# Traverse

Software Requirements Specification

Draft

Samuel Armstrong

COSC 3403 – Software Requirements Specification

## 1. **Scope**.

1.1 **Identification**. This document references the general computer video game implementation of the board game Traverse.

1.2 **System overview**. Traverse is a game similar to Chinese checkers. Two to four players control their own pieces and attempt to get those pieces to the other side of the board. Pieces may jump over other pieces if they would normally be able to move into the other pieces location, should the other pieces not have been there, and the opposite space of the other piece does not have a piece. This may occur multiple times with one piece in a player's turn. Each player has four piece types each with different movement patterns. The video game implementation includes computer players and at most one human player. The game is controlled through the command line. The board states are represented only with printable ASCII characters. The user may use a menu to set game settings and monitor the state of the current game including when there are no human players. The board may be displayed at the end of the game, after a set number of rounds, or after every turn.

1.3 **Document overview**. Section 2 lists the documents used in the creation of this SRS. Section 3 describes the specific requirements of the CSCI. Section 4 describes the four qualification methods. Section 5 contains a table describing how section 3 fulfills the requirements of the methods in section 4. Appendix A contains a use case diagram for the CSCI. Appendix B contains a class diagram for the CSCI. Appendix C contains state transition diagrams.

## 2. **Referenced documents**.

DI-IPSC-81433; SOFTWARE REQUIREMENTS SPECIFICATION (SRS), Unknown 05 December 1994
N/A; COSC3403 Project #3 Requirements, Unknown
N/A; COSC3404 Project #2 Requirements, Unknown

## 3. **Requirements**.

### 3.1 **Required states and modes**.
Traverse shall have states for the game controller, board, and pieces. The controller shall be receiving player settings, displaying the board, waiting for the player's move, causing the computers' moves, and announcing when the game is complete. The board shall have two, three, or four sets of pieces on it and will calculate whether a piece movement is valid. The pieces shall either exist on the board or will be absent from the current game. For state diagrams on these classes, see Appendix C.

### 3.2 **CSCI capability requirements**.

#### 3.2.1 **Players**.

3.2.1.1 **Player count**. Traverse shall allow two, three, or four players to play in one game. Additionally, at most one player may be human.

3.2.1.2 **Computer opponents**. The up to four non-human players must be able to perform moves independent of a human or outside computer's intervention.

3.2.1.3 **Player Turns**. One piece of the player's color must be moved on a given player's turn. If a player has no valid moves, their turn is skipped.

#### 3.2.2 **User controls**.

3.2.2.1 **User menu**. Traverse shall allow the player to have an interactive menu which can start the game, determine the type and kind of players, set a maximum amount of rounds for the game, and view the board state during the game, even if there is no human player in the current game.

3.2.2.1 **Non-human game board monitoring**. When there are no human players in a game, the user shall have the option of displaying the board after each turn, after the maximum number of rounds, or at the end of the game.

#### 3.2.3 **Starting the game**.
Upon starting the game, each player is assigned a color and must arrange all of their pieces along the edge of the board closest to them without using the corner spaces of the board.

#### 3.2.4 **Pieces**.

3.2.4.1 **Piece movement**. Pieces on the board shall be able to be moved by the game controller either under the instruction of the user, if the pieces are under human control and it is the human player's turn, or as a part of the non-human players' turns. A piece may not take a movement which would place it in one of the board's corner spaces.

3.2.4.2 **Piece Types**. Pieces shall have a color to determine their respective player as well as a shape. Square pieces may be moved orthogonally. Diamond pieces may be moved diagonally. Circle pieces may be moved orthogonally or diagonally. Triangle pieces may be moved diagonally away from the player's starting edge or orthogonally towards their starting edge.

3.2.5 **Jumping**. A piece shall have the ability to move to the opposite side of another piece (referred to as "jumping") provided that the other piece is in a board space which the jumping piece would normally be able to move on to and that there is an empty board space on the directly opposite side of the other piece. A piece may jump as many times as its player chooses that turn provided the piece has not jumped over the same other piece and across the same board spaces that turn.

3.2.6 **Ending the game**. The game shall end at the end of a player's turn if that player has all of their pieces on the edge of the board opposing the edge on which their pieces started or if the number of rounds specified by the user pass. The user shall be notified that the game has ended and the CSCI shall close.

