Team PJ-b

CODENAMES GAME

Team Members

Name	ID Number
Ashesh Patel	40018519
Benjamin Thérien	40034572
Bilal Rana	40013408
Christophe Savard	40017812
Daniel Thibault-Shea	40073133
Michael Wilgus	29206388
Mordechai Zirkind	27206151
Rezza-Zairan Zaharin	40003377
Shereece Victor	40105094
Steven Zanga	40000797

Table of Contents

Project Analysis and Development Plan

1.1 Introduction

The purpose of this document is to detail the high-level requirements and features of the Codenames Game developed by team PJ-B. The Codenames Game is a game of 4 human and AI players. This project is an adaptation of the Codenames Game designed by Vlada Chvátil and published by Czech games.

The specifics of how the Codenames Game fulfills these needs will be detailed in the use cases which more will be detailed in the upcoming design phase

1.2 Purpose

This document will describe the specifications entailed by the development of Codenames in compliance with the requirements of COMP 354. It will outline the high-level requirements encompassing user interfaces, product functions, user descriptions, assumptions and dependencies, constraints, specific requirements and an analysis model. The analysis model will hold use case diagrams, class diagrams, sequence diagrams and state transition diagrams.

1.3 Scope

This document only addresses the high level requirements of Codenames that the design phase entails. The analysis model will contain UML diagrams used to serve multiple purposes. The Use case diagrams will give an overview of the functions of the project and how users will interact with it. The class diagrams will show the interactions between different objects in the game.

1.4 Definitions and Abbreviations

1.4.1 Definitions

Board

The main playing area will be composed of a 5 by 5 grid of words. Each cell of the grid are representative of Codenames of agents. Each cell in the grid will be defined in this document as a *Card*.

Card

The card will hold two values indicative of it's state: hidden or revealed. In its hidden state, it will present a word taken from the game. In its revealed state, it can either be a blue or red team's *Spy* card, a *Civilian* card or the **Assassin** card.

Assassin

The Assassin is a card type that when revealed, it would cause the team that revealed it to lose.

Spy

The Spy is a card type that belongs to either the red or blue team. Once all spy cards of a team are revealed, the team wins.

Civilian

The Civilian is a card type that occupies leftover space on the board that is not occupied by *Spy* cards or the *Assassin* card. When revealed, it only skips the revealing team's turn.

1.5 Overview

The rest of this document outlines the problem description and the development plan.

The problem description will describe the game user's interfaces, product functions, user descriptions, assumptions and dependencies, constraints, specification requirements, and the analysis model.

Problem Description

2.1 Project Purpose, Scope, and Objectives

The objective of this project is to simulate a multi-user tabletop game named Codenames game by by Vlada Chvátil and published by Czech games. The rules of the game will intimately follow Vlada Chvátil with slight variations to accommodate the digital conversion. This project will be a multi-user game of up to four players; these players can be either human or computer players.

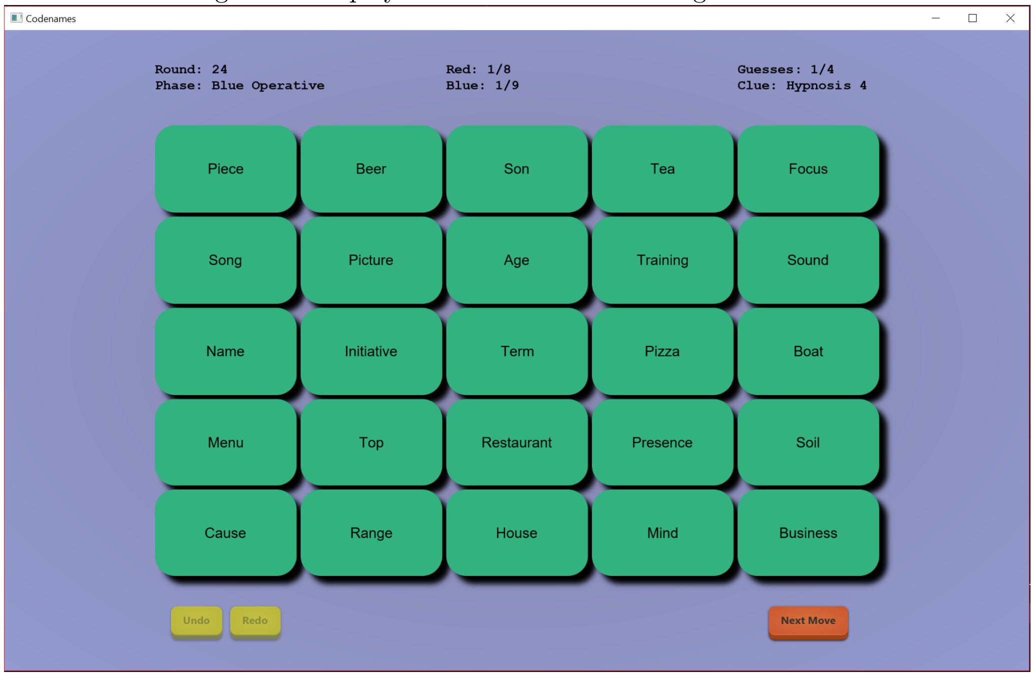
By working on this project, Team PJ-b can experience the software engineering process entailing the difficulties in managing a group project.

