- Rationale for each chosen domain and their relationships (if any)
- Were there any design choices that you had to make while modelling the domain and WHY?
- Explain any assumptions you have made, as well as any other part of your domain model that you feel warrants a justification as to WHY you have modelled it that way.

Assumptions

Mill Checker -> Position
To check that after each position a mill has been formed. And the mill checker checks through all the positions in the board. This happens after each move is performed.

MovablePositionFinder -> Position
To check through all the positions on the board to find empty adjacent positions, or in the case of flying pieces, empty positions.

Rationale for each chosen domain and their relationships

Player
Reason for choosing this domain
Since both HumanPlayer and ComputerPlayer will later have the same attributes (such as an arrayList of tokens) and methods (such as the basic moves of moving the token), Player is introduced as a parent class for both HumanPlayer and ComputerPlayer. It defines the standard for all the actions any type of player can take during the game, hence, Player will be defined as an abstract class so that there is no repetition of code in the HumanPlayer and ComputerPlayer classes. Having an abstract super class for Player, ensures that each type of player can have its own implementation details within the same domain.

Reason for the relationships

- Player-Move : Player can use Move to move their tokens at each turn to any valid position.
- Player-Token : Each Player has a total of 9 tokens at the start of the game. It might be reduced slowly one by one to a minimum of 2 tokens in which the game will end.

HumanPlayer
Reason for choosing this domain
The 9 Men's Morris game requires at least one HumanPlayer for it to be runnable. The HumanPlayer domain defines the implementation of any behaviours that any user of the game does

Reason for the relationships

- HumanPlayer-Player : HumanPlayer is chosen to be a child class to the Player in order to prevent repetition of items in multiple player classes. It extends from the abstract Player class.
- HumanPlayer-Game : A Game will have at least one HumanPlayer (ComputerPlayer Mode) and a maximum of 2 (HumanPlayer Vs HumanPlayer) depending on the game mode chosen.

ComputerPlayer (Advanced Feature)

Reason for choosing this domain

The user can select to play against a bot whose actions and behavioural implementation is defined by the ComputerPlayer domain.

Reason for the relationships

- ComputerPlayer-Player : ComputerPlayer is chosen to be a child class to the Player in order to prevent repetition of items in multiple player classes. It extends from the abstract Player class.
- ComputerPlayer-Game : A Game might have a ComputerPlayer is a user chooses to play with a bot.

Game (RMB TO EXPLAIN THE COMPOSITION)

Reason for choosing this domain

Reason for the relationships

MessageBox

Reason for choosing this domain

A message box is required mainly for 2 functions. Message box is displayed when a mill is formed by either of the players including the computer as an indication for the player to remove a token from the opponent. Other than that, message box is displayed when the game ends.

Reason for the relationships

- Game-MessageBox : The Game will display a minimum of 8 messages (7 mills minimum to win a game formed by the same player + 1 winning message).

Token

Reason for choosing this domain

Token is the main element in the game for the players to move.

Reason for the relationships

- Token-MovablePositionFinder : Each Token when being selected will have to check for its movable positions before a move is made on it.

MovablePositionFinder

Reason for choosing this domain

MovablePositionFinder is used to locate the empty positions that a selected token can be moved to considering the types of Move valid at the point in game (Sliding,Placing,Flying).

Reason for the relationships

- MovablePositionFinder-Position : MovablePositionFinder will have to go through all 24 Position(s) to locate for empty positions for tokens to be placed

Position

Reason for choosing this domain

Position is required in the game to form the basic design structure of 9 Men's Morris's board so that tokens can be placed.

Reason for the relationships

Position-Token : Each Position can have either 0 or 1 Token at any point in the game.

Board

Reason for choosing this domain

The board is the place where all the tokens can move and where ultimately, the actual game will take place.

Reason for the relationships

- Board-Position : The board has 24 positions where a token could be placed. This relationship is chosen to be represented in composition as Position is part of the Board and will not be capable of performing anything without the Board.

Display

Reason for choosing this domain

The game uses the display to show the current state of the board.

Reason for the relationships

- No Relationship with any other domains

Move

Reason for choosing this domain

Using move, actors can reposition tokens at different positions on the board.

Reason for the relationships

Move-MillChecker : After every move, the MillChecker checks if a mill has been formed.
Move-Position : Move updates the position of a token when it is repositioned.
Move-Token :
Bro check the msges in google docs rachit is msging

SlidingMove

Reason for choosing this domain

This move provides a way for tokens to move to adjacent positions, which is the basis for the game itself.

Reason for the relationships
SlidingMove-Move : SlidingMove is the most important move in the game.

FlyingMove
Reason for choosing this domain
This move provides a way for tokens to move to any unoccupied position on the board.

Reason for the relationships
FlyingMove-Move : Using this move, actors with three or less tokens can move their tokens to any currently unoccupied position on the board.

PlacingMove
Reason for choosing this domain
This is a way for pieces to be placed on the board during the start of the game.

Reason for the relationships
PlacingMove-Move : Using this move, actors can summon a new piece to the board for a maximum of 9 pieces each.

RemoveMove
Reason for choosing this domain
This is a way for the tokens to be removed from the board.

Reason for the relationships
RemoveMove-Move : Using this move, the mill checker can remove the opponent's token from the board if a player forms a mill.

MillChecker
Reason for choosing this domain
The mill checker checks all positions to find a mill.

Reason for the relationships
MillChecker-Position : The MillChecker interacts with the occupied positions to find three in a row.