Department of Computer Science

College of Science and Mathematics
Montclair State University, Richardson Hall

Course: CSIT 415 Software Engineering II– Sp 2015 Dr. H. Johnson

Project Due Dates:

Week 1: [1/22] Team organization. Submit team name and list of team members*

Week 3: [2/5] Test Plan Draft

Project:

Create a Dynamic Web-based Course Assessment Software System– (Implementation of project-design during Fall 2013 semester)

.

Content:

Title page

This, as well as **all other documentations submitted throughout the semester**, should be turned in with a title cover page that includes the following information:

- A title that identifies the document
- The name of your team (such as "Get Rich Quick, Inc.")
- The team members who contributed to the document
- Course name, instructor's name, date

Summary

In about two pages, you should briefly summarize the project you are working on. Describe what it does and who will use it. What needs will your system satisfy? How will it help the users? Outline the most important features of your system. Describe the physical environment in which your system will be used, including any other system with which the new system will interface. Are there any important performance goals for your system: time or space efficiency, security, or reliability?

Details concerning system user interactions

The main point of this section (which should be approximately ten pages) is to describe the functions that the system will perform from the point of view of the system user. You need to cover the kinds of inputs your system expects, the actions it will take on both expected and unexpected inputs, and the types of outputs that the user will see in those cases. Part of this section will eventually be developed into a user manual. First we'll consider the content, then the form, of these descriptions.

The inputs, state changes, and outputs should be described in reasonable detail. You need not stick to the exact wording of messages, but you should make definite statements. You can negotiate major changes later. Describe what legal values or ranges your system will accept for inputs, what precision or accuracy constraints you will follow, and how you will handle errors.

Description should employ some of the following techniques:

Sample transcript. A very important part of the description is a sample interaction with the system in the form of a transcript of a dialogue between a user and the system. Use upper and lower case or some similar convention to distinguish between what the program types and what the use types

.

Due Dates

Week 4: [2/12] Updated Test plan documents

Overview

Testing is **one of the most important phases** of a software project. The software team must therefore carefully plan and document the order in which components are to be integrated and the order in which the individual modules are to be completed and tested in isolation. Testing and debugging methods must be agreed upon, documented, and eventually carried out. **The grade given here will be tentative. A final grade will be assigned after you have completed the testing phase.**

Your goal here is to use the test plan to convince the management (in this instance, **me**) that a feasible test plan has been designed and also to provide the project members with directions for the testing phase of the project. This assignment outlines some constraints on the test plans. Several stages of tests must be scheduled, and several testing procedures must be explored. Scheduling is required for unit tests, integration testing, "functional" testing, performance evaluation, and an acceptance test (the demonstration).

These tests must include techniques of walkthroughs, extensive logic testing, input/output testing, and optionally verification. The following sections give details of the contents and style of the test plan document, and the final section lists some reading material that may aid you in designing a better test plan.

Design the test plan

The test and evaluation plan should contain the following components: statement of objectives and success criteria, integration plan, test and evaluation methodologies, and responsibilities and schedules. These components are described in more details in what follows.

1. ***Objectives and Success criteria***: The test plan document should contain a statement of the overall testing objectives of the individual tests that are planned.

2. ***Integration Plan***: An important decision to be made about testing is the order in which modules are to be combined and therefore the order in which they will he tested individually. You must plan for individual component tests, the combination of components during integration testing, a functional test, and an acceptance test.

   The test plan document should describe and defend your integration method. The "clasp your hand and pray" (which is similar to the "big bang") method is not acceptable. Various methods, or a combination of methods of integration are acceptable if convincingly defended.

3. ***Testing Methodologies***: You should design a plan to test all of the components of your system. Your test plan document should name at least one module to which each [testing] technique is to be applied;

   You may decide the testing technique for the other modules later on. Each person on the software team is to perform at least one set of tests and turn in a couple of pages describing the process at a later date.

4. ***Responsibilities and Schedules***: The test plan should provide a schedule describing dates and responsibilities

.

- First, determine an order in which to perform integration and functional tests. In scheduling, you should make use of the dependency graphs based on the external functions that the modules require.