## 3.3 **CSCI external interface requirements**.

3.3.1 **Interface identification and diagrams**. The CSCI shall interface with the computer's command line interface (hereafter referred to as the "terminal"). Information is passed between the terminal and the game controller.

3.3.2 **The terminal**. The terminal shall send the user's keyboard input to the game controller through the terminal's standard input, and display menu, board, and other information from the game controller as needed. The interface is fixed. The game board, pieces, and menu shall only be output using printable ASCII characters. The terminal output's character width shall not exceed 70 columns of characters. The terminal output shall scroll after the user enters input rather than change the output using (X,Y) positioning of the cursor.

3.4 **CSCI internal interface requirements**. Any internal interface requirements are left to the design.

3.5 **CSCI internal data requirements**. TBD

3.6 **Adaptation requirements**. N/A

3.7 **Safety requirements**. N/A

3.8 **Security and privacy requirements**. TBD

3.9 **CSCI environment requirements**. The CSCI shall operate on users' personal computers.

3.10 **Computer resource requirements**.

3.10.1 **Computer hardware requirements**. The computer must have a keyboard or equivalent which can accept ASCII input and a visual output device, such as a monitor, with a minimum resolution of 360p.

3.10.2 **Computer hardware resource utilization requirements**. The CSCI shall not use more than one core while operating.

3.10.3 **Computer software requirements**. The user computer must have installed a standard command line interface such as the Windows Command Line or BASH.

3.10.4 **Computer communications requirements**. The CSCI shall not require a network connection to operate.

3.11 **Software quality factors**. TBD

3.12 **Design and implementation constraints**. The Traverse video game must follow the same rules as the physical board game.

3.13 **Personnel-related requirements**. TBD

3.14 **Training-related requirements**. Users of the CSCI shall have a minimum of a basic understanding of how to use their computer's terminal.

3.15 **Logistics-related requirements**. TDB

3.16 **Other requirements**. N/A

3.17 **Packaging requirements**. N/A

3.18 **Precedence and criticality of requirements**. The highest priority requirements are those in 3.2 regarding the rules of the game and 3.3 and 3.10.3 which state that the CSCI must run on a terminal. 3.10.1 and 3.10.2 are of lower priority should other requirements be deemed more important and mutually exclusive.

4. **Qualification provisions**.

a. Demonstration: The operation of the CSCI, or a part of the CSCI, that relies on observable functional operation not requiring the use of instrumentation, special test equipment, or subsequent analysis.

b. Test: The operation of the CSCI, or a part of the CSCI, using instrumentation or other special test equipment to collect data for later analysis.

