQi92yB949M&list=PL3qblbBtHwheSN3kppupYoMG1StSQ46bY&index=2

Unity Technologies. 2022. ScriptableObject. Read on 11.2.2024. https://docs.unity3d.com/Manual/class-ScriptableObject.html

Vaccarino, D. X. 2013. The Secret History of Dominion. Read on 27.11.2023. https://boardgamegeek.com/thread/996671/secret-history-dominion

Wrobel, L. 2023. GAME DEFINITION: DECK BUILDER. Read on 27.11.2023. https://engagedfamilygaming.com/parent-resources/gaming-glossary/game-definition-deck-builder/