THE SOGGY ZEBRAS

Rollerball

Jonathan Cowles, Angela Root, Dawson Canby, Courtney Torres, Jordan Peterson

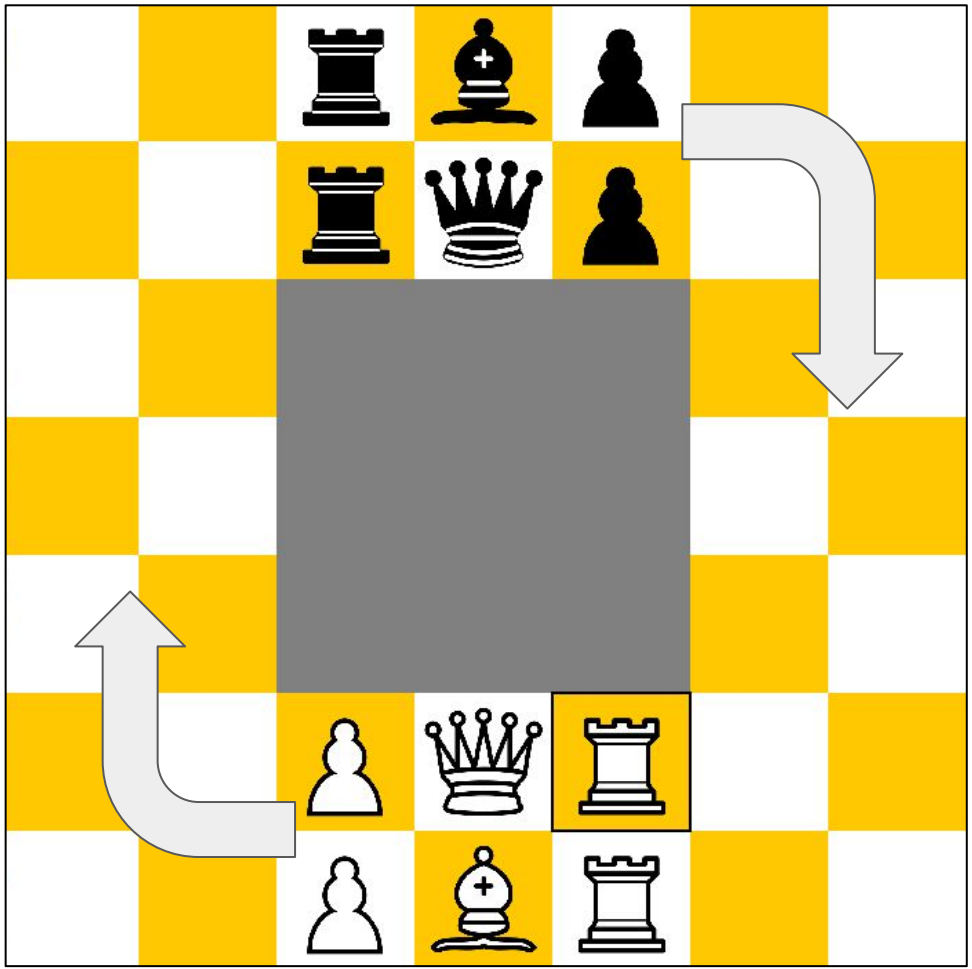
Deliverables:

- Use case document
- Domain model document
- Design document
  - Package diagram (Angela is working on this one :))
  - Full design class diagram - DONE
  - 3 sequence diagrams (Jonathan will auto generate these once the code is all merged)
- Testing document
- Development manual
- Refactoring and design pattern list
- Traceability link matrix
- Challenges and lessons learned

Here is her notes from our P3:

1. [Design document]
    a. • Class diagram
        i. ○ PieceDrawer, GUIRunner and Client are missing from the client class model
        ii. ○ The composition relationships between ⟨GameGUI , RollerballPanel⟩ and ⟨MenuPanel , MenuGUI ⟩ are inverted
        iii. ○ I couldn't find the implementation for:
            1. § the dependency between MenuGUI and GameGUI
            2. § The aggregation between CardContainer and MenuPanel
        iv. ○ What's the difference between the arrows between ⟨Player, Game⟩ and ⟨Game, Piece⟩?
        v. ○ The Server class is missing from the server class model
        vi. ○ The wire format classes are not represented in any diagram
    b. • Sequence diagrams
        i. ○ Good, although the format of all the diagrams should be consistent
    c.
2. [Testing document]
    a. • The tests should be more specific regarding steps to reproduce. They should provide precise test data.
    b.
3. [Source code]
    a. • Documentation can be improved
4.
5. [Development manual]
    a. • Instructions to run tests are missing
    b. • As a general recommendation, the manual should contain a section on software required to build, run and test the system, as well as the steps to install that software (or at least a link to the installation steps in the official websites). It should also include preferred IDE(s) to develop and steps to import the code into that IDE.
6.
7. [Traceability link matrix]
    a. • A lot of classes are missing. The TLM should be as comprehensive as possible.

Rollerball Review



Recap

- 7x7 Game Board with 3x3 hole in the center
- Pieces for each color
  - 1 King
  - 1 Bishop
  - 2 Rooks
  - 2 Pawns
    - If a pawn reaches the other teams starting locations of their pawns, they can be promoted to a Bishop or Rook.
- Player wins by putting other teams King in checkmate.

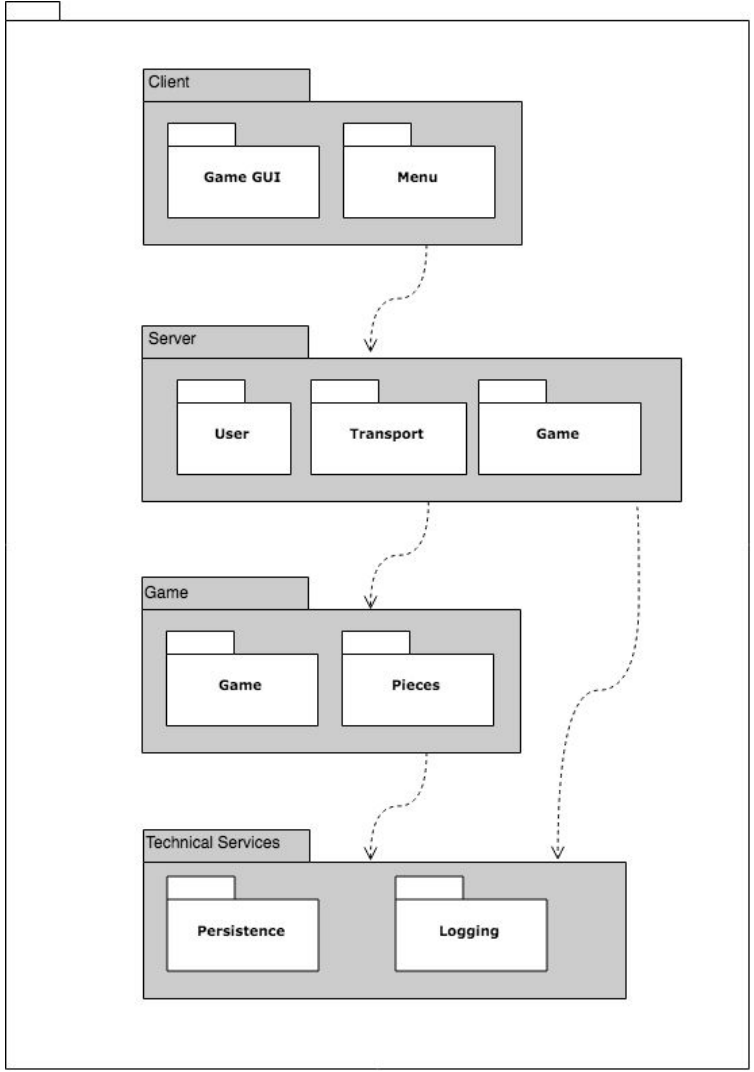
New Use Case: Playing with an AI Bot

Use case id:	12
Use case name:	Playing with an AI Bot
Brief description:	Player play a game with an AI instead of another Player
Type:	User Goal
Primary actors:	Player
Secondary actors:	None
Pre-conditions:	None
Main flow:	1. Player starts game with AI
Alternate flow:	1. Player wants to play game with other player
Post-conditions:	1. Player is able to play with AI