The project will be worked on over 3 iterations whereby they are the requirement phase, the design phase, and the test phase.

2.1.1 User interface

As this is the requirement phase, we are keeping the interface to a minimum design. They will not preface any visual grace but present the most useful information at the time. The visual representation of the game is as shown below:

Figure 1:The playable menu of the current game version



Board

The board is represented by a 5 by 5 set of cards, each possessing a word that when revealed, it will change in color to present what is the card type. When revealed, the card will be of a specific color to determine its value:

- Red is a red spy.
- Blue is a blue spy.
- Black is *the Assassin.*
- Peach is a Civilian.

Round

On the top left, the "*Round*" text represents the number of rounds that have passed in the game in accordance to the numerical value next to it.

Phase

On the top left, the "*Phase*" text details the current player in action. This value would be either one of these at a time:

- Blue Spymaster
- Blue Operative
- Red Spymaster
- Red Operative

Score-Keeping

In the middle top, Two sections with the text "*Red*" and "*Blue*" denotes the number of spies revealed for each team to the number of cards left to revealed. This is done in the format of:

"Number of spies revealed / Total Number of spies"

Guesses

On the top right, the text "*Guesses*" denotes the number of guesses left to be made by the *Operative* given to by the *Spymaster.* The value is formatted in:

" Number of Guesses Left / Maximum Number of Guesses "

Clue

On the top right, the clue given by the *Spymaster* will be displayed here.

Undo/Redo

On the bottom left, Two buttons of "*Undo*" and "*Redo*" functions as a point to click to control the turns elapsed. The *Undo* button reverses a turn so long as it is made. The *Redo* button replays a turn that was affected by *Undo.*

Next Move

On the bottom right, the "*Next Move*" button serves as a platform to click to progress the game forward by a turn.

2.2 Constraints

Assuming that a majority of player's PCs will run by the Windows OS, the project must be written in this platform that supports GUI.Thus, the decision has been made to use :

- JAVA as the programming language.
- SQLite as chosen for data storage.
- JUnit for unit testing.
- Eclipse as the integrated development environment(IDE).

2.3 Analysis Models

2.3.1 Use Case Diagrams

The following diagarams will help provide an overview of the functions in the game. They describe the action that a player can perform, as well as the interaction between some of the system functions which are not directly controlled by the player.

These use case diagrams are included due to their importance in defining the user-to-software interactions, the requirements, and the scope of the system.

The section below will detail the actors involved in the game:

User

The User represents the human player. For this iteration there is only one User. The User is able to control the chronological flow of the game via the use cases detailed below which are "*Undo*", "*Redo*", "*Next Turn*", and "*Start Game*".