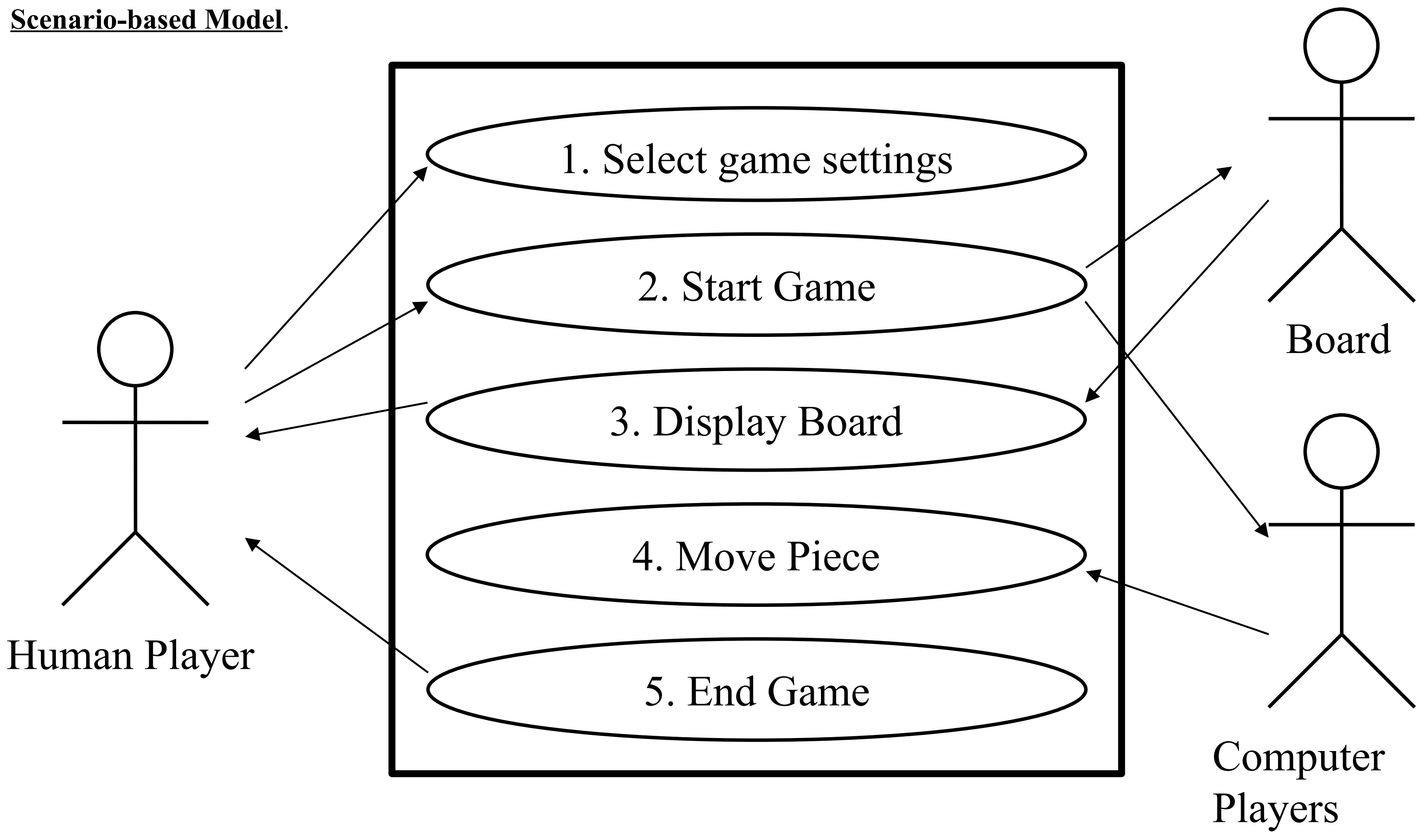
c. Analysis: The processing of accumulated data obtained from other qualification methods. Examples are reduction, interpretation, or extrapolation of test results.

d. Inspection: The visual examination of CSCI code, documentation, etc.

## 5. **Requirements traceability**.

| SRS Paragraph | Short Description | Qualification Model |
|---|---|---|
| 3.2.1.1 | There are limited computer and human players | Attempt to begin the game with greater, fewer, and incorrect values of players. |
| 3.2.1.2 | Computer opponents must operate independently | Observe the progress of a sandboxed CSCI game with only computer opponents. |
| 3.2.1.3 | A player must make only one move or is skipped if there are none. | Attempt to enter illegal moves of multiple pieces in a turn and artificially set the board in a state where a player cannot move. |
| 3.2.2.1 | There must be a functioning user menu. | Attempt every option in multiple ways on the implemented menu. |
| 3.2.4.1 | A piece may not end its turn in a corner space. | Attempt to land every kind of piece in every corner on the board. |
| 3.2.4.2 | There are different pieces with different movement patterns. | Attempt many kinds of movement, including illegal kinds, with every kind of piece. |
| 3.2.5 | Pieces can jump other pieces. | Artificially set the board to attempt jumps involving one piece, multiple pieces, the same piece in different directions, or the same piece in the same direction. |
| 3.2.6 | The game ends if a player's pieces make it across the board or if a set number of turns pass. | Attempt to have a human and computer player win and let the rounds run out. |
| 3.3.2 | The terminal shall only use valid output. | Count various lines for width and send the output to a program to check for non-visible ASCII. |
| 3.10.2 | The CSCI shall only use one core. | Attempt the CSCI on a single core computer and check its usage on a multicore computer. |

A. **Scenario-based Model**.



**Traverse Use Case Diagram**

**Use Case #1: Select Game Settings**

Actor: Human Player

Description: I, the human player, determine the rules of the game including: which of four colored pieces has a human, computer, or no player with a maximum of two pieces with no player, what is the maximum number of rounds, when to display the board during the game, and starting the game.

