Project Milestone #2:
Detailed Planning, Specification, Design, and Test Plan

Goal: To refine the scope and feature set of the accepted project ideas. To refine the architecture and to develop a detailed design. To propose a staged product delivery plan, including a test plan for the team's product.

Due Dates:

Part I: Emailed presentations – Tuesday, July 11, by 10pm (to be presented on Wed, July 12, in class)

Part II: Electronic submission of all documents – Thursday, July 13, by 10pm

Introduction

The purpose of this group assignment is to accomplish the detailed planning, specification, and design required before you implement your ideas in code. In addition to the deliverables described in Boehm [2], we ask that you prepare a test plan [5] that reflects your vision of *what* you are going to test and *how* you are going to test it. This is needed because testability is often found to be highly correlated with design quality – it you can't test it easily, there is likely a flaw in the design.

Your team's results at the end of this stage will be presented as the Lifecycle Architecture (LCA) review milestone. You need to convince your audience that:

(a) you understand well what it is that you are building;
(b) you have a solid idea how to approach building it; and
(c) you have the necessary resources (time, personnel, technology, etc.) to build it.

For more background material on the content of the LCA review, refer to the in-class discussions on the topic, including requirements gathering and specification, and design techniques, as well as the references cited below.

Deliverables

Much of the material for this review is an elaboration of the material used in the LCO review. You can draw on that as a starting point but do not feel compelled to stick too close to it, especially if you believe that changes are necessary to improve its focus and/or scope. Since your goal at this stage is to accurately define the actual product to be built, the result of the LCA milestone should leave fewer options and open items (as compared to the LCO deliverable), and contain more decisions and detail.

As part of your deliverable we will expect to see the following documents:

1. Overview presentation (5 points). A set of PowerPoint presentation slides summarizing the LCA elements for your product. This is the pitch that your team will give in class.

Tip: In wrestling about what to include versus what to skip in your presentation you will likely ask yourself the same kinds of questions that your audience may wonder about as they listen to your pitch. One such question is, "What are the key aspects that the stakeholders (managers, developers, customers, etc.) need to understand in order to decide if this project will succeed in bringing value to the company and to its customers?"

2. Specification document (5 points). This document should accurately and as completely as possible reflect the product you are building from the point of view of both the customer(s) (what they want to see built) and the product administrator(s) (what service modules they will need in order to correctly run your product, even if those modules are not directly visible to the customers). See the lectures and the article by Joel Spolsky [3] for suggestions on content. Also, see the articles on writing specifications by Alistair Cockburn, cited on the course web under 'Useful Resources'.

3. Architecture document (5 points). This document is a detailed definition of the system and software components. It should carefully and clearly identify the modules and interfaces between modules required to implement the system [4]. (The modules should be specific, not just "client", "server", or "GUI".) This section should address the design of the system from the customer's viewpoint, as well as from that of developers and administrators. Use the lectures and the article by David Parnas, also cited under 'Useful Resources', for content suggestions.

Tip: Good interface definitions are invaluable at shedding light on the quality of a design. Diagrams are an important tool to understand components and their relationships.

4. Team structure, schedule, task assignments, and risk assessment (5 points). This document should describe your team structure (how you have organized the team, what the members' roles and responsibilities are), elaborate on milestones (external and internal), define tasks (broken down to a reasonable level of detail – e.g., not just "server functionality", but "server logging functionality and keeping track of game scores across all clients") and initial task deadlines, and specify the team member(s) responsible for each individual task. This should reflect your *actual* plan of work, possibly including items that your team has already completed (e.g., while preparing for the LCA milestone). Also, identify the high risk areas of the project and provide a brief analysis showing why you believe these will not become "show stoppers" for the project, as well as what your risk mitigation plans are for each risk area.

5. Test plan document (5 points). This document should describe what aspects you plan to test, why that would be sufficient for your product from the point of view of the customer(s), as well as how specifically you plan to test those aspects in a disciplined way (rather than merely "playing with" the latest version of the product in hopes of seeing if it eventually breaks somewhere). This implies outlining strategies for unit testing and system (acceptance) testing, along with a set of actual test cases (to be later expanded) by following a methodology such as SFDPO [5].

Note: Some of the above issues have not been discussed in class yet, so it is normal that they may sound unfamiliar to you at this point. By the time the LCA milestone arrives, we expect to have covered all the relevant material.

Mechanics

For this and all remaining group assignments in the course, you will be working in your team of 4.

While you are not strictly limited to any fixed (maximum) number of pages for each individual document, please keep in mind that conciseness is a virtue. (Your audience – managers and other developers – should not find it daunting to read long documents and as a result altogether skip reading what you have written.)

Note that the LCA presentations will *precede* the actual LCA submission deadline by a day. The intent is that you prepare your materials and do a presentation in class, then get preliminary feedback and make adjustments based on the concerns that have been raised by the audience, before finally submitting all the documents.

As there will be two teams to present in 1 hour, rehearse and time your pitches to last about 20 minutes each, allowing adequate time for follow-up questions and discussion.

Following the LCA submission we will schedule 30-minute informal feedback meetings with each project team on Monday, July 17 or Wednesday, July 19.

Evaluation
In our evaluation of your work, we will be looking to see that you have addressed all the necessary elements of an LCA review and have made reasonable decisions related to each. As before, we expect you to organize and present your work well too. (Refer to the lectures, the Boehm paper [2], and the test plan article [5] for more information about what is needed.)

Technological Resources
The department gives us access to web servers and database servers for your use, if needed.
If your product will depend on the availability of special software, be sure to ask upfront as we may or may not be able to obtain access to it and will certainly need time to check and make any arrangements.

Turn-in
Send us your presentations by 10pm on Tuesday, Jul 11 – the night before the actual in-class presentation. Please email those to *both* Valentin and Lincoln, so that we can load them on the presentation tablet. Remember to avoid using animations. Less busy backgrounds will be appreciated too.

For the final submission, by 10pm on Thursday, Jul 13, please have one person from each team do the turn-in so that all files from your group appear in the same place. The turn-in area will be activated shortly. The URL to it is already available on the course web.

Note: eSubmit, the UW Catalyst tool we use for managing electronic submissions, accepts files of size up to 2MB each, so plan accordingly. (If your files are larger in size, consider archiving them or, as a last resort, creating an archive with multiple volumes, each of which is no bigger than the allowable size.)

Resources
[1] *Rapid Development*, Steve McConnell.
[2] *Anchoring the Software Process*, Barry Boehm (USC)
http://citeseer.ist.psu.edu/boehm95anchoring.html
[3] *Painless Functional Specifications*, Joel Spolsky
http://www.joelonsoftware.com/articles/fog0000000036.html
[4] *Software Architecture*, David Garlan
http://www-2.cs.cmu.edu/afs/cs/project/able/ftp/encycSE2001/encyclopedia-dist.pdf
[5] *How Do You Spell Testing? – A Mnemonic to Jump-Start Testing*, James Bach
http://www.satisfice.com/articles/sfdpo.htm