Spymaster

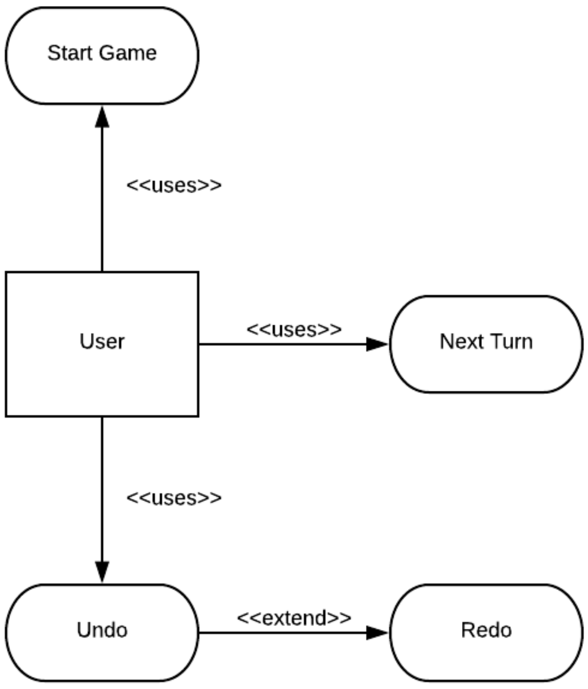
The Spymaster is involved in the game by doling out clues. As there are only two teams, there will only be two Spymasters in the game. The Spymaster hands out clues via the "*Reveal Clue*" use case detailed below.

Operative

The Operative is involved in the game by revealing cards in accordance to the clue and number of guesses given to by the Spymaster actor. As there are two teams, there will be only two Operatives in the game. The operative reveals cards through the use case "*Reveal CARD*" use case as detailed below.

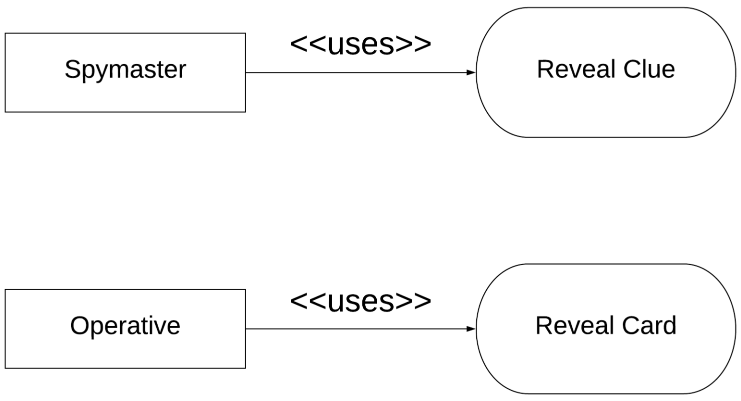
User Use Case

Figure 2: A use case diagram showing the use case concerning the actor "User"



Spymaster and Operative Use Case

Figure 3: A use case diagram showing the use case concerning the actors "Spymaster" and "Operative"



2.3.2 Use Cases Details

Use Case 1: Start Game

Description	The Player commences the game
Actors	User
Pre-Conditions	None
Basic Path	1. The User clicks ”Start Game”. 2. 4 Dummy players are generated and each are: • Deposited into a team. • Distributed a role. 3. The board is initialized. 4. The card layout is randomized. 5. The team turn order is randomized. 6. The first turn order is given to the leading team’s Spymaster.
Alternative Paths	None
Post-Conditions	• The board has been initialized. • 4 Dummy players are generated with their own team and role. • The first player can play his/her turn.
Related Use Cases	
Used Use Cases	None
Extending Use Cases	None

Use Case 2: Next Turn

Description	The game moves forward a turn.
Actors	User
Pre-Conditions	• The Board is initialized. • The game has not ended.
Basic Path	1. The user clicks the button, "Next Turn". 2. The current acting player plays his/her turn. 3. The turn passes to the next acting player.
Alternative Paths	None
Post-Conditions	A player has acted on his/her turn.
Related Use Cases	
Used Use Cases	None
Extending Use Cases	None

Use Case 3: Undo

Description	The game reverses to a state before the current turn.
Actors	User
Pre-Conditions	• The user clicks the button,"Undo". • The Board is initialized. • At least a turn has been made.
Basic Path	1. The game reverses the actions done for the current turn. 2. The turn is passed to the previously acting player.
Alternative Paths	None
Post-Conditions	A player has has his/her turn redacted.
Related Use Cases	
Used Use Cases	None
Extending Use Cases	Redo