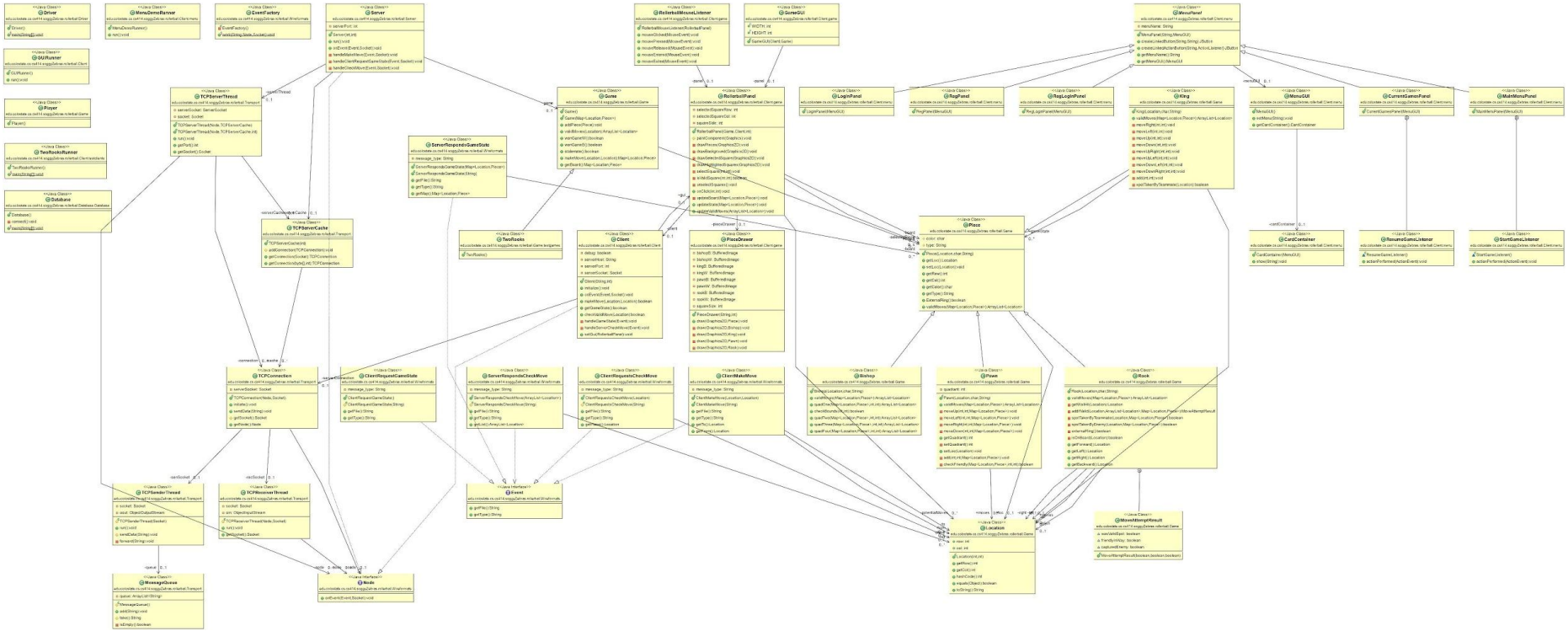
Traceability Matrix

	1_LOGIN	2_LOGOUT	3_UNREGISTER	4_REGISTER	5_RESUME_GAME	6_QUIT_GAME	7_ACCEPT_INVITE	8_PLAY_GAME	9_CREATE_GAME	10_INVITE_PLAYERS	11_REJECT_INVITE	12_PLAYING_AI
gameGUI	x			x	x	x		x				x
RollerballPanel	x			x	x	x		x				x
CreateInvitePanel	x			x					x	x		
LoginPanel	x			x								
MainMenuPanel	x	x	x	x	x		x	x	x	x		x
MenuGUI	x	x	x	x	x		x	x	x	x		x
PendingInvitesPanel	x			x			x	x		x	x	
RegLoginPanel	x			x								
RegPanel				x								
Database	x	x	x	x	x	x	x	x	x	x	x	x
Bishop					x	x		x	x			x
Game					x	x		x	x			x
King					x	x		x	x			x
Location					x	x		x	x			x
Pawn					x	x		x	x			x
Piece					x	x		x	x			x
Rook					x	x		x	x			x
GameCache	x			x	x	x	x	x	x	x	x	x
Invite	x			x					x	x		
Server	x	x	x	x	x	x	x	x	x	x	x	x
User	x	x	x	x	x	x	x	x	x	x	x	x
Transport	x	x	x	x	x	x	x	x	x	x	x	x
ClientMakeMove	x			x				x				x
ClientRequestGameState	x			x	x	x		x	x			x
ClientRequestsCheckMove	x			x				x				x
ClientRespondsInvite	x			x			x			x	x	
ClientSendsUnregister	x		x	x								
ClientSendsInvite	x			x					x	x		
ClientSendsLogin	x			x								
ClientSendsLogout	x	x		x								
ClientSendsRefresh	x				x			x				x
ClientSendsRegistration				x								

