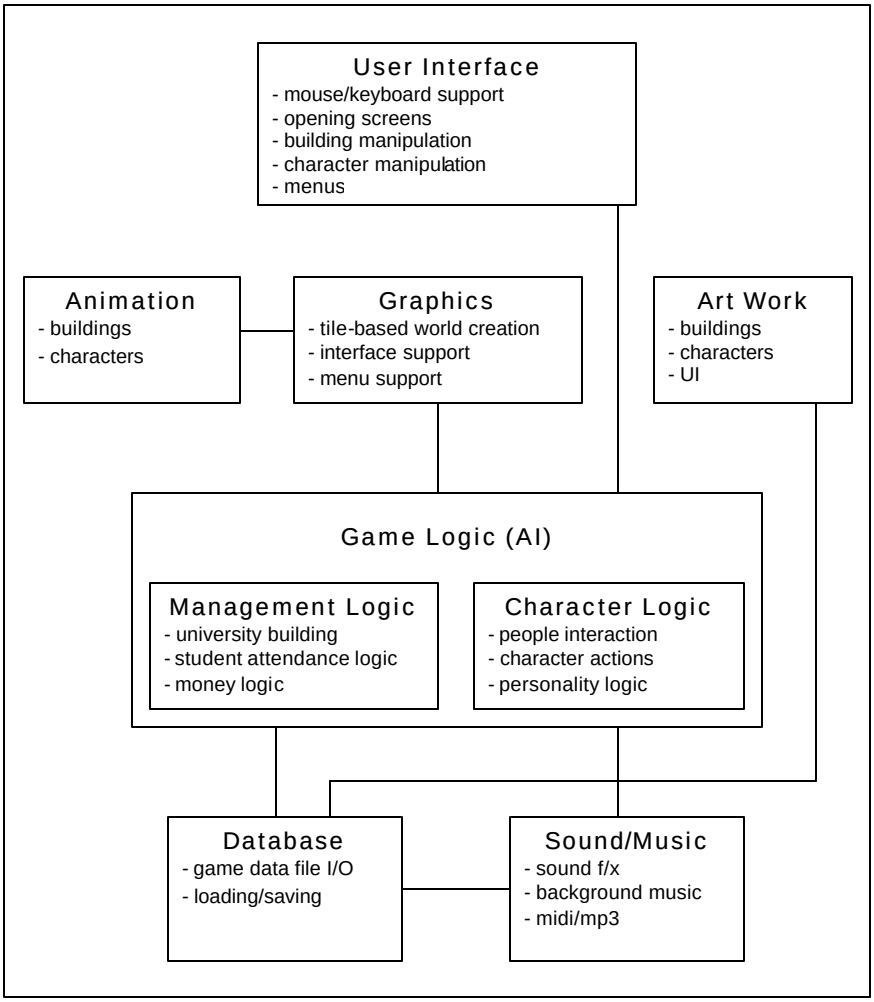
BROWN UNIVERSITY EDITION

Top-Level Design Document

Richard Hsieh (rhsieh) . 02.21.02

High-Level Component Diagram & Description



Game Logic (AI)

- Control the main aspects of player control in the game and decide the results of decisions the player makes.
- Management Logic & AI
    - Determines how building certain buildings and hiring professors will affect the how the game is played.

    - Manages where buildings can be placed and what can be built.
    - Controls how much money the player will get from student enrollment and how money is spent.
- Character Logic & AI
    - Determines how people will interact with another based on their personality ratings.
    - Controls how players will react if told to perform a certain action.
    - Decides what actions characters will take based on the schedule that the player creates for them (or that is automatically generated for them).
    - Determines where players will walk if told to walk to certain places on the campus, avoiding other players and buildings (collision detection).

Graphics
- Top-down view of the campus using an isometric tile-based map.
- Interface for manipulating buildings, people and the map.
- Menu support to allow user to set up game.

Animation
- Animate buildings to show activity going on inside of them.
- Animate people to have gestures, walking/running around campus.

User Interface
- Perform actions primarily by mouse clicks on graphical iconic menu on main map screen (like Warcraft III and many graphic adventure games) but provide keyboard shortcuts.
    - Clicking on people or buildings in game will bring up different features. When clicking on a building the user can select various options (upgrade, etc), or when clicking on a person options for modifying characters' lives will show up.
- Support for menus popping up during gameplay, and as a stand-alone screen.