Use Case 4: Redo

Description	The game repeats action redacted by the "Undo" use case.
Actors	User
Pre-Conditions	• The user clicks the button "Redo" • The Board is initialized. • The "Undo" use case has been used at least once.
Basic Path	1. The game proceeds a turn with action undone by the use case "Undo". 2. The turn is passed to the next acting player.
Alternative Paths	None
Post-Conditions	A player has has his/her turn redone.
Related Use Cases	
Used Use Cases	None
Extending Use Cases	None

Use Case 5: Reveal Clue

Description	The Spymaster issues a clue
Actors	Spymaster
Pre-Conditions	• The Board is initialized. • The turn belongs to the Spymaster. • The team Operative has not started his/her turn. • There are still cards on the board to reveal.
Basic Path	1. The Spymaster issues a clue with a number of guesses. 2. The turn is passed to the team's Operative as detailed in the use case, "Reveal Card".
Alternative Paths	Alternative 1: 1. After step 1, if the clue not considered valid by the system (i.e. One or more words contains the clue word), the turn is passed to the Spymaster. 2. The turn is also passed for the team Operative as detailed in the use case, "End Turn"
Post-Conditions	• A clue has been revealed • The Spymaster's turn has ended • The turn is passed to the team's Operative.
Related Use Cases	
Used Use Cases	Reveal card, End Team's turn.
Extending Use Cases	None