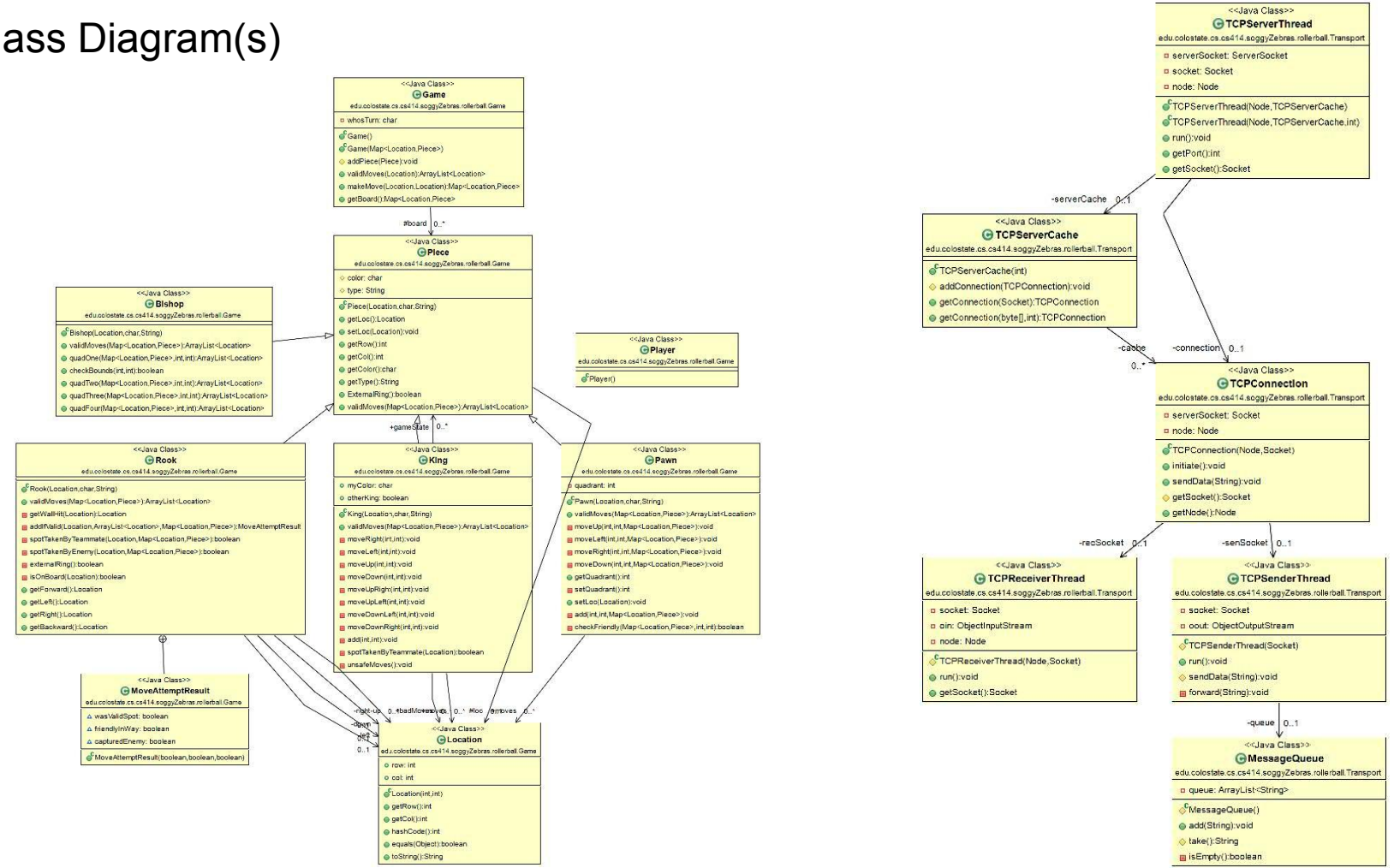
Package Diagram



