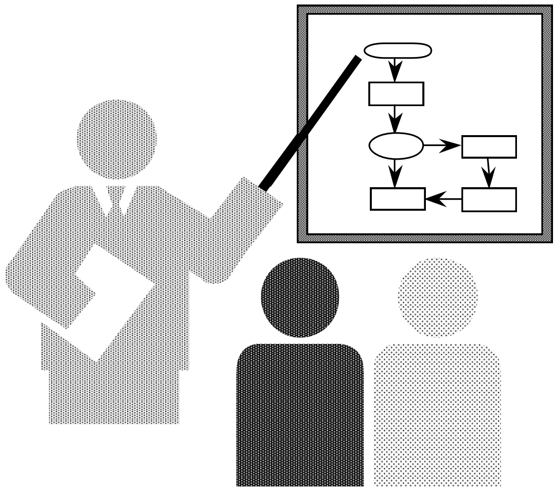
TTT Project
Requirements Document / Use Case Analysis



JUNE 7, 2016

Team Members:

Name	**ID**	**Email Address**
Shen Li	27913907	shenli2015@yahoo.com
Isha Mehta	27376030	ishakmehta04@gmail.com
Zhihao li	27252331	Lizhihao1109@gmail.com
Akjil Kumar	27711794	reddyakhil24@gmail.com
Chenyang Li	27588801	tomli8chenyangli@gmail.com
Xiangshiyu Li	27320841	lixiangshiyu@gmail.com
Meng Yao	26847005	yaomenghncn@gmail.com
Singh Mehare	40012584	birdevinder1705@gmail.com
Golnoush Lotfi	26753906	golnoush.lotfi@gmail.com

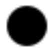
Contents

1. State Machine Diagram- Introduction

A state diagram is a UML diagram that depicts the dynamic aspects of the system and describes different state of an object during its lifetime.
A state diagram has States, Transitions and Actions. For which a State of an object is depicted by rounded rectangle and shows the set of values of its attributes at some point of time. A class can have at least one state. A transition happens when a signal or event fires and a change is state happens. A state has exactly one associated action, executed upon entry to the state.
Following notations are used to model a State Diagram:-

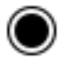
- **Initial State Vertex**

  It is indicated with the filled circle and represents the object's initial state.



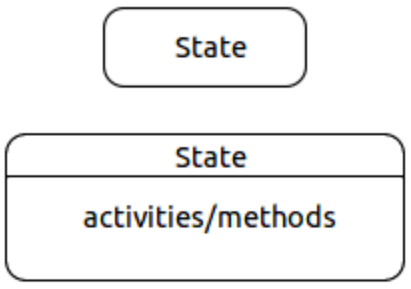
- **Final State Vertex**

  It is indicated by two concentric circle with inner circle filled representing object's final state.



- **State**

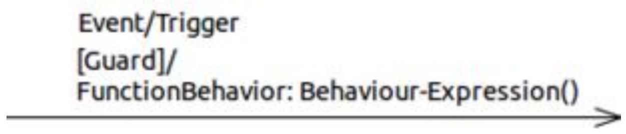
  A state of an object represents a set of values of its attributes at certain point of time. It can be a simple state or a state with internal activities. It is indicated by a rectangle with rounded corners.



- **Transition**

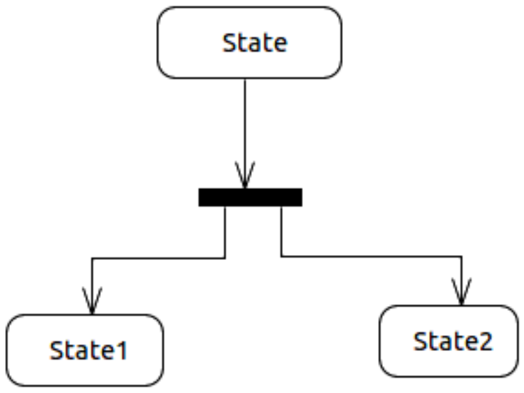
  Transition represents a change of state in response to a signal or event and is considered to occur instantaneously. The labels on each transition are signals that cause change of state. Those labels can have 3 parts:
  a) Event/Trigger- It specifies the event that induces the state transition or state change.

b) Guard- It specifies the boolean expression in terms of triggering event.
c) Behavior expression- It is a method that is executed when the transition is fired.



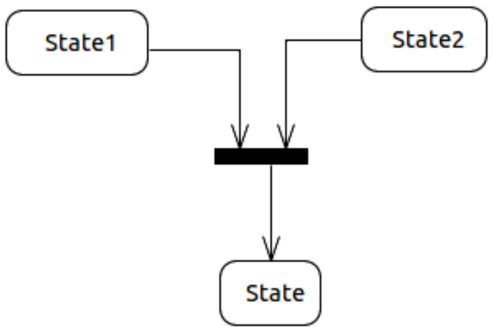
- **Fork**

  Fork is used when a transition is split into concurrent multiple transitions. It is indicated by a solid line.



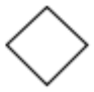
- **Join**

  Join is used when concurrent multiple transition is merged back into one transition. It is indicated by a solid line.



- **Decisions**

  When conditional situation occurs in the state transition, decision is used. It is indicated by a diamond.