**Use Case #2: Start Game**

Actor: Human Player, Computer Players, Board

Description: I initiate the game with the set rules and generate the board and computer players.

**Use Case #3: Display Board**

Actor: Human Player, Board

Description: I am given the current state of the board.

**Use Case #4: Move Piece**

Actor: Board

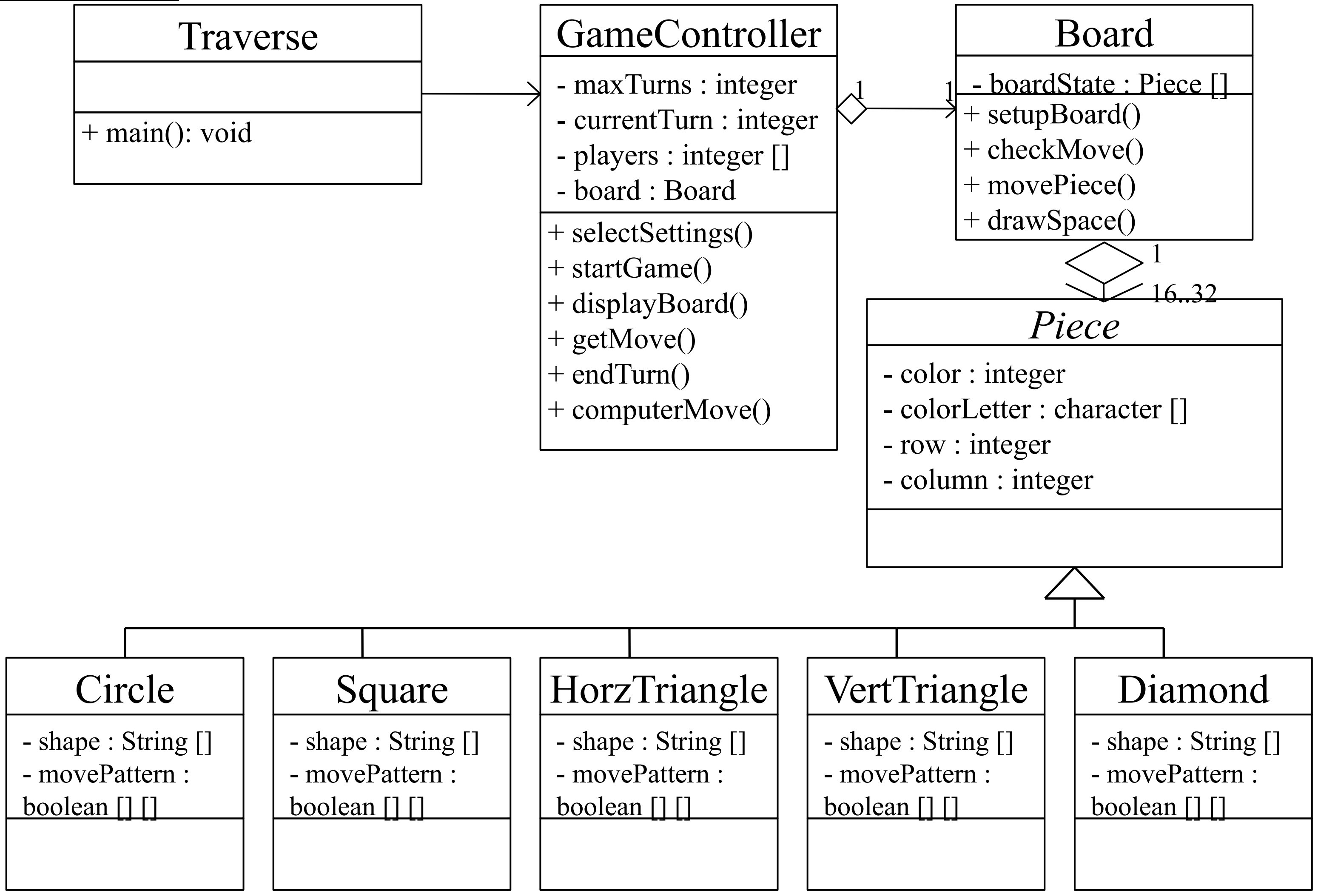
Description: The board changes its state as determined by a computer player or myself.

