Moriah Tolliver

October 30, 2019

CMSI-401

**Pilone and Miles Book Assignment**

**Problem 1**

The two major concerns of any software project are how much will it cost and how long will it take. I feel that the latter is more important, because if a project does not get finished on time or worse, never gets finished, then it is a waste of time, money, and resources that could be allocated to other projects. While an expensive project may cost more up front, the finished product will have more time to make up for the costs spent than a project that was really cheap but took too long to complete.

**Problem 2**

The four main phases that occur in every iteration of agile development are analysis, design coding or constructing, and testing. There may be times, like if a project is very small, when it is more beneficial to do analysis once at the beginning of the project then keep building until it is fully finished and ready for testing. This may save on overall time when the project requirements are very clear, so meeting to rehash them would not be efficient.

**Problem 3**

The main phases of waterfall development are requirements analysis, product design, system design, coding, testing, and maintenance. These are different from the phases in the agile method, because they do not happen iteratively, but in one sequence. Other phases that can occur in the waterfall development cycle include risk assessment, prototyping, quality assurance, production, marketing and exit strategy. Depending on the project these other phases may be needed in waterfall. For example, it may become necessary to assess whether or not the entirety of the project is worth working on, and while this process is not built into the agile method, it may need to be included.

**Problem 4**

* A “user story” is a story about how a user will interact with the software being built.
* “Blueskying” is brainstorming without any restrictions in order to capture every single idea.
* User stories should describe one thing that the software needs to do for the customer, be written in language that the customer understands, be written by the customer, and be concise.
* User stories should not be long, use technical jargon that is unfamiliar to the customer, or mention specific technologies.

**Problem 5**

I agree with the statement, “All assumptions are bad, and no assumption is a ‘good’ assumption,” because even if the assumption turns out to be correct, it is always better to ask or research to gain certainty especially if the point is arguable. I also agree with the statement, “A ‘big’ user story estimate is a ‘bad’ user story estimate,” because if there are too many details in a user story, it may be encompassing multiple users at once, so the story could be broken up and better estimated.

**Problem 6**

* 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: 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

I agree with the book answers. The only part I slightly contest is that Blueskying and observation cannot completely capture everything, but as iterations are done, the pieces that were missing from user journeys will come up.

**Problem 7**

A best-case estimate is an estimate that maximizes the value of taking a given action, therefore, a “better than best-case” estimate is one that may leave you with extra time or resources at the project’s completion.

**Problem 8**

In my opinion, the best time to tell the customer that I will not be able to meet their delivery schedule is after a couple of days of making sure this is case or right away if the deadline is coming soon. I feel that this is the best time, because if I rush as soon as I think I might not make it to say that I will not be able to make the deliverable, the account may be given away and I may find a solution pretty quickly that still makes the delivery date doable. However, I would also want to tell the customer pretty soon after I suspect that I will not be able to make the delivery date, because if they need to hire someone else, I would want them to do that in the time they need. This would definitely be a difficult conversation, as it would most likely cost me or my team money, but I would rather be honest about that and save the customer money and save myself some time.

**Problem 9**

Branching in a software configuration is good when there is a released version of the project that needs to be maintained away from development. However, most changes could probably be made from that one development branch instead of creating a lot of branches.

**Problem 10**

I have used npm as a build tool in my development. Its good points are that it can handle dependencies and run scripts, and it can handle these things over different branches. One of its drawbacks is that it heavily depends on packages that may need different operating system features to run properly and it cannot account for these differences.