HW Assignment #1

Problem 1.1, Stephens page 12

Requirements gathering, high-level design, low-level design, development, testing, deployment, maintenance, and wrap-up.

Problem 1.2, Stephens page 12

Descriptions of tasks listed above:

* **Requirements gathering:** learn the customer’s wants and needs
* **High-level design:** describe the major pieces of the application and how they interact
* **Low-level design:** provide more detail about how to build the pieces of the application so that the programmers can actually implement them
* **Development:** write code to implement application
* **Testing:** use app under different circumstances to try and detect bugs or flaws
* **Deployment:** roll out applications to users
* **Maintenance:** implement any bug fixes, addition, future versions to the program
* **Wrap-up:** evaluate projects history to determine what went right and what went wrong so that you can use that knowledge in the future

Problem 2.4, Stephens page 26

Here is a link to the Google doc with some of our answers for this homework: <https://docs.google.com/document/d/1akoqikIoFRisS0sKtywwaMKl_ARGkd797ykoXJnnS20/edit?usp=sharing>.

You can view the version changes!

Problem 2.5, Stephens page 26

Just Barely Good Enough, the philosophy that one shouldn’t write any more code, documentation, or comments than are absolutely necessary.

Problem 3.2, Stephens page 51

The critical path is Start → A → D → E → M → Q → Finish. The total expected duration of the project is 31 working days.

Problem 3.4, Stephens page 51

The Gantt chart is stored as Homework01GanttChart.pdf in the same repository as the document.

Problem 3.6, Stephens page 51

One way to handle unpredictable problems is to add extra time to estimates to account for these unpredictable hiccups. You can also represent the time that lost time as its own task when planning. For example, create a task to represent sick time, and if sick time unexpectedly comes up, the time from the “sick time” task can be allocated to the task that may fall behind as a result.

Problem 3.8, Stephens page 51

The two biggest mistakes you can make while tracking tasks are ignoring lost time then hoping it will be made up later and adding more developers to a task that is already behind.

Problem 4.1, Stephens page 82

Five characteristics of good requirements:

1. Clear
2. Unambiguous
3. Consistent
4. Prioritized
5. Verifiable

Problem 4.3, Stephens page 82

Time Shifter program: list the audience-oriented categories for each requirement. Are there requirements in each category?

FURPS (Functionality, Usability, Reliability, Performance, Supportability)

1. Allow users to monitor uploads/downloads while away from the office (F)
2. Let the user specify website log-in parameters such as an Internet address, a port, a username, and a password (F, U, S)
3. Let the user specify upload/download parameters such a number of retries if there’s a problem (F, U, S)
4. Let the user select an Internet location, a local file, and a time to perform the upload/download (F, U, S)
5. Let the user schedule uploads/downloads at any time (R)
6. Allow uploads/downloads to run at any time (R)
7. Make uploads/downloads transfer at least 8 Mbps (P)
8. Run uploads/downloads sequentially. Two cannot run at the same time (F)
9. If an upload/download is scheduled for a time when another is in progress, it waits until the other one finishes (F)
10. Perform schedule uploads/downloads (F, U)
11. Keep a log of all attempted uploads/downloads and whether they succeeded (F)
12. Let the user empty the log (F, U, S)
13. Display reports of uploads/download attempts (F, U)
14. Let the user view the log reports on a remote device such as a phone (F, U)
15. Send an e-mail to an administrator if an upload/download fails more than its maximum retry number of times (F, U, S)
16. Send a text message to an administrator if an upload/download fails more than its maximum retry number of times (F, U, S)

There are requirements in each of the FURPS categories.

Problem 4.9, Stephens page 83-84

Brainstorm the Mr. Bones application and see if you can think of ways you might change it. Use the MOSCOW method to prioritize your changes.

* **Ads (M):** Most apps, especially games, make money off of displaying ads. The Mr. Bones app could do this by playing an ad in between each round of the game.
* **Game Levels (S):** The game currently has no method of incentivizing the user to continue playing, other than continuing to solve puzzles. Instead, the game should implement levels by starting on easy words and then progressing to longer words or phrases as the user progresses throughout the game.
* **Themes (C):** The app could feature different themes depending on the season and upcoming holidays. In this regard, Mr. Bones could have festive attire and the background of the game could vary depending upon the time of year.
* **Hints (C):** The app could give hints upon the user’s request if they need a clue. The user would have a limited amount of hints and they would only increase if the user solved X number of puzzles correctly without having to use one.
* **Time Constraint (W):** Force the user to pick a letter in a certain amount of time or they forfeit that turn.