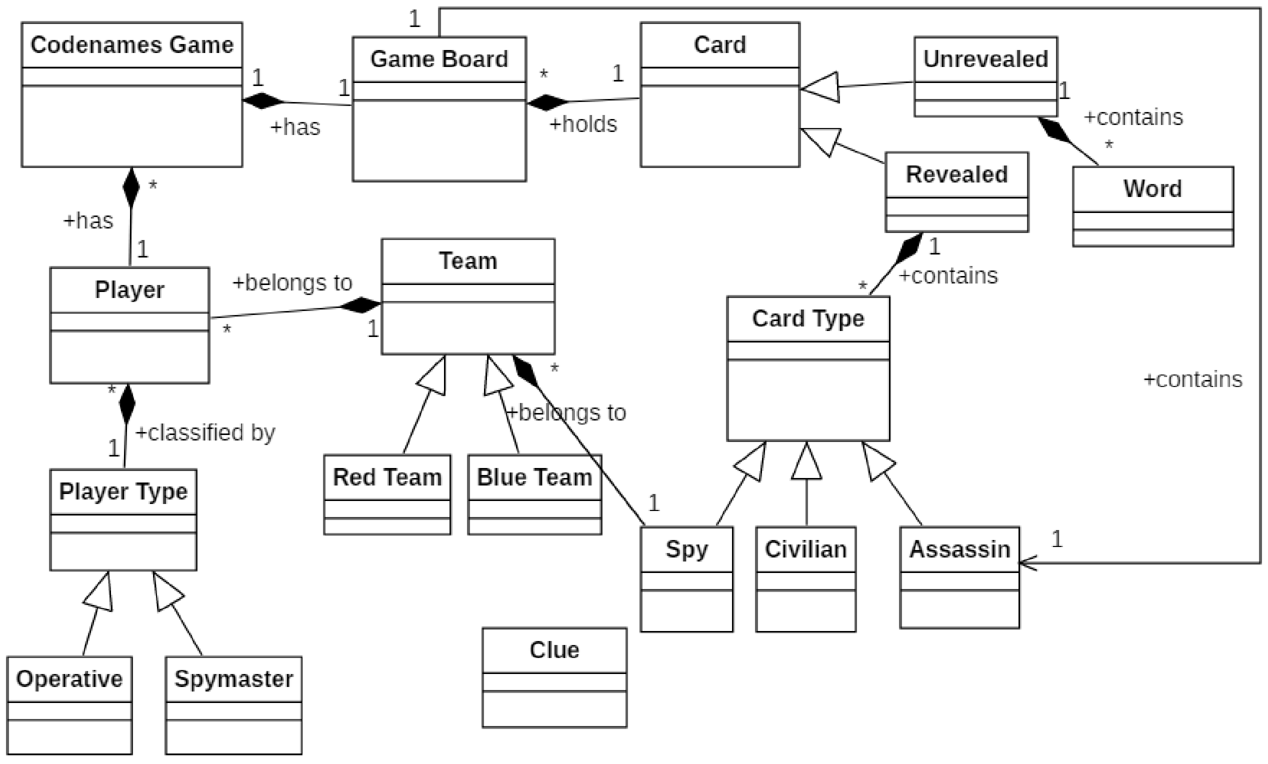
Use Case 6: Reveal Card

Description	The Operative reveals a card
Actors	Spymaster
Pre-Conditions	• The Board is initialized. • The team's Spymaster • The turn belongs to the Operative. • The team Operative has not started his/her turn. • There are still cards on the board to reveal. • The number of guesses given by Spymaster is at least one.
Basic Path	1. The Operative chooses to either: • Continue to make guesses, proceed to step 2. • End his/her turn, proceed to step 4. 2. The Operative reveals a card on the board. 3. The number of guesses for the Operative is depleted by one. 4. If there are still more than one guess left and there are cards to be revealed, repeat step 1. 5. The turn is passed to the enemy Spymaster detailed in the use case, "Reveal clue".
Alternative Paths	Alternative 1: • After step 2, If card revealed is an *Enemy Spy* or *Civilian*, the turn is passed to the enemy Spymaster detailed in the use case, "Reveal clue". • If all of the enemy team's *Spy* cards are revealed, the game ends with the enemy team winning. Alternative 2: • After step 2, If card revealed is the *Assassin* card, the game ends with the enemy team winning. Alternative 3: • After step 2, If all of the Operative's *Spy* cards are revealed, the game ends with the Operative's team winning.
Post-Conditions	• A card is revealed on the board. • The turn is passed to the enemy Spy.
Related Use Cases	
Used Use Cases	Reveal Clue
Extending Use Cases	None

2.3.3 Class Diagrams

Simplified Domain Model

Figure 5: A domain model of the Codenames game.