Art Work
- Designing buildings from an isometric top-down view.
- Drawing different types of characters to show variety of students and to differentiate them from professors.

Database
- Method of storing data in a compact, efficient way.
- Retrieving information on each student/professor building efficiently.
- Creating load and save formats for storing campus information.

Sound/Music
- Creation of sound effects to play during the game.
- Support for background music which matches mood of game (mp3/midi).

External Dependencies

The largest external dependency is obtaining the art work that will show the buildings, people and interface in the program. It would be easiest if the art work were readily available, but it appears that someone would have to take on the role of obtaining, converting, and creating the art work for the game. This can be complex

(creating highly detailed buildings and people) or relatively simple (taking models from online sites such as 3dcafe.com or creating buildings from photos of buildings on campus). It is entirely up the person who takes on this role.

Another dependency is the graphics engine that the program will use. Currently, OpenGL seems to be the ideal choice for this as many other games have been written under this engine.

For sound, there is a Linux library available at http://www.opensound.com/ which allows a programmer to add sound/music to any C++ program. It is called the Open Sound System (OSS).

Task Breakdown & Group Organization

Project Manager. The project manager is responsible for the overall progress of the project. His duties involve making sure that each team member is communicating with the rest of the group and accomplishing the milestones in the schedule. The project manager makes the final decision for all decisions.
# of people: 1
suggested person(s) to fill role: rhsieh

Architect. The architect is responsible for knowing the design and working with team members to flesh out each person's role in the design.
# of people: 1
suggested person(s) to fill role: mplin, nige, cphartne

User Interface Architect. The UI architect is responsible for designing all aspects of the UI. This includes the menu system, the main interface panel during gameplay and any other interactions that the user makes with the program using his mouse or keyboard.
# of people: 1
suggested person(s) to fill role: awhull, msh, lrosenhe, tmohamed

Coder. The coder will implement and realize the design of the program. Each coder will be given a specific component to implement, given the detailed specifications worked on by the architect. The components are: Graphics, 2 Game Logic, Database, Sound/Music. Since graphics is such a large component, it would be helpful for the UI architect to work closely with the graphics coder. These two people together should find some way to get the art work for the game.
# of people: 5
suggested person(s) to fill role: bpeng (graphics), awhull (graphics), tyoon (graphics & logic), kfardig (logic & database), tneal (database), eboroson (music), dshue (music), rhsieh (music & graphics)

Documenter. The main duty of the documenter is to write the user manual. The additional role that the documenter will have is to act as the librarian. The documenter can work with the tester to do this, since during the design stages, these roles do not have as much work as others.
# of people: 1
suggested person(s) to fill role: hyazawa, lmolinar

Tester. During the programming process, the tester will help the coders test each of their components. The tester is responsible for recording each and every bug he finds and must organize some method to report these bugs to the other team members. After integration, the tester will make sure the entire program works.
# of people: 1
suggested person(s) to fill role: rmckenzi, dshue, hyazawa

Schedule

3/9:
- Create web page for team members to post messages and their progress.
- Meet together with whole team once before this point.
- Complete Final Top-level Design.

3/12:
- Coders and architects complete interface proposals for each component of the system.
- Begin setting up directory structure for CVS.

3/17:
- Provide feedback for interface proposals.

3/21:
- Complete interface definitions.

4/4:
- Complete detailed designs for each component and write mod specs for all modules.
- Begin programming each of the components.

4/16:
- By this date, a beta version of the program should be complete and integrated.

4/27:
- Integrate everything by this point.

5/2:
- Fix all bugs and demo project.

5/9:
- Submit final copy of program.

Assumptions

- Still haven't fleshed out the details of each aspect of the game. I believe some people are still unclear as to how exactly the control of flow will work.
- Each team member should write out his own schedule and have it approved by the program manager. Hopefully, each member will have specific milestones that he will have to complete rather than abstract ideas.
- I'm hoping that art work will not be too difficult to create. There are last resorts (such as taking photos of buildings on campus and creating OpenGL boxes of

them), so creating the art work in the game will vary depending on how people want the final product for this semester to look.