Requirements and Analysis Document

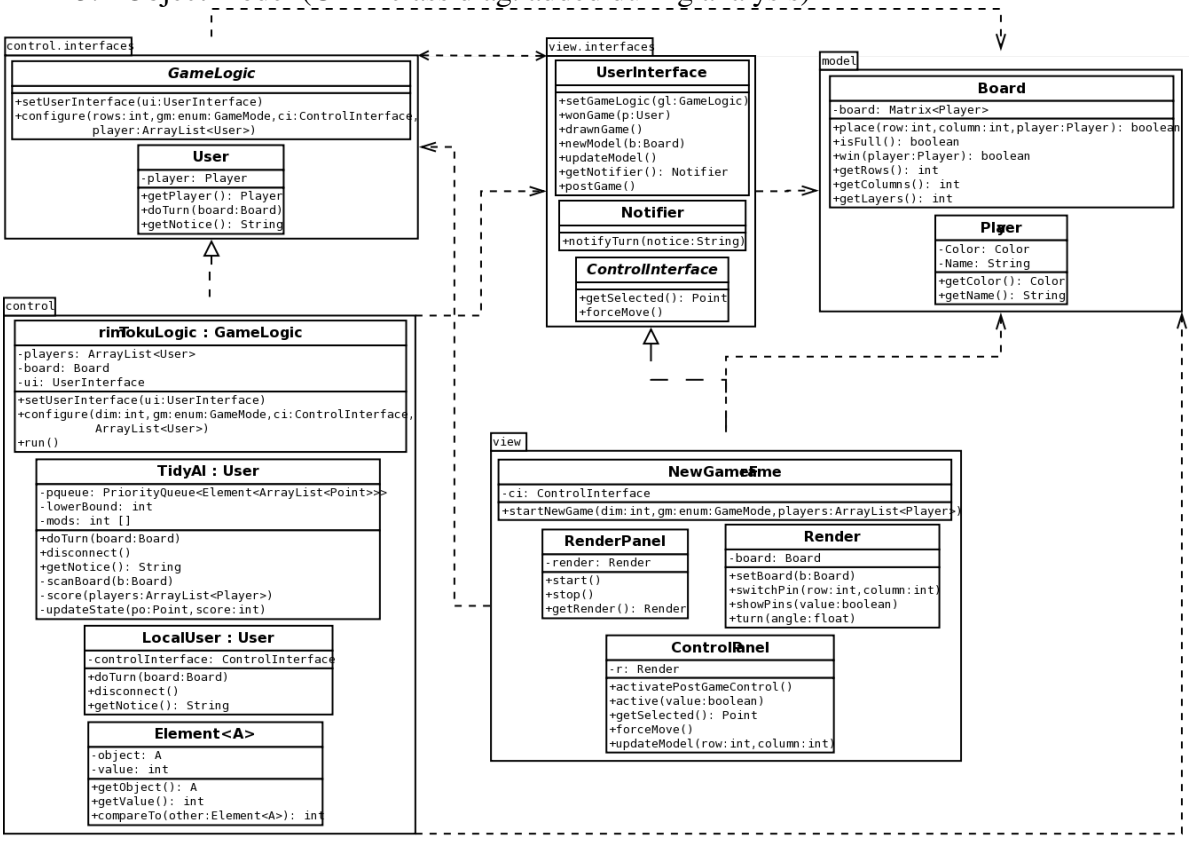
(Note: This is part of the contract between the group and the customer (assistant). May not be changed without agreement from the customer.)

1. Introduction
    1. Purpose of application (system)
       A simple game that should easy to play, but hard to master.
    2. General characteristics of application
       Single application. A board game. 3D graphics! Fairly competent AI. Should work under Windows, Linux and OS X.
    3. Scope of application
       Play a round to completion (win, lose or draw).
       Play either against an AI, or in a “hot seat” manner between two human players.
    4. Objectives and success criteria of the project
       A working application that hopefully exceeds the demands for passing the course.
    5. Definitions, acronyms and abbreviations
2. Proposed application
    1. Overview
       We intend to implement Gomuku (five in a row) in 3D. In addition to the two normal dimensions there is also height. This adds several possible ways to win. There are also several new strategies that the players have to use, for instance stacking your pieces to achieve five pieces in a diagonal row.
    2. Functional requirements
       Start a new game.
       One game at a time.
       Only two players in a game.
       Some manner of 3D representation of the playing board so the players can tell what is going on. Being able to make columns opaque so that the inner parts of the board can be inspected.
       No possibility to undo moves.
       No time limit on moves.
    3. Non-functional requirements
        1. Usability
           The game should be accessible for most people, so the user interface has to feel intuitive and the number of controls kept to a minimum.
        2. Reliability
           Won't have any network functionality, so the potential problem area is the 3D graphics. This will be very basic, so the application should be quite reliable. Some synchronization issues might arise, since we may not have full control over the rendering.
        3. Performance
           The algorithm for check if a player has won might become heavy for larger boards, but