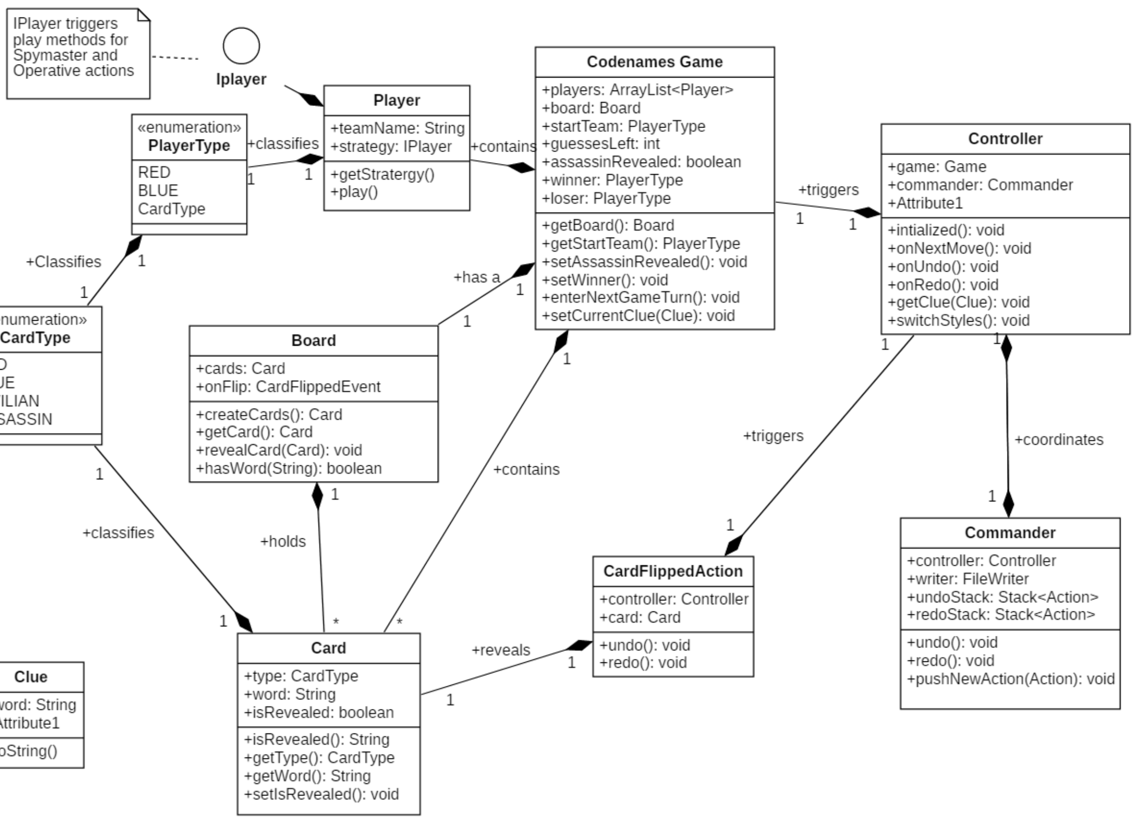


The Game consists of a game board with cards which can either be revealed or unrevealed during the game. When cards are not yet revealed they show a word. Each card has a type, it is either a spy, a civilian or the Assassin. When cards are revealed they show their type. There can only be one assassin card per game. Each spy belongs to a team, red or blue.

The game also contains players which interact with the game. Each player, like the spies, belong to either the red or blue team.

Class Diagram

Figure 6: A class diagram of the Codenames game.



This class diagram for the Codenames game does not show all the components of the final classes and interfaces. Instead it shows how some of the concepts presented in the domain model can be implemented.

The commander contains most of the functions the user interacts with, the controller carries out those functions by triggering methods in other parts of the system. The card flipped event for example, can be triggered when the user presses the 'undo' button. This is received by the commander, which passes it to the controller, which triggers the card flipped event, revealing or hiding a crad on the game board within the game.

Team Meeting Minutes

3.1 Meeting 0

3.1.1 Administrative

Wednesday, 10th January 2019 |8:15pm – 8:45pm |H Building (Lab near COMP 354 classroom)

Attendees:

- BT - Benjamin Thérien
- DT – Daniel Thibault-Shea
- Leo (No longer a team memeber)
- MZ - Mordechai Zirkind
- RZ - Rezza-Zairan Zaharin
- SV - Shereece Victor
- SZ – Steven Zanga

3.1.2 Summary of Discussion

- We met and introduced ourselves.
- We decided which roles we will play for the first iteration. BT, SZ, and a new unknown team member would be coders. Leo, RZ and SV would be documenters; and MZ, and DS would be organizers.
- We discussed the areas we have experience in and what we're confident in.
- We discussed Maven and the different software tools which can be used to great a game.
- We agreed that Discord would be the messaging application of choice.
- We agreed to have a meeting before the lab the following week.

3.1.3 Agreed upon goals

- Read the project outline.
- Review and get to know all the tools to be used for the project.
- Confirm time for next meeting.
- Think through how you would do the project on your own.
- Ensure we each have access to the GitHub repository and Discord server.

3.2 Meeting 1a

3.2.1 Administrative

Wednesday, 16th January, 2019 |8:15pm – 9:15pm |H Building 7th Floor Common Area

Attendees:

- BT - Benjamin Thérien
- DT – Daniel Thibault-Shea
- MZ - Mordechai Zirkind
- RZ - Rezza-Zairan Zaharin
- SV - Shereece Victor
- SZ – Steven Zanga

3.2.2 Summary of Discussion

Play Codenames

- We played Code Names the card game. DT (Blue Team) and MZ (Red Team) were the Spy Masters. RZ and SZ comprised the Red Team. BT and SV comprised the Blue Team.
- During play the rules, were clarified and discussed. We also confirmed and questioned what taunts and interactions are allowed by players.

The game is quite fun. We understand the game play better now.

Approaches to Game Architecture

- SV presented a possible method using a 2x2 matrix for the game board representation, assigning each block a number from 1 to 25 and using random functions to assign red and blue card placement along with other cards on the board.
- MZ lead a discussion of the logical design of the game using objects.
- SZ and BT explained the concept of MVC (Model View Controller)
- We discussed whether the Controller or the Model contains the logic of the system.
- Discovered that the group was expanded and thus we proceeded to the lab to meet the rest of the group.

We need to ask the lecturer to clarify MVC.

3.2.3 Agreed upon goals

- Ask the lecturer about MVC in class.

