NAU–CS Team Project Self-Reflection Worksheet

Overview: At the end of a project, it's useful to go back and reflect on how the project went, how the team functioned, how effectively you used tools, and so on. This worksheet is designed to guide you in this process, and capture the outcomes.

How to fill this out: Hold a final team meeting, after you've turned in the last deliverable and the heat is off. Order a pizza, crack open a beverage. Then sit down as a team and go through the following worksheet, discussing and filling in each section. Type up and the result, and email the document to your team mentor.

Grading Metrics: You will not be graded on the *content* of this document per se. That is, if for instance, your self-assessment concludes that you "didn't use version control tools effectively", then this shortcoming won't affect your grade; the point is that it should be an honest assessment. What you *will* be graded on is *how well* you fill in this document: thoughtful self-analysis gets a perfect score; cursory/lame/vague self-analysis will score low. We instructors use this document to help us think about how to encourage more learning and better teaming on projects, so please help us out!

Team Name: Lunar Pit Patrol

Team members: Evan Palmisano, Ibrahim Hmood, Alden Smith, Caden Tedeschi, Levi Watlington

Course number and name: CS-480: Capstone Experience Fall

Semester: **Fall 2024** Date this reflection completed: 12/12/24

Software DESIGN PROCESS

How did your team structure the software development process? Did you choose a particular formal model (SCRUM, Agile, etc.). If so, which one and why? If not, did you explicitly agree on an informal process…or was it just pretty random. Explain briefly.

We used the Agile development process progressing through multiple iterations of GUIs and utilized accessibility to a large population of astrogeologists for user testing so we had an effortless experience deploying our product for testing and addressing the issues that popped up.

How did it go? Now briefly discuss how satisfied you were with this process. Did it work well for this project? Why or why not?

As a team, we thought this process worked well since we were able to hone in on the big problems that we experienced throughout the development process. Since we were frequently reviewing what we accomplished, we could observe issues more frequently and provide solutions.

What changes might you make in your development process if you have to do it again? More structure? Less? Different process model?

Agile worked excellently for the team. The only changes we think would be applicable would be maybe an extra cycle for application deployment to sponsor desired stretch goals.

Software DEVELOPMENT TOOLS

What software tools or aids, if any, did your team members use to support or organize software development? For each of the following categories, list the tool(s) used, and briefly describe how the tool was used. If you didn't use a formal tool, explain how you handled the matter with informal means.

- Source creation tools: IDEs, text editors, plugins, anything used to edit/create sources.
    - VSCode was the main IDE used across the team. No plugins were needed aside from the standard python formatting extensions.

- Version control: How did you manage your codebase?
    - We used GitHub as our version control.

- Bug tracking: How did you keep track of bugs, who was working on them, and their status
    - We used GitHub issues to track our bugs mainly derived from user testing.

- UML modelers and other miscellaneous tools:
    - Draw.io was a critical resource in developing UML and Architecture diagrams

How did it go? Comment on any problems or issues related to organizing the coding process. How might you have managed this better? Were some tools you used superfluous or overkill? What tools or mechanisms would you try next time to deal with those issues better?

TEAMING and PROJECT MANAGEMENT

Without getting caught up in detailed problems or individual blame, take a moment to think about how your team dynamics worked overall. Here are a few questions to guide you:

How did you organize your team? Did you have some clear distribution of team roles (leader, technical lead, documentation lead, etc.) upfront? Or was it more just "everyone does everything as needed"?
Team Lead: Evan Palmisano
Customer Communicator: Ibrahim Hmood
Architect: Caden Tedeschi
Release Manager: Alden Smith
Recorder: Levi Watlington

How did you communicate within the team? Comment on each of the following communication mechanisms:

- Regular team meetings? If so, how often?

Team meetings were hosted weekly in addition to our weekly mentor meetings. If there was a class lecture scheduled we would meet either before or after the lecture session otherwise for non-lecture days we would meet remotely over Discord.

- Impromptu team meetings? If so, roughly what percent of total team meetings were of this sort?

Only a couple impromptu meetings were held. These occurred if we needed to record for a video or presentation dry runs.

- Emails to all members? If so, explain briefly: about how often, what used for?

Emails were used to communicate with the team's mentor

- Software tools? Were any of the software tools you mentioned above (e.g. bug/issue tracking) using to communicate and organize tasks, e.g., in lieu of emails or other discussion?

GitHub was a critical tool used in our software development process. It allowed us to develop new features in isolated environments, pronounce bugs and issues, and keep track of version control.

- Other communication channels used? Facebook, wiki, text messages, phone conferences, etc.

We used a discord server for team communication and we all shared a Google Drive so we could distribute deliverables that the team needed to work on to everyone.

How did it go? Did you feel that intra-team communication overall went well? Were there breakdowns, e.g., where someone didn't know something was due, didn't realize a task had been assigned to him/her, did not know about a deadline, etc.? Without getting into details, simply comment on whether such breakdowns occurred, what the overall cause was, and how serious (if at all) the consequences were.

The communication among the team worked out well. There were a few times where some team members forgot information that we discussed, but when that happened we were able to refresh their memory or point them to the channel that the team was using so they could go back and read what was discussed. Anytime a lack of communication was present, the discord '@' feature was used to grab attention.

What could you do better? More structured leadership? A more formal task assignment/tracking system? Using better/other communication mechanisms? Generally, just think about what you all would do next time to improve communication and avoid breakdowns mentioned.

Overall, the Lunar Pit Patrol had a clear scope of the capstone project given. While the project went smoothly, there is always room for improvement. Overall, the workload could have been more distributed amongst team members when it came to the development of the application and the filing of the weekly task report. Other than the two major factors. The team could have done better at keeping track of updating the Gantt chart

Nice work! Congratulations on finishing your project! Please enter all of your answers in this electronic document and send it off to your instructor or team mentor.

Some closing thoughts…

Spend a little more time on your own percolating on the answers you gave in this self-reflection exercise. Being effective as a project team is ***not easy*** (!!), and is a skill that we all have to work on continuously. There is rarely any single or simple reason why a project was a bumpy ride; usually it's a combination of factors…of which is YOU. Regardless of project or team, there are things that could have been done differently to make it flow better. Recognizing those things through thoughtful reflection post-facto is the key to improvement!

The Lunar Pit Patrol project highlighted the importance of teamwork, communication, and adaptability. Our Agile approach worked well, helping us address challenges iteratively, though we could improve task distribution and project tracking in the future. Minor communication breakdowns were quickly resolved, reflecting our commitment to keeping the project on track. Overall, this experience taught us valuable lessons about collaboration and problem-solving, which we'll carry into future endeavors.

Thank god it's over. - Alden