1 Storyboard and Conceptual Images

Provide a visual description of your project. Highlight the core ideas (compelling game idea and innovative concepts).

2 Project Software Engineering

Your course project is to develop and/or integrate a set of technologies that realize new features, capabilities, or avenues for interactive entertainment. At the end of the project, you will have created a working playable game that demonstrates (as a proof-of-concept) the novelty of your innovation. With high probability, you will not write the greater majority of code for this project. You will likely integrate existing code, libraries, online services, etc. with custom code that you will write.

Integrating innovative but unrefined techniques with external implementations is a tenuous proposition. Planning ahead and managing risk are the best means to address this uncertainty. To this end, we expect that you have a plan for the software engineering of your final project. Below are exerpts from Burback's dissertation [1] describing the basic software engineering process and phases.

The software engineering of a system follows four phases:

> There are four fundamental phases in most, if not all, software engineering methodologies. These phases are analysis, design, implementation, and testing. These phases address what is to be built, how it will be built, building it, and making it high quality.

You will define a specific plan for analyzing, designing, implementing, and testing your final project. Consider your customers to be: 1) the game consumer (i.e., you and people you know) who is looking for something new and interesting to play and 2) your investors (i.e., the course staff) who want to ensure the project is compelling for consumer interest and feasible given your available resources. Remember, the course staff will be very critical in evaluating potential failing points in your proposal.

On a related note, we highly encourage creative and nonstandard choices for external support resources (e.g., game engines, input devices, external libraries, etc.). However, the course staff will only support the Dojo engine for G3D.

[1] http://www-db.stanford.edu/ burback/watersluice/node2.html

2.1 Analysis Phase: Statement of Requirements

The analysis phase defines the requirements of the system, independent of how these requirements will be accomplished. This phase defines the problem that the customer is trying to solve. The deliverable result at the end of this phase is a requirement document. Ideally, this document states in a clear and precise fashion what is to be built. This analysis represents the "what" phase.

2.2 Design Phase

In the design phase the architecture is established. This phase starts with the requirement document delivered by the requirement phase and maps the requirements into an architecture. The architecture defines the components, their interfaces and behaviors. The deliverable design document is the architecture. The design document describes a plan to implement the requirements. This phase represents the "how" phase. Details on computer programming languages and environments, machines, packages, application architecture, distributed architecture layering, memory size, platform, algorithms, data structures, global type definitions, interfaces, and many other engineering details are established. The design may include the usage of existing components.

2.3 Implementation Phase

In the implementation phase, the team builds the components either from scratch or by composition. Given the architecture document from the design phase and the requirement document from the analysis phase, the team should build exactly what has been requested, though there is still room for innovation and flexibility. For example, a component may be narrowly designed for this particular system, or the component may be made more general to satisfy a reusability guideline... The implementation phase deals with issues of quality, performance, baselines, libraries, and debugging. The end deliverable is the product itself.

2.4 Testing Plan

The test plan defines the testing necessary to establish quality for the system. If the system passes all tests in the test plan, then it is declared to be complete. If the system does pass all test then it is considered to be of high quality. The more complete the coverage of the system, the higher is the confidence in the system: hence the system's quality rises.

The test plan could be considered as part of the design, which is the position taken here, or it could be considered as the first accomplishment in the testing phase. One of the goals of the design phase, is to establish a plan to complete the system, thus it is very natural to include the test plan. Also the trade-offs between alternative architectures can be influenced by differences in their test plans.

One single test will exercise only a portion of the system. The coverage of the test is the percentage of the system exercised by the test. The coverage of a suite of tests is the union of the coverage of each individual test in the suite. Ideally, 100 percent test coverage of the entire system would be nice, but this is seldom achieved. Creating a test suite that covers 90 percent of the entire system is usually simple. Getting the last 10 percent requires significant amount of development time.

3 Project Timeline/Milestones

Provide a timeline for achieving project milestones and realizing deliverables. List specific dates as deadlines for completing milestones. Tangible deliverables (in the form of weekly updating and demos) are expected to occur throughout the semester.

Note: Friday class meeting times during the semester are allocated for project weeklies (2-5 minute presentations).

4 Project Rubric: expectations for grading

Based on the stated software engineering plan, provide a rubric for evaluating your project. The rubric should provide a breakdown the various components of your project in terms of the percentage of their importance to the project. For each entry in the breakdown, specify the criteria for judging the quality of each component in the final implementation. This criteria should itemize what consists of quality at the 100

Note: that this rubric is only a suggestion for grading by the course staff. The actual assignment of grades is subject to the evaluations of the course staff.

The default rubric consists of:

Novelty of the project concept 20Thoroughness of the game implementation 30Thoroughness of the innovative component implementation 30Quality of weekly updates 20

This rubric is very nebulous and leaves grading open to the interpretations of the course staff. **It is to your advantage to propose concrete rubric with detailed specification of your expectations.**

4.1 Group Workload Breakdown

This breakdown should set individual rubrics for each person in the team and cover all aspects of the overall project.

5 Selection and approval of a mentor

The proposed project must have a mentor who is committed to provide guidance and conceptual support. The mentor can be a member of the course staff or another individual, approved by the instructor, with expertise in the area. The mentor must approve of the project before it will be accpeted by the course staff.

The course website maintains a list of potential projects suggested by individuals available to serve as mentors.