Ben Kern
Dr. B.J. Johnson
CMSI 401
10/30/2019

Assignment 2

Problem 1

Write a short paragraph to answer these three questions:

- What are the two major concerns of any software project?
    - Cost
    - How long it will take
- Which do you feel is more important?
    - The time constraint is more important. Many software projects need way more time than they are given, and result in stressed out workers and poor quality of end product.
- Where does the idea of complete functionality fit with these two concerns?
    - The idea of complete functionality means the usually unattainable standard that both cost and time are on/under budget and on/under time, with everything that is needed & works great.

Problem 2

- In the Agile method for software development, what are the four main phases that occur in each and every iteration? Do you feel that any of them could be done at the start of the project and not be repeated in every iteration? Do you feel that would save time overall on the project? Justify your answers with a brief explanation.
    - The four main phases of the Agile method are requirements, design, code, and test. I feel that the requirements should be not repeated for every iteration because that will lead to code that is constantly changing if the requirements change every time. I feel that the requirements should be laid out at the beginning of the project with minimal changes. This would save time because the developers would know what they need to build and not have to worry so much about any impending changes. It would also save cost, as requirements that change usually imply that the cost will increase.

Problem 3

- In the Waterfall method for software development, what are the main phases that occur? How are they different from the phases in the Agile method? What other phases are in Waterfall that are left out of Agile? Do you think these are needed in Waterfall? Describe a situation using Agile in which one of these extra Waterfall phases might me needed.
    - The main phases of the Waterfall method of software development are requirements analysis, design, code, test, and maintenance. This is the same as the phases of the Agile method except for the maintenance phase. This extra phase is needed in Waterfall because the Waterfall method is done all in essentially one sprint, so there is not any allotted time to perform maintenance until the end. Agile methods might benefit from having a maintenance phase, however, as with

Agile, small mistakes in code or design can get through to the end of the sprint. A fully formed maintenance shift may catch these smaller mistakes and bugs if implemented.

Problem 4
Write one-sentence answers to the following questions:

- What is a "user story"?
    - A user story is the description of how the customer imagines interacting with the software you are producing for them.
- What is "blueskying"?
    - Blueskying is a grand brainstorming session between the customer and contractor where every one of the customer's grandest ideas are documented so that all the potential needs of the customer are laid out in front of everyone.
- What are four things that user stories SHOULD do?
    - All from pg 39:
    - They should describe one thing that the customer needs the software to do
    - They should be written in language that the customer understands
    - They should be written by the customer
    - They should be concise
- What are three things that user stories SHOULD NOT do?
    - All from pg 39:
    - They should NOT be long
    - They should NOT contain technical jargon that the customer might not understand
    - They should NOT mention specific technologies

Problem 5
What is your opinion on the following statements, and why do you feel that way:

- All assumptions are bad, and no assumption is a good assumption.
    - I disagree. While it is true that making assumptions in many cases is bad, however sometimes making assumptions are necessary to solve subsets of problems. An example is in many logical problems such as mathematical problems, it is necessary to assume some things about the input data, or about the nature of the problem parts.
- A big user story estimate is a bad user story estimate.
    - I agree with this statement. In software development, it is necessary to be able to think big, but not too big. If you are busy thinking about what could be in a project, you are not spending your time actually implementing these ideas.

Problem 6
Fill in the blanks in the statements below, using the following things [you can use each thing for more than one statement]: Blueskying; Role playing; Observation; User story; Estimate; Planning poker.

- *You can dress me up as a use case for a formal occasion*: User Story
- *The more of me there are, the clearer things become*: User Story

- *I help you capture EVERYTHING*: Blueskying, Observation
- *I help you get more from the customer*: Blueskying, Role Playing, Observation
- *In court, I'd be admissible as firsthand evidence*: Observation
- *Some people say I'm arrogant, but really I'm just about confidence*: Estimate
- *Everyone's involved when it comes to me*: Blueskying
- NOTE: when you have finished, check your answers with the result in your text on page 62. Do you agree with the book answers? If you disagree with any of them, justify your preferred answer.
    - For" I help you get more from the customer," I think blueskying fits with the statement better because if you can ask for ideas from the customer(s).

Problem 7

- Explain what is meant by a better than best-case estimate.
    - A better than best-case estimate is an unrealistic estimate that has a greater strain than the best-case estimate and has many underlying assumptions with it. There are many assumptions that tie into this estimate, and these are only the ones the programmer is aware about. This expectation is incredibly unrealistic and more demanding than the best-case estimate.

Problem 8

- In your opinion, when would be the best time to tell your customer that you will NOT be able to meet her delivery schedule? Why do you feel that is the best time? Do you think that would be a difficult conversation?
    - I believe the best time to tell the customer that the delivery schedule will not be able to be met is as soon as it is known that the delay is imminent. It being a difficult conversation depends on the temperment of the customer, but it is the best to do it this way because the longer that you wait to tell the customer, the more stressed and angry they are going to be that their demands are not met.

Problem 9

- Discuss why you think branching in your software configuration is bad or good. Describe a scenario to support your opinion.
    - Branching is necessary for minimizing the amount of bugs that get into the code of the overall product. It is therefore good and helpful for software development. Branching allows the software engineer to be able to roll back to old versions to be able to compare if bugs do all of a sudden appear, and they also allow project managers to see exact changes that are implemented in the project.

Problem 10

- Have you used a build tool in your development? Which tool have you used? What are its good points? What are it's bad points?
    - Yes, we use a build tool called Github Actions that does continuous integration and testing for our project. Its good parts are that it ensures that only successful builds pass into our Github repository and that all tests pass for every branch. It does not have bad parts as far as we are concerned.