FIT3077: Software Engineering: Architecture and design

Semester 1, 2024

Domain Model Rationale

Group No: MA_Wednesday_04pm_Team690

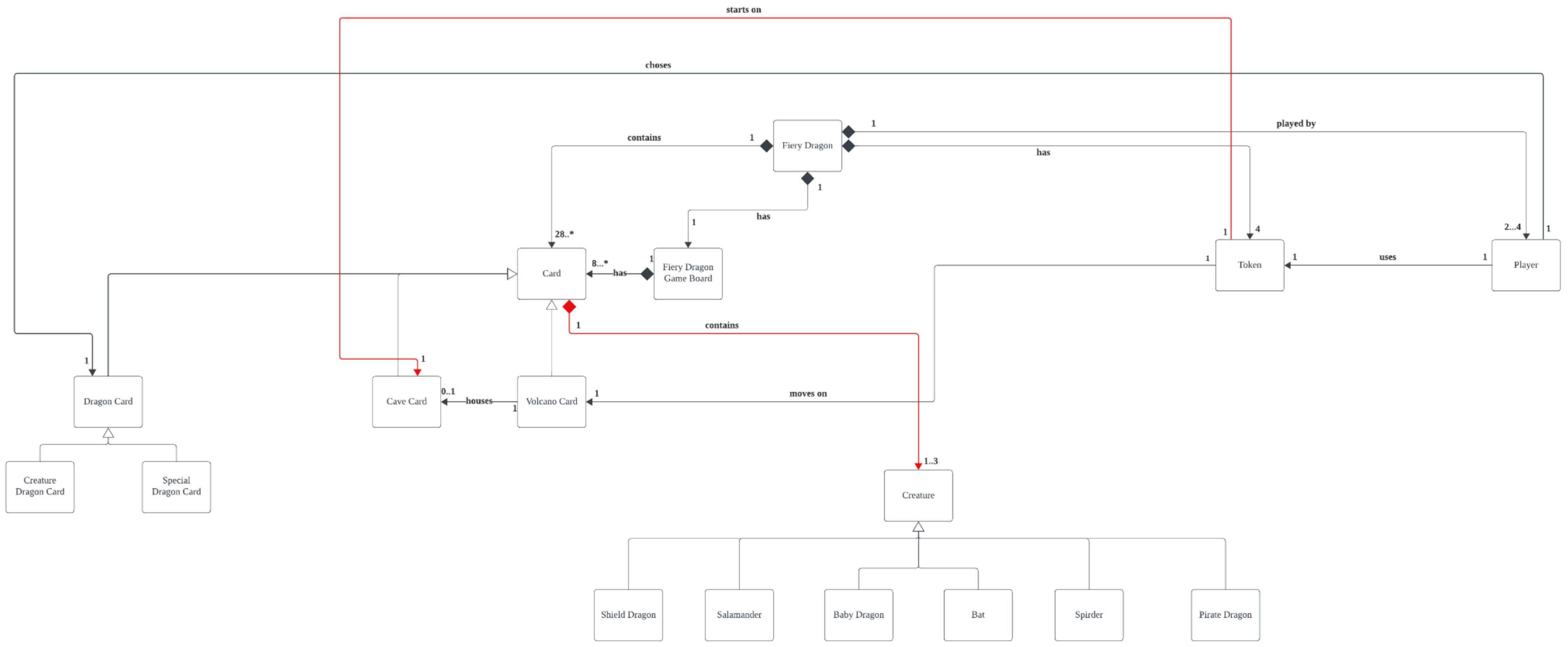
Prepared by:

Tong Jet Kit	32632177	jton0028@student.monash.edu
Mandhiren Gurdev Singh	32229828	mgur0007@student.monash.edu
Nisha Kannapper	31121993	nkan0018@student.monash.edu

Content

Domain Model

Link to domain Model:
https://lucid.app/lucidchart/fb08fa5b-01d3-4ab3-8422-b9800109875a/edit?viewport_loc=-1775%2C-749%2C3454%2C1598%2C0_0&invitationId=inv_f097129c-da99-4a21-97cf-2efd5211da27



Justification

When creating the domain Model, our team decided to come up with all of the physical entities of the game then try to eliminate any redundant or extra domain entities. For the domain Model for Fiery Dragon we first added the Fiery Dragon as a domain entity since it is the game itself. Additionally, we added the Fiery Dragon Game Board and Card as a domain entity in the model. This is because the Fiery Dragon Game Board itself is the physical entity that the players will play on and the cards are the items that will form the game board itself. The Fiery Dragon domain entity will have a composition relationship with the Fiery Dragon Game Board and Card domain entity as the game itself cannot exist without the cards and the game board itself. The cardinality between the Fiery Dragon and the Card domain entity is 1 to 28 or more since our initial idea for the extension is to have more than 8 volcano cards to increase the duration of the game. Moreover, there will also be a composition relationship between the Fiery Dragon Game Board domain entity and the Card domain entity since the Game Board is made out of the Cards. The cardinality of this relationship is 1 to 8 or more rather than 1 to 8 as we decided to add an extension where the number of volcano cards can be more than 8 to be able to build a bigger ring to prolong the game more.