**Use Case #5: End Game**

Actor: Human Player

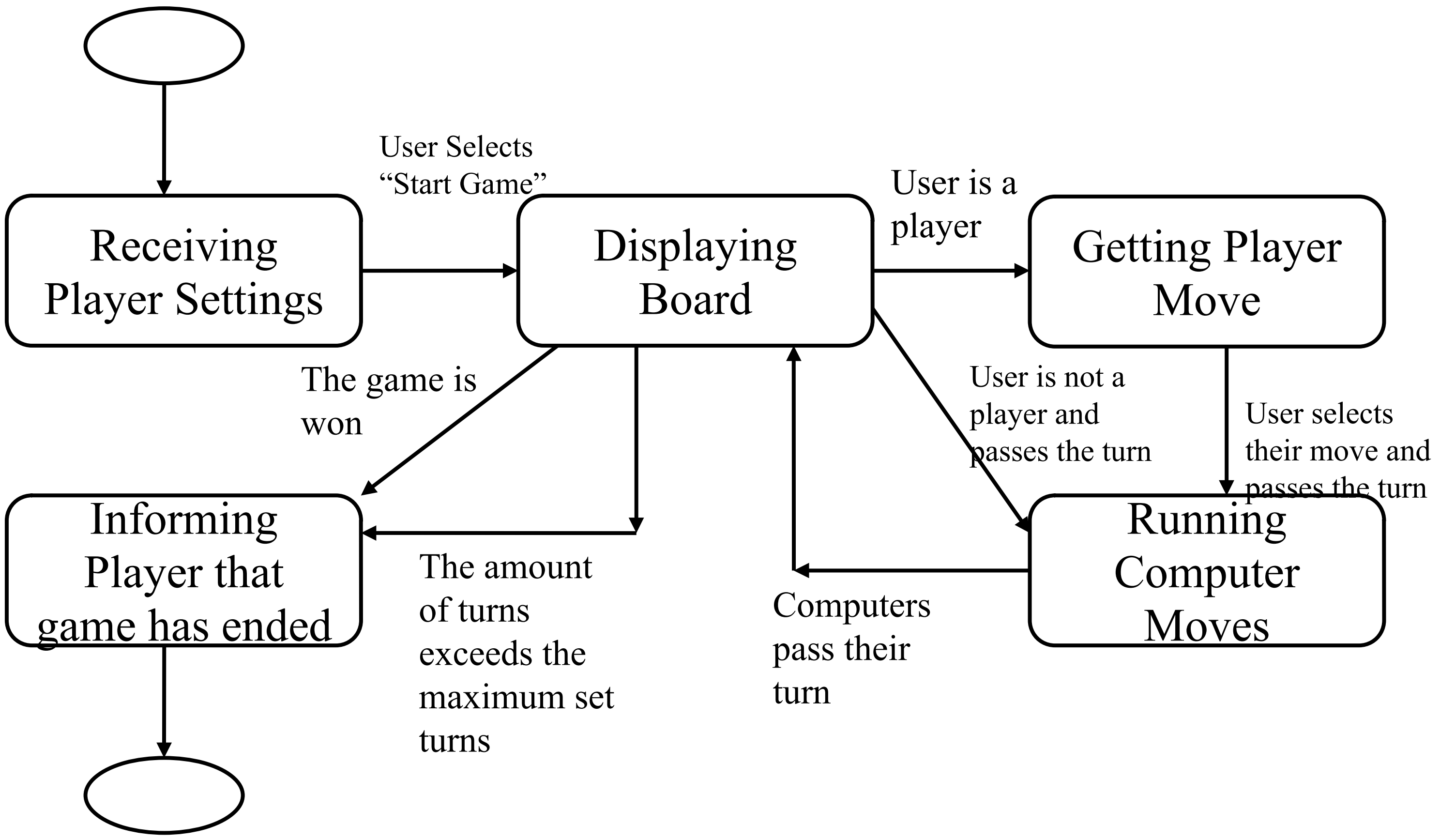
Description: A board end state is reached, I am informed of the end of the game, and the program terminates.

