System Design Document

1. Introduction
    1. Design goals

        We value simplicity over an extendable design and plug-ins, but we'll try to keep our interfaces as general as possible, so that you could “easily” write a new graphics subsystem, for instance.

    2. Definitions, acronyms and abbreviations
    3. References
2. Proposed software architecture
    1. Overview (see RAD 1.2)

        We're doing a simple game and therefor intend to use the MVC design principle.

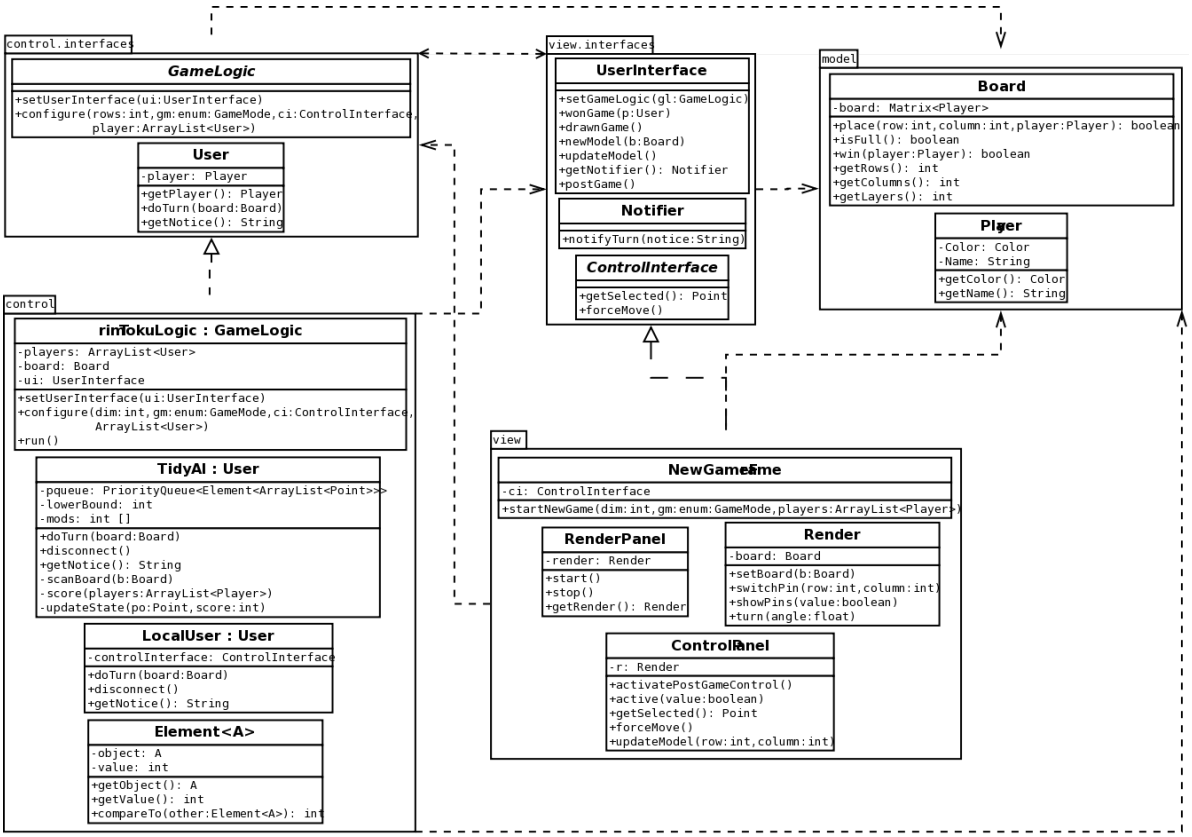
    2. Hardware/software mapping (UML Deployment diag.)

        PC (linux, windows, osx) with a graphics card and the Java runtimes.

    3. Software decomposition
        1. Decomposition into tiers (UML Deployment diag.)

            N/A

        2. Decomposition into subsystems / Layering / Dependency analysis (UML Package diag.)



4. Concurrency issues

   Our 3D things will run in a separate thread from the game logic, so there is the potential for concurrency issues. The risk that anything will break in such a limited and unrelated environment is low.

5. Persistent data management

   In the most basic version of the game there won't be any features of this sort. One possible extension of the project could be to include online play, in which case a ranking system with leader boards and such could be interesting. It is doubtful that we'll have time to do such an extension.

6. Access control and security

   N/A

7. Boundary conditions

   Start-up: Initialize the 3D stuff.

   Shutdown: Garbage collection should take care of everything for us.

   Error behavior: Garbage collection should take care of everything for us.

8. Points of variation

Possible changes, try to forsee

Selection of 3D graphics library. Some libraries may not support all platforms.

9. Glossary