3.3 Meeting 1b

3.3.1 Administrative

Wednesday, 16th January 2019 |9:30 pm – 10:40pm (approx.) |H 903 Lab

Attendees:

- AP – Ashesh Patel*
- BT - Benjamin Thérien
- CS - Christophe Savard*
- DT – Daniel Thibault-Shea
- MW – Micheal Wilgus*
- MZ - Mordechai Zirkind
- RZ - Rezza-Zairan Zaharin
- SP – Saad Patel*
- SV - Shereece Victor
- SZ – Steven Zanga

* - New team members

3.3.2 Summary of Discussion

Re-introductions

- Met the new members of our group AP, SP, CS, MW,
- Met and spoke to the TA.
- Discussed the different roles: Coders, Documenters, Organizers, Quality Assurance and the importance of keeping track of everyone's tasks within their roles.
- Established that everyone will be involved in testing and coding.
- Established the skill sets of team members: Who's familiar with Java, Unit testing with JUnit GitHub, SQL, Latex, Maven, Package Mangers, Discord.
- Discussed how to get all these tools.

We need to all get familiar with the tools we will be using for the project.

Iteration 1 Roles

- We grouped into the different teams
- Coders: CS, SZ, BT
- Documenters: RZ, SP, SV
- Organizers: DT, MZ,
- Quality Assurance: AP, MW

Work to be done

- Recap of MVC
- Listed all the documents to be hosted on GitHub, code, diaries, testing.
- What are we storing in our database? Words, hints, game boards, game states, game statistics
- Should the game boards be saved or randomly generated at the beginning of each game
- Game Stats will include: Game history, Number or rounds, number of each colour spy revealed, innocent bystanders revealed, winners, losers, whether the assassin was revealed
- Data structure of the words, their hints and their associations; graph, database, do we need a bridge table, how many tables?
- How intelligent would the interactions with the word database be?
- Should we have user names? Player 1 and 2, or Bot 1 and 2
- We'll be treating team players in the game as one single entity per team, plus another entity for the spy master.
- Our user controls a player guessing.
- The Spy Masters are programmed.
- Ways to map word associations, storing meanings with each word? Graphs?
- Game cycle.
- Initial loop: Choose players and identities, create board, arranging board.
- For iteration 1, guessers should be an array of random words
- How do the spy masters see the game board? Toggle view?
- Assigned tasks due next week

3.3.3 Agreed upon goals

- Download the Java version used in the lab (8.161)
- Download JetBrains
- Sign up for GitHub Education Pack
- Install SQLite
- Download Latex
- Coders: Do a rough sketch of UML and ER diagrams, set up objects and program structures, discuss plans on Discord, Bare bones of the project, and record the logic
- Documenters: Prepare and share minutes, ER diagrams from coders, Get Familiar with Latex, Table of contents
- Record meeting in personal diaries, get familiar with GitHub, add personal diaries to GitHub, Read over the project description
- Looking to Unit Testing

3.3.4 Miscellaneous

Labs aren't mandatory. They're only a scheduled meeting time for teams. We need to be there for demos, however.

3.4 Meeting 2

3.4.1 Administrative

Wednesday, 23rd January 2019 |9:30 pm – 10:40pm (approx.) |Capstone Project room

Attendees: Via Discord:

- BR - Bilal Rana
- CS – Christophe Savard
- SZ - Steven Zanga

Present:

- AP – Ashesh Patel
- BT - Benjamin Thérien
- DT - Daniel Thibault-Shea
- MW – Micheal Wilgus
- MZ - Mordechai Zirkind
- RZ - Rezza Zairan
- SV - Shereece Victor

3.4.2 Summary of Discussion

Recap

- What did each person do this week?
- Organizers, pestered people for code, scheduled meetings
- Documenters: Learned Latex,
- SP is no longer a part of the documenters team and team in general
- Quality Assurance: Read up on unit testing, and JUnit
- Coders: Set up repo, and GitHub tools; Formatted MVC, made classes, some of the UML, some ER

We need to plan before coding too much so that we're all on the same page

Discussion of Game play

- We discussed the creation of the following objects: Board (2d array), Card (Type, State-Overturned or not, Word)
- Strategy Framework
- The MVC basically consists of three java packages
- Game Controller contains: 'start()' and the 'main_loop()'
- When someone clicks something in the view, which contains the cards, board, it (the event) gets sent to the controller as a request translates it to an operation and passes that to the model
- JavaFX has been added to our toolset
- What do we want the game to look like?
- Java Fx has been added to our toolset.