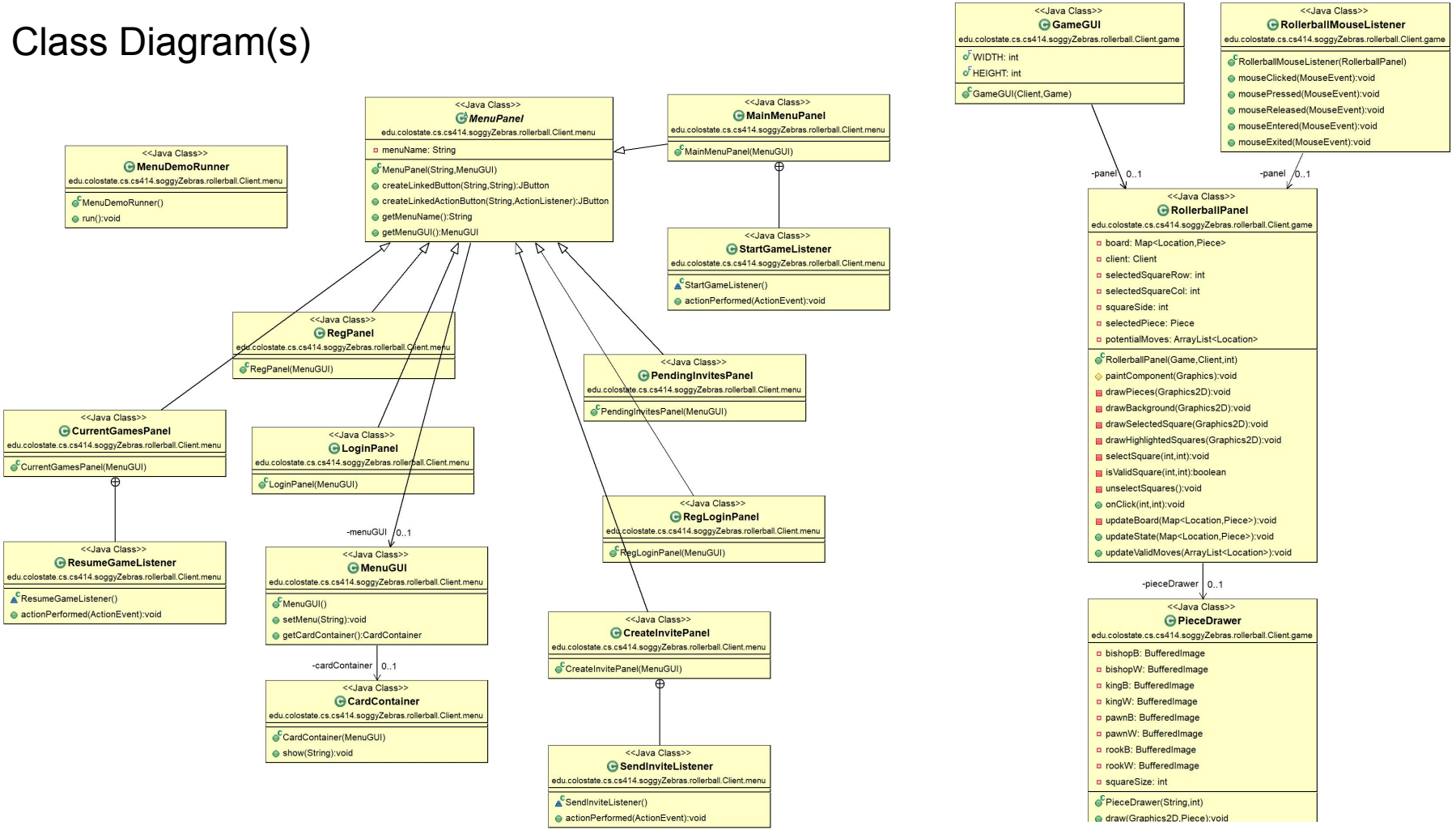
Class Diagram(s)