B. **<u>Class-based Model</u>**.



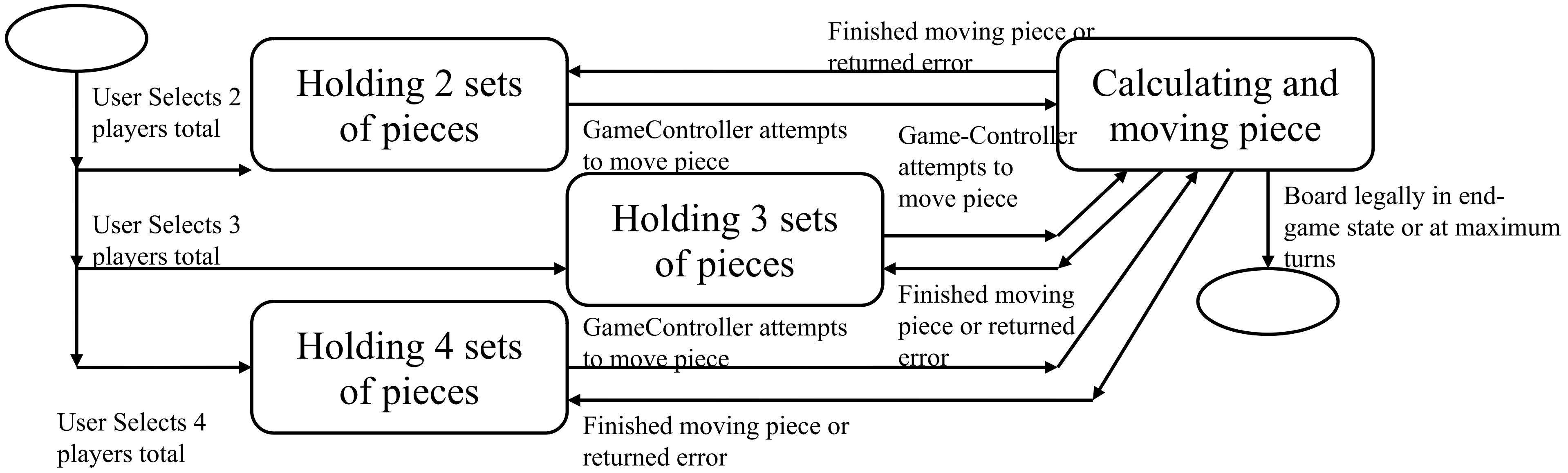
**Traverse Class Diagram**

C. **Behavioral Model**.

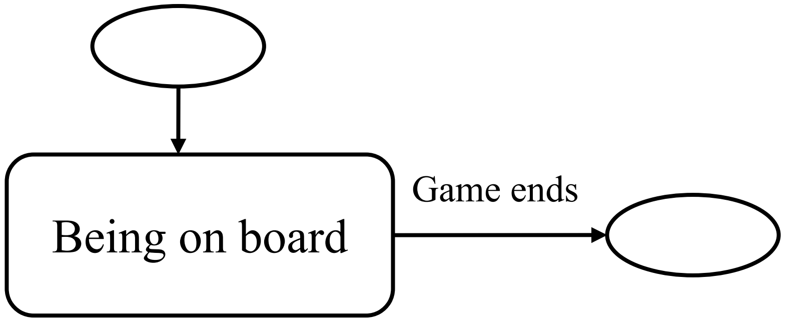


User Selects "Start Game"
User is a player
Receiving Player Settings
Displaying Board
Getting Player Move
The game is won
User is not a player and passes the turn
User selects their move and passes the turn
Informing Player that game has ended
Running Computer Moves
The amount of turns exceeds the maximum set turns
Computers pass their turn

**State Diagram for GameController Class**


Holding 2 sets of pieces
Holding 3 sets of pieces
Holding 4 sets of pieces
Calculating and moving piece
User Selects 2 players total
User Selects 3 players total
User Selects 4 players total
GameController attempts to move piece
Game-Controller attempts to move piece
GameController attempts to move piece
Finished moving piece or returned error
Finished moving piece or returned error
Finished moving piece or returned error
Board legally in end-game state or at maximum turns

**State Diagram for Board Class**



**State Diagram for Piece Class**