CMSI 4071: Pilone & Miles Book Assignment
Assignment 2

Problem 1

Software projects revolve around making sure the product works as intended and ensuring that it runs efficiently. “Working well” means the program delivers the results it was designed for and meets user expectations. “Efficiency” on the other hand deals with how smoothly it performs things like speed and memory use. As a group, we believe functionality is slightly more important, because even the fastest or most optimized system is meaningless if it fails to do what users actually need. Still, efficiency matters too, since good performance enhances usability. In the end, strong functionality supported by solid efficiency is what makes a project truly complete.

Problem 2

In Agile development, every iteration includes the same five main phases: planning, designing, developing, testing, and reviewing. These steps repeat so the team can continuously adapt to feedback and improve the product. Some planning, however, happens at the start, Agile values short, ongoing cycles instead of long, one-time phases. It helps teams stay flexible and fix mistakes early. Trying to do everything at the beginning might look efficient but often causes more rework later. Teams can gradually build reliable software and save time overall by repeating these small steps.

Problem 3

The Waterfall model follows a strict process that moves through requirements, design, implementation, testing, deployment, and maintenance. Each phase must be completed before the next begins. It focuses heavily on documentation and planning, unlike Agile, which encourages frequent change and iteration. Waterfall can be slower but works best when stability and safety are priorities like in construction or government systems where you can't risk skipping documentation or introducing last minute changes. Even though it lacks Agile's flexibility, its structure ensures clarity and accountability for large and regulated projects.

Problem 4

A user story is a brief explanation of a software feature told from the user's point of view describing what they need and why. Blueskying means brainstorming freely without limits or practical restrictions, which allows creative ideas before narrowing down what's realistic. Good user stories will focus on clear goals and simple testing to encourage collaboration between the team and stakeholders. They shouldn't be overly technical or vague. The Waterfall method doesn't use user stories. Instead, it depends on detailed requirement documents written before development begins.

Problem 5

We don't agree that all assumptions are bad. Assumptions can definitely cause confusion or missing features, but small, discussed assumptions can help teams keep progress moving instead of waiting for every detail to be confirmed. Communication makes the difference. Assumptions can help when used as starting points, but never as conclusions.

As for the statement that a “big user story estimate is a bad user story estimate”, we agree. Oversized stories usually mean the task isn’t specific enough. Breaking them into smaller stories makes it much easier to estimate and complete them successfully.

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: Estimate

I help you capture EVERYTHING: Blueskying

I help you get more from the customer: Role playing

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: Planning poker

Problem 7

A “better-than-best-case” estimate means predicting an outcome that’s unrealistically optimistic. It assumes no bugs, no slowdowns, and no unexpected changes will occur. It’s a positive claim, but it usually sets false expectations and puts unnecessary pressure on the team. Realistic estimates always include uncertainty because no project ever goes perfectly according to plan.

Problem 8

If we knew we wouldn’t meet a customer’s deadline, the best approach would be to communicate that effectively ASAP. Being transparent early builds trust with the customer and shows professionalism. It’s not easy to deliver bad news, but transparency with the customer helps prevent larger issues later, especially the lack of trust.

Problem 9

Branching in software is helpful because it allows different parts of the project to develop at the same time without interfering with each other. Developers can develop safely in separate branches to keep the main version stable. Once the work is tested and ready, it can be merged back. In our group, we used branching to work on different parts of the project at the same time while keeping the main branch stable.

Problem 10

In our group, some of us have used CMake while others have used npm in past projects. CMake helped organize source files and generate build configurations across different platforms, while npm managed dependencies and automated scripts for web-related work. Both tools saved time by handling setup and repetitive tasks that would otherwise be done manually. The main challenge was keeping configurations consistent, but once everything was set up, both tools made collaboration and version control much smoother.