The basic code 'skeleton' has been set up by the coders.

Miscellaneous

- What data structure show we use for the words and their hints? Or do we do a database?
- How much does each member need to know? SV raised an issue with a lack of communication of tasks accomplished among the various task groups.
- A story/ use case is what we expect to happen when we objects

The basic code 'skeleton' has been set up by the coders.

3.4.3 Agreed upon goals

- Documenters: Do a detailed UML of the classes to be created.
- Coders: Set up database, JUnit tests
- Quality Assurance: JUnit tests

3.5 Meeting 3

3.5.1 Administrative

Wednesday, 30th January 2019 |9:30 pm – 10:30 pm |H 903 Lab

Attendees:

- AP – Ashesh Patel
- BR - Bilal Rana
- CS – Christophe Savard
- BT - Benjamin Thérien
- DT - Daniel Thibault-Shea
- MW – Micheal Wilgus
- MZ - Mordechai Zirkind
- RZ - Rezza Zairan
- SV - Shereece Victor
- SZ - Steven Zanga

3.5.2 Summary of Discussion

Pre-Demo - Requirements for Iteration 1

- The TA reviewed our progress and listed the things we were missing or needed to fix.
- These became our goals for the coming week.

3.5.3 Agreed upon goals

To be done:

- Table of contents.
- Domain model before use cases
- Diagrams for each use cases
- A MVC diagram
- Glossary of key terms
- Format the document properly
- State what relationships exist between objects
- Make sure there's a latex version of documents
- Code should be structured, remove unnecessary code, have an jar file in addition

- Note who created each class, and method
- Comments comments comments!
- Make sure code works
- Complete unit tests, comment, check
- Diaries, minutes etc
- Evidence of individual effort.
- Submit, documents and diaries as pdf
- Submit Individual diaries.

3.6 Meeting 4

3.6.1 Administrative

Wednesday, 6th February 2019 |9:46 pm – 10:33 pm |Capstone Rm 961-03

Attendees:

- AP – Ashesh Patel
- BT - Benjamin Thérien
- DT - Daniel Thibault-Shea
- MW – Micheal Wilgus
- MZ - Mordechai Zirkind
- RZ - Rezza Zairan
- SV - Shereece Victor

Via Discord:

- BR - Bilal Rana
- CS – Christophe Savard
- SZ - Steven Zanga

3.6.2 Summary of Discussion

Demo Status

- Recapped the events in the lab leading up to the cancellation of the days demo

Recap

- What have we done this week:
- AP: Has been trying to get familiar with GitHub
- MW: Understanding codes, wrote tests, code looks great, had questions about testing the GUI and undo, Answered by CS
- RZ: Use cases, formatting Latex
- SV: Model diagram, minutes; To do: class diagram, edit latex, create Latex version of minutes
- BT: Observers, game loop, method in each players, play turn, showed the game, already working, and demoed it to us, explained the next play button,
- CS: enforced standards, maintained GitHub repository, cleaned up the code, linked to gui,
- SZ: Commenting, Javadocs
- MZ: GUI
- BR: Coordinated with MZ to motivate the team

Work to be done

- Documenters: Proper error-free latex, images, annotations, diagrams, stylize documents, convert minutes to latex docs
- Coders: Cleaning out, making sure it runs smoothly, finish commander module so that it reaches the model, testing
- QA: user experience testing, model unit tests,

Things to be Learned:

- What do we need to learn by the completion of the project:
- MW: Latex
- BT: Unit tests, Latex
- RZ: Latex, code,
- AP: Git Kraken, JUnit
- MZ: Become familiar with the code, the document and UML
- SV: GUI,JUnit
- DT: JUnit, GitHub, become familiar with the code
- BR: Git Kraken, code base, Latex, JUnit
- CS: JUnit, Use cases,
- SZ: Latex

3.6.3 Agreed upon goals

- Think possible roles for the next iteration