- Next, this order should be used to determine the order in which the individual modules are to be tested. Determine the dates by which tests are to be completed and the individuals responsible for testing. Prepare a master test plan schedule and include it in the document.

- Finally, you should define a monitoring procedure to ensure that tests are designed and carried out on schedule. Someone will have to keep a record and report lack of compliance to the other team members. Similarly, there must be a procedure for reporting and correcting bugs.

Writing the document

The test plan document is to contain an introductory section that summarizes the whole document in a page or two and discussion sections that cover the plan in more detail. A total number of pages between 15 and 25 is about right.

Remember that this document is intended for several audiences. The document should be organized so that it is easy to find schedule summaries and monitoring plans and easy directions on personnel responsibilities. The test plan should not be very long; if you think that the writing of the document is taking significantly longer than the design of the test plan, consult with me. Representative samples of your tests are to be turned in later (as scheduled in the test plan).

The introductory section of the test plan document, which should only be about three pages long, should include the following:

- Overall objective and success criteria
- Summary of the integration plan - a list of tests and dates and people responsible
- Summary of the module-to-test-technique mapping for the four required testing techniques
- Summary of the monitoring, reporting, and correcting procedures
- Proposed dates for submission of individual test reports

The discussion section should contain:

- Defense for the integration plan (**about** half a page)
- Details of the tests with objective and success criteria for each test or group of tests (if you proposed most of the tests in your design document, there should not be more than 20 entries in this list, with each entry less than one page long; if you did not have good tests in your design document, however, you must include them here)
- Details of monitoring, reporting, and testing procedures (a page or so should be enough)
- Details of individual team member assignments (just a page or so, this is essentially cross reference list)

Week 6: [2/26] Initial Test Suite

Week 7: [3/5] Final version of test suite

Week 9 [3/19] Final version of the test plan, hand in **two copies**, one returned with comments, one for our files.

Design Documents (Revisited)

Week 10: [3/26] Detailed design due (2 copies) (major grade) module drivers for kernel due (compiled but not tested) reapportion and review remaining tasks in section.

The design process

To help you along with the design, a few milestones have been established for you. During the design process you should focus on the components of the design essential to producing a core system and merely outline extensions to a more complete system. One of your major goals is to have something worthwhile by your deadline, even if it isn't the full system. Hence, you should determine what needs to go into a skeletal system and schedule the design of those components first.

The end goal in the design process is a detailed design document that is complete enough to be a reference from which any competent [software engineer] computer scientist can produce code, test plans, or a user manual. (This document should eventually evolve into the code and system maintainers' guide.)

An intermediate goal is the overall design specification. This can be considered a draft of the detailed design document with some of the details missing. Interfaces between modules should be clearly defined. Once the design has been written, everyone in the group should read the entire document to make sure they understand the design. Individuals should then be assigned responsibility for particular modules/components and should design these in more detail.

Each person should carefully review at least one other person's section (and all sections should be reviewed by someone other than the author). Pick the one(s) with which your module has the strongest interaction(s). Based on the examination of the module interactions, you should prepare a set of interface definitions..

Finally, prepare the detailed design document. Most of the work will be done by individuals filling in the details of their modules, but you must review the document as a whole to make sure you are all using the same interface definitions, and to make sure that everything has been covered and that the parts of the document hang together.

Writing the documents

Your goal is to provide a recursive (top down) description of your system in a form something like the following. For each level you should cover the following:

Overall Design Document

For the overall design document, you should define your major data abstractions and modules, focusing on the abstract, design, and sub-modules, on error handling and exports and importance of the major modules/components. If you are in doubt as to
whether something belongs in the overall or detailed design, put it in the overall design with a note that you aren't sure. Fill in as many other details as possible - it will be to your advantage to see it all in writing and get some feedback. You should also include sections discussing the modification and the management questions. The document should be about 30-50 pages. **A table of content is a must for this document.**