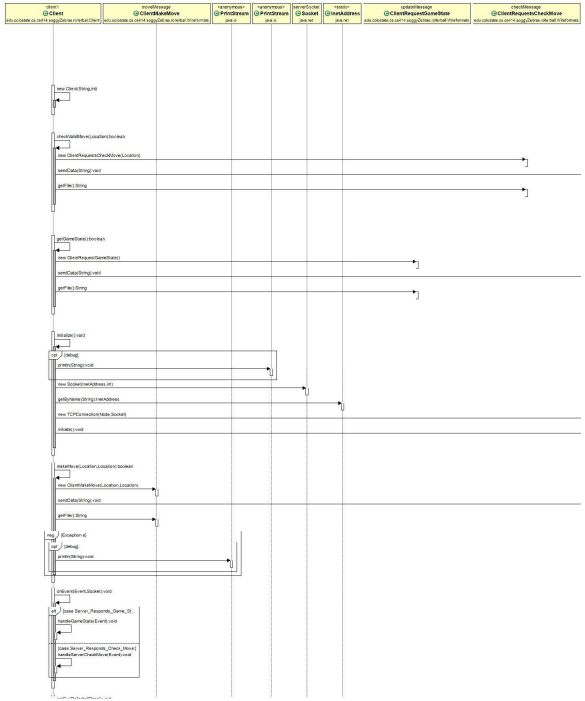
Class Diagram(s)

