Introduction to Homework 4

The purpose of homework 4 is to do the detailed planning and specification development for the application you will build during the rest of the quarter. This work will be compiled and turned in as the second Life Cycle Objectives milestone (LCO2) for your product.

Outline

Project go-ahead has occurred! Management and your fellow developers have bought all the glossy promises you made about the vast audience for your product and the feasibility of implementing it, and so now it's time to:

1. Get organized and decide who will do which tasks.
2. Decide exactly what this product is going to do and how it will appear.
3. Decide how to actually implement it, including support functions.

This is another turn around the development spiral. The milestone at the end of this turn is the Life Cycle Objectives review #2. A good LCO2 package will clearly define the product to be built, the tasks that need to be done, and the plan to accomplish these tasks.

Deliverables

The deliverables for this homework are all documentation. There is no code deliverable. However, you should not hesitate to write small prototypes as needed in order to clarify the capabilities and design of your application.

1. Specification document. For the client side of the product, the spec should contain specific views of the application as it will appear to the user in each state and details of the commands to go from state to state. For the server side, it should contain information about the application setup files, how to change them, and the most important user controllable parameters. This document defines the product you are building, so spend some quality time deciding what it should say!

2. Preliminary architecture document. The primary mission of this document at this stage is to identify the major functional blocks of the application, and identify and specify the interfaces between them.

   Of particular interest for the client side is a definition of the interface with the server. Also, any information that will be stored and retrieved from local persistent storage should be identified. For the server side, the interface to the client application is very important. Also, the interface to live data sources and persistent databases as well as setup and configuration controls should be given.

3. Task assignments. Task descriptions for the project and the specific team member responsible for each task.

Turnin

One of the team members should turn in all the deliverables **before midnight, Thursday May 1.**

References

Lectures

4 – Life Cycle
5 – Life Cycle Objectives
6 – System Requirements
7 – Project Teams

Papers

Anchoring the Software Process, Barry Boehm, USC
http://citeseer.nj.nec.com/boehm95anchoring.html

Painless Functional Specifications, Joel Spolsky
http://www.joelonsoftware.com/printerFriendly/articles/fog0000000036.html

Life Cycle Objectives. Description copied from Boehm.

The "Top-level system objectives and scope" part of the LCO milestone involves establishing the system boundary: the set of key decisions on what will and will not be included in the system to be developed. The part that will not be included will therefore be in the system's environment: key parameters and assumptions on the nature of users, data volume and consistency, workload levels, interoperating external systems, etc. These should be characterized not just at their initial operating levels, but in terms of their likely evolution, in order to avoid the point-solution difficulties discussed in the Introduction.

The "Operational Concept" involves working out scenarios [Carroll, 1995] of how the system will be used in operation. These scenarios may involve prototypes, screen layouts, dataflow diagrams, state transition diagrams, or other relevant representations. If the ability to perform in off-nominal situations (component failures, crisis situations) is important, scenarios for these should be developed as well. Scenarios for software and system maintenance need to be worked out, including determination of which organizations will be responsible for funding and performing the various functions. These organizations are some of the key stakeholders whose concurrence is needed for realistic and supportable system definitions.

The "System Requirements" in the next part of the LCO definition in Table 1 are not absolute cast-in-concrete specifications as in the waterfall or related contract-oriented models. Instead, they record the collective stakeholders' concurrence on essential features of the system, whose detail can be modified easily and collaboratively as new opportunities (reuse opportunities, strategic partners), problems (budget cuts, technical difficulties), or developments (reorganizations, divestitures) arise.

The definition of "System and Software Architecture" should be at a sufficient level of detail to support analysis of the architecture's feasibility in supporting the system's objectives and requirements. Having more than one feasible choice of architecture is acceptable at the LCO stage; an example would be the existence of two feasible central commercial-off-the-shelf (COTS) products with different architectural implications. However, if no architectural option can be shown to be feasible, the project should be canceled; or its requirements, scope and objectives reworked. A record of infeasible options which were considered and dropped should be kept as insurance that these options will not be adopted in ignorance later.

A critical component of the initial "Life-Cycle Plan" is the identification of the major stakeholders in the system to be developed and evolved. These frequently involve system user, customer, developer, and maintainer organizations. If the system is closely coupled with another system, the interoperator organization is a key stakeholder. If system safety, privacy, or other general-public issues are important, a representative of the general public should be a stakeholder. These are stakeholders whose concurrence on the system requirements is needed; otherwise, the system may not reflect their needs and will not be a success. Another critical component of the life cycle plan is the identification of the process model(s) to be used (waterfall, evolutionary, spiral, incremental, design-to-cost/schedule, or hybrid combination of these and others).

For the main part of the Life-Cycle Plan, an organizing principle is needed which scales down to provide simple plans for simple projects. A good approach is the WWWWWHH principle, which organizes the plan into Objectives (Why is the system being developed?); Milestones and Schedules (What will be done by When?) Responsibilities (Who is responsible for a function? Where are they organizationally located?); Approach (How will the job be done, technically and managerially?); and Resources (How much of each resource is necessary?). Using this approach, the essential decision content of a life cycle plan for a small, straightforward project can be packed into one page or two briefing charts.

*The most important thing to achieve for the Life Cycle Objectives milestone is the conceptual integrity and compatibility of its components abov*e. The element which assures this is the "Feasibility rationale." It uses an appropriate combination of analysis, measurement, prototyping, simulation, benchmarking, or other techniques, to establish that a system built to the life cycle architecture and plans would support the system's operational concept. A further key element of the rationale is the business case analysis, which establishes that the system would generate enough business value to be worth the investment.