Detailed Design Document

The detailed design document should include answers to all of the questions for as many levels as possible. Write an outline of the pseudo-code. Update the sections describing coding responsibilities. **A Table of Contents is necessary in this document**. If your document seems to be getting too long (more than 70 - 100 pages) consult me. **Good luck!!!!!**

(Test Plan Assignment)

Week 11 [4/2] Top level code for main modules, tested with drivers and stubs, due

User Manual

Week 12. [4/9] Draft due

Week 13. [4/16] Final version of User Manual

Week 15. [4/30] Final version of Project and Presentation

The User Manual should be a **self contained** description of how to install and use your system. A user manual, and ALL documents, should be **polished, professional pieces of technical prose** that a software company is proud to have accompany one of its products (Moreover, **this is a handy AND a proud accomplishment to show off at job interviews**).

The document should have a structure that is evident to someone who is reading it straight through and someone looking for a particular topic or fact. A table of contents is required; and the organization that it reflects carefully considered. An index and appendix might also be helpful.

Remember that the document should be completely self explanatory. Do not assume the reader has your specification (problem statement). You may of course edit the sections of prose from your previous documents. Do not discuss any implementation unless it directly affects the user's interface with the system

Documentation

You should prepare about 15 minutes of material that exhibit the best features of your system and allow about five minutes for questions and feedback from the audience. If your system has been properly designed, you can let the audience tell you what to input to show off your error handling and user help facilities..

Final Project Evaluation

The purpose of this document is to evaluate your final project. The document provides an opportunity to step back and put things in perspective and appreciate how much you have accomplished. If your group does not agree on evaluation and recommendations, feel free to write individual versions.

Final Week of Classes: On the last Thursday class meeting for the semester you (each team) will give a presentation/demonstration (approximately 15-20 minutes) of the functionality of the software system that was developed.

Demo Description

Required Submission:

1. On the day you give your presentation, you are required to turn in a f**older/binder/bounded - a polished, professional piece of technical prose-** containing:

   - the complete specification and design (workflow diagrams, use case realization, etc) for your system
   - a transcript of your test plan
   - test suites etc, along with the rest of the evaluation.
   - the code for your system
   - the user manual

2. Along with the hard copy of the materials specified in (1) you must also hand in a CD containing all the material specified above – spec, design, the code for the system, etc, along with instructions for loading and running the system (user's manual!!!)

NOTE: On the day you give your presentation, **each member of each team must have a complete (printed) set of all the material specified in (1) and (2)** for me to sign.

"Functional" Performance

Discuss how well your project satisfies your original specification. What has been added or subtracted? What advice would you give to the next group of students about writing specifications? Suggest improvements to the assignment description.

Now, think about the expected performance of your system and compare this to your earlier predicted performance (LOC, memory requirement, response time, calculation time, etc.). Can you come up with reasons for the discrepancies (if any)?

The Design Process

Were you satisfied with your design process? How helpful was the process of writing out the specifications? How useful were the use cases and activity diagrams? How would you go about the design process next time? What differences, if any, would you propose for the overall and detailed design assignments?

Testing

Did the order of integration that you chose seem to work out reasonably well? Did you stick to your test plan schedule? If not, why, and what would you change if you had to do it again? What method did you use for recording test results?

Was debugging easier or harder than you expected? Do you think that the design process significantly reduced the amount of time needed for debugging? Discuss the types of bugs you encountered. Were they limited to one component? Were they the result of errors in the interface specifications? How did you handle fixing bugs?

Management

What was the management structure within your group? Is this the structure you had originally planned? What problems (if any) did you have getting people to do their share of the work? What suggestions (if any) do you have on how this could have been scheduled better?

Compare the actual time spent designing, testing, debugging, and managing the project with your predicted times. Have you learned anything that will help you to make better predictions next time?
What suggestions would you give to next semester's students?

The Real World

Discuss what you might have done differently if this were a real world project – in an organizational setting
How much additional work would be required to complete the project to the level described in the specification?
How much would you consider selling the system for? Do you think the system is worth its cost? Justify your response.

Finally:

Submit a written critique of the course – what you liked/disliked and why. What are your suggestions for improving what you disliked (if anything)