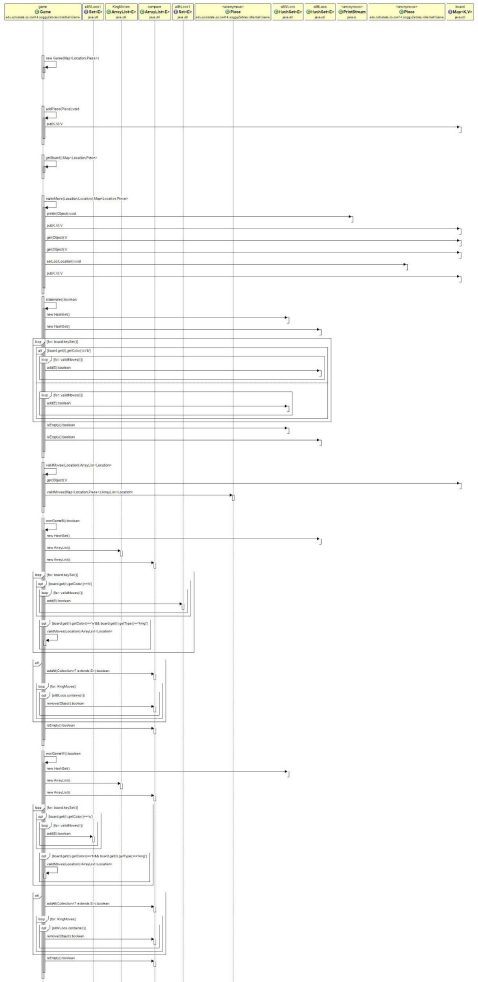


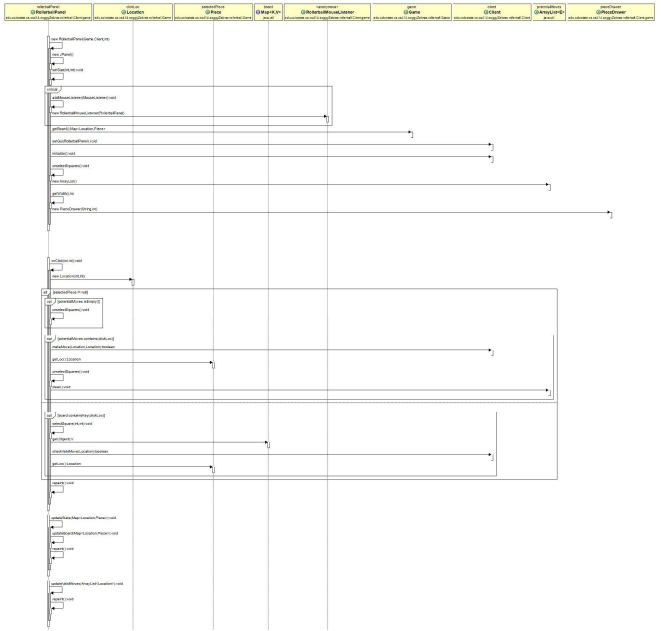
Class Diagram(s)