In terms of alternatives, we initially decided to not have the Fiery Dragon domain entity as we were going to define the internal component of the game itself instead. However, after careful discussion and prototyping, we decided to add the domain entity as it helps to signify the game itself which is part of the overview of the game itself.

The Fiery Dragon cards consist of 3 types, the Dragon Card, the Cave Card and the Volcano Card. The Dragon Cards are the chit cards that the player uses to choose to determine their movement during their turn. The Cave Cards are the starting point for the players while the Volcano Cards are the cards that will make up the game field that the game token will move on. Since these cards will be interacted with by the player, a domain entity is created for each of them. These card domain entities will have a generalisation relationship with the Card domain entity created previously as they all stemmed from it as they are cards. Between the Cave Card and Volcano Card, there is an association relationship between them. This is because a Cave Card is housed in a Volcano Card so there is a relationship between them where the cardinality is 1 Volcano Card to 0 to 1 Cave Card as a Volcano Card may house a Cave Card.

In terms of the alternatives, initially the team debated combining the Dragon Card, Cave Card and Volcano Cards into a single domain entity. However, we soon realised that merging all types of cards into one generic “Game Card” entity would only lead to non-modularised functionalities that are not specific to the different types of Game Cards. Thus, we chose to separate domain entities for Dragon, Cave, and Volcano Cards to accurately represent their unique roles and interactions in gameplay.

Moreover, other than the card itself, we created a domain entity called Creatures. This domain entity represents the type of symbols that can appear in the cards. The creatures are Bat, Spider, Baby

Dragon, Pirate Dragon, Salamander and Shield Dragon. This Creatures domain entity will then have a composition relationship with the Cards domain entity since each card contains 1 to 3 types of creature on it and without the creatures it will not be the game card. For example, a Volcano Card has 3 types of creatures while Dragon Cards and Cave Cards have 1 type of creatures.

Furthermore, the Dragon Cards have different types too, there are dragon cards with creatures on them that allow users to move forward and special dragon cards that provide special effects so we create the Creature Dragon Card and Special Dragon Card domain entities. Thus, we create a domain entity for both of these Dragon Cards since although they are Dragon Cards they have different behaviours. The Creature Dragon Card will allow the player to move forward if they choose the correct corresponding creature with the Volcano Card they are on or not move at all if they choose the dragon card with the wrong creature on it. In the Special Dragon Card, it will contain either the Pirate Dragon Card or the Shield Dragon Card. The Shield Dragon Card is our extension where it will shield the player from one Pirate Dragon Card effect. The Pirate Dragon Card is part of the normal game where it will force the player to move backwards according to the number of pirate dragons in the card. Thus, the Special Dragon Card is a domain entity in the domain model as they are special cards that the player can choose during the game. Since both the Creature Dragon Card and Special Dragon Cards are dragon cards too, it will have a generalisation relationship with the Dragon Card domain entity.

In terms of the alternatives, initially we contemplated creating each of the Volcano Card, Dragon Card and Cave Card entities For example, the Bat Dragon Card, Salamander Cave Card, Shield Dragon Card and so on. However, we realised that this would be redundant as they are all just cards that do the same thing as its parent, Dragon Card, Volcano Card and Cave Card respectively. Therefore, for clarity, we decided to opt for the separation approach to create a Symbol domain entity which can ensure clearer representation of the Fiery Dragon game,

Besides that, we also created a domain entity called Token, the Token is a pseudo-physical (moves virtually on the game board) entity that acts as a representation for the Player who uses the Token. and this has an composition relationship with the Fiery Dragons game itself, because the game cannot exist without the Token. There is a 1 game to 4 Tokens cardinality because of the fact that there can only be up to 4 Players in the game, and that 4 Tokens will always exist to be chosen by the Players. The Player itself is another domain entity that the game itself creates, thus explaining the composition relationship between them. Each game can have 1-4 Players, which explains the cardinality between the Fiery Dragons game and the Player which plays the game. The main reason why we chose the Token to be a standalone entity is for further generalisation later on when implementing the game. For example, we currently have 4 different types of Tokens in the game that share similar attributes of the Token domain entity. By generalising in this way, our rationale was that it will be able to reduce repetitive code. 1 Token starts at 1 Cave Card as well, and the association relationship between the Token and the Cave Cards is one such that the Cave Card houses the Token at the beginning of the game. The Player also has an association with the Dragon Cards themselves, as the Player is supposed to choose a Dragon Card in order to progress through the game in each turn. There is a 1 to 1 cardinality between the Player and the Dragon Card because 1 Player can only open 1 Dragon Card per move in their turn. Lastly, the Token has a 1 to 1 cardinality between itself and the Volcano Card because 1 Token can be on 1 Volcano card at any time in the game, and each Token moves on the Volcano Cards to progress through the game. The domain model we've implemented accurately captures the dependencies between the game, players and tokens.

Some alternatives considered and discarded were embedding Token attributes within the Player domain entity itself but we discarded this idea because Tokens are supposed to represent individual Players and each of them have their own unique attributes, such as the position on the current game board. Therefore, separating Tokens as a distinct entity lets us better organise and extend the game in the future if need be.