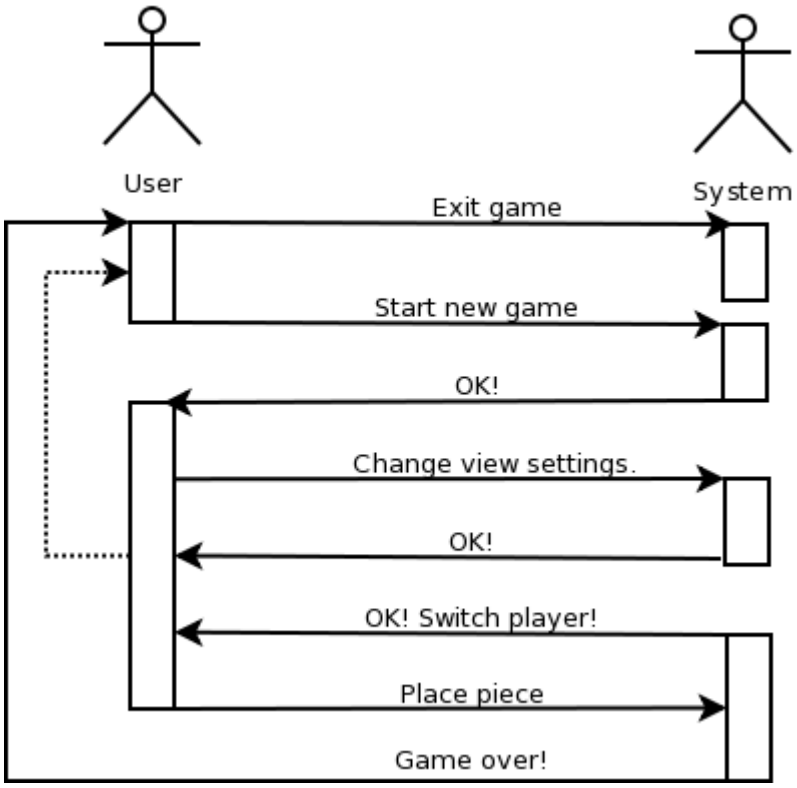
since reaction time isn't important for the game, this is of less concern. The 3D elements will be fairly basic, so this shouldn't be a problem for all but the most basic machines.

4. Supportability
   No support will be given once the application is completed.
5. Implementation
6. Interface
   Standard window layout. One large part devoted to showing the 3D board and a smaller part to house controls. Separate windows for some features (game won etc.).
7. Packaging
   In accordance with the MVC principle.
8. Legal
   The game is ancient and belongs to the people. As will our application.

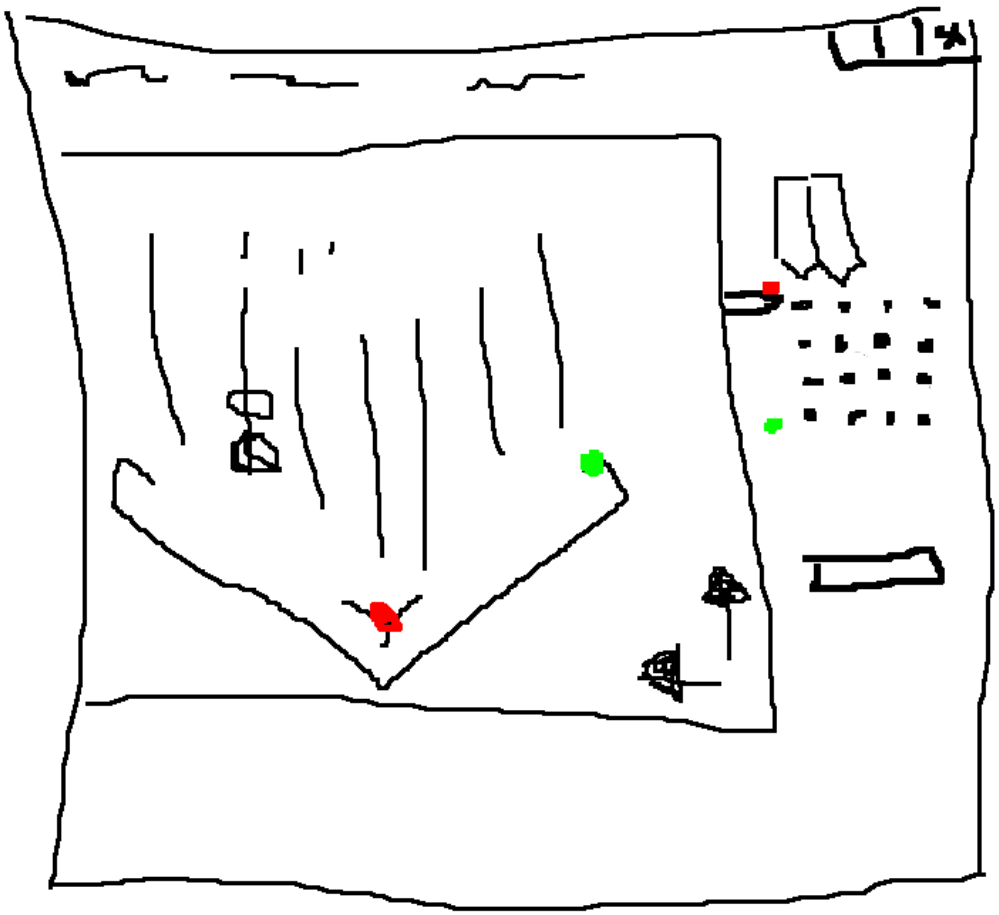
4. Application models
   1. Scenarios (added during requirement)
   2. Use case model (added during requirement)
   3. Object model (UML class diag. added during analysis)



4. Dynamic model (UML sequence, added during analysis)



5. User interface
    1. Screen mock-ups (added during requirement)



    2. Navigational paths (added during requirement)
5. Possible future directions
   Rematch capabilities (the users can of course play repeatedly, but no win/loss statistics are saved). Random players assignment for a more fair game. Piece customization (shape, texture selection). Pin selection in the 3D view.
6. Glossary
7. References