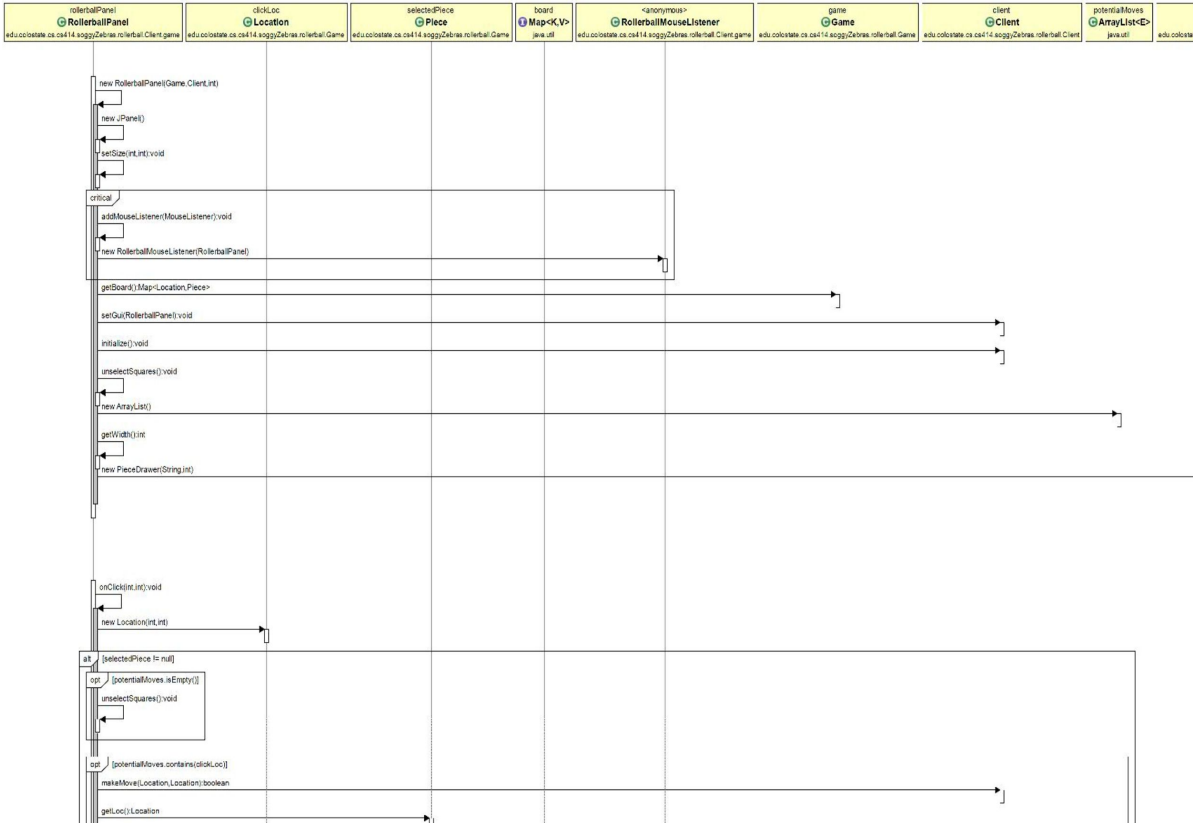
Sequence Diagram(s)

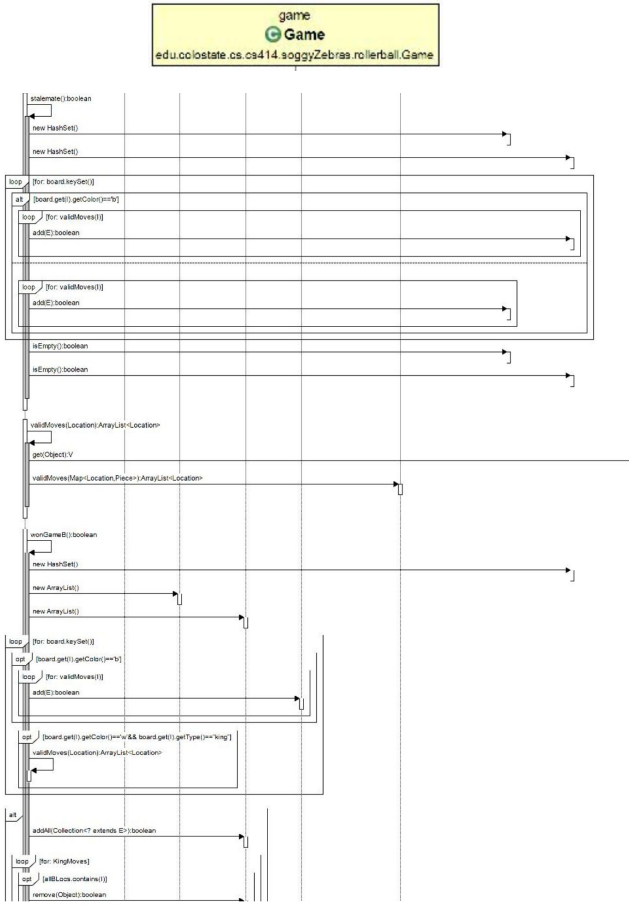






Sequence Diagram(s)





Challenges and Lessons learned

- **Object Serializability**
- **Databases**
- **Code Design**
- **Time Conflicts**
- **GUI programming**
- **Migration from single instance to distributed system**