2. State Machine Diagram

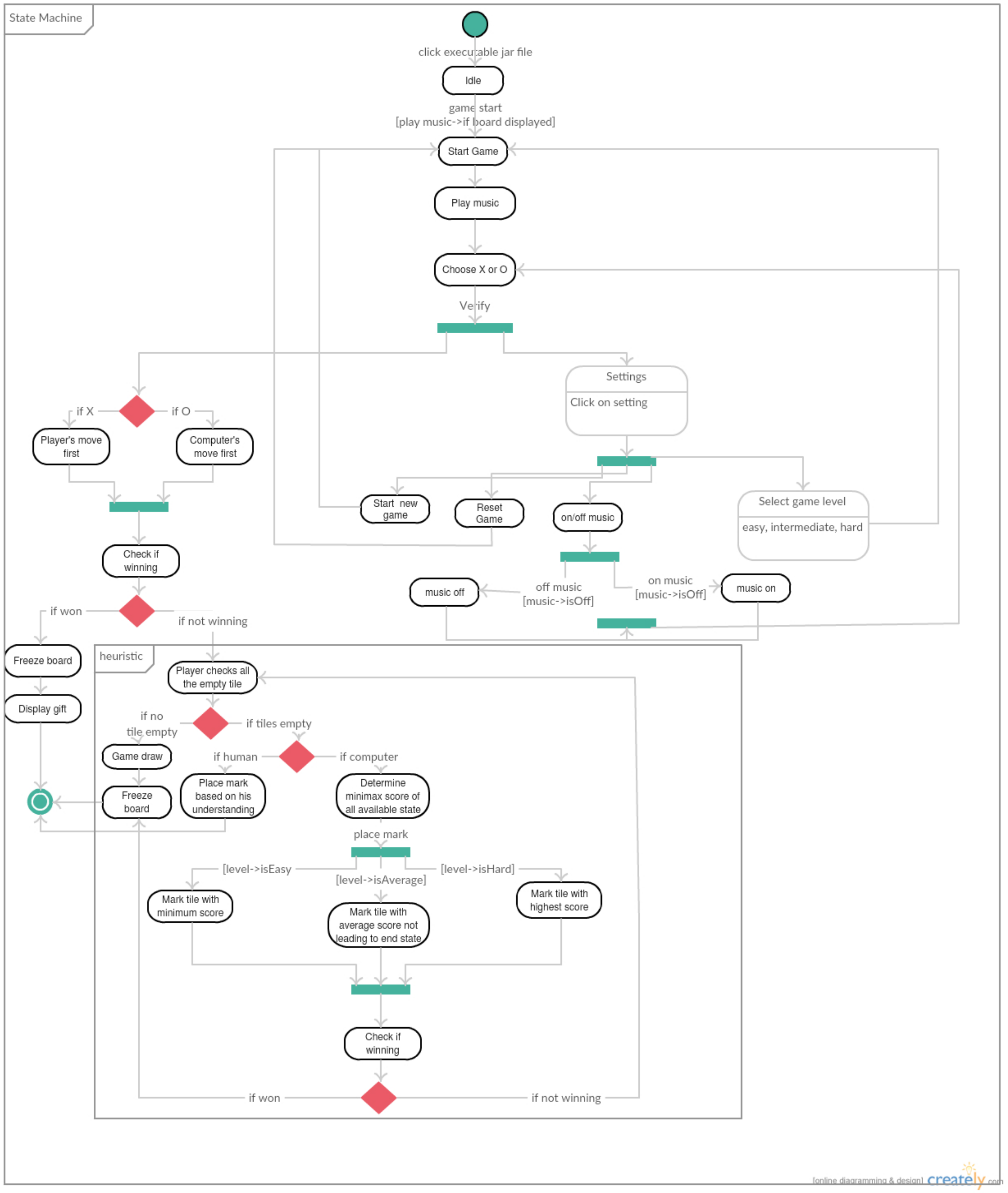


Figure 1 State Machine Diagram for Tic-Tac-Toe

3. Explanation of Tic-Tac-Toe state machine diagram

Figure 1 above shows the state machine diagram for the Tic-tac-Toe game. As seen the, the game begins when a human player clicks an executable jar file. As the game starts, a 3*3 board is displayed which is the guard condition for the system to play the music. The system then allows the player to pick 'X' or 'O'. If the player selects X, then player has the first move. This is because a Tic-tac-Toe game always begins with an X. Similarly, if player selects O, then computer has first move.

After each move the system checks if the winning condition is satisfied. If a player or the computer, any of one wins, then the board freezes. If the human player wins then the system issues a gift to the human player. This is only if the winner is a human player. This acts as a guard condition for display gift. After that the game ends.

If the winning condition is not satisfied, the player checks all the empty tile. A human player cannot be guessed. It depends on player where to place the next mark. For the computer player, it follows minimax algorithm. This is the main heuristic of this game. The computer will compute the minimax score of all the available state. Now, the computer's move depends on the level selected for the game by the human player. If the player is playing on the easy level then the computer will try not to win and place the mark on the tile with the minimum score. Likewise, if the level selected is average, the computer places the mark on the tile that leads to maximum score but not immediately. That means, the mark is placed on the tile which eventually leads to winning condition in next one or two or more steps. If the level selected is hard then the computer places the mark on the tile with the maximum score. It is important to note that, not always the computer will find the highest score. In such a case, computer will follow the average level rule. After this step again the winning condition is checked. If won then freeze board and stop the game if not then repeat the above step from checking the empty tiles.

The human player also has a settings button from where he/she can select the difficulty level of the game and on or off the music. Music can be on only if it was off previously. Similarly it can be turned off if it was on previously. This acts as a guard condition.

References

1. Hamou-Lhadj, P. A. (2016). *Modeling with state diagrams.* Retrieved from https://moodle.concordia.ca/moodle/pluginfile.php/2186025/mod_resource/content/1/LEC4-ModellingBehaviour.pdf

2. Creately, Online UML Modeling tool,[Online] Available at: https://creately.com/app/#

3. Sparx System, UML2 state machine diagram, [Online] Available at: http://www.sparxsystems.com/resources/uml2_tutorial/uml2_statediagram.html